THESIS

APPLICATIONS OF SIMULATION IN THE EVALUATION OF SCADA AND ICS SECURITY

Submitted by

Brandt R. Reutimann

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2020

Master's Committee:

Advisor: Indrakshi Ray
Co-Advisor: Joseph Gersch

Peter Young

ABSTRACT

APPLICATIONS OF SIMULATION IN THE EVALUATION OF SCADA AND ICS SECURITY

Power grids, gas pipelines, and manufacturing centers provide an interesting challenge for cybersecurity research. Known as supervisory control and data acquisition systems (SCADA), they can be very large in scale and consist of hundreds to thousands of physical controllers. These controllers can operate simple feedback loops or manage critical safety systems. Following from this, cyber-attacks on these controllers can be extremely dangerous and can threaten the distribution of electricity or the transmission of natural gas that powers electrical plants. Since SCADA systems operate such critical infrastructure, it's important that they are safe from cyber-attacks. However, studying cyber-attacks on live systems is nearly impossible because of the proprietary nature of the systems, and because a test gone wrong can cause substantial irreversible damage. As a result, this thesis focuses on an approach to studying SCADA systems using simulation. The work of this thesis describes considerations for developing accurate and useful simulations as well as concerns for cyber vulnerabilities in industrial control environments. We describe a rough architecture for how SCADA simulators can be designed as well as dive into the design of the SCADA simulator built for research at Colorado State University. Finally, we explore the impact of falsified sensor readings (measurement attacks) on the safety of the natural gas pipeline using simulation. Our results show that a successful measurement attack on a gas system requires a sophisticated plan of attack as well as the ability to sustain such an attack for a long period of time. The results of this work show that a gas system reacts slower than would be expected of a typical electrical system.

ACKNOWLEDGEMENTS

I would like to send a deep thanks to Jerry Duggan, Dan Zimmerele, Dr. Peter Young, and the employees of the CSU Powerhouse and METEC site for their contributions and dedication in creating a safer world for industrial control systems. Thank you to Dr. Ray for always encouraging me in my work and for always believing that I could achieve something great. Thank you to Dr. Gersch for helping me see the big picture, for keeping me focused on what really matters, and reminding me to "not let my schooling get in the way of my education". A deep thanks to my colleagues Luis Rodriguez and Saja Alqurashi for expressing interest in this work and helping it continue into perpetuity. Finally, I would like to thank my friends and family for encouraging me and asking about my work even if you did not always understand it.

DEDICATION

I would like to dedicate this thesis to my grandfathers Melvin B. Mattison and Robert Reutimann.

Thank you for teaching me to always take pride in my work. I love and miss you both.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

Industrial control systems are an increasingly target-rich environment for cyber-criminals, terrorists, and advanced persistent threats. Those entities who would wish to do harm to government entities or corporations can exploit the complex nature of these systems to their advantage. As control technology becomes more connected through advances in networking technology and design, the physical systems underlying these control topologies also become connected to clever malicious actors. Additionally, the critical nature of systems such as power plants or gas pipelines can make them high priority targets for those wishing to ransom, extort or cause damage to a nation's critical infrastructure. As a result, the security of these systems is tremendously important.

Work in industrial control system security (ICS) focuses mostly on a handful of critical embedded systems. Usually when it comes to security this centers around the embedded control systems that operate power grids, gas pipelines and manufacturing centers. These systems can be incredibly sophisticated. Modern industrial control systems are typically operated by large networks of interconnected controllers and operators. These networks are colloquially defined as supervisory control and data acquisition systems (SCADA). Hundreds to thousands of process logic controllers are connected to SCADA and historian severs as well as operators of the over arching system operators. Sometimes these SCADA systems may even be connected to corporate networks where typical IT security can become just as much of a concern as SCADA system security.

The size and temperament of large SCADA systems leaves several vulnerable attacks surfaces. For instance, controllers in large scale SCADA systems tend to be heterogeneous as it can be logistically infeasible to replace or update all of the controllers at the same time. The switching costs involved in replacing or modifying controllers can also be high as gas and electrical systems typically must maintain a high level of availability through out their life cycle. As a result, in any SCADA system it is likely that there can be entry points for a clever attacker who can take advantage of the "weakest" links in the system. It follows from this that most defenses in SCADA

systems are to isolate the SCADA network from the outside world. This is an extremely effective strategy but not impervious. Often attackers can find ways to tunnel into the SCADA network either through a corporate network, or through other clever methods.

The most popular example of exploiting a SCADA system and its controllers is the Stuxnet worm used to attack the Iranian nuclear program. The worm did not get into the system using conventional methods of exploiting network security. Rather, the malicious software was loaded on to flash drives which were inserted into the system by an unsuspecting user. In addition, local area networks in the system were used for propagation. This worm is particularly interesting because it is an early recorded case of directly attacking the controllers to cause physical damage to the components in the system [1]. This attack gives insight into how potentially dangerous a compromise of controllers or SCADA components can be.

As SCADA systems are large, extremely complex, and critically important for our daily lives studying them for cyber security is of obvious value. However, the process of studying these systems is challenging and sometimes potentially dangerous. A large portion of data on SCADA systems is proprietary, leaving many researchers guessing about how to best model and design these systems. In addition, studying live systems can cause safety hazards such as gas pipelines and electrical grids being put under unnecessary stress. That's why the work of studying these systems requires the effort of not only expert computer scientists but also engineers and safety and security experts. The goal of our work, then, is to demonstrate how a SCADA system can be studied safely and accurately using simulation techniques. By using a simulated virtual test bed, we can easily iterate and redesign our simulations to be as accurate as possible. This also allows us to model cyber attacks without ever harming a real system.

In this thesis, we ponder several research questions pertinent to the design of SCADA simulations and test-beds. Firstly, we ask how can we design a system in such a way that it is scalable enough to model thousands of miles of electrical grid or gas pipeline, while at the same time being realistic enough that it shows behavior similar to it's real world counterparts? Additionally, when modeling these systems how can we validate their behavior in terms of the physical dynamics,

networking, and control behavior? Another question is: by developing simulations for SCADA environments, can we create interesting cyber-security scenarios that will help answer real world dilemmas? One problem, of particular interest to us, is what happens when the control layer starts lying to the supervisory system about the state of the physical world (what has been coined in literature as a measurement attack) [2]? Is there a way to ensure that the reported state of system sensors is consistent with the context of the rest of the system from a physical perspective?

In our research we will explore these research questions with a particular focus on measurement attacks. Our contributions include defining a set of requirements and considerations for developing useful SCADA simulations and test beds. Then we introduce another method for the simulation of SCADA systems, as well as describe a modular architecture that allows for the integration of real hardware or physical models. This method will mainly focus on the development of a control layer that can integrate with enterprise software for the simulation of physical systems. Following the design of this simulation we demonstrate an attack on a simulated gas pipeline model. Finally, we discuss strategies for large scale SCADA simulation, hardware in the loop simulation using a mix of real and virtual controllers, as well as requirements for the virtualization of SCADA controllers.

The rest of the thesis is organized as follows. Chapter 2 discusses related work where we look into previous literature on SCADA simulation and how it has been used to study industrial control systems. Chapter 2 also reviews the MATLAB based Simulink Tool that we use in our work to model gas systems. Chapter 3 proposes critical requirements for the design of SCADA simulators. Chapter 4 explores cyber security threats to industrial control systems and how they should be considered for simulation. Chapter 5 explains the design of the SCADA simulator we created for studying measurement attacks and its limitations and capabilities. Chapter 6 is where we test measurement attacks on simulated SCADA systems and observe their impacts. In Chapter 6.7 we discuss the results of our measurement attack experiments and how they are unique for gas systems. Following the discussion, in Chapter 7, we look into important avenues for improving our simulations and experiments that we would like to run in additional trials. Finally, in Chapter 8 we conclude this thesis by summarizing the important findings and insights from this work.

Chapter 2

Background and Related Work

2.1 Literature Review

In order to ensure the security of SCADA systems blue teams, white hat hackers, and those wishing to employ cyber-defenses must be able to anticipate and defend against possible attacks. The best way for them to do this is to probe these systems for flaws on their own. Unfortunately, the testing of industrial control systems can be a complex, difficult and sometimes dangerous process. Traditional methods of testing these systems usually include some sort of traditional engineering validation methods. However, when testing the cyber-security of these systems previous research has shown that there are some serious risks. In one report, the use of ping sweeps caused a robotic arm to swing on a factory floor and in another case caused a system failure that resulted in over $50,000 worth of damage to equipment [3]. As a result of the risks of cyber-security and non-physical testing, a recent paradigm in research has focused on simulating industrial control environments - described in this paper as SCADA (supervisory control and data acquisition) systems. Researchers from Mississippi State University have successfully created several physical SCADA systems for their test bed [4]. This test bed includes a water tank, water tower, small gas pipeline, factory conveyor belt, and smart grid system. The same authors were also able to create virtual models of their water tank, and gas pipeline systems. These virtual models were validated against the physical implementations of the systems [5]. Although these simulations models have actual physical implementations, the physical test beds are extremely simple. These models have single feedback loops and simple control sequences. As a result, it can be hard to do any security testing that relies on attacking the control system algorithms. Instead these models focus on attacks on network or physical components of the test bed.

Prior research has shown successful tests of various network-based attacks on virtual SCADA test beds [2]. While several other attempts have been made to model large scale behavior of

SCADA systems from a test bed or network-based perspective [6,7]. However, the aforementioned research either only models the network behavior of SCADA systems or is limited to a small scale physical test bed. As a result there is a shortfall in this previous work. A SCADA system model is either lacking in granularity by only focusing on networks, or lacks larger scale perspective from only focusing on physical control systems. In addition, systems that do have physical implementations are limited by the fact that most test beds only have the resources to acquire so many real components. Survey research of this problem has shown that hybrid and hardware in the loop simulations tends to lead to the best of both worlds [8]. However, these solutions are not as cost effective as software modeling and virtualized systems. This shows that scalability, accuracy, and fidelity to the real world are critical components of these systems. Not only for the development of test beds that can be used for security testing but, also for the design of system simulations from an engineering standpoint. In some sense, the use of simulating full scale SCADA systems can also be considered as a tool for designing and reasoning about large scale gas pipelines and electrical grids.

In addition to studying SCADA simulation in general, the interactions between gas and electrical systems has also been a popular area of study. Particularly when it comes to the security of these systems. In certain urban areas, gas and electrical systems can be tightly coupled together in the sense that gas-fired power plants drive electrical operations, and electrical power can supply gas compressors which handle the transmission of gas to power plants. As a result, failure of these gas systems can cause failure in electrical systems and vice versa. Some literature has tried to tackle this interaction by using mathematical models to show the interdependencies of the systems described as a mathematical optimization problem [9, 10]. Other work has expanded upon this by creating models to simulate this interaction. These simulated systems have shown that failure in a handful of gas lines can lead to cascading failures in a corresponding electrical system [11]. However, these models can only consider a limited set of variables as opposed to a high fidelity simulation like the ones we will use in this paper. Previous work in SCADA simulation has focused mainly on the operation of single facilities or models of electrical systems [12].

Although SCADA simulation is a topic that does have a wealth of literature behind it, we believe our work fits into a unique role. At the time of writing we are not aware of any work in SCADA simulation that studies gas systems or their effects on the electrical systems. Especially when it comes to cyber attacks and vulnerabilities. Although this work does not cover the interaction between electrical and gas systems in depth, we do explore vulnerabilities in simulated gas systems that have the potential to harm a hypotheisized electrical system. We leave the complex feedback interactions between these two systems for future work. In addition to being some of the first simulation work to explore gas transmission systems, our work also explores measurement attacks in a new and unique way. Based on our review we have only seen a limited body of work on measurement attacks and how they can applied to large scale systems. The following research explores this research question by testing several measurement attacks on gas pipeline models. Finally, we believe that our design for simulating SCADA systems provides a unique and simplified architecture that can be expanded in future work to create modular simulations that can be used to study all kinds of cyber physical systems. To our knowledge, most work before this has mostly focused on creating simulations for specific types of systems (mainly electrical) and has neglected a method for creating a more expansive simulation model.

2.2 Simulink

Simulink is a MATLAB based tool for modeling the dynamics of systems. It uses a graphical block based interface that allows users to piece together components of a simulated system. The Simulink programs developed for our SCADA simulation are built upon the Simscape library. Simscape is a toolbox for Simulunk systems that allows you to build more sophisticated systems on top of Simulink. Both Simulink and Simscape are tightly integrated with MATLAB making it easy to integrate programmatic component into physical models when necessary. We chose Simulink and Simscape because of their wide variety of tools and libraries for making scalable and accurate simulations. However, the MATLAB/Simulink/Simscape package is not open source, and is being leveraged based on Colorado State University student licenses. The core value in using

Simulink for our project is that it is not limited to simulating gas systems, it can also be used to model electrical systems, micro grids, or even the dynamics of something such as a heavy truck. By being able to model so many different physical systems it is possible that in future project iterations this simulation architecture could be expanded for cyber security research in several embedded system domains.

Chapter 3

Requirements for Developing Realistic SCADA Simulations

3.1 Integrating with existing process simulation software

Although it is possible to create in-house process simulations for SCADA models, doing this disregards a strong body of work in the world of simulating the nature and dynamics of physical systems. As a result of this, many of the virtualized SCADA systems built for cyber-security are backed by complex mathematical simulators that accurately model the state of real world systems. Two of the most popular of these systems in the world of SCADA testbeds are PowerWorld and Simulink. When designing the control layer for a SCADA simulation it is important it can integrate with several different physical simulators. In this sense, it is necessary to design control layers with a modular API in mind. Therefore allowing the emulated control layer to act much like real PLCs do in real world systems. In order to design this API, we reason that emulated PLCs have three main objectives: communicate with the control software over ICS protocol such as MODBUS or DNP3, perform I/O operations on a real or simulated process systems, and perform limited logic operations on their own, such as activating pumps when pressures get too low. We argue that the best way to ensure the modularity of this API between emulated PLCs and simulated or real models is to abstract the behavior of these PLCs into either reading from sensors, or writing to actuators. The behavior of writing to an actuator or reading from a sensor is then converted by some middleware into valid operation for one of the integrated simulation software or hardware in the loop components.

