Baiting For Defense Against Stealthy Attacks on Cyber-Physical Systems

by

David B. Flamholz

Bachelor of Science, Massachusetts Institute of Technology (2015)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2019

© Massachusetts Institute of Technology 2019. All rights reserved.

Signature redacted

Author .

✓
Department of Mechanical Engineering
January 15, 2019

Signature redacted

Certified by.....

.....
Anuradha M. Annaswamy
Senior Research Scientist
Thesis Supervisor

Signature redacted

Accepted by

Nicolas Hadjiconstantinou
Chairman, Department Committee on Graduate Students

Baiting For Defense Against Stealthy Attacks on Cyber-Physical Systems

by

David B. Flamholz

Submitted to the Department of Mechanical Engineering
on January 15, 2019, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

The goal of this thesis is to develop a defense methodology for a cyber-physical system (CPS) by which an attempted stealthy cyber-attack is detected in near real time. Improvements in networked communication have enabled vast and complex dynamic control systems to exploit networked control schemes to seamlessly integrate parts and processes. These cyber-physical systems exhibit a level of flexibility that was previously unavailable but also introduce communication channels that are vulnerable to outside interference and malicious intervention.

This thesis considers the effects of a type of stealthy attack on a class of CPS that can be modeled as linear time-invariant systems. The effects of this attack are studied from both the perspective of the attacker as well as the defender. A previously developed method for conducting stealthy attacks is introduced and analyzed. This method consists of injecting malicious actuation signals into the control input of a CPS and then designing a sensor attack to conceal the effect of the actuator attack. The result is an attack that cannot be detected upon inspection of the Kalman filter residual. Successful implementation of this attack is shown to require the attacker to attain perfect model knowledge in order for the attack to be stealthy.

Based on the execution of past attacks on CPS, this thesis proposes an attacker who starts their attack by "fishing" for critical and confidential system information such as the model parameters. A method is then proposed in which the defender attempts to feed the attacker a slightly falsified model, baiting the fishing attacker with data that will make an attack detectable. Because the attacker's model is no longer correct, their attack design will induce a mean-shift in the Kalman filter residual, breaking the stealthiness of the original attack formula. It is then shown that the defender can not only detect this faulty attack, but use observations of the Kalman filter residual to regain more accurate state estimates, mitigating the effect of the attack.

Thesis Supervisor: Anuradha M. Annaswamy
Title: Senior Research Scientist

Acknowledgments

I would like to thank my advisor, Dr. Anuradha Annaswamy for her guidance and support throughout my research process and for teaching me new tools and ways of thinking along the way. I appreciate the time she has taken to smooth my transition into academic research and help me reason out how to handle the challenges I faced. I would like to thank Dr. Eugene Lavretsky for his advice and support along with the support of the Boeing Strategic University Initiative.

I owe special thanks to a number of friends and colleagues who supported me throughout this process and spent many hours listening to me practice presentations, helping me understand tough concepts, and helping me de-bug problems with my simulations. I would like to extend special thanks to my colleagues in the Active Adaptive Control Laboratory, especially Joseph Gaudio, Benjamin Thomsen, and Seyed Mehran Dibaji for their invaluable help and input. My brother, Avi Flamholz, as well as my girlfriend, Amit Schechter, also provided significant technical feedback.

Among many other friends, I would like to thank my family whose support has given me so many opportunities. They have made it their priority to ensure that no resource is out of reach and loved me unconditionally.

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

1	Introduction	13
1.1	Background	13
1.2	Contributions	18
2	Models of the Dynamic System and the Attack	21
2.1	Dynamic System Model	21
2.2	Attack Model	23
2.3	Design of a Stealthy Attack	24
3	Baiting For A Detectable Attack	29
3.1	Effect of Model Perturbation on Attack	29
3.2	A Defender's Perspective of the Baiting Approach	31
3.3	Least Squares Implementation	34
3.4	Steps of the Baiting Defense Recipe	38
3.4.1	Baiting: Step 1	38
3.4.2	Baiting: Step 2	39
3.5	Estimating Time of Attack Onset	40
4	Simulations	43
4.1	Preliminary Details	43
4.2	Simulation of a Non-Linear Dynamical System	44
4.2.1	Attack Comparisons and Results of Baiting Step 1	47
4.2.2	Results of Baiting Step 2	49

4.3	Estimating Time of Attack Onset for Improved State Estimation . . .	53
5	Concluding Remarks	59
5.1	Summary and Conclusions	59
5.2	Future Work	60

List of Figures

1-1	Vulnerability of an Aircraft CPS	14
1-2	Fundamental Pillars of Securing and Compromising CPS	14
1-3	Deception Attack	17
1-4	Attacker Fishing for Model Data	19
2-1	Deception Attack With Anomaly Detection	23
2-2	Stealthy Attack Design	27
3-1	Baiting a Fishing Attacker with a False Model	31
4-1	Comparison of Healthy and Attacked CPS	47
4-2	Comparison of State and State Estimates When Using a 99th Percentile Threshold Detector	49
4-3	Comparison of State and State Estimates When Attack is Detected at 0.2 Seconds	51
4-4	Comparison of State and State Estimates When Using an Ideal Detector	52
4-5	Residual RMSE as a Function of Estimated Time of Attack Onset . .	54
4-6	Estimated Actuator Attack as a Function of Estimated Time of Attack Onset	55
4-7	Comparison of State and State Estimates Upon Estimating Time of Attack Onset	56

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

4.1	Mean Percent Change in RMSE With Updated Estimates Over 50 Simulations	53
-----	---	----

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

1.1 Background

Over the past few decades, and with increasing intensity over the past few years, vast and complex dynamic control systems have relied on networked control schemes to seamlessly integrate parts and processes. These systems have come to be called Cyber-Physical Systems (CPS) and can be found in fields such as power, manufacturing, and transportation. As a result of this networked communication and control, components that were once separate or physically connected could no longer be treated as isolated from other parts of the system and the world, but rather in constant communication with them over potentially insecure channels. Thus, while networked communications enable flexibility in CPS, they introduce vulnerabilities in the form of potential entry points for corrupted data. As an example, corruption of vulnerable actuator and sensor channels of an aircraft CPS, as shown in figure 1-1, has the potential to induce instabilities in the system as well as significant damage. While in the context of classical cyber-attacks, problems related to cyber security sought to contain the damage that could be done to information and cyber-processes, vulnerabilities in CPS now give us cause for concern about damaging and potentially unsafe outcomes in the physical world.

Concerns of cyber security arose out of the older field of information security (InfoSec) which developed significantly over the course of the 1970s. During this

Figure 1-1: Vulnerability of an Aircraft CPS

period, some of the first personal computers were introduced, as well as a variety of cheap and standardized software packages [2]. In 1975 Salzer and Schroeder classified three fundamental threats to information security: “unauthorized information release, unauthorized information modification, and unauthorized denial of use [19].” In 1987, the Johnson Space Center labeled the corresponding security concerns as the CIA-triad, standing for Confidentiality, Integrity, and Availability [3]. These three pillars are commonly viewed as the backbone of a secure system. However, from the perspective of an adversary attempting to break security, these three pillars have corresponding threats that compromise the system according to vulnerabilities in each pillar. These threats or attacks are concisely summarized by Salzer and Schroeder’s words and can be referred to as Disclosure, Deception, and Disruption attacks [1] or DDD attacks. [6] offers more in depth coverage of how CIA and DDD play out in CPS but it is not hard to see that the same defense and attack concepts that once applied to information security, now apply to cyber-physical systems.

Defender’s Perspective	Attacker’s Perspective
Confidentiality	Disclosure attack – ex. eavesdrop
Integrity	Deception attack – corrupt signals
Availability	Disruption attack – block, delay

Figure 1-2: Fundamental Pillars of Securing and Compromising CPS

An issue of interest is how insights from systems and control can be used to detect,

prevent, and mitigate attacks on CPS. In particular, much attention has been paid to fault and anomaly detection [13] as well as design for robust response [5] in the fields of estimation and control. While control theoretic methods of the past have treated anomaly detection and response in the cases where anomalies were viewed as exogenous disturbances, a control theoretic response for defense against CPS-attacks needs to account for a malicious actor who cannot be simply modeled as a random disturbance. Therefore, further research is warranted in designing systems that can detect and react to a malicious attacker. In this thesis, we develop a method by which a worst-case attack can be detected through a suitable analysis and design of a dynamic system.

In order to motivate the focus of the cyber-attack in this thesis, we return to two major attacks that have garnered significant attention. The first of these, Stuxnet, was a cyber-physical attack on an Iranian uranium enrichment plant resulting in damage to approximately 1000 nuclear centrifuges [7]. In targeting a commercially available programmable logic controller (PLC) operating under a narrow set of conditions, the attackers were able to ensure that the attack reached its desired recipient with limited fallout [17]. Stuxnet was designed to spy on industrial systems [15] and employed a cyber-attack tool, known as a rootkit, which enabled the malware to lie dormant in the system and go undetected [17]. This presents an opportunity for a malicious attacker to perform reconnaissance, sitting quietly in a system undetected while, perhaps, observing critical and confidential system data. Upon observing sensor outputs of the system under stable conditions, the Stuxnet attack was able to replay those measurements, while injecting malicious actuation signals which did significant damage to a number of centrifuges [17].

The second major attack of interest is a cyberattack on the entire power grid of the country of Ukraine. In 2015 and 2016, a complex series of cyber attacks were launched on essential Ukrainian power distribution and transmission networks causing outages as well as lasting damage [8]. The 2015 attack was introduced to the network via phishing emails containing the BlackEnergy malware. Once it infiltrated the system it enabled the attacker to steal critical system data and study the system environment,

enabling what would become a catastrophic attack [4] with effects on the Ukrainian power grid that have lasted to this day.

Motivated by the execution of the aforementioned attacks, this thesis envisions an attacker that starts their attacks by collecting useful and confidential data about the system. As mentioned above, the Ukraine attackers gained entry into the system through phishing emails, enabling access to confidential data. We propose a broader notion of an attacker "fishing" for confidential data, not restricted to emails as the entry point of access. Both of the above examples show an attacker fishing for confidential data which they can then use to construct critical system knowledge and as a result, develop a much more dangerous attack. For the duration of this thesis, we examine the security of CPS from both an attacker's perspective and a defender's perspective. The attacker's perspective is utilized to address how the attacker not only wants to infiltrate the system, but to gain critical and confidential system knowledge, such as the model parameters, in order to enable the design of an effective attack. As will be shown in subsequent sections, the stealthy and effective nature of this attack essentially causes the system to be driven to an unanticipated and possibly dangerous state with the defender unaware of the presence of the attacker. From the perspective of the defender, on the other hand, it is useful to design a system that not only keeps an attacker out, but alerts the defender to an attacker's presence and perhaps mitigates the effect of an attacker's actions. In what follows, we will refer to a successful, undetectable attack as stealthy and an unsuccessful attack as detectable.

The specific cyber-attack model that we focus on in this thesis is that proposed in [11] which outlines a recipe for a stealthy attack on a flight control system. The authors of [11] assume that the attacker has access to a subset, if not all, of the actuator and sensor signals and can manipulate them, resulting in the deception attack shown in figure 1-3. In addition, the authors also assume that the attacker has full and accurate knowledge of the parameters of the linear dynamic system. It is then shown that the attack recipe ensures that the attacker can remain undetected and cause the estimation error to grow without bound. While the assumptions made in this paper are arguably unrealistic, one can view the attack recipe as a worst case.

Figure 1-3: Deception Attack: Malicious injection of control and sensor signals. Injected actuator attack, given by $\mathbf{a}_c$, and sensor attack, given by $\mathbf{a}_o$, cause the control input that reaches the plant, $\mathbf{u}_a$, and the sensor measurement that reach the controller, $\mathbf{y}_a$ to be different from the uncorrupted $\mathbf{u}$ and $\mathbf{y}$

In this thesis, we propose a defense mechanism to address this worst case, which is constructed so as to *bait* the attacker to reveal themselves.

Defense mechanisms that cause the attacker to be detected have been studied in a number of papers. The most notable of these are [17] and [20], and have been labeled as watermarking, since they correspond to authenticating signals of interest. Replay attacks of the kind conducted in the Stuxnet attack are considered in [17], and the *watermarking* based defense mechanism consists of adding a random perturbation to a standard feedback control input. In the presence of such a perturbation, any replayed signal will not be capable of producing a detection signal that corresponds to the correct distribution, and therefore enable attack detection. Care is taken to ensure that such a detection is enabled under minimal departure from the nominal feedback controller which is designed in an optimal manner. In [20], Satchidanandan and Kumar study the effects of this added perturbation signal on a more general set of sensor attacks and show that a set of uncompromised or “honest” actuators, each injecting their own added perturbation signal, can be used to check the honesty of the sensors which should report back measurements that contain some history of the perturbation signals’ effects. More specifically, they analyze a number of systems including SISO and MIMO linear systems with gaussian noise, SISO ARX models, and SISO ARMAX models and show that the asymptotic behavior of a system with the perturbation signal constrains the damage a sensor-spoofing attacker can do while going undetected.

A second approach that has been used for attack detection is termed “moving target approach”, an example of which is presented in [24]. The attacker is assumed to have access to input-output channels, similar to [11], and therefore is capable of obtaining the true model parameters. The defense approach proposed then consists of augmenting the state vector with pseudo-states, the dynamics of which is created using a linear time-varying system and therefore difficult to carry out system identification on the associated model parameters. This in turn causes the underlying system to be a moving target and therefore difficult to stealthily attack. The specific use of such a time varying dynamics can be viewed as generating a watermark on the system which will enable authentication.

1.2 Contributions

In contrast to the watermarking approaches considered in the papers above, the method proposed in this thesis focuses on a parametric perturbation rather than an input perturbation or the addition of extraneous states. In contrast to replay attacks or the assumption that the attacker has knowledge of the system model, we assume the worst case recipe as in [11] which not only corresponds to a known system model but also full access to sensors and actuators which are then used to engage in the attack in a stealth mode. The parametric perturbation proposed will then be shown to bait the attacker into revealing themselves.

This thesis envisions an attacker that "fishes" for model parameter data in confidential documentation with the goal of using this model data to construct a stealthy attack. The objective of this attack is to generate large state estimation error while going undetected. To go undetected, the attacker's goal is to maintain a Kalman filter residual that looks identical to that of a healthy system. To ensure the attacker does not have a correct model, it is proposed that a defender anticipating this type of malicious intervention plants a falsified model in their documentation as bait for the fishing attacker. This falsified model contains the parametric perturbation discussed above. Upon employing the falsified model in their attack design, the baited attacker

Figure 1-4: Attacker Fishing for Model Data [14]

will reveal themselves by generating a Kalman filter residual that has a mean-shift from that of a healthy system.

While this first step of baiting enables the attack to be detected, this thesis proposes a second step in which observations of the residual combined with recursive least squares estimation offer an updated state estimate. Therefore, baiting will not only act to detect the presence of this class of stealthy attacks, but to enable more accurate state estimation in the presence of these attacks.

The baiting approach proposed here can be viewed as similar to a cyber-security tool termed honeypot [21]. This denotes an information system resource to guard against any probes, attacks, or compromises [22]. For example, someone anticipating a cyber-attack, may use a secondary computer as a honeypot, acting as bait for an attacker to infiltrate. In this way, the honeypot can be used to detect attacks and even learn about the attacker’s methods and capabilities [22]. A special type of honeypot, called a honeytokens, is a smaller, and often fabricated piece of data, rather than a larger system such as a computer [21]. In accessing and probing the honeytokens, which was not meant to be accessed or probed, the attacker might alert the defender to the intrusion [21]. While the intended effect of the proposed baiting approach in this thesis is quite similar to these honeytokens, the application, as well as specific details associated with the methodology are quite distinct.

Motivated by a technique in which aircraft attempt to evade detection by enemy radar through the use of decoy targets, the authors of [9] suggest a cyber-attack

defense method in which intentionally planted decoy exploits are placed in code. Military aircraft sometimes scatter aluminum strips, creating many sources of reflection for an enemy radar and ultimately making it hard for an enemy to detect an aircraft among a host of decoys. These aluminum strips are called chaff. Therefore [9] refers to the bugs it uses as decoy exploits to stymie an attack as *chaff bugs*. Since attackers often look for bugs that can be used to exploit a system’s vulnerabilities, in introducing a significant number of intentionally placed benign chaff bugs, [9] proposes that an attacker will be forced to waste time and resources on trying to exploit many non-exploitable parts of the code. Like honeypots, this method is also conceptually similar to the baiting defense proposed in this thesis and reinforces how similar techniques are being explored in the field of cyber-security. We now extend our focus to the defense of CPS against stealthy attacks.

This thesis proceeds as follows: In chapter 2, we present a model of the dynamic system we will be studying as well as an attack model. Then, in section 2.3, we present a stealthy attack design. Chapter 3 develops a method to ensure that attacks are detectable and updated state estimates can be computed. The effects of the stealthy and detectable attacks on the lateral directional dynamics of an aircraft are simulated in chapter 4 and conclusions as well as suggested next steps are given in chapter 5.

Chapter 2

Models of the Dynamic System and the Attack

2.1 Dynamic System Model

The starting point for our discussion is a dynamic system whose model is given by

$$\begin{aligned}\mathbf{x}(k+1) &= A\mathbf{x}(k) + B\mathbf{u}(k) + E_1\mathbf{w}(k) \\ \mathbf{y}(k) &= C\mathbf{x}(k) + E_2\mathbf{v}(k)\end{aligned}\tag{2.1}$$

with the state $\mathbf{x}(k) \in \mathbb{R}^n$, the input $\mathbf{u}(k) \in \mathbb{R}^m$, and the measured output $\mathbf{y}(k) \in \mathbb{R}^p$. In many cases $n > p$. Further $\mathbf{w}(k) \sim \mathcal{N}(\mathbf{0}, Q)$, and $\mathbf{v}(k) \sim \mathcal{N}(\mathbf{0}, R)$ are uncorrelated process and output noises respectively, where $\sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ denotes a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix Σ . A, B, C, E_1 , and E_2 are all matrices of the appropriate dimensions and the system is assumed both controllable and observable. It is assumed that the state $\mathbf{x}$ is not measurable, and therefore needs to be estimated. Suppose that a standard Kalman filter is employed to estimate $\mathbf{x}$

based on the measurements $\mathbf{y}$ using the following equations:

$$\begin{aligned}\hat{\mathbf{x}}(k+1|k) &= A\hat{\mathbf{x}}(k) + B\mathbf{u}(k) \\ \hat{\mathbf{x}}(k+1) &= \hat{\mathbf{x}}(k+1|k) + L\mathbf{r}(k+1) \\ \mathbf{r}(k+1) &= \mathbf{y}(k+1) - C\hat{\mathbf{x}}(k+1|k)\end{aligned}\tag{2.2}$$

L is the steady state Kalman gain and is given by:

$$L = P_a C^T (C P_a C^T + E_2 R E_2^T)^{-1}\tag{2.3}$$

where $P_a = \mathbb{E}[(\hat{\mathbf{x}}(k|k-1) - \mathbf{x}(k))(\hat{\mathbf{x}}(k|k-1) - \mathbf{x}(k))^T]$ is the steady state a priori error covariance computed by solving the discrete time algebraic Riccati equation:

$$P_a = A P_a A^T + E_1 Q E_1^T - A P_a C^T (C P_a C^T + E_2 R E_2^T)^{-1} C P_a A^T\tag{2.4}$$

The Kalman filter drives the steady state estimation error, $\mathbf{e}(k) \triangleq \mathbf{x}(k) - \hat{\mathbf{x}}(k)$, to a zero mean noise signal: $\mathbf{e}(k) \sim \mathcal{N}(\mathbf{0}, P)$. P is the steady state estimation error covariance given by:

$$P = (I - LC)P_a\tag{2.5}$$

Meanwhile, the difference between the measurement and the predicted measurement known as the residual, $\mathbf{r}(k)$, can be shown to be a zero-mean white noise signal of known variance [16]: $\mathbf{r}(k) \sim \mathcal{N}(\mathbf{0}, \Sigma_r)$ where Σ_r is given by:

$$\Sigma_r = C P_a C^T + E_2 R E_2^T\tag{2.6}$$

While $\mathbf{e}(k)$ is the error in state estimate, $\mathbf{r}(k)$ is the error in the measurement prediction. It makes intuitive sense that these signals are both zero-mean noise terms in that an ideal filter of an observable system would converge to an unbiased estimate whose accuracy would be limited by the noises inherent to the system.