3.2 Validating Simulations

The process of ensuring that a simulated SCADA system acts like a real SCADA system is challenging and complex. The solution to the problem may seem trivial in a small system. Recent research has shown that for small models, a simulated system can be compared to a real one by using captures of packet logs, and comparing the recorded behavior of a real system with the captured simulated behavior of a physical model [4]. However, as simulated systems grow to represent large scale models of power grids or gas pipelines it becomes very difficult to validate a simulation with this strategy. Mainly, as a real world system grows it becomes increasingly complex with multiple layers of feedback between different components. Trying to capture the behavior of this system on a summary level and compare it to a simulation might give false intuition as to how the two compare. One approach to this problem could be to validate large simulations on a component by component basis, however this method would disregard the interactions between these complex systems. As achieving accurate simulations is integral to using simulation technology for SCADA research this validation problem is a key area of any research in SCADA simulation.

3.3 Scaling Simulations

Most of the current attempts to simulate SCADA systems have dealt with fairly small scale systems [7, 13, 14]. These simulations deal with systems composed of tens of controllers and single purpose process models. However, having a simulation that could represent the power grid of an entire city or thousands of miles of gas pipeline would be particularly useful for testing the nature and cyber-security of large systems. The challenge with scaling a simulation is that in order to meet real time demands systems often lose granularity. This means that most large scale simulations overlook real time interaction with emulated controllers. An interesting research challenge is to find a way that a system can support thousands of emulated controllers without introducing unnatural delays and errors due to overloading the simulation software. Part of this problem could be solved using either higher powered machines or by virtualization and distributing the load of simulating controllers over several machines. But sometimes adding more compute

power may not be the best solution. Another approach could be to have some controllers that are dummy models while others are full interaction high fidelity emulated controllers. This strategy, in theory, could allow us to interact with controllers of interest while other controllers follow a set of predefined, or process defined, behavior. Scalability is certainly another interesting and complex problem when developing and evaluating SCADA simulations.

3.4 Simulating Long Term System Behavior

Another interesting problem and value proposition for software and virtual simulations is the idea of simulating behavior in the long term. For example, instead of looking at the instantaneous impact of a controller being compromised it might be of interest to explore the resulting changes in the system over a period of several days or weeks. This functionality, nonetheless, adds another challenge to the development of a SCADA simulation. One aspect of this challenge is synchronizing the rate of simulated time to real time across the controller layer, process layer, and control software (HMI) in such a way that all the systems speed up time in an appropriate manner. Another challenge is how to ensure existing controller emulators and control software that can handle a speed up of commands that would occur by running in a ratio of, 10:1, of simulated to real world time. To go along with this, how do you determine when network communication becomes a bottleneck to how fast you can speed time up in a simulated environment? This makes simulating systems in the long term another interesting but valuable challenge that faces SCADA simulation. One possible approach to this problem is synchronizing the various components of the simulations to a shared global clock. This global clock would act as a timing oracle in the simulation that ensures that the simulations components follow a consistent time schedule by either slowing, pausing, or speeding up specific parts of the architecture. However, the implementation of this clock can be exceptionally difficult as many implementations of physical system dynamics are not necessarily discrete. Some of these physical systems may be computationally difficult to iterate in a discrete manner, or it just may be difficult to interface with existing software in such a way.

3.5 Using Hardware in the loop (HIL) to increase accuracy

A common approach to increase the fidelity of SCADA simulations is to add hardware controllers into virtual simulations. In our case we propose using some virtual and some physical controllers. This approach has numerous benefits. First of all, it increases the quality of simulating a particular controller since you have an actual controller. With cyber-security research you can focus on attacks on these physical controllers. In addition, any vulnerabilities found on a hardware in the loop controller can then be replicated in other emulated controllers in the system. This will allows for the benefits from using both physical controllers and virtual controllers: physical controllers provide high fidelity and accuracy and the virtual controllers make it cost effective and scalable. Having hardware and virtual controllers has been considered in some literature as a hybrid approach [8]. However, one possible drawback of this approach is that it may not integrate nicely with the long term simulation ability that is mentioned earlier in this paper, but this has yet to be researched. It is much easier to reconfigure virtual controllers than it is to retrofit physical ones.

3.6 Simulating Network Behavior

The modern approach to designing large scale SCADA systems is to use the implementations of existing network protocols in order to control traffic via MODBUS or other industrial protocols. However, because of security and performance concerns it is usually prohibitive to use the Internet to carry control system traffic. Since gas pipelines or power grids use wide area networks with a broad scope of differing protocols, the networking protocols used in a SCADA system themselves can be a point of attack. Oftentimes if given access to these networks an attacker can use denial of service to prevent the transmission of valuable control or sensor data. Although these types of network attacks are not unique to SCADA systems, they can have catastrophic effects when it comes to the timing and transmissions of data in a functional safety environment. As a result, being able to model networks between control systems like HMI's (human machine interface) and networked PLCs is an extremely valuable tool for testing the security of SCADA systems. How-

ever, the model of these networks is a non-trivial problem. Mainly, this network behavior can be modeled by setting up actual physical networks, by using some software defined mechanism for these networks, or by creating a module that simulates the behavior of the network but does not actually implement it's protocols. Each one of these approaches carries its own drawbacks, for example setting up a physical network may be too heavy weight and difficult to observe, while using a completely simulated network prevents testing of certain attacks. Network and simulation and testing is not a primary focus of our research but it is something that is an important consideration in improving models of SCADA simulation.

Chapter 4

Cyber-Security Threats to SCADA Systems

4.1 Compromising a SCADA network

Attackers can use multiple strategies to gain access to the SCADA network. Common paths to compromise in recent years include SQL injection, phishing and spear-phishing, weak authentication, and removable media [15]. It is common for SCADA networks to be connected in some way to greater corporate networks with firewalls and network intrusion sensors guarding entry points into the corporate network and into the SCADA network [16]. However, field agents who operate both on the corporate network and SCADA network performing maintenance, migrating laptops and devices between both networks, provide an interesting side channel for attack. If the device of one of these agents were to be compromised then it could be used as an entry point for an attack or point for deploying malware onto the SCADA network. Once an attacker has gained access to the SCADA network, they have several options for causing damage to the SCADA system components or for attacking the physical system they control. Hypothesized network based attacks include denial of service, measurement based attacks, and command injection. Denial of service can also be expanded to either saturate the SCADA network or take down switches or controllers on the network. Nonetheless, the amount of possible attacks once on the network are not limited to these. SCADA devices have strong real time constraints. They are often left without updates to their software. This is because it can be too costly to take down the whole system to do software updates. As a result, these devices are also vulnerable to attacks that prey on their out of date software and protocols.

4.2 Command Injection Attacks

Command injection attacks are another extremely dangerous flaw that exists in traditional SCADA systems, particularly those communicating over the MODBUS protocol. The MODBUS

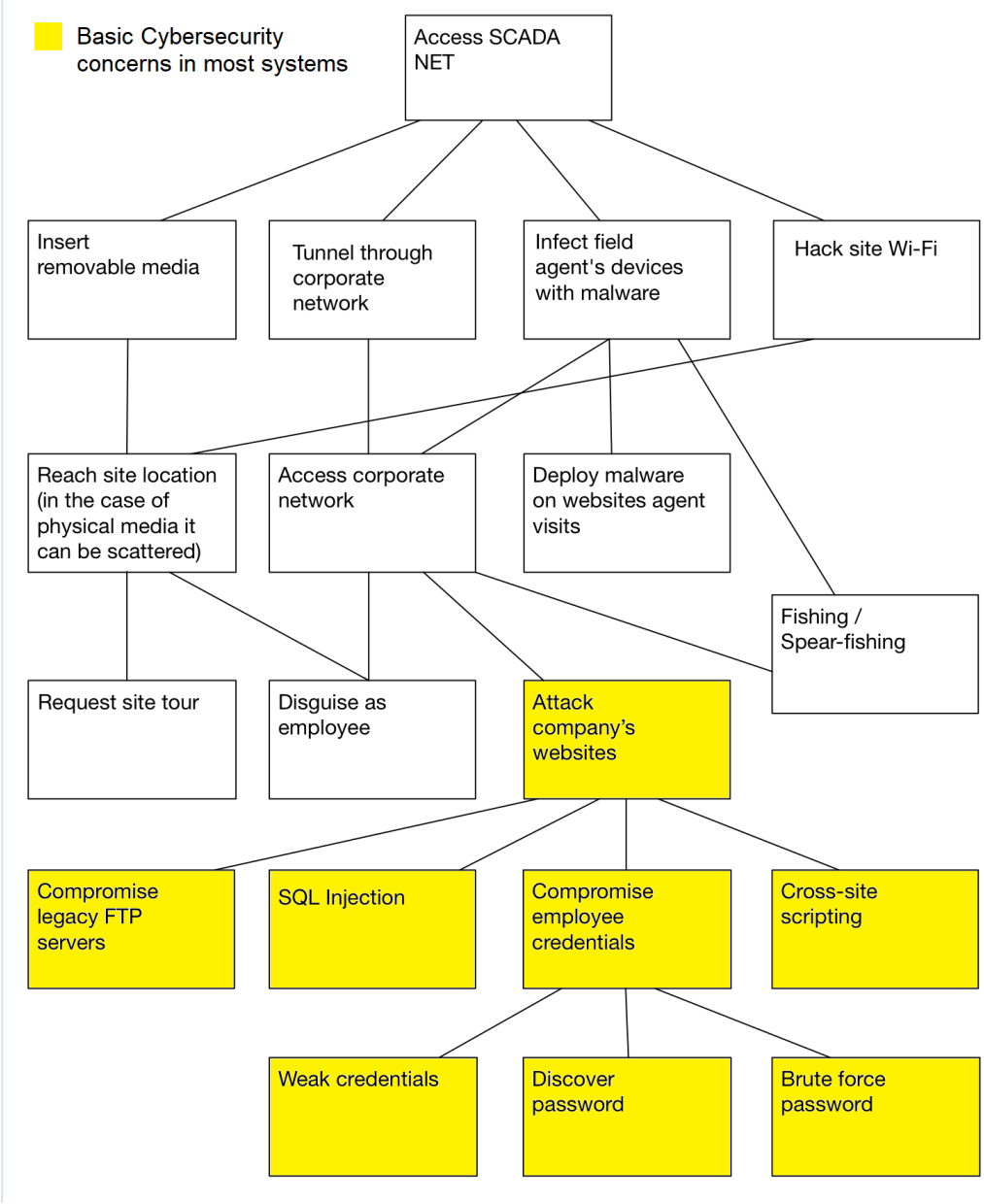


Figure 4.1: Example Attack tree for accessing a SCADA network. Information from [5].

standard was not originally designed for security, or even to be used with TCP/UDP, as original implementations were for a serial RTU based approach. As a result, many controllers on SCADA networks are open to listening to any commands regardless of origin or access control. This can cause attackers to exploit the "trusting" nature of the network. For example, an attacker may be able to send a MODBUS command cancelling an action from the control center. This kind of attack could have a catastrophic effect on the system as it could prevent a operator from taking a safety oriented action. A skeptic of this kind of assault might argue that the problem could simply be fixed by using cryptography or access control. This is a very fair point for considerations from a security standpoint, however it is complicated by the fact the distributing keys to sensors, especially those already in place, is a non-trivial process. In addition, there are a lot of SCADA hardware and sensors which are not equipped to use the strong encryption algorithms that would be required to secure the system. As a result, command injection attacks can be extremely dangerous and difficult to defend. Following from this, many strategies that mitigate risk for this kind of attack might focus on preventing an attacker from gaining access to the control network in the first place.

4.3 Ransomware

Ransomware attacks have recently come into light as a very dangerous and common weapon used by malicious actors. Typically ransomware attacks start by a system being compromised by some malware, this malware will encrypt sensitive data and then hide or throw-away the encryption keys. This makes it nearly impossible to recover the data. These attacks get their name because the attackers will typically offer a decryption key to the locked data in exchange for a ransom price. A recent case of a ransomware attack on a US gas pipeline has shown that when securing SCADA systems these types of attacks must be considered as well [17]. Ransomware attacks on SCADA systems might target historical data stores, or files required to run system software. In the case of this recent attack, the polling servers, human machine interfaces and historical data servers lost availability as a result of a commodity ransomware attack. Although ransomware attacks are

an important consideration for SCADA and ICS security, our simulator has not been designed to simulate or handle these kinds of attacks. Although future research could reveal avenues on how to extend our models such that we can simulate such scenarios.

4.4 Network Based Attacks

The networks used to distribute SCADA traffic are interesting, mainly because they transport a voluminous amount of geographically dispersed data. As discussed early, these networks are now messaging systems that are the backbone of most distributed control. As result, attacks that could compromise the network, or prevent traffic from being transported can result in a loss of control, a loss of view, or both. There are several ways attackers might consider jeopardizing a SCADA network. Once approach could be a straightforward denial of service attack, while others could be more clever attacks that exploit network protocols like token rings. However, the main focus of our research is not on network based attacks since this is a highly explored area of network security.

4.5 Measurement Attacks

In our research, we focus on an attack in the literature that has been called a "measurement attack". The attack requires a set of assumptions. Firstly, that an adversary has found a way to either impersonate or compromise a controller in the SCADA network. This compromise would allow the attacker to send sensor readings and MODBUS responses as if they were the controller. In a trivial scenario, the attacker may send fuzzed responses such as large numbers, NaN values, and negative numbers. However, we theorize that this type of fuzzing could be easily detected, as this type of behavior is very abnormal for a standard SCADA controller. Another attack vector might be to report a malfunction of a perfectly functioning component in a control system. For example reporting a functional gas sensor as malfunctioning or in a more dangerous example reporting a malfunctioned gas as functioning normally. This type of attack could cause a catastrophic failure in a gas pipeline as the control system, believing the gas load is lower than it is, might make decisions to balance the believed gas supply by upping pressures. If pressures get too high then valves will

trip off or gas pipe may even fail. Of course, for this to work an adversary would have to sabotage a gas sensor before reporting it as functioning normally. We theorize that measurement attacks can be done with no such requirement. Measurement attacks work by manipulating the sensor response of control devices in the system. Similar to the previous example, the aim of this attack is to cause a malfunction in the system by feeding the control software compromising values. We hypothesize that in certain situations this attack could be done by compromising only one controller.