2.2 Attack Model

In [11], Kwon et. al assume that the system given by Equation (2.1) is attacked both at the input and at the output using attack vectors $\mathbf{a}_c(k)$ and $\mathbf{a}_o(k)$ so that the attacked system is of the form

$$\begin{aligned}\mathbf{x}_a(k+1) &= A\mathbf{x}_a(k) + B\mathbf{u}(k) + B_c\mathbf{a}_c(k) + E_1\mathbf{w}(k) \\ \mathbf{y}_a(k) &= C\mathbf{x}_a(k) + B_o\mathbf{a}_o(k) + E_2\mathbf{v}(k)\end{aligned}\tag{2.7}$$

$\mathbf{a}_c(k) \in \mathbb{R}^{m_c}$ represents an injected attack on a subset of the actuators with $B_c \subseteq Ra(B)$ and $\mathbf{a}_o(k) \in \mathbb{R}^{p_o}$ represents an injected attack on a subset of sensors with $B_o \subseteq Ra(C)$. The attacked system is assumed to have a state $\mathbf{x}_a$ and the corresponding output is denoted as $\mathbf{y}_a$. Further, it is assumed that the attacks are not injected until the system has reached steady state.

Suppose that a Kalman filter is used for the attacked system as

$$\begin{aligned}\hat{\mathbf{x}}_a(k+1|k) &= A\hat{\mathbf{x}}_a(k) + B\mathbf{u}(k) \\ \hat{\mathbf{x}}_a(k+1) &= \hat{\mathbf{x}}_a(k+1|k) + Lr_a(k+1) \\ r_a(k+1) &= \mathbf{y}_a(k+1) - C\hat{\mathbf{x}}_a(k+1|k)\end{aligned}\tag{2.8}$$

Figure 2-1: Deception Attack. An anomaly detector probes the Kalman filter residual to see if it exhibits the characteristics of a healthy system.

It follows that the corrupted residual $\mathbf{r}_a(k)$ differs from $\mathbf{r}(k)$ due to the insertion of $\mathbf{a}_c(k)$ and $\mathbf{a}_o(k)$. It also follows that under the unattacked case, the residual $\mathbf{r}(k)$ is guaranteed to remain as $\mathcal{N}(\mathbf{0}, \Sigma_r)$. It therefore remains to be analyzed as to

how $\mathbf{r}_a(k)$ differs from $\mathbf{r}(k)$ because of the attack inputs. As an example, we will consider the system under attack to be an aircraft. Figure 2-1 presents a schematic of the attack on an aircraft. The attacker injects $\mathbf{a}_c$ into the control input to enforce that the input entering the control system, $\mathbf{u}_a$, is different from that of the nominal control input, $\mathbf{u}$, inducing unintended aircraft behavior. The attacker can also inject $\mathbf{a}_o$ into the sensor outputs to feed the controller a corrupted output, $\mathbf{y}_a$, instead of the true output, $\mathbf{y}$. The Kalman filter residual, which is the difference between the measurement $\mathbf{y}_a$ and the expected measurement $\hat{\mathbf{y}}$, is then analyzed by an anomaly detector to check if it has the expected characteristics of a healthy system.

Moving forward, we would like to analyze the potential effects of this attack from the perspective of the attacker. Let us assume that the attacker's objective is to generate non-zero state estimation error, $\mathbf{e}_a$, while going undetected. This stealthy attack will be achieved by designing the output attacks to mask the accumulated effect of the injected actuator attacks and therefore ensure that the resulting residual, $\mathbf{r}_a$, has a distribution identical to the residual of a healthy system, $\mathbf{r}$ [11]. This results in potentially dangerous aircraft behavior that goes undetected by the defender. In the following subsection, we will investigate the effects of the actuator and sensor attacks and an attack design to achieve this stealthy attack.

2.3 Design of a Stealthy Attack

A common anomaly detection method for linear systems is to analyze the residual distribution. As previously discussed, in a healthy system, the defender expects a white residual distribution of zero-mean and known variance, Σ_r . As is done in [11], algebraic manipulations of Equations (2.7) and (2.8) lead to the derivation of the residual of the attacked system as

$$\mathbf{r}_a(k+1) = CA\mathbf{e}_a(k) + CB_c\mathbf{a}_c(k) + CE_1\mathbf{w}(k) + B_o\mathbf{a}_o(k+1) + E_2\mathbf{v}(k+1) \quad (2.9)$$

where $\mathbf{e}_a(k) \triangleq \mathbf{x}_a(k) - \hat{\mathbf{x}}_a(k)$ is the attacked system's state estimation error and has the following dynamic equation:

$$\mathbf{e}_a(k+1) = G\mathbf{A}\mathbf{e}_a(k) + GB_c\mathbf{a}_c(k) + GE_1\mathbf{w}(k) - LB_o\mathbf{a}_o(k+1) - LE_2\mathbf{v}(k+1) \quad (2.10)$$

where $G \triangleq I - LC$.

Equation (2.10) can be taken $k+1$ steps back and be rewritten to reflect its complete history:

$$\begin{aligned} \mathbf{e}_a(k) = (GA)^k\mathbf{e}(0) &+ \sum_{i=0}^{k-1} (GA)^i GB_c\mathbf{a}_c(k-1-i) + \sum_{i=0}^{k-1} (GA)^i GE_1\mathbf{w}(k-1-i) \\ &- \sum_{i=0}^{k-1} (GA)^i LB_o\mathbf{a}_o(k-i) - \sum_{i=0}^{k-1} (GA)^i LE_2\mathbf{v}(k-i) \end{aligned} \quad (2.11)$$

whereas a healthy system error, $\mathbf{e}(k)$, does not have the attack terms:

$$\mathbf{e}(k) = (GA)^k\mathbf{e}(0) + \sum_{i=0}^{k-1} (GA)^i GE_1\mathbf{w}(k-1-i) - \sum_{i=0}^{k-1} (GA)^i LE_2\mathbf{v}(k-i) \quad (2.12)$$

Note that because the the attack does not begin until $k=0$, at which point the system has reached steady state, $\mathbf{e}_a(0) = \mathbf{e}(0)$ and therefore $\mathbb{E}[\mathbf{e}_a(0)] = \mathbb{E}[\mathbf{e}(0)] = \mathbf{0}$. Therefore, upon further inspection of Equations (2.11) and (2.12), $\mathbf{e}_a(k)$ can be written as the sum of the healthy system's state estimation error, $\mathbf{e}(k)$, and some attack terms:

$$\mathbf{e}_a(k) = \mathbf{e}(k) + \sum_{i=0}^{k-1} (GA)^i GB_c\mathbf{a}_c(k-1-i) - \sum_{i=0}^{k-1} (GA)^i LB_o\mathbf{a}_o(k-i) \quad (2.13)$$

By combining Equations (2.9) and (2.11), the attacked system's residual can be

written to reflect its complete history as well:

$$\begin{aligned}
\mathbf{r}_a(k+1) = & CA(GA)^k \mathbf{e}(0) + CA \sum_{i=0}^{k-1} (GA)^i GB_c \mathbf{a}_c(k-1-i) + CA \sum_{i=0}^{k-1} (GA)^i GE_1 \mathbf{w}(k-1-i) \\
& - CA \sum_{i=0}^{k-1} (GA)^i LB_o \mathbf{a}_o(k-i) - CA \sum_{i=0}^{k-1} (GA)^i LE_2 \mathbf{v}(k-i) \\
& + CB_c \mathbf{a}_c(k) + CE_1 \mathbf{w}(k) + B_o \mathbf{a}_o(k+1) + E_2 \mathbf{v}(k+1) \quad (2.14)
\end{aligned}$$

and can similarly be written as a combination of the healthy system's residual and the effect of the attack terms:

$$\begin{aligned}
\mathbf{r}_a(k+1) = & \mathbf{r}(k+1) + CA \sum_{i=0}^{k-1} (GA)^i GB_c \mathbf{a}_c(k-1-i) \\
& - CA \sum_{i=0}^{k-1} (GA)^i LB_o \mathbf{a}_o(k-i) + CB_c \mathbf{a}_c(k) + B_o \mathbf{a}_o(k+1) \quad (2.15)
\end{aligned}$$

By designing the sensor attack to satisfy:

$$B_o \mathbf{a}_o(k+1) = -C(AE[\mathbf{e}_a(k)] + B_c \mathbf{a}_c(k)) \quad (2.16)$$

the attack terms in Equation (2.9) cancel and $\mathbf{r}_a(k+1) = \mathbf{r}(k+1) \sim \mathcal{N}(\mathbf{0}, \Sigma_r)$ [11]. Equation (2.16) can be written more explicitly:

$$B_o \mathbf{a}_o(k+1) = -C \sum_{i=1}^{k+1} A^{i-1} B_c \mathbf{a}_c(k+1-i) \quad (2.17)$$

Equation (2.16) is the pivotal equation proposed by Kwon et. al. which represents a recipe for a stealthy attack design and Equation (2.17) offers an algebraic view of this that will be useful for later manipulation. Note that just as the design given by Equation (2.16) canceled out the attack terms in Equation (2.9) for the residual dynamics, Equation (2.17) will cancel out the attack terms in the other representation of the residual given by Equation (2.15), again resulting in $\mathbf{r}_a(k+1) = \mathbf{r}(k+1) \sim \mathcal{N}(\mathbf{0}, \Sigma_r)$. That is, by this choice of sensor attack, the residual distribution of the

attacked system is identical to that of the healthy system, preventing the system defender from detecting the presence of an attack. Meanwhile, this choice of sensor attack, given by Equation (2.16), can be combined with the expectation of Equation (2.10) to give the mean state estimation error dynamics described by the following equation:

$$\mathbb{E}[\mathbf{e}_a(k+1)] = A\mathbb{E}[\mathbf{e}_a(k)] + B_c \mathbf{a}_c(k) = \sum_{i=1}^{k+1} A^{i-1} B_c \mathbf{a}_c(k+1-i) \quad (2.18)$$

Taken together, Equations (2.18) and (2.16) represent the recipe for an attacker to choose $\mathbf{a}_c(k)$ at each step so as to generate some desired and admissible error according to their attack objective and then recursively design the sensor attack, $\mathbf{a}_o(k+1)$, which in turn cancels out the effect of the actuator attack on the residual, enforcing that the attacked system has a residual distribution that is identical to that of the healthy system. In other words, an attacker chooses $\mathbf{a}_c(k)$ according to Equation (2.18) to generate state-estimation error and then designs $\mathbf{a}_o(k+1)$ according to Equation (2.16) so as to erase any history of the actuator attack and prevent the defender from detecting the presence of this estimation error. This process can be viewed in figure 2-2.

Figure 2-2: Stealthy Attack Design. The attacker injects $\mathbf{a}_c$ in order to induce damaging system behavior and generate state estimation error. The resulting measurement $\mathbf{y}$ is composed of a healthy signal, $\mathbf{y}_h$, which would be the output of the system without $\mathbf{a}_c$, and a signal resulting from the input attack, $\tilde{\mathbf{y}}$. The attack recipe then prescribes a sensor injection, $\mathbf{a}_o$, which cancels out the accumulated effect of the actuator attack resulting in a residual that is identical to that of a healthy system.

Note that Equation (2.16) can only be satisfied if the following condition holds

[11]:

$$Ra(C \times ctrb(A, B_c)) \subseteq Ra(B_o) \quad (2.19)$$

where $ctrb(A, B_c)$ denotes the controllability matrix formed by the pair (A, B_c) . Further, if the pair (A, B_c) is controllable, the attacker has the potential to generate arbitrary state estimation errors [11].

Upon inspection of Equations (2.16) and (2.18), it becomes apparent that for an attacker to successfully implement the stealthy attack design, a number of things have to be available to the attacker. The first is that the attacker has the ability to modify the states using an attack input of the form $B_c \mathbf{a}_c(k)$ at each k , so as to modify system behavior. Second, the attacker has sufficient access to the sensors, as given by Equation (2.19), to be able to modify the output using $B_o \mathbf{a}_o(k)$ at each k , concealing the effect of the actuator attack. In addition, Equation (2.16) also assumes that the attacker has full knowledge of the matrices A and C which in turn amounts to knowing critical components of the system model. A major question of this thesis regards how robust this attack recipe is. That is, what happens when the attacker does not have perfect knowledge of the system model.

Chapter 3

Baiting For A Detectable Attack

3.1 Effect of Model Perturbation on Attack

In the previous section, a method was offered for an attacker with perfect model knowledge to induce potentially arbitrary state estimation error while going undetected, and quantified in the form of Equations (2.16) and (2.18). It is reasonable to raise the question as to how they attain this model knowledge, how accurate their knowledge is, and what the effect of an attack would be with a slightly inaccurate model. In this chapter, we investigate the effect of an attacker with an imperfect model. We will analyze how an imperfect model can lead to the attack becoming detectable. From the defender's perspective, it is then of interest to introduce such an imperfection into the attack-model. We will show that if the attacker is baited into such an inaccurate model, the defender can not only ensure that an attack is detectable, but can regain accurate state estimates.

We return to Equations (2.16) and (2.17). Suppose that there is an uncertainty Δ in A so that if we define $A_b \triangleq A + \Delta$, the attacker uses A_b instead of A in Equations (2.16) (2.17). This perturbation modifies Equation (2.17) as

$$B_o \mathbf{a}_o(k+1) = -C \sum_{i=1}^{k+1} A_b^{i-1} B_c \mathbf{a}_c(k+1-i) \quad (3.1)$$

Using the fact that:

$$A_b^{i-1} = (A + \Delta)^{i-1} = A^{i-1} + \sum_{j=0}^{i-2} A^{i-2-j} \Delta (A + \Delta)^j \quad (3.2)$$

Equation (3.1) can be expanded as:

$$B_o \mathbf{a}_o(k+1) = -C \sum_{q=1}^{k+1} A^{q-1} B_c \mathbf{a}_c(k+1-q) - C \sum_{q=2}^{k+1} \sum_{j=0}^{q-2} A^{q-2-j} \Delta (A + \Delta)^j B_c \mathbf{a}_c(k+1-q) \quad (3.3)$$

A comparison of Equations (2.17) and (3.3) shows that the effect of Δ introduces an additional second term not present in the attack model. More generally, we can say at any instant $k - i$:

$$B_o \mathbf{a}_o(k-i) = -C \sum_{q=1}^{k-i} A^{q-1} B_c \mathbf{a}_c(k-i-q) - C \sum_{q=2}^{k-i} \sum_{j=0}^{q-2} A^{q-2-j} \Delta (A + \Delta)^j B_c \mathbf{a}_c(k-i-q) \quad (3.4)$$

Now, by inserting Equation (3.3) into the final term on the right hand side of Equation (2.15), and inserting Equation (3.4) into the third term on the right hand side of Equation (2.15), we get an equation for the resulting faulty residual, $\mathbf{r}_b$, with some nominal attack terms that cancel each other out by design, and two terms generated by the modeling error introduced by the attacker:

$$\begin{aligned} \mathbf{r}_b(k+1) = \mathbf{r}(k+1) + CA \sum_{i=0}^{k-2} (GA)^i LC \sum_{q=2}^{k-i} \sum_{j=0}^{q-2} A^{q-2-j} \Delta (A + \Delta)^j B_c \mathbf{a}_c(k-i-q) \\ - C \sum_{q=2}^{k+1} \sum_{j=0}^{q-2} A^{q-2-j} \Delta (A + \Delta)^j B_c \mathbf{a}_c(k+1-q) \end{aligned} \quad (3.5)$$

Equation (3.5) shows that the effect of an uncertainty Δ in the attack model causes the residue to change from $\mathbf{r}$ to $\mathbf{r}_b$ instead of $\mathbf{r}$ to $\mathbf{r}_a$ as in (2.15). That is, an attacker with a modeling error in A will induce a time varying shift in the mean of a healthy system's residual given by the last two terms in Equation (3.5). Because these two terms are a continuous function of Δ , there exists some $\zeta > 0$ such that

for a sufficiently large $\Delta > \epsilon$, $\mathbb{E}[\mathbf{r}_b(k+1)] > \zeta$ and the faulty attack is detected. Note that the deviation from the nominal residual induced by Δ is reminiscent of the effect of a watermarked input, similar to [17, 20]. Instead of an exogenous design of an additional perturbation signal in the control input on the part of the defender, any endogenous presence of a modeling error Δ on the part of the attacker may cause the attack to be revealed. In the next section, we discuss the effects of a defender enforcing such a Δ in the attack, thus intentionally "watermarking" the model.

3.2 A Defender's Perspective of the Baiting Approach

In order to ensure an attacker does not have a perfect model, the defender needs to anticipate how the attacker might acquire the model in the first place. It will be assumed that, as seen in a number of other attacks on CPS, the attacker fishes for model knowledge by infiltrating confidential systems, stealing enough data to construct the model, while remaining in a stealth mode. We propose that the defender, anticipating this type of infiltration, plants a deliberate perturbation Δ suggesting a slightly falsified model, A_b , thereby baiting an attacker fishing for data. This faulty model acts like a honeypot, only to be used by an intruding attacker. The subscript b indicates a model that is used as bait for the attacker. This offers the defender the ability to choose Δ such that the use of A_b in the sensor attack design, given by Equation (3.1), is likely to result in a detectable attack.

Figure 3-1: Baiting a Fishing Attacker with a False Model [14]

Additionally, the combined effects of the attack and the faulty model yield the following equation for the state estimation error induced by a detectable attacker, $\mathbf{e}_b$:

$$\mathbf{e}_b(k) = \mathbf{e}(k) + \sum_{q=2}^{k+1} A^{q-2} B_c \mathbf{a}_c(k+1-q) + \sum_{i=0}^{k-2} (GA)^i LC \sum_{q=2}^{k-i} \sum_{j=0}^{q-2} A^{q-2-j} \Delta (A+\Delta)^j B_c \mathbf{a}_c(k-i-q) \quad (3.6)$$

The true state as a result of this baited attack is given by $\mathbf{x}_b$ and the defender's naive estimate, before attempting to account for the effect of the attack, is given by $\hat{\mathbf{x}}_b$. Note that $\mathbf{e}_b = \mathbf{x}_b - \hat{\mathbf{x}}_b$ is composed of the healthy system's state estimation error which is a zero mean noise series, a term yielding the effect of the stealthy attack, and a decoupled term induced by the fault in the model. Given that the defender can see $\mathbf{r}_b$, given by Equation (3.5), and more specifically, approximate the sum of the non-zero mean terms induced by Δ , it is of interest to see if it is possible to use that knowledge to reconstruct $\hat{\mathbf{e}}_b(k)$, an estimate of $\mathbb{E}[\mathbf{e}_b(k)]$ in Equation (3.6), and therefore reconstruct a new more accurate state estimate, $\hat{\mathbf{x}}_{new}$, using:

$$\hat{\mathbf{x}}_{new}(k) = \hat{\mathbf{x}}_b(k) + \hat{\mathbf{e}}_b(k) \quad (3.7)$$

Equation (3.7) implies that not only does the use of Δ by the defender have the potential to make the attack detectable but it also provides an opportunity to get an updated and more accurate estimate $\hat{\mathbf{x}}_{new}$ which includes the error $\hat{\mathbf{e}}_b(k) \approx \mathbb{E}[\mathbf{e}_b(k)]$. In other words, A successfully baiting defender's choice of Δ makes the attack detectable, but knowledge of that Δ makes it much easier to estimate Equation (3.6) using Equation (3.5), improving the defender's state estimate.

Using Equations (3.5) and (3.6) to estimate $\mathbf{e}_b(k)$ from observations of $\mathbf{r}_b(k+1)$ can be viewed as attempting to estimate the actuator attack, $\mathbf{a}_c(k)$ for all prior k , enabling the calculation of the cumulative effect of the actuator attacks on the system dynamics. We will break up this estimation task into two cases. The first and more general case assumes that $\mathbf{a}_c(k)$ is time varying, and therefore many, potentially distinct, values of $\mathbf{a}_c(j)$ have to be estimated for all prior time $j = 0 \dots k$. The second case of interest assumes that the actuator attack does not vary over time, but

rather is a constant bias $\mathbf{a}_c(k) = \mathbf{a}_c$. For the general case where the attacker injects a time-varying actuator attack, $\mathbf{a}_c(k)$, under appropriate conditions, weighted least squares can be used to solve a time-series representation of Equation (3.5) for $\mathbf{a}_c(k) \forall k$. However, this method requires batch processing which will increase in complexity on each step and may therefore be intractable for real-time computations.

Let us proceed under the assumption that the attacker injects a constant bias as an actuator attack, that is, $\mathbf{a}_c(k) = \mathbf{a}_c$. Then, using Equation (3.5):

$$\mathbf{r}_b(k+1) \sim \mathcal{N}(M(k+1)\mathbf{a}_c, \Sigma_r) \quad (3.8)$$

where

$$M(k+1) = C(A \sum_{i=0}^{k-2} (GA)^i LC \sum_{q=2}^{k-i} \sum_{j=0}^{q-2} A^{q-2-j} \Delta(A+\Delta)^j - \sum_{q=2}^{k+1} \sum_{j=0}^{q-2} A^{q-2-j} \Delta(A+\Delta)^j) B_c \quad (3.9)$$

Therefore, $\mathbf{a}_c$ can be estimated using recursive weighted least squares estimation under suitable rank conditions.