However, this hypothesis is something we will test in our research. We hope to research the impact of measurement attacks on a simulated gas system. Previous research has modeled the effectiveness of measurement attacks on a small scale electrical system [2]. We want to see if we can produce results consistent with prior research, as well as expand upon models of these types of attacks on larger simulations. In addition to exploring the impact of measurement attacks on a gas system we wish to explore strategies for defending against these types of attacks. Of course defending against these attacks is principally a two-part research question. Firstly, how can the control software detect that the system is either under attack, or that one of the controllers is faulty? Secondly, if the control software suspects it is under attack then what kind of actions should it take to prevent failure or long term damage to the physical process?

In some respects, the problem of process controllers lying is a "Byzantine Generals" problem in an abstract sense. As the number of compromised controllers increases it will become harder to detect a liar or set of liars [18]. However there are some strategies we hypothesize that can be used to detect liars when their number is fairly small. One preliminary strategy is to use historical data to model the 'normal' behavior of the system, and to configure the control software such that it makes decisions in a manner that is conservative when taking into consideration historical information and the current state of the simulation. Another is to use simulation and modeling of the supervised SCADA system, the same type we use for security testing, as a sanity check for the current state of the system. If the state of the real world system is inconsistent with the simulation's estimate of how it should be, then this might be a scenario where the control system has to make decisions in a manner that will conserve system resources and prevent failure. Finally, a simple

approach might be to employ a set of constraints on the reported system measures such that single points of failure are less common. For example, a simple constraint might be that if the pressure decreases at a specific point in the system we should also see a drop in temperature (as result of the ideal gas law). Another simple constraint is that power stations in similar downstream situations should have a variance in pressure and temperature together.

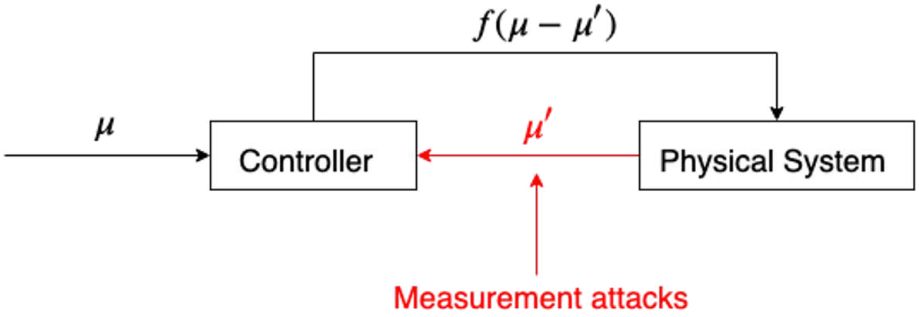


Figure 4.2: Measurement attacks occur within the response part of a feedback loop. In the case of our model the feedback loop occurs between virtual controllers and the system operator.

Chapter 5

Design of the Colorado State University SCADA Simulator

5.1 Architecture of a SCADA Simulator

We view SCADA systems as composed of three main interacting parts. The foundation of any system is in the physical process, in a gas system, this is the actual pipes, valves and compressors that move and distribute gas. In an electrical system, this is the electrical lines, transformers, and power storage systems. On top of this physical process there is usually a control layer, this layer is composed of sensors, and micro-controllers that monitor and actuate the state of the physical system. This control layer may take predefined action, or wait for commands from an outside source. Finally, in order for humans to be able interact with this control layer there is some sort of human machine interface. This interface communicates with the control layer through a SCADA network protocol such as MODBUS or DNP3. The human controller may react to system wide state changes or allow for control action that can take into context the entire system instead of the smaller data views that a single controlling PLC is limited to.

Using this modular approach to SCADA system design affords several capabilities. Primarily this approach allows us to separate the responsibilities of different simulation components in an agnostic fashion that allows them to be interchanged. In our model, we use a MATLAB/Simulink model for simulating the gas system behavior, while an interfacing layer allows that model to interact with virtual controllers. These controllers can in turn interact with either a simulated worker, or an actual real time GUI interface. In other cases, virtual controllers could be swapped with real PLCs, or hybrid PLCs built on top of raspberry pi's. In addition, since each layer is indifferent to the nature of its counterparts, the physical layer could be swapped with a designed physical model or another simulation software.

We also see this three layer design as something that shares commonalities with all embedded systems. For example, a car or heavy vehicle may consist of electronic control units (ECUs) that interact with sensors or display data to the driver. These systems could be modeled or simulated in a similar manner by using a physical process model of the vehicle's mechanics, a control model of the system ECUs, and a user interaction model of the dashboard display or other human interface.

5.2 Design of the Gas Pipeline Model

The gas pipeline model used in our simulation is fairly simplistic, but offers enough fidelity to demonstrate the efficacy of measurement attacks. The gas pipeline model consists of two modeled compressor stations, four simulated power plant loads, two gas distribution loads, two externally controllable gas line valves and several thousand miles of simulated gas pipe. Each power plant and compressor station has sensors for pressure and temperature, as well as sensors that communicate the online / offline state of the power plants. The output pressures on the compressor stations can be adjusted from the control systems in order to meet gas demand. The pipeline model mathematically models the flow rate, pressure and temperature of gas in the system at any point along the pipeline. Modeling the gas system using Simulink in this way makes it simple to change or update the pipe to certain specifications, or reset the simulation in the event of a catastrophic failure caused by a cyber attack.

Although the physical model of the gas pipeline is fairly elementary, the design of the model requires engineering considerations to be authentic. Mainly, the parameters of the model must be considered in a way that replicates actual gas systems. For example, in order to provide a realm of realism to the gas load requirements we must determine how much power must be produced by the power plants in our model and the amount of gas needed to produce that power. In early demonstrative models, such as our first two experiments these properties are less important as the experiments are designed to demonstrate basic principles of measurement attacks. For our third experiment, the power and load requirements are generated based on recorded measures from the City of Fort Collins. Power requirements are converted based on the heating value of natural

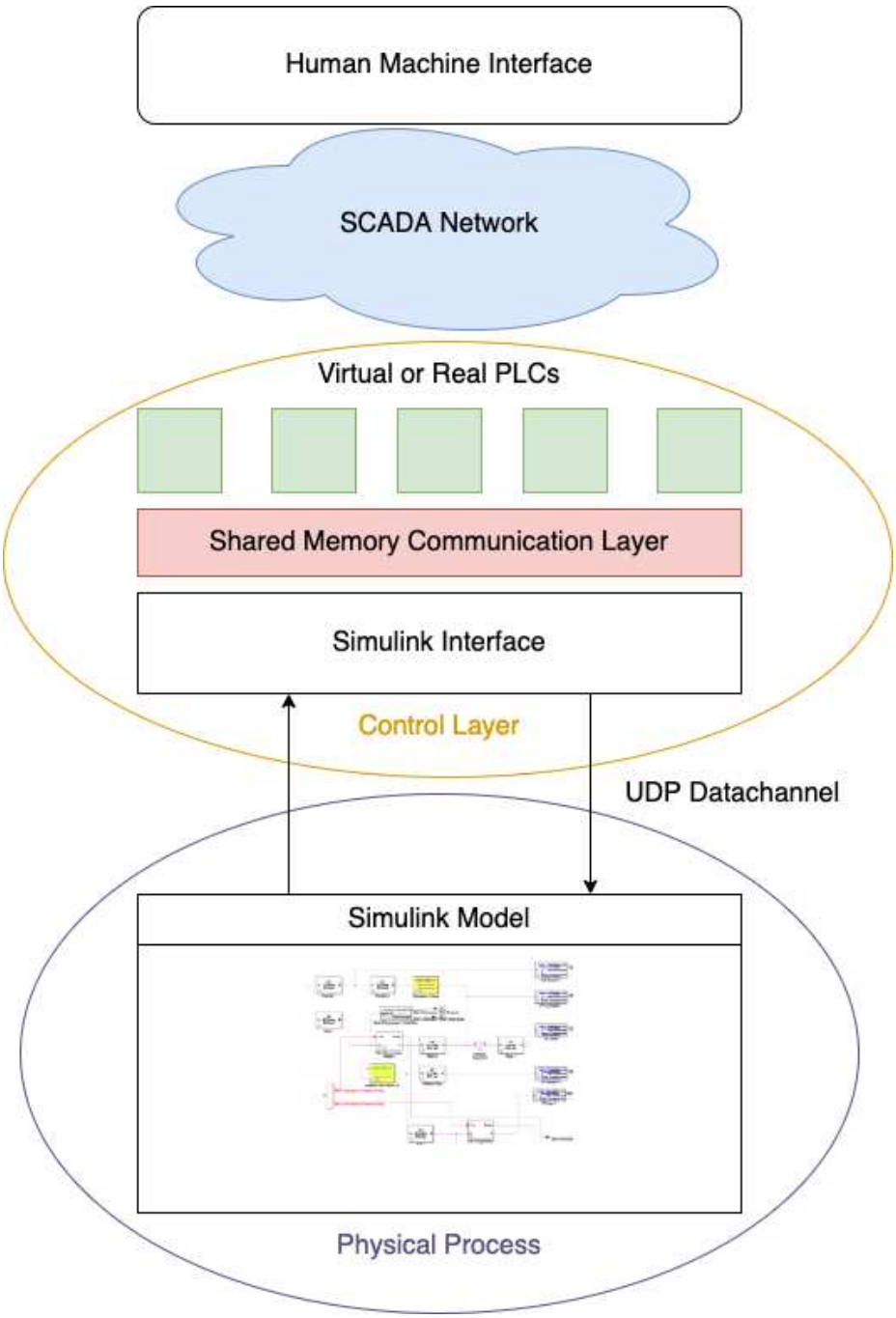


Figure 5.1: The three layer architecture of a SCADA simulation.

gas (22500 Btu/lb, 45357 kJ/kg), and an estimated efficiency of 15% [19, 20]. As a result the conversion function for watts to kg of gas is:

$$kgs\_of\_gas = \frac{watts\_energy}{(45357 * 1000 * .15)}$$

5.3 Design of the Simulink Interface

The Simulink interface is the part of the simulation responsible for connecting the virtual controllers to the simulated gas pipeline model. This interface communicates with the physical process model on behalf of any virtual controllers, therefore acting as a pseudo-API. When the Simulink interface is started it reads a YAML file containing configuration data for its PLCs. The interface then spawns each PLC as a process. These PLC processes in turn are their own MODBUS servers that communicate with the human operator or human machine interface (HMI). In addition, each PLC spins up worker threads for each sensor in their jurisdiction. These worker threads in turn register to receive data from the Simulink interface. In a sense, the Simulink interface is analogous to the wire that connects PLCs physically to sensors and actuators. However, in this model the Simulink interface handles the ability to read and write from the MATLAB simulation.

The Simulink interface is at its core a select server running a single process. This server receives data from the physical model and relays that data using a shared memory queue, and a publish-subscribe mechanism. We chose to use this publish subscribe mechanism for scenarios where we wanted multiple PLCs to view data from the same sensors. Once such scenario is the use of an "Oracle PLC" which will be described later. Additionally, this publish-subscribe mechanism simplifies the process of adding new PLCs. Essentially, the Simulink interface listens for all data coming in from the simulation, and then it will dispatch that data to the appropriate listening process queues. In order for PLCs to respond through the Simulink process, they must be able to communicate to it in an efficient manner. To achieve this, those PLC workers which are configured to send data back to the simulation are given a pipe between the Simulink interface and the PLC. This allows the worker processes to send data to the Simulink interface without having to use a

network connection, and in addition these pipes can be listened to by the Simulink interfaces main select loop. As a result of using shared memory for inter-process communication, the virtual PLCs are closely coupled with the Simulink interface in order to increase performance. We decided on this approach because previous attempts at simulation had demonstrated to us that this was a more efficient method of parallelism. Previous attempts at simulation had spawned all PLCs as threads respective to the main python process, this led to substantial performance issues that affected the realism of the simulation.

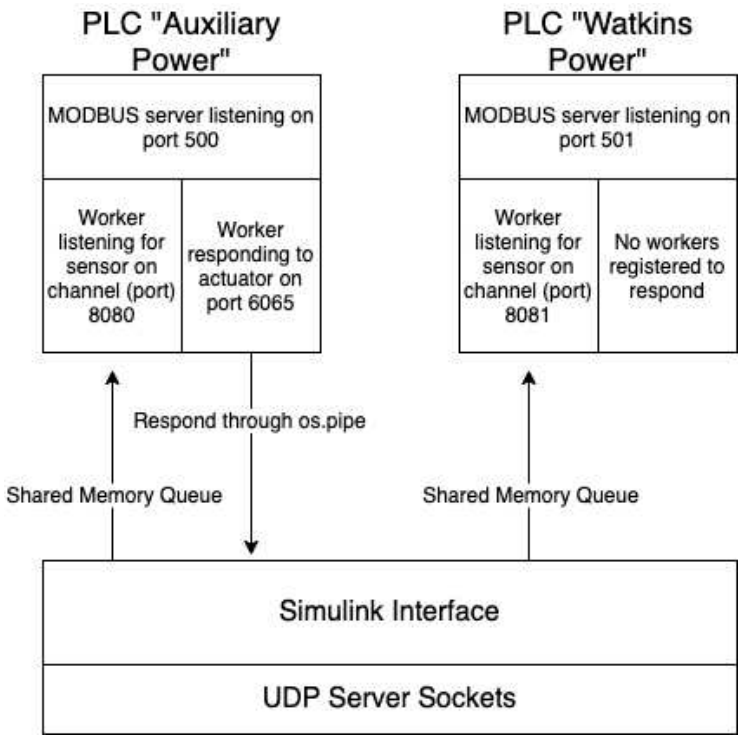


Figure 5.2: Design of the Simulink Interface.