As a result of Equation (3.6):

$$\mathbf{e}_b(k) \sim \mathcal{N}(\mathbb{E}[\mathbf{e}_b(k)], P) \quad (3.10)$$

where

$$\mathbb{E}[\mathbf{e}_b(k)] = \left(\sum_{q=2}^{k+1} A^{q-2} + \sum_{i=0}^{k-2} (GA)^i LC \sum_{q=2}^{k-i} \sum_{j=0}^{q-2} A^{q-2-j} \Delta(A+\Delta)^j \right) B_c \mathbf{a}_c \quad (3.11)$$

Therefore, using the least squares estimate $\mathbf{a}_c$ at each step, $\hat{\mathbf{e}}_b(k)$ can be estimated at each step and consequently, using Equation (3.7), $\hat{\mathbf{x}}_{new}(k)$, the updated and more accurate state estimate, can be computed at each step. The result of this analysis is that a defender that successfully baits an attacker fishing for model data can induce elements in the residual that not only make the attack detectable but re-enable accurate observations of the states, mitigating the effects of the attack.

3.3 Least Squares Implementation

In the previous section, we developed the baiting defense which enabled attack detection followed by updating the state estimates using recursive weighted least squares. In this section, we will give the steps that enable practical implementation of that recursive least squares update.

In the following, the time at which the attacker injects the actuator attack, $\mathbf{a}_e$, will be treated as $k = 0$. In practice, the defender will have to detect the presence of the attack based on some change in the residual, which will typically not be immediate. Therefore, the defender, seeking to estimate the effect of the attack on the states based on observations of the residual, will not know exactly when the attack was inserted. In other words, the defender does not know exactly when $k = 0$, only when the attack was detected. Attack detection will be discussed in further detail in Section 3.4.1.

In the following, it is assumed that the defender has an ideal detector such that the attack is detected immediately and they therefore know exactly what time corresponds to $k = 0$. In figure 4-2, we will see an example of how state estimation plays out in simulation when the time of detection is treated as $k = 0$ despite that not being the case. We will then look at cases where it is assumed that the defender has a better detector and therefore has a detection time that is closer to $k = 0$. This can also be thought of as the defender coming up with an estimate of how many steps prior to detection the attack must have occurred based on recordings of past residuals.

We will now develop the practical implementation of the recursive weighted least squares estimation method in which we use the observations of the baited residual to estimate $\mathbf{a}_e$ and therefore enable improved state estimation. Upon inspection of Equations (3.8-3.9), we see that in that in order to execute each step of the recursive least squares update, we need to compute $M(k+1)$ at each step. This computation can be done efficiently by using a recursive formulation that computes $M(k+1)$ based on a prior computation of $M(k)$.

We can rewrite Equation (3.9) as follows:

$$M(k+1) = C(AJ(k+1) - Z(k+1))B_c \quad (3.12)$$

where

$$\begin{aligned} J(k+1) &\triangleq \sum_{i=0}^{k-2} (GA)^i LC \sum_{q=2}^{k-i} \sum_{j=0}^{q-2} A^{q-2-j} \Delta(A+\Delta)^j \\ Z(k+1) &\triangleq \sum_{q=2}^{k+1} \sum_{j=0}^{q-2} A^{q-2-j} \Delta(A+\Delta)^j \end{aligned} \quad (3.13)$$

If we define the following variables:

$$\begin{aligned} F(k+1) &\triangleq \sum_{i=0}^{k-2} (GA)^i LC \sum_{j=0}^{k-i-2} A^{k-i-2-j} \Delta(A+\Delta)^j \\ V(k+1) &\triangleq \sum_{j=0}^{k-2} A^{k-2-j} \Delta(A+\Delta)^j \\ W(k+1) &\triangleq \Delta(A+\Delta)^{k-2} \end{aligned} \quad (3.14)$$

then the following recursive formulation can be used to solve for J at each step:

$$\begin{aligned} W(k+1) &= W(k)(A+\Delta) \\ V(k+1) &= AV(k) + W(k+1) \\ F(k+1) &= GAF(k) + LCV(k+1) \\ J(k+1) &= J(k) + F(k+1) \end{aligned} \quad (3.15)$$

with the following initial values:

$$\begin{aligned} J(1) &= J(2) = 0 \\ W(3) &= \Delta \\ V(3) &= \Delta \\ F(3) &= LC\Delta \\ J(3) &= LC\Delta \end{aligned} \quad (3.16)$$

Note that (3.15) along with (3.16) enable the iterative computation of $J(k+1)\forall k$.

Similarly, if we define the following:

$$\begin{aligned} H(k+1) &\triangleq \sum_{q=2}^{k+1} (A + \Delta)^{q-2} \\ T(k+1) &\triangleq (A + \Delta)^{k-1} \end{aligned} \tag{3.17}$$

then the following recursive formulation can be used to solve for Z at each step:

$$\begin{aligned} T(k+1) &= T(k)(A + \Delta) \\ H(k+1) &= H(k) + T(k+1) \\ Z(k+1) &= AZ(k) + \Delta H(k+1) \end{aligned} \tag{3.18}$$

with the following initial values:

$$\begin{aligned} Z(1) &= 0 \\ H(2) &= I_{n \times n} \\ T(2) &= I_{n \times n} \\ Z(2) &= \Delta \end{aligned} \tag{3.19}$$

Note that $I_{n \times n}$ is the $n \times n$ identity matrix and as a reminder, $G \triangleq I - LC$. Equations (3.15,3.16,3.18,3.19) combined with Equation (3.12) enable a recursive calculation of matrix $M(k+1)$ based on the matrix at the previous step, $M(k)$.

Now that we can construct $M(k+1)$ on each step, and we know the distribution of the baited residual is given by Equation (3.8), we have enough information to recursively update our estimate of $\mathbf{a}_c$ on each step upon observation of the baited residual using the following recursive weighted least squares formulation [10]:

$$\begin{aligned} S(k+1) &= S(k) + M^T(k+1)\Sigma_r^{-1}M(k+1) \\ \hat{\mathbf{a}}_c(k+1) &= \hat{\mathbf{a}}_c(k) + S^{-1}(k+1)M^T(k+1)\Sigma_r^{-1}(\mathbf{r}_b(k+1) - M(k+1)\hat{\mathbf{a}}_c(k)) \end{aligned} \tag{3.20}$$

with the following initialization:

$$\begin{aligned} S(2) &= M^T(2)\Sigma_r^{-1}M(2) \\ \hat{\mathbf{a}}_c(2) &= S^{-1}(2)M^T(2)\Sigma_r^{-1}\mathbf{r}_b(2) \end{aligned} \tag{3.21}$$

Now that we have a way of estimating and recursively updating $\mathbf{a}_c$, we want to use our estimate, $\hat{\mathbf{a}}_c$ to estimate the state estimation error, $\mathbf{e}_b$ using Equation 3.11. Equation 3.11 can be rewritten as:

$$\mathbb{E}[\mathbf{e}_b(k)] = (N(k) + J(k+1))B_c\mathbf{a}_c \tag{3.22}$$

where $J(k+1)$ is as defined in Equation (3.13) and N is given by:

$$N(k) \triangleq \sum_{q=2}^{k+1} A^{q-2} \tag{3.23}$$

$J(k+1)$ can be recursively computed based on computations of J at past steps using the iterations and initial conditions given by Equations (3.15-3.16). If we define

$$Y(k) \triangleq A^{k-1} \tag{3.24}$$

then $N(k)$ can be recursively computed using the following formulation:

$$\begin{aligned} Y(k) &= AY(k-1) \\ N(k) &= N(k-1) + Y(k) \end{aligned} \tag{3.25}$$

with the following initial conditions:

$$\begin{aligned} Y(1) &= I_{n \times n} \\ N(1) &= I_{n \times n} \end{aligned} \tag{3.26}$$

Now, looking back at Equation (3.22), we can get an estimate of the state estimation

error $\hat{\mathbf{e}}_b(k)$ by plugging our most up to date estimate of $\mathbf{a}_c$ into the following:

$$\hat{\mathbf{e}}_b(k) = (N(k) + J(k + 1))B_c\hat{\mathbf{a}}_c(k) \quad (3.27)$$

Finally, we can get the most updated estimate of the state, $\hat{\mathbf{x}}_{new}$, by adding this new estimate of the state estimation error to the naive Kalman filter estimate of the state that we attained before this update, $\hat{\mathbf{x}}_b$, using Equation (3.7).

3.4 Steps of the Baiting Defense Recipe

The aforementioned baiting defense can be thought of as broken down into two steps. The following elaborates on these steps and their execution to achieve the baiting defense.

3.4.1 Baiting: Step 1

The first step of the baiting defense consists of planting the false model given by $A_b \triangleq A + \Delta$ and then detecting an attempted attack in the resulting residual shift given by Equations (3.5, 3.8).

Over the course of this thesis, we have focused on the residual as the signal that contains information about how close the system behavior is to nominal and developed conditions under which the residual will suggest departure from this nominal behavior. It is important that we discuss more specifically how we will determine a significant departure has occurred in simulation and therefore detect an attack. In [11], Kwon et. al choose to use a method called compound scalar testing which will be used in this thesis as well. This method computes a scalar function of the residual, referred to in [11] as the power of the residual, and given by:

$$\mathbf{r}^T(k)\Sigma_r^{-1}\mathbf{r}(k) \quad (3.28)$$

This scalar metric acts a weighted norm of the residual. For a zero-mean residual, as

we expect in a healthy system, the power of the residual has an expected value of p , the dimension of $\mathbf{r}$. A threshold $h > p$ can then be chosen such that if $\mathbf{r}^T(k)\Sigma_r^{-1}\mathbf{r}(k) > h$, the defender considers an attack to have occurred. In general, the choice of the detection threshold should depend on how we prioritize the detector's sensitivity to attacks against its sensitivity to false alarm. For the case of a baited attacker, the resulting residual power, $\mathbf{r}_b^T(k)\Sigma_r^{-1}\mathbf{r}_b(k)$, should exhibit an increase compared to that of a healthy system. For a large enough increase in the residual power and an appropriate choice of h , attacks should be detectable with low likelihood of false alarm.

3.4.2 Baiting: Step 2

Once the attack is detected, the residual of the Kalman filter can be observed to recursively estimate the injected actuator attack, $\mathbf{a}_c$, as explained in Section 3.3. Equations (3.12-3.21) give the necessary information to construct the matrix M at each step and use recursive weighted least squares so as to update the estimate of $\mathbf{a}_c$ based on the observed residual at each step. For a sufficiently timely attack detection and a sufficient number of observations as well as a full rank $M(k)$, the estimate of $\mathbf{a}_c$ should converge to the true $\mathbf{a}_c$.

Similarly, Equations (3.22-3.27) give the necessary information to get an estimate of the induced shift in the state estimation error, $\hat{\mathbf{e}}_b(k) \approx \mathbb{E}[\mathbf{e}_b(k)]$, by plugging the most updated estimate of $\mathbf{a}_c$ into Equation (3.11). Before the update, the Kalman filter estimate is naive to the effect of the attack injections on the state estimation error. Equation (3.7) shows how at each step, this naive Kalman filter estimate, $\hat{\mathbf{x}}_b(k)$, can be added to the updated estimate of the state estimation error, $\hat{\mathbf{e}}_b(k)$ to yield the updated state estimate at that step, $\hat{\mathbf{x}}_{new}(k)$. As the accuracy of $\mathbf{a}_c$ improves with the number of observations of the residual, the accuracy of the state estimates should improve accordingly. Note that since recursive estimation is used, updated estimates are iteratively built on previous estimates and this technique can efficiently update state estimates in real-time.

3.5 Estimating Time of Attack Onset

In Section 3.3 we discussed how in practice, the defender does not know the exact time at which the attack was introduced, only the time at which the attack was detected. This, as we will see in Section 4.2.2, has the potential to result in an unideal step 2 state estimate. Ideal use of Equations (3.20-3.21) requires the defender to know when the attack is introduced, so that the defender knows what step in the attack sequence, k , has been reached. We will see in Section 4.2.2 that as the attack detection time approaches the time of onset, the accuracy of state estimates resulting from step 2 of the baiting defense generally improves.

In this section we propose a method by which the defender can estimate the time of attack onset by looking at a recording of the residual signal for some prior period of time. In this method, the defender will record the residual until some time $k = l$ where l occurs after the attack is detected. It is assumed that time l is sufficiently long after the attack onset, that there are sufficient observations of the residual to enable accurate estimation of $\mathbf{a}_c$. We will define $\hat{\mathbf{a}}_{c0}(l)$ to be the estimate of $\mathbf{a}_c$ resulting from baiting step 2 when the defender assumes the time of detection corresponds to $k = 0$, the time the attack is introduced. As discussed in Section 3.3, the defender can reasonably assume that their detection was not immediate and may therefore wish to estimate how many steps prior to detection the attack must have occurred. We therefore define $\hat{\mathbf{a}}_{cj}(l)$ which is the result of baiting step 2, based on observations of the residual up to $\mathbf{r}_b(l)$, and assumes the attack was introduced j steps before it was detected. That is, the defender has recordings of the residual signal for k up to $k = l$, and at time l can get an estimate of $\mathbf{a}_c$ using Equations (3.20-3.21) assuming that the time of attack onset is j steps before the time of detection. Unlike the method discussed in Sections 3.3-3.4, this method does not update the state estimates at each step but after a period of observation ending at time $k = l$.

At time $k = l$, the defender will estimate $\hat{\mathbf{a}}_{cj}(l)$ for a range of choices of j . This corresponds to a range of guesses of how long before attack detection the attack onset took place. Then by replacing $\mathbf{a}_c$ with $\hat{\mathbf{a}}_{cj}(l)$ in Equations (3.8-3.9), we can

define $\hat{\mathbf{r}}_{\mathbf{b}j}(k+1)$ to be the resulting estimate of mean residual, $\mathbb{E}[\mathbf{r}_{\mathbf{b}}(k+1)]$, over the defender's assumed attack time period. We can compare $\hat{\mathbf{r}}_{\mathbf{b}j}(k)$ to the recordings of $\mathbf{r}_{\mathbf{b}}(k)$ over that assumed time period to get a root mean square error of our expected residual when using $\hat{\mathbf{a}}_{\mathbf{c}j}(l)$. This will result in a root mean square error for every choice of j . The choice of $j = j^*$ that leads to the lowest root mean square error will then be chosen as an estimate of the number of steps before attack detection when the attack was introduced.

Now $\hat{\mathbf{a}}_{\mathbf{c}j^*}(l)$ is the defender's best estimate of $\mathbf{a}_{\mathbf{c}}$ resulting from estimating the time of attack onset prior to detection. Replacing $\mathbf{a}_{\mathbf{c}}$ with $\hat{\mathbf{a}}_{\mathbf{c}j^*}(l)$ in Equations (3.10-3.11) results in an estimate of the state estimation error for all k up to $k = l$. Finally, this estimate of the state estimation error can be plugged into Equation (3.7) to yield an updated set of state estimates that takes into account the time of attack onset as distinct from the time of attack detection.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 4

Simulations

4.1 Preliminary Details

In this section we carry out simulation studies to validate the baiting approach proposed in Section 3. For this purpose, we use a nonlinear model of an F-16 outlined in [23] and further developed for simulation in [18]. The model in Equation (2.1) will correspond to the linearized model of an F-16, and includes lateral-directional dynamics, assumed to be decoupled from the longitudinal dynamics as discussed in [12]. A healthy system, that is one that is not under attack, will be simulated and compared to a system undergoing a successful stealthy attack as well as a system in which the attacker has been baited with a false model.

In Section 3.4.1, we discussed using the residual power as a metric for detecting attacks. A threshold can be defined such that if the residual power increases beyond that threshold, an attack is declared. In the following simulations, we will use a smoothed version of the power of the residual, resulting from a causal moving average filter. This smoothed signal is more sensitive to trends in the detection metric and in many cases makes it easier to detect the attack.

4.2 Simulation of a Non-Linear Dynamical System

The lateral-directional dynamics model of the F-16 operating near a straight and level steady cruise condition with an altitude of 15000 feet and a velocity of 500 feet per second can be approximated by the following linearized dynamic equation:

$$\dot{\mathbf{x}}_{cont} = \begin{bmatrix} \dot{\phi} \\ \dot{\beta} \\ \dot{p} \\ \dot{r} \end{bmatrix} = A_{cont}\mathbf{x}_{cont} + B_{cont}\mathbf{u}_{cont} \quad (4.1)$$

where the subscript *cont* denotes continuous dynamics. A_{cont} and B_{cont} are assumed to represent the plant with an inner loop control. The control input $\mathbf{u}_{cont}$ represents the aileron and rudder inputs of the aircraft:

$$\mathbf{u}_{cont} = \begin{bmatrix} \delta_a \\ \delta_r \end{bmatrix} \quad (4.2)$$

We will choose the following measurements:

$$\mathbf{y}_{cont} = \begin{bmatrix} p \\ r \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \phi \\ \beta \\ p \\ r \end{bmatrix} \quad (4.3)$$

Though the dynamics are continuous, our technique is discrete and so sampling will be required. Using a sampling time of $T_s = 0.01$ seconds and adding zero-mean Gaussian process and measurement noises, Equation (4.1) is converted to a discrete time state space model of a form similar to the one given in Equation (2.1). In our application, there will be zero-mean Gaussian noise of $1deg/sec$ standard deviation on both sensor measurements and the process noise will be given as a Gaussian input disturbance with a standard deviation of 1% of the respective control surface maximum deflection. In order to deal with a particularly dangerous attack, we will look at the case where

the attacker can corrupt all actuator and sensor channels. In other words, $B_c = B$ and $B_o = I_{2 \times 2}$ where $I_{2 \times 2}$ represents the 2×2 identity matrix. This results in a discretized model of the system under attack with the following form:

$$\begin{aligned} \mathbf{x}_a(k+1) &= A\mathbf{x}_a(k) + B(\mathbf{u}(k) + \mathbf{a}_c + \mathbf{w}(k)) \\ \mathbf{y}_a(k) &= C\mathbf{x}_a(k) + \mathbf{a}_o(k) + \mathbf{v}(k) \end{aligned} \quad (4.4)$$

where $\mathbf{w}(k)$ and $\mathbf{v}(k)$ are the process and sensor noises discussed above. Equation (4.4) is a specific form of Equation (2.7). These choices of B_c and B_o satisfy the rank condition given by Equation (2.19), therefore enabling the stealthy attack. To generate state estimation error, the attacker will inject a constant 3 deg attack into the rudder control input and attempt to go undetected using the attack recipe given by Equation (2.16). Therefore, $\mathbf{a}_c = \begin{bmatrix} 0 & 3 \end{bmatrix}^T$ when the system is under attack. Further, a baited attacker will use a false model of $A_b = 0.9A$ for their attack design given by Equation (3.1), resulting in a 10% model error.

In order to develop an attack detection threshold h , 100 20-second simulations of the healthy system were run and the maximum of the smoothed residual power was recorded on each run. The choice of threshold h can then be determined by our desired false alarm rate. Our choice of threshold will correspond to the 99th percentile choice of h , a high threshold with a low likelihood of false alarm. In general, one could choose a lower threshold, increasing the chance of detection, or enabling earlier detection, but also increasing the likelihood that a healthy system undergoes a false alarm.

For the following figures it will be useful to name the elements of relevant vectors. The healthy system's state estimation error will be denoted by:

$$\mathbf{e} = \begin{bmatrix} e_\phi \\ e_\beta \\ e_p \\ e_r \end{bmatrix} \quad (4.5)$$

A system undergoing a stealthy attack has a state estimation error that will be

denoted by:

$$\mathbf{e}_a = \begin{bmatrix} e_{a_\phi} \\ e_{a_\beta} \\ e_{a_p} \\ e_{a_r} \end{bmatrix} \quad (4.6)$$

And a system undergoing a baited attack has a state estimation error that will be denoted by:

$$\mathbf{e}_b = \begin{bmatrix} e_{b_\phi} \\ e_{b_\beta} \\ e_{b_p} \\ e_{b_r} \end{bmatrix} \quad (4.7)$$

Additionally a system undergoing a baited attack will have a true state $\mathbf{x}$, an original Kalman filter state estimate $\hat{\mathbf{x}}_b$, and an updated estimate, $\hat{\mathbf{x}}_{new}$, which is computed upon detection using observations of the baited residual signal in step 2 of the baiting defense, all given by:

$$\mathbf{x} = \begin{bmatrix} \phi \\ \beta \\ p \\ r \end{bmatrix} \quad \hat{\mathbf{x}}_b = \begin{bmatrix} \hat{\phi}_b \\ \hat{\beta}_b \\ \hat{p}_b \\ \hat{r}_b \end{bmatrix} \quad \hat{\mathbf{x}}_{new} = \begin{bmatrix} \hat{\phi}_{new} \\ \hat{\beta}_{new} \\ \hat{p}_{new} \\ \hat{r}_{new} \end{bmatrix} \quad (4.8)$$

Simulations of the non-linear model are conducted and their results are shown in figures 4-1 - 4-4.