5.4 Resolving Timing Issues in the Simulation

The process of interfacing with the Simulink gas pipeline model is in some respects trivial, but also lends certain challenges. In order to exchange data with the Simulink model we use a set UDP "send/receive blocks" these blocks allow the model to interact with our interface in a non-blocking manner. However, the non-blocking nature of this communication can lead to timing issues. This

occurs mainly because the MATLAB/Simulink model can switch between operating continuously and operating discreetly. What this means is that the MATLAB model will make certain jumps in time - for performance reasons and in order to simplify the process of solving the underlying differential equations. Then the model may switch to discrete operation in sections where data is being received from the interface. This can lead to certain timing challenges. One of the largest being the ratio of simulated time to real time.

When the simulation is advancing in large time steps, it can dramatize the delay in communication from when data is sent out of the model to when the model receives new control actions. For example, during the 20 - 40 ms that it takes for data to exit the model and for the model to get a response, simulated time may have advanced by up to 5 - 10 minutes. This makes it very difficult to model delay between an HMI and physical model, as well as give control actions that are appropriate for the current state of the system. Currently this problem is addressed by adding a timer thread to each PLC and a timer block to the physical model. Once every simulated minute the timer block sends out a simulation time reading. Upon receiving these readings each PLC updates it's clock to the current simulated time. This solution has two primary benefits. The first is that by having a timer block send out data every minute, the simulation cannot make a jump in simulation time greater than a minute. In addition, by reporting it's time to each PLC the control layer is able to maintain an understanding of where in the simulation it is currently. This becomes useful when trying to model PLC compromises later on.

5.5 Simulating Compromises in Sensors

The main value proposition of being able to simulate SCADA environments is so that we can replicate and test the effects of theoretical cyber attacks. In order to do this we need to be able to model specific attack scenarios, or the nature of what certain attacks may look like. Currently our system is only able to model one type of attack, the one we have previously described as a measurement attack. In order to model these types of attacks any sensor worker in the virtual PLC can be marked as compromised. Sensor compromises are a mathematical function of the original

sensor reading, x, and the current simulation time, t, giving us $f(x,t)$. A sensor compromise can something as simple as always reading 10% too high, modeled as $f(x,t) = 1.1 * x$. However, more complex compromises may require a function that is gradually decreasing with respect to t. Either way, by having this flexibility it makes it easy to model different scenarios where compromised PLCs could be lying about sensor readings. Compromised sensors are specified in the configuration file for the virtual PLCs, then a special type of worker is created that wraps the original worker and modifies it's sensor readings (see Figure 5.3). We chose to implement compromises in this way in order to simplify the process so that any worker in the model can be marked as compromised.

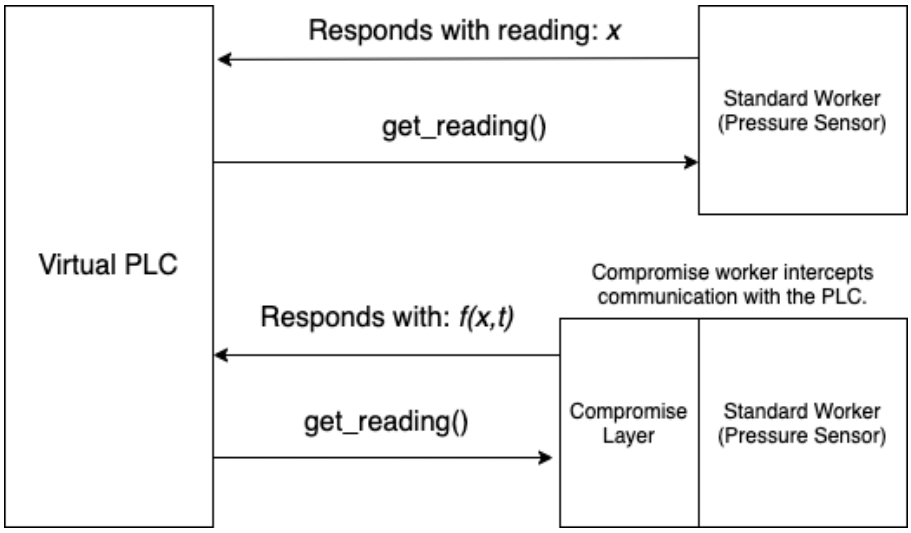


Figure 5.3: Designing worker compromises.

5.6 The Oracle PLC

The Oracle PLC is a critical component of experimenting on a simulated system when compromises are occurring. Any control decisions made by our fake gas system employee or HMI are made on information coming from our virtual PLCs, affording us the ability to model cyber attacks. However, it is also important to understand the disparity between the information the HMI is receiving from these PLCs and what the actual information in the model is. This way it is possible to accurately reflect on the results of the attack scenario and see where exactly the physical model

started to fail. In order to satisfy this need, we designed a special virtual PLC called the Oracle PLC. The Oracle PLC operates identical to any other virtual PLC, except that it is registered on the data channel of every sensor in the physical model. By design, none of the sensors in the oracle PLC should be compromised. As a result, querying the Oracle PLC can allow you to access any sensor data in the model the same way you would with any other virtual PLC. By creating this oracle it is easy to model scenarios where there is only ground truth information, and also allows us to put any controller utilities in one consistent place. For example, the Oracle PLC can also report data about the current simulation time so that any HMI GUI or modeled gas control worker has some sense of the current simulation time, without having to modify it's normal MODBUS behavior.

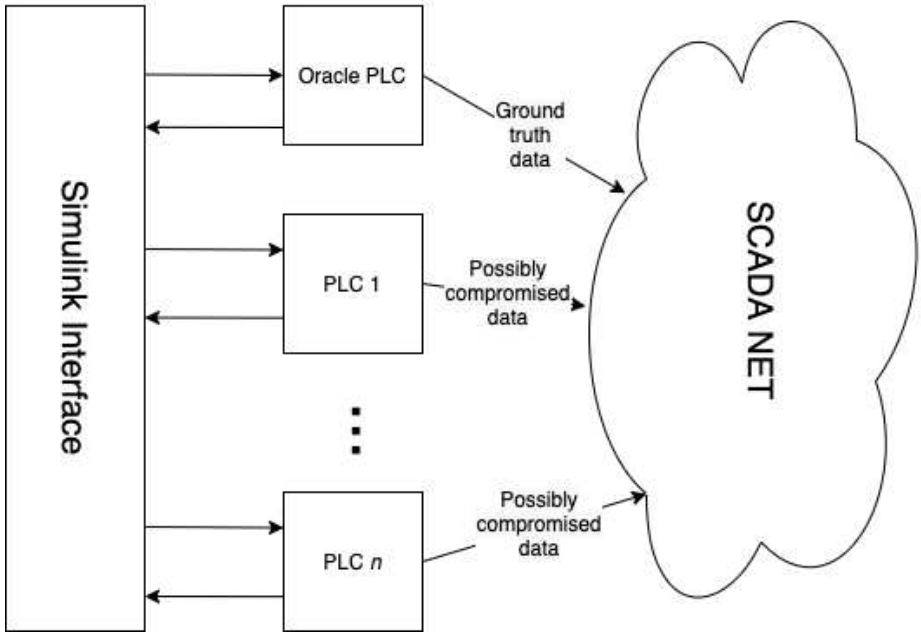


Figure 5.4: Oracle PLC sends ground truth information over the SCADA NET.

5.7 Modeling a Pipeline Operator

The final step in modeling the SCADA control system is in some respects the most challenging. Modeling an operator of the system is complex because these operators must take certain control

actions based on the state of the system, and they should be modeled in such a way that these control actions are realistic behaviors for an actual gas control operator (in our case). For the purpose of our simulation we think of a gas control operator as a simple control flow type diagram. An example of this flow would be something like:

1. Check the pressure sensors at Power Plant X
2. If the pressure is below 500 psi then increase upstream pressure by 50 psi.
3. If pressure exceeds 650 psi decrease pressure by 50 psi.

A simple control behavior like this can be easily and quickly coded in a python script. However, control flows for real gas operators are far more complex involving several variables and safety procedures. The field of control systems theory, however, is out of scope for the infant stages of our simulation project. Hence, we decided to keep control behaviors simple for our model so that we can demonstrate the basic theory and application of a measurement attack. For the most part controller behaviors in our system are dependent on the physical model. For our models, the controller attempts to maintain distribution pressure at all power plants and distribution loads by adjusting the output pressure of upstream compressor stations. In addition, the controller monitors for pressure or temperature dropping below certain thresholds on power stations and trips them off after a certain point. This is to model the scenario where a power plant will have to go offline either because it does not have enough gas to meet its power demand, or because it is no longer safe for the plant to operate with the current gas input.

5.8 Limitations and Capabilities

The current SCADA simulation is designed to test the effectiveness of measurement attacks in its simulated environment. As a result a handful of simulation considerations have been left out in the first iteration of the project. The current simulation is able to integrate with existing Simulink software to extract data and initiate control function in simulated time. Simulations typically run over the course of 1 - 2 minutes of real time, and simulate about 4 - 5 days of simulated time.

Virtual controllers integrated with the simulink model can alter the state of the system while it's running, and are capable of modelling the behavior of faulty or manipulated sensor readings. These manipulations occur within the virtual controller, so the faults are only visible to the fake control operator interacting with the virtual controllers. The operator can send MODBUS requests to virtual controllers in order to alter the state of simulation.

However, being able to extract data, simulate PLC behavior and simulate operator behavior are not the only important considerations within a SCADA simulator. Previously we discussed some important considerations for the development of useful and accurate SCADA simulations. Our simulation does not currently have any capabilities for modeling ransomware, virus/worm propagation, network based attacks, or hardware in the loop simulation. Despite this, the modular nature of the simulation means that it can be easily modified to include these capabilities. Our team's current research focuses mainly on demonstrating a proof of concept for the usefulness of SCADA simulation and on demonstrating the impact of measurement attacks in SCADA systems. As a result, many of the aforementioned simulation capabilities are left out of our model.

A final and important note on the limitations of our system has to do with the process of validating a model's accuracy and fidelity to the real world. Validating our model is extremely challenging, as we have limited empirical data to compare the nature of our simulation to one in the real world. As a result, we have relied on the testimony of gas system experts at the CSU Powerhouse who have taken active part in the development of the gas model. Consultation with expert opinions has given us some confidence in the physical side and gas dynamics of our model. However, an important shortcoming to note is that typically most gas system models are larger and much more complex than the ones we are using for the experts in our current research. As a result, these larger systems are composed of redundant sensors and specialized safety systems that are used to prevent Byzantine failures. Although our model does not currently make use of redundant sensors, we argue that since the measure attack occurs at the PLC level it may not make much of a difference.

Chapter 6

Testing Measurement Attacks in a Simulated Environment

6.1 Experiment Setup

For this project we have decided to run three main experiments. For the first two experiments we will be using a simple gas model consisting of two gas lines, one main compressor, and two power plant loads. The loads for these plants will be predetermined and there will be no cascading failures. This gas model will have 3 main PLCs, one on the compressor station, and one on each power station load. For the final experiment we will use a larger model designed by CSU's powerhouse. The second gas pipeline model consists of two modeled compressor stations, four simulated power plant loads, two gas distribution loads, two externally controllable gas line valves and five main sections of simulated gas pipe. This model also includes a dynamically adjustable load that changes the gas requirements of each power plant based on the amount of required power output, and the amount of power stations that are currently online. The second model will have a virtual PLC for each of its 6 simulated loads as well as one for each of the compressor stations. Both gas models will run for 7 days of simulated time before stopping. Each model will have its own simulated operator who will know the structure of the physical system and attempt to maintain gas distribution pressure by adjusting upstream compressors and will trip off power stations for safety issues such as low gas pressure or temperature.

6.2 Data Collection

Data will be collected in several places of the simulation to demonstrate consistency and to show the effects of the compromises in the controllers. The Simulink model logs all data during the simulation, and we will also track data that is coming from PLCs to the simulated operator.

This data will be marked on the operator side. As a result we will have the operator query both data from the simulation PLCs and from Oracle PLC so that there is a copy of authentic data as well as a copy of compromised data. We can compare these two sets of data in order to show how the real state of the system is affected by the control actions taken on the compromised data. As a result, we have the capacity to demonstrate that data received by the operator is the same as the data in the model, as well as showing the compromises are affecting the real state of the physical model.

6.3 Experiment 1: Plant Failure on Gas Pressure Loss

6.3.1 Scenario

For our first experiment, we explored the impact of a slowly decreasing pressure sensor on a power plant. The purpose of this experiment was to demonstrate the impact that a compromised PLC can have on a single point of failure system. For safety reasons most power or gas systems will have redundant sensors and data points to ensure decisions are made consistently with the nature of the system. However, in our work we take the stand that a clever attacker should be able to compromise more than one PLC in the SCADA network. As a result, we see these types of experiments as an interesting starting point to study the impact of measurement attacks.

The physical model for our pressure failure experiment is extremely simple in order to easily demonstrate the theory behind this kind of attack. The physical model for this experiment consists of two 48" diameter 50 mile gas mains with a compressor in between. At the end of this line there is a power station consuming up to 80 kg/s of natural gas (producing about 500MW of power). The system operator for this model adjusts the compressor output in order to ensure that gas is delivered to the power plant at 600 psi. The operator uses the following feedback function to adjust the upstream pressure: $ps_{t+1} = ps_t + [(600 - p) * 1.5]$. Where p is the downstream power plant pressure, t is time, and ps is the current pressure setting.

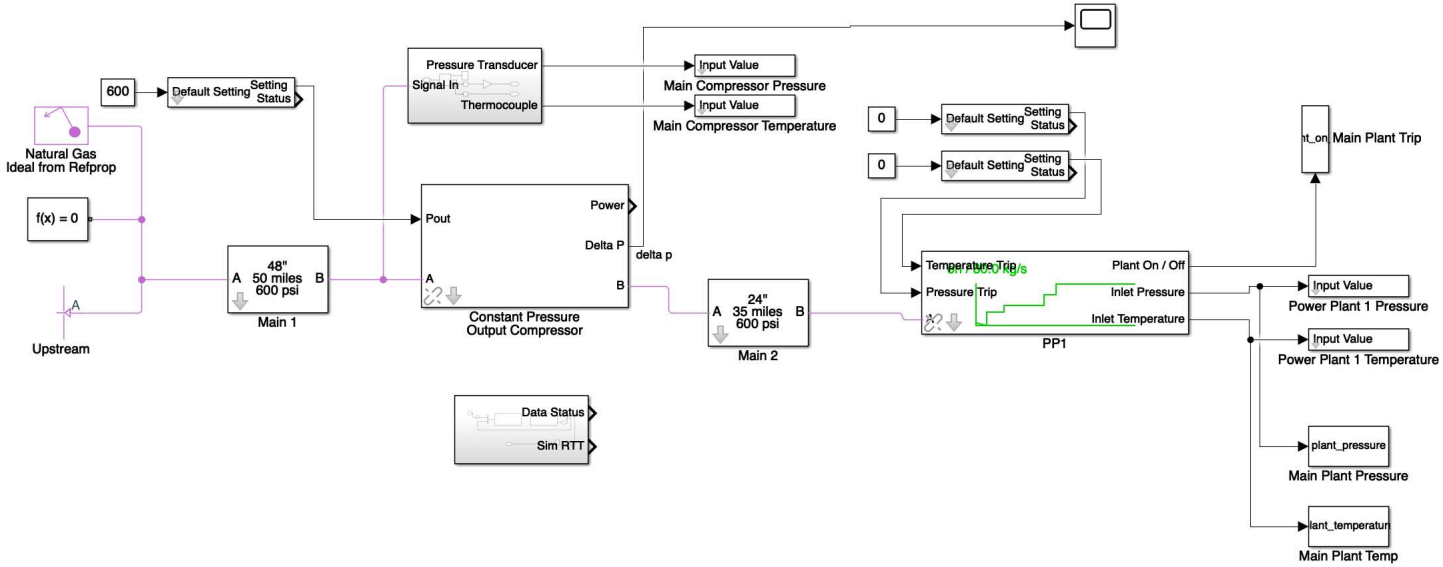


Figure 6.1: The physical model designed for experiment 1.

6.3.2 Results

The results of the first experiment show a dramatic disparity between the actual pressure reading and what reading is being sent to the system operator. In 6.2 you can see this disparity. In the beginning of the experiment we can see that the compromised sensor is reading about 500 psi while the true reading is 600 psi. Then we see the initial spike in pressure in both readings that occurs as the operator is trying to bring up the low pressure from the initial reading of the compromised sensor. As the simulation goes on we can see that the compromised pressure gradually decreases over the next few days of the simulation. The operator has no means of increasing the pressure as the upstream compressor is already maxed out at 800psi. Not pictured in 6.2 is that sometime between days 3 and 4 the power plant load trips off because of low pressure, as a result you can see a small spike in pressure. It is also important to note that temperature is not varying in an expected manner with the falsified pressure readings. Though you can see an expected variation in the physical model results 6.3b. With any rapid drop in pressure we should also have a fairly rapid drop in temperature due to ideal gas laws. As the experiment goes on the load is also increasing rapidly over the first 2 days up to 80 kg/s (6.3c). However, when the power plant trips we end up with an over-pressured pipe at 800 psi and the demand of our power load not being met.

6.4 Colorado Gas Model

6.4.1 Model structure

The Colorado scale gas model used for the next two experiments is a substantially sophisticated gas model that requires special design considerations. The main objective in designing these larger models is to make them realistic enough that the system fails because of a cyber-attack and not because the engineering inherent to the model is poor. In order to do this our team consulted with engineers at the Colorado Powerhouse as well as gas system experts to define what constraints the model must meet. The main concern was getting data to scale our model around. However, the process of acquiring accurate data on control systems like the ones we study is very challenging.

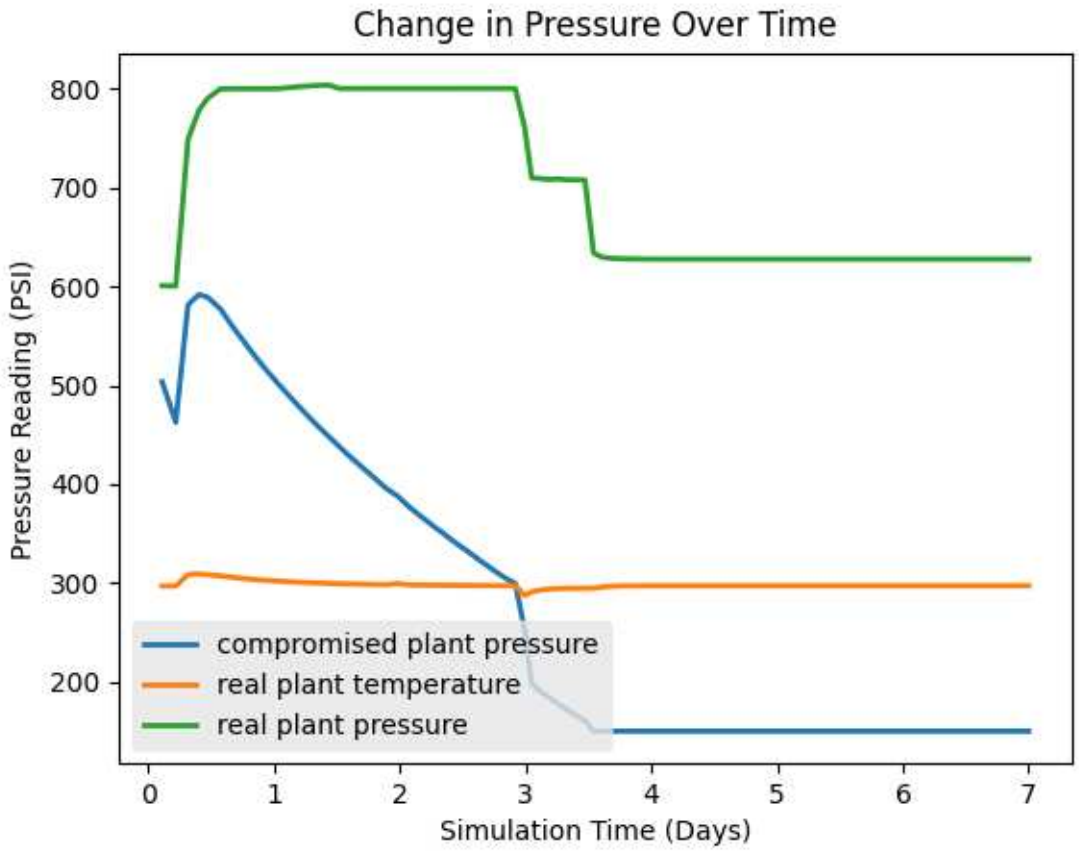
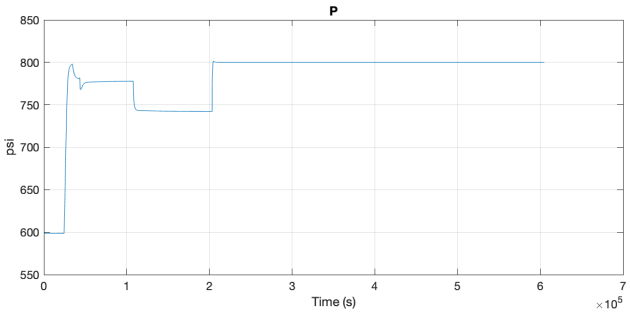
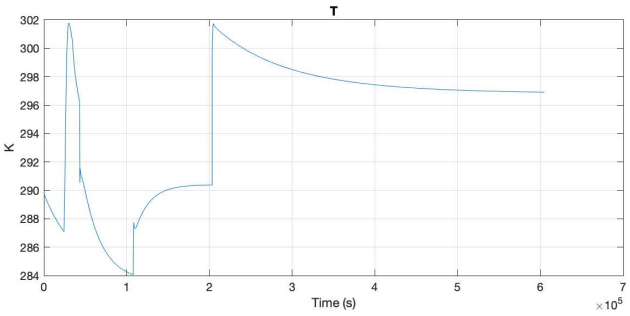


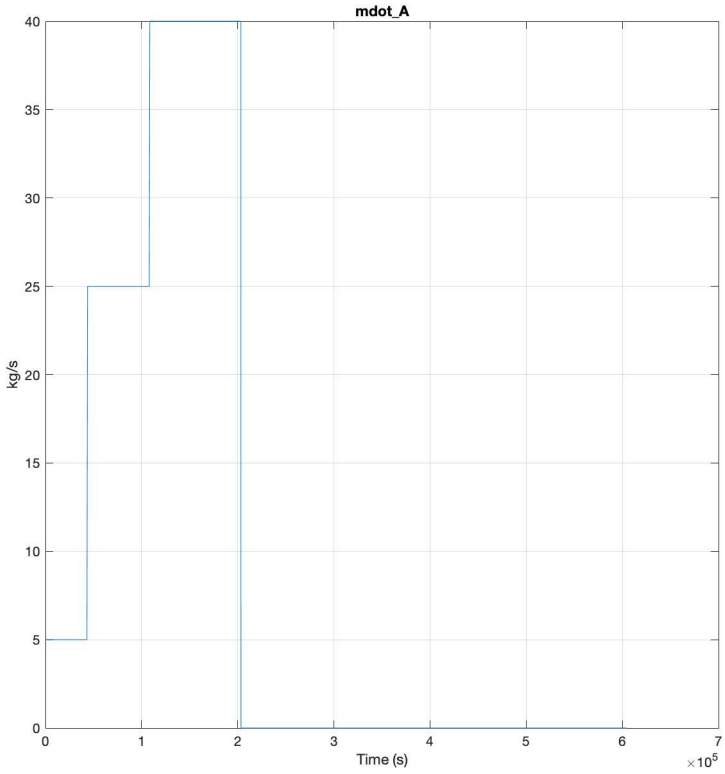
Figure 6.2: Compromised plant pressure, real plant pressure, and plant temperature from experiment 1.



(a) Pressure



(b) Temperature



(c) Gas Load

Figure 6.3: Readings taken from within the physical model during experiment 1.

For this system we combine data from several sources, as well as using some principles of gas dynamics in order to determine accurate measurements. We used a Kinder Morgan system map to discover the general area of where gas pipes and power plants exist in the state of Colorado [21]. We then extrapolated this map to the nearest large cities and determined rough distances for placement of the gas pipe. We then placed moderate sized power plants near medium sized cities in Colorado and a large power plant in Denver. We also sprinkled a handful of distribution loads to add complexity to the system. Once we determined the placement of power plants, compressors and distribution loads we referenced Xcel Energy in order to discover the peak load capacity in Megawatts for power plants around the state [22]. We converted these peak capacities from Megawatts to required gas in kg/s using the gas heating values from experiment 1.

Once we determined the peak mass flow gas loads for each power plant (in kg/s) we were able to design the diameter of the gas pipes to meet this demand at a nominal pressure of 800 psi using Bernulli's Equation (ignoring gravity / height differences). As gas loads in this system and real life are very high the pipes have to be very big in order to distribute as much gas as needed. Typically gas pipes come in nominal sizes that might not exceed 36 inches in diameter, and when engineers need to distribute more gas along a line they will lay several pipes in parallel. Currently our system does not consider this and uses a single large pipe. This may have adverse effects on some of the properties of the gas. For example, there is more friction when using several small pipes than there is in using a single large pipe. In addition, in smaller pipes gas may be flowing at a higher velocity which means that temperature changes can be more dramatic.

6.4.2 Automated Switch-Off

In Experiment 1 we used an external control system to switch off a power plant when its pressure or temperature reached a certain point. For the Colorado gas model we wanted to keep the switch-off control inside of the process model to represent engineers at a power station tripping the station off when the gas reached a certain set pressure or temperature. In the Colorado gas system we add a MATLAB function block that sets the power stations load to zero and switches

it off for a number seconds specified by a parameter at the beginning of the simulation. For this model we used 3600 seconds (one hour) as the shutoff duration for our power plants. This time represents how long after a shutoff it takes the power plant engineers to run through their safety procedures and ramp the system back up to meet the current load. The goal of most of the trials in experiment 2 are to get these power plants to trip their automatic shutoff by dropping the gas pressure in the system.

6.4.3 Dynamic Load Distribution

The Colorado gas model uses a dynamic load distribution algorithm that allows it to simulate a system wide load that different power plants in the system have to unite to meet. In previous experiments every load was specified individually, whereas in this experiment a single load can be specified in order to model a realistic scenario. The advantage of this approach is that when one power plant in the system trips off, the load switches onto other plants in the system. This is effective for showing scenarios where the loss of a power plant can possibly cause a cascading failure as demands get increased very rapidly on other parts of the system.

Each power plant in the system is specified with a load capacity in kg/s. When determining the load of a given power plant you use it's ratio to the total system capacity to express its proportion of the current load. Let the plant capacity be p_c, the total system capacity be s_c and the current load at time t be l_t. Then the load on the a given plant, i, will be $\frac{p_{c,i}}{s_c * l_t}$. Also we must consider a scenario where the total load on the system exceeds the capacity of the power plants available. In this case, each plant will output at its maximum load and the system's demand will not be met. This scenario can be considered one where the gas system is not generating enough electrical power to meet demand.

6.4.4 Control Design

The control system for experiment 2 is substantially more complex than for experiment 1. The compressor stations in experiment 2 are modified so that the power capacity of the station can be modified. When compressing gas, the compressor station inspects a value δ_p which is the

difference in pressure between the desired set point (in the case of this model it is 800 psi) and the current pressure reading. As δ_p increases, more power is required to compress the gas to meet the pressure differential. In order to make the system realistic there has to be a limit on the amount of power that a compressor can use. This limit is specified in megawatts, with the base value being 5 megawatts. When the system requires more power than is available δ_p is modified to the maximum compression available for the current maximum amount of power. This plays into the control system as the operators of the system will want to minimize the amount of power being used by compressor stations whenever possible.