4.2.1 Attack Comparisons and Results of Baiting Step 1

Figure 4-1: Comparison of absolute state estimation errors and residual power signals of an un-attacked healthy system (Column 1) to a successful stealthy attack (Column 2) and a detectable (baited) attack (Column 3). State estimation errors are the result of a standard Kalman filter that is unaware of the attack inputs.

The results in figure 4-1 are categorized into three systems. The healthy system is represented in the first column of figure 4-1. It exhibits small state estimation errors, as expected, as well as a baseline for the smoothed residual power signal in equation (3.28), which stays under the detection threshold. The second column of figure 4-1 shows the effect of a stealthy attack in which the attacker uses the recipe given by Equation (2.16) with $\mathbf{a}_e = \begin{bmatrix} 0 & 3 \end{bmatrix}^T$. It is clear from observations of the stealthy attack that the attacker is successful in being undetected as the residual power stays below the set threshold while at the same time, the state estimation errors are growing

significantly. The final column shows the effect of an attack in which the attacker was baited with a false model $A_b = 0.9A$. It is clear that baiting is effective in revealing the attacker. This 10% error in A results in a jump in the residual power which crosses the detection threshold 0.6 seconds into the attack, concluding the first step of the baiting defense. In this example, there is substantially less state estimation error for a baited attack than a stealthy one. This difference is induced by the final term in Equation (3.6), which encodes the effect of the model error. It should be noted that while the baiting state estimation errors are not as large as in the successful attack, they still exhibit more than a tenfold increase in the overall root mean square error (RMSE) compared to that of the healthy system over the final three seconds of simulation. The reason errors are compared over the final three seconds of simulation is to enable comparison to estimates in figures 4-2 - 4-4 following a transient period in estimation. In the case of figure 4-1, the baited attack has been detected but the states are still estimated using a standard Kalman filter. This naive Kalman filter does not account for the attack injections. No effort has yet been made to update state estimates upon detection. This case will act as a baseline for the state-estimation error when a baited attack has occurred. In an effort to further reduce this error, in figure 4-2 we take the next step, attempting to update our state estimates based on observations of the baited residual, as outlined in Equations (3.7 - 3.11) and further elaborated upon in Section 3.3. The following are the results of step 2 of the baiting defense.

4.2.2 Results of Baiting Step 2

Figure 4-2: Comparison of true state, baseline state estimate under conditions of detectable attack using a standard Kalman filter, and updated state estimate for all four elements of state vector. The attack threshold h is chosen as the 99th percentile of the maximum smoothed residual powers resulting from 100 20 second simulations of a healthy system. Using this threshold, the attack is detected after 0.6 seconds. The root mean square errors in the updated estimates of β , p , and r decrease by 21%, 18%, and 60% when compared to the baseline estimate using a standard Kalman Filter. The updated estimate of ϕ has increased by 37% relative to the standard Kalman filter estimate.

Figure 4-2 shows the resulting state and state estimates of a baited attack with a 99th percentile detection threshold. Initially, there is no ability to get an updated estimate, but when the detection threshold is crossed after 0.6 seconds, updated estimates are computed. These updated estimates are the sum of the original naive Kalman filter

estimate of step 1, $\hat{\mathbf{x}}_b$, and the step 2 estimate of the state estimation error computed upon observation of the baited residual, $\hat{\mathbf{e}}_b$, given by Equation (3.7). Note that most of the updated estimates display substantially decreased state estimation error, however, the estimate of ϕ shows a significant shift from the true state. While the root mean square error (RMSE) in β , p , and r decrease by 21%, 18%, and 60% respectively compared to that of the baseline baited error prior to the update, the root mean square error in ϕ has increased by 37% relative to the uncorrected Kalman filter estimate. As we discuss below, this error in estimating ϕ is likely due to delay in detection. Therefore, baiting with the 99th percentile threshold has led to detection and improvement in all individual state estimates except for ϕ .

All of the above results shown in figures 4-1 and 4-2 correspond to a the case where an attack is declared when the 99th Percentile detection threshold is crossed, in this case at 0.6 seconds. This brings up an interesting question as to how estimation would improve if the attack could be detected earlier. We will show the results of step 2 estimation for two more scenarios which assume that we have a better detector. The first scenario assumes that the attack is detected 0.2 seconds after it is injected. The results of this case are shown in figure 4-3. The second scenario assumes a hypothetical ideal detector that could detect an attack instantly, the estimation results of which are shown in figure 4-4.

Figure 4-3: Comparison of true state, baseline state estimate under conditions of detectable attack using a standard Kalman filter, and updated state estimate for all four elements of state vector. It is assumed here that we have a better detector that can catch the attack after 0.2 seconds. When compared to the baseline state estimation errors resulting from a standard Kalman filter, the RMSE in estimates of ϕ , β , p , and r decrease by 87%, 84%, 82% and 83% respectively.

Figure 4-3 shows the resulting state and state estimates of a baited attack if we had a detector capable of detecting the attack after 0.2 seconds. This quicker attack detection enables the defender to get more accurate state estimates in step 2 when compared to the baseline baited error resulting from a standard Kalman filter in step 1, shown in the final column of figure 4-1. Figure 4-3 shows how the updated estimates significantly improve when compared to the baseline estimate in which no update is made. There is a significant reduction in the root mean square errors associated with each individual updated state estimate.

Further investigation of detection methods may be required to find a detector that will detect this attack faster than the method used in this paper and therefore enable improved estimation results such as those in figure 4-3. The estimation results of a hypothetical ideal detector that could detect an attack instantly are shown in figure 4-4. Though this detector is unrealistic, it gives us a goal to move towards.

Figure 4-4: Comparison of true state, baseline state estimate under conditions of detectable attack using a standard Kalman filter, and updated state estimate for all four elements of state vector. The improved estimates are the result of an ideal situation in which the attack could somehow be detected immediately. When compared to the baseline state estimation errors resulting from a standard Kalman filter, the RMSE in estimates of ϕ , β , p , and r decrease by 75%, 97%, 86% and 89% respectively.

Figure 4-4 shows the resulting state and state estimates of a baited attack if we had an ideal detector that could detect an attack immediately. When compared to

the baseline case of baiting where state estimates result from a standard Kalman filter in step 1, the step 2 update with this ideal detector improves estimation accuracy significantly as demonstrated by the substantial reductions in root mean square errors.

It is of interest to see how representative the above simulation is of an average case. Therefore, the same set of simulations were conducted 50 times and average results comparing the step 2 updated state estimates to the step 1 baseline baiting estimates resulting from a standard Kalman filter are reported in table 4.1. The average detection time for our 99th percentile detector over these 50 simulations was 0.55 seconds.

Table 4.1: Mean Percent Change in RMSE With Updated Estimates Over 50 Simulations

Detection Condition	ϕ	β	p	r
99th Percentile Threshold	+45%	-30%	-23%	-62%
Detection at 0.2 Seconds	-77%	-86%	-82%	-85%
Ideal Detector	-75%	-96%	-86%	-90%

The above simulation results show the behavior of a healthy system, a successfully attacked system, and a system undergoing a baited attack. We see that baiting leads to a detectable attack. Once the attack is detected, state estimates can be updated by a recursive least square procedure. As is shown in table 4.1, the accuracy of the updated state estimates generally improves with faster attack detection.

4.3 Estimating Time of Attack Onset for Improved State Estimation

In the previous section we saw that the 99th percentile threshold residual power detector we used detected the attack 0.6 seconds after it was introduced. We also saw that faster detection led to more accurate state estimates. Though the defender using a particular detector knows when the attack is detected, they may wish to know how many steps prior to this detection was the attack introduced. In this section we wish

to employ the method developed in Section 3.5 in order to improve our estimate of the time of attack onset, therefore enabling more accurate state estimates.

In the cases of Figures 4-1-4-4, the period of observation after the attack onset is 6 seconds. Therefore, in the following estimation of the time of attack onset, we will use residual observations up to $k = l = 6$ seconds in order to reconstruct estimates of the mean residual dynamics, $\hat{\mathbf{r}}_{bj}(k)$, for varying estimates of $\hat{\mathbf{a}}_{cj}(6)$, as discussed in Section 3.5. We will take the best choice of the attack estimate, $\hat{\mathbf{a}}_{cj^*}(l)$, to be the one that results in the lowest root mean square error when compared to the actual residual observations over the associated time period. This is the result of treating the attack as having been injecting j^* steps before it was introduced.

Figure 4-5: Root mean square error of estimate of mean residual dynamics when compared to actual residual observations. The attack is introduced 0.6 seconds before it is detected, indicated by the vertical black line at -0.6 in the figure. The root mean square error is minimized at -0.62. Therefore the attack is estimated to have started 0.62 seconds before it was introduced.

Figure 4-5 shows how the root mean square error of the estimated mean residual dynamics change with different guesses for the number of time steps by which the initial attack injection preceded detection. In this case, 101 evenly spaced guesses are made in increasing increments of 0.01 seconds for 0 to 1 seconds before the attack is detected. The root mean square error of the estimated mean residual dynamics is minimized when the attack is assumed to have been introduced 0.62 seconds before detection, quite close to the actual time of 0.6 seconds. This estimate of the time of attack onset corresponds to guessing that the attack was introduced $j^* = 62$ time steps prior to detection. The resulting estimate of the actuator attack injection using 6 seconds of attack observations, and 5.6 seconds of observation after detection, is given by $\hat{\mathbf{a}}_{c62}(6)$.

Figure 4-6: Estimates of injected aileron and rudder attack injections varying with estimated time of attack onset. The red lines indicate the true values of the attack injection and the black vertical line indicates that the attack was introduced 0.6 seconds before detection. The estimate of $\mathbf{a}_c$ which minimizes the RMSE in Figure 4-5 is given by $\hat{\mathbf{a}}_{c62}(6) = \begin{bmatrix} -0.07 & 2.88 \end{bmatrix}^T$ whereas the true value is given by $\mathbf{a}_c = \begin{bmatrix} 0 & 3 \end{bmatrix}^T$.