The control algorithm for the Colorado gas model increases the power available to upstream compressors based on the current pressure reading at a downstream station or compressor. The controller iteratively checks the state of the system and updates the power available to each immediate upstream compressor. Power is updated in 5 megawatt increments, if the pressure is less than 750 psi then power is updated +5 megawatts while if power is at 800 psi available power is decreased -5 megawatts to save energy. Upstream compressors are identified by consulting a directed graph that represents that gas flow in the system. Each node in the graph is either a power station or a compressor. Every power station is a leaf node by default, as gas does not flow through power stations to other nodes in the system.

6.4.5 Gas System Scenario

For the purpose of the next two experiments the Colorado gas model is set up in a specific scenario. The scenario chosen represents a time when the gas system is put under a high level of strain. This is important because instances where the system is under high strain are great opportunities for a cyber attack. The current scenario is a 3 day simulation where for the first day the system is running under a moderate amount of load. These situations can happen when renewable energy like wind is carrying a large portion of the electricity. Then suddenly, on the second day we have a loss of this wind power. As a result, the natural gas power plants have to ramp up to a high load to generate enough electricity to meet the demand. After a 12 hour period

the required load drops back down to model either the wind power coming back on or the demand for electricity going down (see Figure 6.4 for a profile of the system load in this scenario). It is important to note that when these power plants ramp up the demand for natural gas throughout the system also dramatically increases. If the compressors in this system are not adjusted to meet demand we can see power plants start to fail as they are not being delivered gas at a high enough pressure (see Figure 6.5 for an example of what happens when the control system is unresponsive) . Thus the goal of measurement attacks in this scenario is to prevent the proper control actions from being taken during this window of high strain, and therefore causing failures of power plants in the system. In the following experiments we say an attack is successful if it causes a power plant to trip off within the window of high strain on the system.

6.5 Experiment 2: Single Point of Failure

6.5.1 Experiment Setup

For this experiment we explored the effect of a single point of failure on the system during the 12 hour window of high strain. When gas loads on the system ramp up pressure starts to dip at all the power plants as they start evacuating their upstream gas lines. When this pressure dips the controller should notice and make more power available to upstream compressors. Therefore, for attacks in this experiment we compromised power plant controllers to falsify their pressure readings. We hypothesized that compromising pressure readings at these power plants would prevent upstream compressors from ramping up and therefore preventing the power plants from getting the gas pressure they need to continue operating.

The compromise used for pressure readings was extremely simple. No matter what the current state of the system is, the compromised power plant will always report 800 psi. For this experiment we ran 5 trials, applying this compromise to each power plant and rerunning the simulation. We can say the attack was successful if a power plant shut offs in the 12 hour window of high strain. The goal of this experiment was to determine if a single compromise could cause a catastrophic effect in the system.

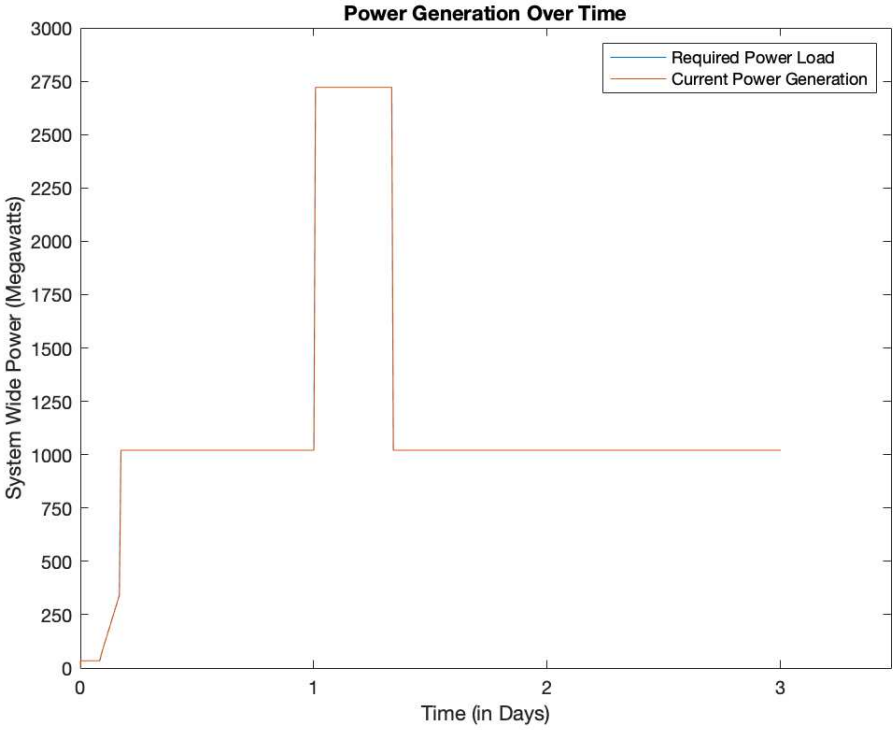


Figure 6.4: Under a period of high stress there is no loss in power if the system corrects properly.

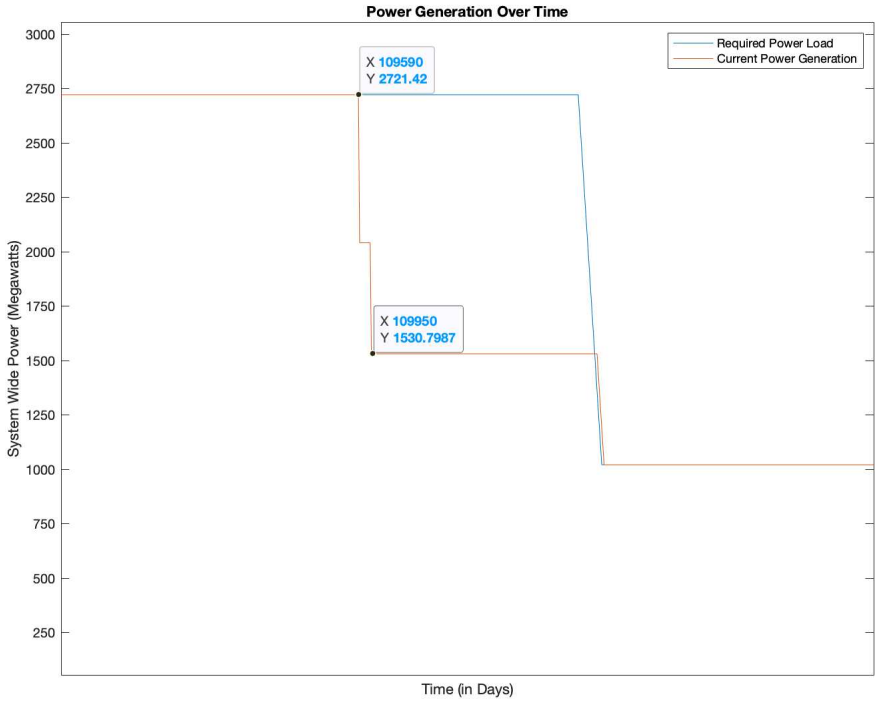


Figure 6.5: Rapid loss of power generation capability can occur without control system intervention.

6.5.2 Results

The single point of failure trials showed interesting results. For 4 out of the 5 trials we saw that even though a measurement attack was taking place the compressor immediately upstream of that attack might not be affected. This is because there can be multiple downstream entities showing low pressure and so a single compromise will not prevent the controller from still updating the pressure at the compressor. For power plants that were isolated the impact of this measurement may have disabled their compressor but since the compressors along the main lines in the simulation were still running there was enough gas being pushed through the system to prevent a failure.

The one trial that caused failure did provide interesting insights. When applying the compromise to the Fort Collins power plant we saw that the Fort Collins compressor did not kick in and this caused the Fort Collins plant to go offline (Figure 6.6 shows the differential between the falsified pressure reading and the actual reading). In addition, the increased demand brought onto the auxiliary line from Fort Morgan also caused the Fort Morgan plant to trip near the end of the 12 hour window (see Figure). An interesting take away from this trial is that Fort Collins compressor is a critical point in the system as it feeds gas to a lot of major power plants downstream of it. If the Fort Collins compressor does not meet the demand of the downstream plants it can cause increased load on the auxiliary lines.

6.6 Experiment 3: Sophisticated Measurement Attack

6.6.1 Experiment Setup

For the second experiment we explored the impact of compromising multiple sensors in the system in order to cause a rapid failure. The objective of this experiment is to prevent a compressor from ramping up when it needs to while encouraging other compressors to ramp up when they may not need to. In a way, this attack is similar to the attack described in experiment 2 but slightly more sophisticated. As we are using falsified readings in an attempt to manipulate the actions of upstream compressors.

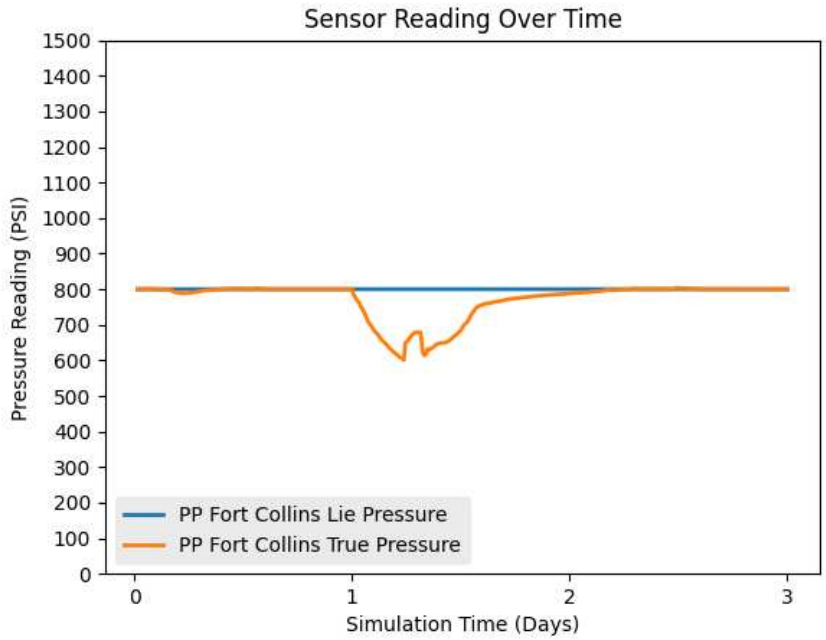


Figure 6.6: Experiment 2: Difference between the actual and falsified readings at the Fort Collins compressor.

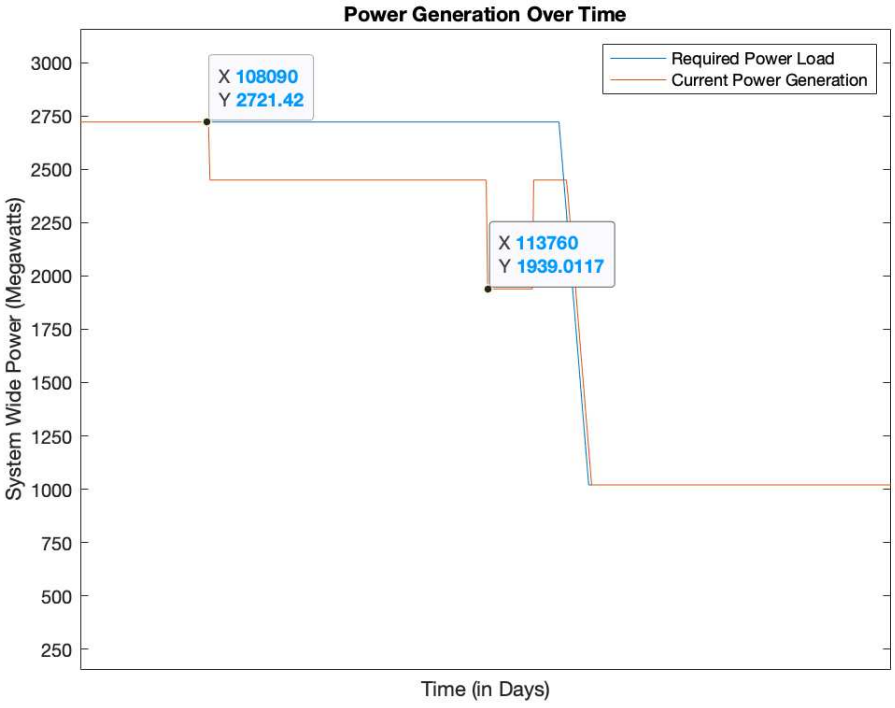


Figure 6.7: Experiment 2: loss in power generation when the Fort Collins and later the Fort Morgan plants go offline.

For this experiment we are compromising three sensors in the system. Based on experiment 2 we know that the Fort Collins compressor is a particularly critical point in the system. So we compromised the immediate downstream neighbors to this compressor, the Fort Collins power plant and the Longmont compressor. We set both the Fort Collins power plant and the Longmont compressor to always read 800 psi regardless of the current actual pressure. In addition, we marked the Denver power plant as reading low so that the Denver compressor would ramp up, pulling gas down through the path from Fort Collins to Denver. The objective here is to pull gas down the line while preventing the Fort Collins compressor from ramping up to meet the new demand. This should hopefully empty the gas lines and cause failures in the system. The resulting disparity between the true readings and the compromised readings can be seen in Figure 6.9.

6.6.2 Results

The second experiment posed interesting results in the realm of measurement attacks. The compromised high readings at the Fort Collins power plant and Longmont compressor prevented the Fort Collins compressor from ramping up during the period of high stress. In addition, the controller offered an increased power to the Denver compressor to compensate for the falsified low pressure reading. This caused the pressure in the Fort Collins to Longmont line to drop (see Figure 6.8) as well as causing an increased demand on the auxiliary line from Fort Morgan. As gas is evacuated out of both the lines coming into Longmont pressure drops rapidly in both the lines. As a result, both the Fort Collins and Fort Morgan plants tripped off in rapid succession of one another. This led to the gas and power generation system not being able to meet its total demand. The total loss of capacity is measured in kg/s of gas load, but translates roughly to a loss of about 800 MW within a period of approximately 5 minutes (see Figure 6.11).

6.7 Discussion

The results of the 3 experiments performed give an insight into the potentially dangerous nature of measurement attacks. The first experiment provided a useful example of how a measurement

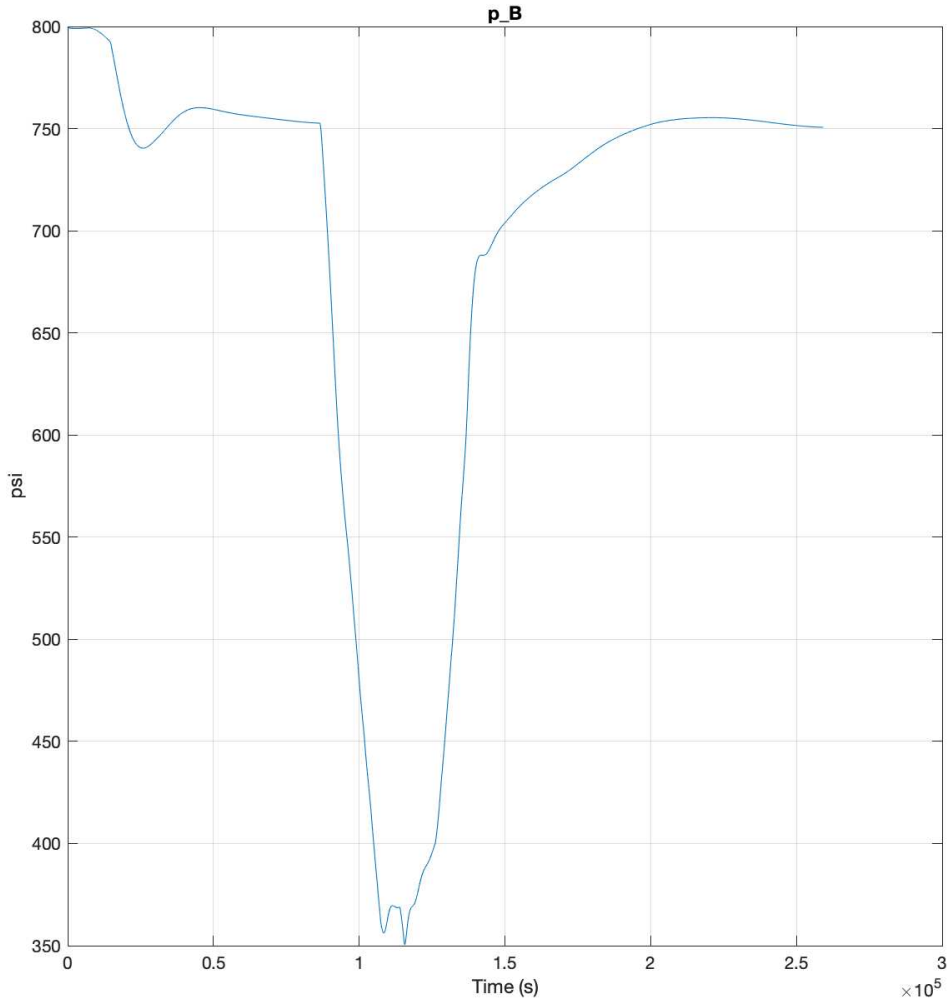


Figure 6.8: Experiment 3: Pressure in the Longmont gas line dips under high stress.

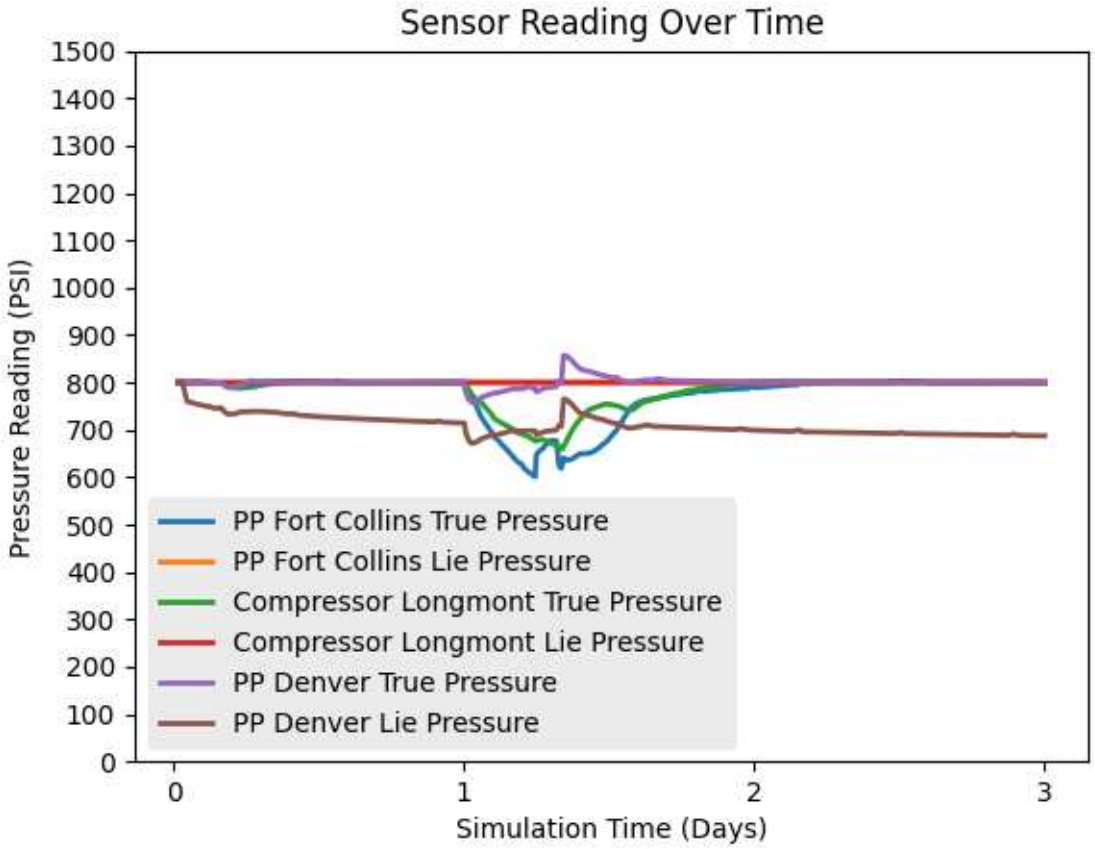


Figure 6.9: Experiment 3: Real and compromised sensor readings.

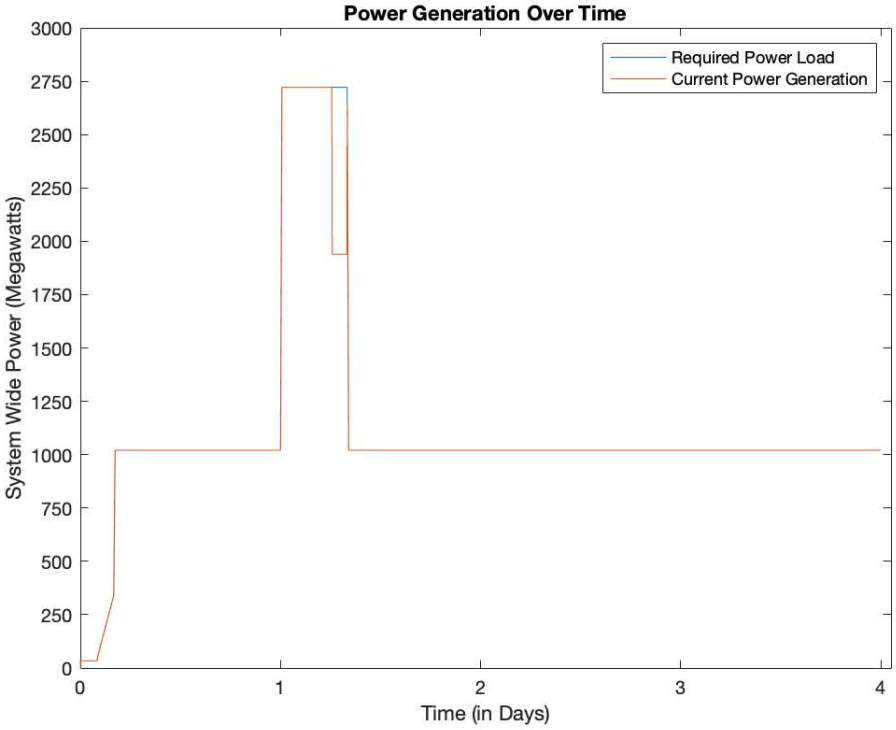


Figure 6.10: Experiment 3: Load profile shows a loss of generation capacity near the end of the window.

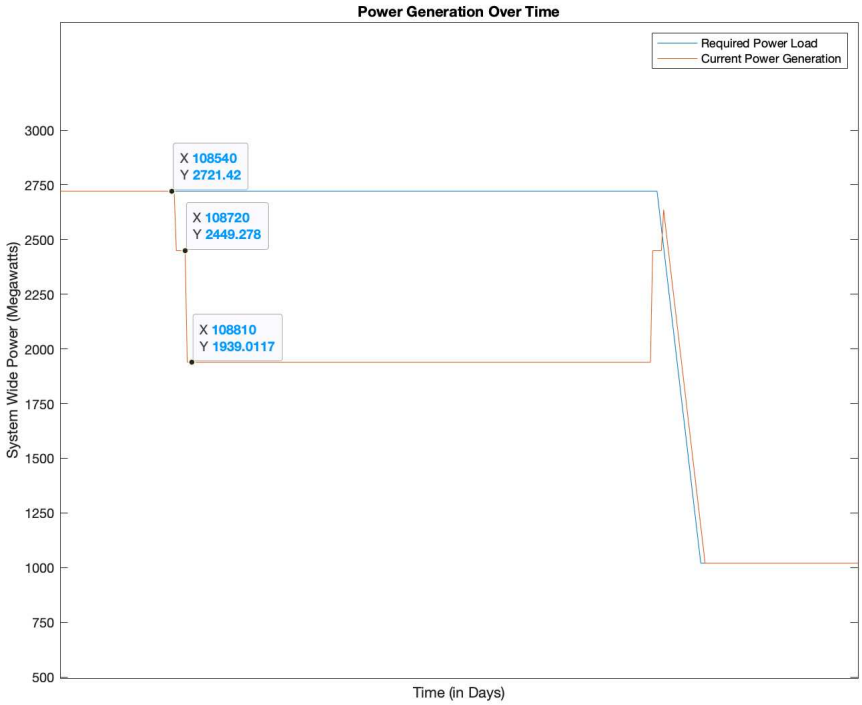


Figure 6.11: Experiment 3: Close up view shows that two plants fell offline in rapid succession.

attack can change control behavior in the system. However, this example was fairly contrived. In the second experiment we explored how a single point of failure can affect a complex system. The single point of failure work seemed to show that although at critical points in the system there could be a catastrophic effect, more isolated areas tended to show little impact on the overall system. One insight that can be derived from this is that defense in these types of systems may not have to be completely blanketing. It may be possible to secure the system from catastrophic failure by identifying critical points and applying efforts to maximize defense at those points.

The third experiment was extremely interesting and slightly frightening in that it demonstrated that an attacker with intimate knowledge of a systems dynamics can use measurement attacks to manipulate the control response to the system. By using measurement attacks the system can be put into a state that causes failures that otherwise would not happen with normal control behaviors. The most interesting insight from this experiment is that as more sensors in the system become compromised the attacks can get increasingly complex. This likely means that they will also be harder to detect. However, for these types of attacks to happen the attacker must understand the way the control system operates, the gas system topology and have knowledge of when this system is in a period of high strain.

Finally, the results of the experiments expose that the nature of gas systems in response to cyber attack may be very different from that of electrical systems. Failures in electrical systems can occur very rapidly. For example, in the famous northeast blackout of 2003 at around 4:06 pm system operators started noticing abnormalities, by 4:10 pm a power surge severed a large system line that caused a cascade of events leading to the blackout [23]. In another instance, the San Diego blackout of 2011, a system suspended primarily by 500 kV line went down in roughly 10 minutes following a maintenance problem at a remote substation [24].

Unlike the fast most moving makeup of electrical systems physical reactions in gas systems can be delayed or slow moving. This results from the fact that it takes large amounts of time to physically move gas through pipes. We also believe that could be why isolated attacks can have less of an effect on the system. In addition, with large gas pipes like the ones used in our system

there is a substantial amount of gas storage retained within a pipe. Even when they are jerks in the system like rapidly increased load, the storage retained within the pipes helps prevent a failure when there is not an immediate reaction to the change in system state. For example, in the scenario where there was no control response the system did not fail until the end of the 12 hour window. This suggests that there is ample time to react when the system is not operating entirely as it is supposed to. Additionally, placement of gas storage tanks along strategic places within gas lines can provide another source of safety for systems trying to avoid failure during periods of high strain. This does not bode well for attackers trying to damage the system, but it also does not mean the system is safe from strategically devised and coordinated attacks.

The main take away from the earlier experiments then is that hackers would have sustain an attack for hours or even days to have an effect on the gas system. For example, in our system which is sustained by one primary source of gas - not unlike the 500 kV line in the San Diego blackout example - a measurement attack had to be sustained for nearly 12 hours before there was any impact. This is encouraging in the sense that it would be very difficult to trick systems operators for this long. Redundant sensors and cross-communication between operators would likely allow them to determine that there was something wrong in the system. However, because cyber attacks are fairly unprecedented on gas pipelines it is also unlikely that the average gas operator would make the assumption that their system is under attack, rather explaining any strange phenomena as faulty sensors or devices. Despite this, if a measurement attack is sustained on a gas system for long enough undetected it can cause a set of rapid cascading failures like we saw in experiment 3.