Replacing $\mathbf{a}_c$ with $\hat{\mathbf{a}}_{c62}(6) = [-0.07 \ 2.88]^T$ in Equations (3.10-3.11) results in an estimate of the state estimation error for all k up to $l = 6$ seconds. Employing Equation (3.7), we can then add this series of estimates of the state estimation error to recorded Kalman filter estimates of the state to get updated estimates of the state trajectories for the observed time period. The result of this is shown in Figure 4-7

Figure 4-7: Comparison of true state, baseline state estimate under conditions of detectable attack using a standard Kalman filter, and state estimate which is updated upon estimation of the time of attack onset for all four elements of state vector. The state estimates are computed upon observing 6 seconds of the attack and estimating the time of attack injection. The actuator attack can then be estimated as $\hat{\mathbf{a}}_{c62}(6)$ which can be used to update the state estimates.

Figure 4-7 shows the resulting state and state estimates of a baited attack which is detected 0.6 seconds after it is introduced. By estimating the number of steps by which the attack onset preceded its detection, the defender can estimate the attack

input and resulting state estimates after a period of observation. This offers a way of getting more accurate state estimates than simply starting the step 2 state estimation (see Sections 3.3 and 3.4.2) at the time of detection as is done in Figure 4-2. It is important to note that since this method requires a period of observation, the resulting state estimates are not attained until after that period is over, yielding an estimate of prior state trajectories as well as a current state estimate. Of course, the defender can still recursively update their estimates in real time during the observation period in the way that is done in Figure 4-2 and then update their estimates again at the end of the designated observation period using the method employed in this section.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5

Concluding Remarks

5.1 Summary and Conclusions

This work studied a type of stealthy attack on Cyber-Physical Systems with linear dynamics and developed a methodology by which the attack is revealed.

In Chapter 1, we laid out the background of cyber-physical systems and the motivation for continued research. We elaborated on the potential threats to cyber-physical systems and discussed some defense mechanisms that have been studied. After presenting two important attack on CPS, we envisioned an attacker whose ultimate goal is to inflict a damaging attack on CPS, but first must "fish" for confidential data about the system which will enable their attack. In the case of the stealthy attack of interest to this thesis, the confidential data the fishing attacker required were the model parameters. We therefore proposed a baiting defense against a fishing attacker, in which a defender anticipating a stealthy attack will plant falsified model data to be accessed and used by a fishing attacker.

In Chapter 2, we introduced a dynamic model of a CPS and the stealthy attack of interest to this thesis. This stealthy attack, introduced by the authors of [11], involved an attacker injecting malicious actuation and sensor signals to affect the control and measurement of the CPS. It was assumed that the attacker's objective was to induce large state estimation errors while going undetected. Based on the work in [11], we showed an attack recipe that enabled an attacker to inject actuation signals that

would cause the state of a CPS to diverge from its intended course while injecting a sensor signal to cancel out the affect of the actuation signal on the observed Kalman filter residual, rendering the attack undetectable. It was noted that to successfully achieve this attack, the attacker needed to have sufficient access to corruptible data channels as well as know the CPS model parameters.

It was then of concern how robust this stealthy attack design is. Chapter 3 began by analyzing how the Kalman filter residual of the attacked system was affected by an attack design that used a faulty model. The result of the fault was that the attacker induced a mean-shift in the Kalman filter residual from that of a healthy system, rendering the attack detectable. We then proposed the baiting defense. Step 1 of our baiting defense was to bait an attacker with false model data and then detect the resulting attack. The next step of the baiting defense was devoted to developing a method by which the defender, upon detecting the attack in step 1, could take the next step and update their state estimates based on observations of the shifted residual. This update is step 2 of the baiting defense and allows for more accurate state estimates than the standard Kalman filter in step 1 offered. We also presented a method by which a defender can estimate the time of attack onset, prior to attack detection, after a period of observation. By estimating the time of attack onset, the defender can get a better picture of past state dynamics as well as a more accurate present state estimate at the end of the observation period.

In Chapter 4, we simulated the lateral directional dynamics of a non-linear model of an F-16 in the presence of a $(1deg/sec)^2$ sensor noise variance and a Gaussian input disturbance given by 1% of the maximum control surface deflections. We showed that a system undergoing an attack in which the attacker has a 10% model error results in attack detection and the ability to update state estimates.

5.2 Future Work

A number of next steps that can be taken to improve upon the work presented in this thesis.

1. The work in this thesis was based on a technique developed for discrete systems. The discrete attack and defense techniques presented were applied to a continuous aircraft dynamics model after discretely sampling data. In general, it is common to design in the continuous domain in aircraft applications. Therefore, for aircraft, and perhaps for a number of other CPS, it is important to see how these techniques can be redesigned for continuous time applications.
2. In Chapter 4, we saw general improvement in the state estimates with earlier attack detection. Further investigation of detection techniques is required to enable earlier detection of the attack and therefore improved state estimates.
3. From the perspective of the defender, it is useful to investigate if there is an ideal choice of baiting parameter, Δ . A good place to start is to find a choice of Δ that is likely to result in attack detection for a given choice of detector. An optimal choice of Δ will lead to the fastest possible attack detection for a given choice of detector. This optimization problem will likely be subjected to a number of constraints. We would like the choice of Δ to minimize the effect on the state estimation error. We should also probably constrain Δ to be within some bounds so that the baited model is not so drastically different from the true model so as to make the attacker suspicious.
4. In Sections 3.5 and 4.3 we saw that the defender can estimate the time of attack onset prior to detection after a period of observation. This was done with the goal of improving state estimates. It was noted that since this is done after a period of observation, the defender is not recursively updating their estimates in real time using this method alone. We then mentioned that the attacker can use the recursive techniques of Baiting step 2 (see Sections 3.3 and 3.4.2) to update the estimates until the end of the observation period and then update the estimates again using the methods of Sections 3.5 and 4.3. It is worth investigating how the attack onset estimation can be efficiently executed at each step rather than just once after a period of observation and what the resulting state estimates look like, so as to enable better real time estimation.

THIS PAGE INTENTIONALLY LEFT BLANK

Bibliography

- [1] Matthew A Bishop. *Introduction to Computer Security*, volume 50. Addison-Wesley Boston, 2005.
- [2] Y. Cherdantseva and J. Hilton. The evolution of information security goals from the 1960s to today. Available Online at <https://users.cs.cf.ac.uk/Y.V.Cherdantseva/>, 2012.
- [3] Yulia Cherdantseva and Jeremy Hilton. A reference model of information assurance & security. In *2013 International Conference on Availability, Reliability and Security*, pages 546–555, 2013.
- [4] Tim Conway, Robert M Lee, and Michael J Assante. Analysis of the cyber attack on the ukrainian power grid. defense use case. Technical report, Technical Report. SANS ICS, 2016.
- [5] Munther A Dahleh and Ignacio J Diaz-Bobillo. *Control of Uncertain Systems: A Linear Programming Approach*. Prentice Hall Englewood Cliffs, 1995.
- [6] Seyed Mehran Dibaji, Mohammad Pirani, David B. Flamholz, Anuradha M. Annaswamy, Karl Henrik Johansson, Aranya Chakraborty, and Eugene Lavretsky. A systems and control perspective of cps security. *Submitted to Annual Reviews in Control*, 2018.
- [7] David P Fidler. Was stuxnet an act of war? decoding a cyberattack. *IEEE Security & Privacy*, 9(4):56–59, 2011.
- [8] Andy Greenberg. How an entire nation became russia’s test lab for cyberwar. *Wired*, June, 20, 2017.
- [9] Zhenghao Hu, Yu Hu, and Brendan Dolan-Gavitt. Chaff bugs: Deterring attackers by making software buggier. *arXiv preprint arXiv:1808.00659*, 2018.
- [10] Michael I Jordan. Notes on recursive least squares. *University of California, Berkeley*. www.cs.berkeley.edu/~jordan/courses/294-fall98/readings/rls.ps, 1998.
- [11] Cheolhyeon Kwon, Weiyi Liu, and Inseok Hwang. Analysis and design of stealthy cyber attacks on unmanned aerial systems. *Journal of Aerospace Information Systems*, 11(8):525–539, 2014.

- [12] E Lavretsky and K Wise. *Robust and Adaptive Control: With Aerospace Applications*. Springer, 2012.
- [13] M-A Massoumnia, George C Verghese, and Alan S Willsky. Failure detection and identification. *IEEE Transactions on Automatic Control*, 34(3):316–321, 1989.
- [14] Kayla Matthews. 6 sure signs someone is phishing you—besides email. Available Online at <https://blog.malwarebytes.com/101/2018/09/6-sure-signs-someone-is-phishing-you-besides-email/>, 2018. Image reference.
- [15] Robert McMillan. Siemens: Stuxnet worm hit industrial systems. *Computerworld*, 14, 2010.
- [16] Raman Mehra. On the identification of variances and adaptive kalman filtering. *IEEE Transactions on automatic control*, 15(2):175–184, 1970.
- [17] Yilin Mo, Sean Weerakkody, and Bruno Sinopoli. Physical authentication of control systems: Designing watermarked control inputs to detect counterfeit sensor outputs. *IEEE Control Systems*, 35(1):93–109, 2015.
- [18] Richard S Russell. Non-linear f-16 simulation using simulink and matlab. *University of Minnesota, Tech. paper*, 2003.
- [19] Jerome H. Saltzer and Michael D. Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278–1308, 1975.
- [20] Bharadwaj Satchidanandan and Panganamala R Kumar. Dynamic watermarking: Active defense of networked cyber-physical systems. *Proceedings of the IEEE*, 105(2):219–240, 2017.
- [21] Lance Spitzner. Honeypots: Catching the insider threat. In *Computer Security Applications Conference, 2003. Proceedings. 19th Annual*, pages 170–179, 2003.
- [22] Lance Spitzner. *Honeypots: Tracking Hackers*, volume 1. Addison-Wesley Reading, 2003.
- [23] Brian L Stevens and Frank L Lewis. *Aircraft control and simulation*. John Wiley & Sons, Inc., 2003.
- [24] Sean Weerakkody and Bruno Sinopoli. Detecting integrity attacks on control systems using a moving target approach. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 5820–5826, 2015.