Chapter 7

Future Work

7.1 Resolving the Timing Problem with a Simulation Clock

As discussed in the design of the model in Chapter 4 timing issues are a complicated problem when dealing with simulations that scale on the order of days rather than minutes. Any delays in the simulated controllers, or any inefficiencies are dramatically magnified. In our work we dealt with timing by trying to make our simulated controllers and operators as efficient as possible. Additionally, we relayed timestamps from the simulation physical model throughout our virtual ecosystem in order to maintain some sense of time. Many of the timing issues arising in our model can be attributed to the way that the simulation uses both discrete and continuous methods of solving differential equations. A more fine grained approach to this timing problem is to model the simulation entirely discreetly, and use an external clock process that advances and updates the physical model as well as the controllers and the HMI at the same time. This would allow for the entire simulation to work on a rigid schedule and prevent any delays in processing from having adverse effects on the simulation. However, our team has determined this problem is out of scope for the current stage of the project and has left this as a future research problem.

7.2 Modeling SCADA Network Behavior

One consideration discussed in Chapter 2 was the ability to model the behavior of the SCADA network. In the current system we have developed network communication is modelled by sending MODBUS packets over UDP between the simulated operator and the virtual controllers. This approach simplifies the communication process for developing the simulation but fails to consider the use of other protocols. In future iterations of simulation work it is important to design methods that SCADA network part of the simulation can be modular and interchangeable. Modeling network traffic in a greater depth would allow for a wider variety of attacks to be modeled, as well as

the ability to explore different ICS protocols. One interesting question that could be answered by this type of network modeling is how attacks on different underlying network protocols could lead to failures in the physical system?

In addition to being able to explore new types of attacks, adding a network modeller to the simulation would allow for fine grained control of network delays. Since control systems for gas pipelines and other ICS systems can span over long geographic distances, there are real delays that can occur within the read/respond feedback loop. Currently our simulation has no way to model these delays. Furthermore, it could be an interesting experiment to explore the cyber security implications of these delays. For example, an attacker may try to race a sensor reading to the operator giving a false reading to the operator without ever compromising the sensor's PLC. Another attack could be to somehow mute a long distance sensor, and masquerade as that sensor to the system operator. Finally, an attacker might be able to induce or increase delays in sensor readings and in the feedback loop such that an operator's actions do not occur quick enough to prevent a system-wide failure. Despite the interesting questions posed with network based attacks on these systems, this is another problem that out-scopes our research and so was not addressed in this work.

7.3 Integrating Hybrid or Real PLCs

One of the largest limitations of the current system is how it simulates the behavior of the PLCs in the SCADA ecosystem. PLCs, often referred to as virtual controllers, in our system are simple modbus servers with limited built in control. Although it is possible to replicate many of the control behaviors of real PLCs, it is a time consuming process that is often more difficult than acquiring a PLC of the specified model. However, one of the main advantages of virtual PLCs is their low cost, quick deployment, and high level of scalability. This advantage comes with the drawback of not having real physical PLCs for penetration testing, or for truly authentic modelling of the control behavior. Oftentimes, stronger approaches to simulation will use both virtual controllers and real controllers [8].

In order to expand this model it would be a wise approach to find a method for integrating physical PLCs or for creating hybrid PLCs using micro-controllers and open source PLC software such as OpenPLC [25]. To achieve this, one approach might be to convert signals from digital to analog which are then read by hybrid PLCs which respond in turn with control behavior. Furthermore, these hybrid PLCs would run MODBUS servers that could respond to real MODBUS commands from simulated operators. Simulink provides a toolbox of software that allows the Simulink software to communicate with controllers through serial or network based communication. However, an interesting research challenge of hybrid and real PLCs in this system is finding ways to resolve simulation timing issues discussed in our work. Mainly this will be a challenge as proprietary PLCs will be much less flexible to software changes that may be required to integrate them into the simulation environment.

7.4 Analyzing and Preventing Measurement Attacks with Machine Learning

In our work we have demonstrated the theory behind measurement attacks, as well as possible dangers they may pose. However, we have not delved into possible methods for detecting or preventing these attacks. As machine learning has become an increasingly popular approach for problems involving large amounts of data, or for deriving insights based on data, it would be interesting to explore methods for detecting faulty or lying sensors in a SCADA system. There are several thoughtful approaches to this problem. One introductory thought may be to use statistical methods for outlier identification. These methods may allow control systems to discover sensor readings as outliers within the time series of data they are contained in. Another approach may be to apply deep learning or classification algorithms towards identifying anomalies within a time series of data. However, even if one could identify outliers using this technique what would be the approach for dealing with these outliers? Would it be wise to block out data from sensors presumed to be lying? Is it still possible to derive useful information from a sensor even when it is lying?

Maybe a more interesting approach to observing the data in these SCADA systems is to identify constraints between and within the data coming from sensors. For example, a constraint within the data may be that a pressure sensor in a normal scenario should never read outside of a certain range. However, a more interesting case may be to identify constraints between sensor readings across a physical system. In the case of one PLC it can be something as simple as temperature and pressure vary together because of underlying properties of gas. In addition, one could explore constraints between PLCs in a model. For instance, several power plants downstream of a large compressor station will likely have pressure readings that vary together in response to changes in the compressors outputs. Being able to identify these constraints could allow an intelligent SCADA system to flag violations in data constraints and report these to system operators.

A primary issue with identifying constraints in these SCADA systems is that as the systems scale, they get increasingly complex and interconnected. The process of manually identifying constraints between different components would require time, clever engineering and would be prone to errors. Researchers at Colorado State University have developed methods for automatically identifying constraints in data [26]. This work has developed a methodology for identifying and verifying constraints in large data sets using a feedback loop with subject matter experts. An expansion of this work could deal with how to apply these automated constraint discovery approaches to time series data. Then applying this approach of constraint discovery to SCADA systems.

7.5 Adding Encryption and Access Control to SCADA Protocols

The legacy nature of SCADA systems means that there is usually no authentication or access control baked into the common protocols used by these systems. Many issues with measurement attacks or command injection could be resolving by adding authentication and access control as basic security fundamentals. However, the process of deploying and developing these systems can be prohibitive, especially if it means there will be a loss of service. Thus, a simulated environ-

ment is an especially appealing scenario for developing and reasoning about the deployment of authentication, encryption and access control in SCADA systems.

Using encryption in SCADA systems is a challenging problem. To begin with, many PLCs and sensors are part of legacy systems and there may be no way of retroactively adding encryption to them. In addition, the gradual integration of security methods may only succeed in providing peace of mind, while the system remains only as strong as its weakest link. Another challenging question is the distribution of keys in large scale SCADA systems consisting of thousands of sensors and actuators. In addition to key distribution, another research question follows of how to secure data between sensors and PLCs and what encryption schemes are best for securing data traffic between PLCs and system monitors.

In addition to deploying encryption in SCADA environments, having access control schemes that prevent erroneous or dangerous command injection should be a primary focus of future research on these systems. In order to have access control, though, a PLC needs to be able to verify and authenticate that the system giving it commands is not a hacker in disguise. Additionally, there are questions that revolve around the real value of access control in these systems. What mechanisms are most effective, and if a system is only managed by a single master administrator is access control a redundant security step? Ultimately, these are questions that could be answered through experimentation and testing in a simulated environment. After rigorous testing in simulation, the application of these security methods may become more clear in real environments.

7.6 Automatic Discovery of Critical Points for Defense

As uncovered in Chapter 5.5 not all points of control are as equally critical to the operation of a SCADA system. Compromise of some controllers can be much more devastating than the compromise of others. In the case of 5.5 the compromise of the Fort Collins power plant controller was able to prevent the Fort Collins compressor from feeding gas to the numerous downstream power plants. This resulted in a partial failure of the system. The research question we can derive from this is: how can we determine which points in the system are critical for operation? Is there an

automated way to do this? One approach may be to model real world systems using simulation like we have in the past, and then iteratively apply measurement or command injection compromises to each controller. This would be similar to what we did in 5.5. Then upon inspecting the results you could see which compromises caused failures within the system and continue exploration from that point.

As well as automatically discovering critical points of operation, there is a body of research that could explore ways to incrementally employ defenses to these critical points. If you have limited resources to secure a system the high level insight is that you want to prioritize how you defend points in the system. By using automatic discovery of critical points you can determine which areas have the most potential for harm. Then you can develop a strategy for deploying defenses in these points. In this way you could step toward system security by hardening the most vulnerable areas first and work your way out towards defending less vulnerable spots later.

Chapter 8

Conclusion

In conclusion, the areas of SCADA simulation, cyber security, and measurement attacks are all interesting paths for research. We believe that improving the realism of SCADA simulations, and investigating the cost of a compromised SCADA system are critical research areas for protecting the current infrastructure of power and gas systems. The hope through this thesis project is to begin growing a foundation for exploring the research questions outlined in this paper, and primarily to explore one question of interest. What are the impacts of measurement attacks in SCADA systems and how can we defend against them? From our work we discovered that an intelligent attacker can cause substantial harm to a system by employing these types of attacks. Additionally, we were able to uncover some interesting observations on the nature of how gas systems respond to cyber attacks. Experiments on the nature of critical points in the system opens a new area of research related to defense of SCADA systems. On top of this, the use of simulated environments is what helped us get to a point where we can start reasoning and asking high level questions about SCADA system security without ever touching a real system. This is an encouraging demonstration then for the use of simulation in SCADA security, as it allows us to experiment with the hypotheticals that we would not be able to explore on real systems.

Bibliography

[1] Ralph Langner. Stuxnet: Dissecting a Cyberwarfare Weapon. *IEEE Security Privacy*, 9(3):49–51, May 2011. Conference Name: IEEE Security Privacy.

[2] A. Ashok, Pengyuan Wang, M. Brown, and M. Govindarasu. Experimental evaluation of cyber attacks on Automatic Generation Control using a CPS Security Testbed. In *2015 IEEE Power Energy Society General Meeting*, pages 1–5, July 2015.

[3] David P Duggan. Penetration Testing of Industrial Control Systems. *Sandia National Laboratories*, page 7, 2005.

[4] Thomas Morris, Anurag Srivastava, Bradley Reaves, Wei Gao, Kalyan Pavurapu, and Ram Reddi. A control system testbed to validate critical infrastructure protection concepts. *International Journal of Critical Infrastructure Protection*, 4(2):88–103, August 2011.

[5] Thomas H Morris, Zach Thornton, and Ian Turnipseed. Industrial Control System Simulation and Data Logging for Intrusion Detection System Research. *7th Annual Southeastern Cyber Security Summit*, page 6, 2012.

[6] Simon Duque Antón, Michael Gundall, Daniel Fraunholz, and Hans Dieter Schotten. Implementing SCADA Scenarios and Introducing Attacks to Obtain Training Data for Intrusion Detection Methods. *arXiv:1905.12443 [cs]*, May 2019. arXiv: 1905.12443.

[7] Bertrand and Olivier Taburiux Masset. Simulating Industrial Control Systems Using Mininet. 2018.

[8] Qais Qassim, Mohd. Ezanee Rusli, Salman Yussof, Roslan Ismail, Fairuz Abdullah, Norhamadi Ja'afar, Hafizah Che Hasan, and Maslina Daud. A Survey of SCADA Testbed Implementation Approaches. *Indian Journal of Science and Technology*, 10(26):1–8, June 2017.

[9] Carlos M. Correa-Posada, Pedro Sánchez-Martín, and Sara Lumbreras. Security-constrained model for integrated power and natural-gas system. *Journal of Modern Power Systems and Clean Energy*, 5(3):326–336, May 2017. Conference Name: Journal of Modern Power Systems and Clean Energy.

[10] Tao Li, Mircea Eremia, and Mohammad Shahidehpour. Interdependency of Natural Gas Network and Power System Security. *IEEE Transactions on Power Systems*, 23(4):1817–1824, November 2008. Conference Name: IEEE Transactions on Power Systems.

[11] Burcin Cakir Erdener, Kwabena Pambour A., Ricardo Bolado Lavin, and Berna Dengiz. An integrated simulation model for analysing electricity and gas systems | Elsevier Enhanced Reader, April 2014. Library Catalog: reader.elsevier.com.

[12] Kostas Mathioudakis, Nick Frangiadakis, Andreas Merentitis, and Vangelis Gazis. Towards generic SCADA simulators: A survey of existing multi-purpose co-simulation platforms, best practices and use-cases. page 7, 2013.

[13] Thiago Alves, Rishabh Das, and Thomas Morris. Virtualization of Industrial Control System Testbeds for Cybersecurity. In *Proceedings of the 2nd Annual Industrial Control System Security Workshop on - ICSS '16*, pages 10–14, Los Angeles, CA, USA, 2016. ACM Press.

[14] Thiago Alves, Rishabh Das, Aaron Werth, and Thomas Morris. Virtualization of SCADA testbeds for cybersecurity research: A modular approach. *Computers & Security*, 77:531–546, August 2018.

[15] Industrial Control Systems Cyber Emergency Response Team. Recommended Practice: Improving Industrial Control System Cybersecurity with Defense-in-Depth Strategies. Technical report, Department of Homeland Security, September 2016.

[16] I. N. Fovino, M. Masera, L. Guidi, and G. Carpi. An experimental platform for assessing SCADA vulnerabilities and countermeasures in power plants. In *3rd International Conference on Human System Interaction*, pages 679–686, May 2010.

[17] US Department of Homeland Security. Ransomware Impacting Pipeline Operations | CISA, February 2020.

[18] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 4(3):20.

[19] US Energy Information Administration. What is the efficiency of different types of power plants?, August 2019.

[20] Engineering ToolBox. Fuel gases heating values., 2005.

[21] N. Schubert. KMI System Map, February 2014.

[22] Colorado Generating Stations – Public Service Company of Colorado, 2017.

[23] Interim Report on August 14, 2003 Black. Technical report, New York Independent System Operator, January 2004.

[24] Jeff McDonald and Morgan Lee. Blackout sparks multiple investigations. *The San Diego-Union Tribune*, September 2011.

[25] Thiago Rodrigues Alves, Mario Buratto, Flavio Mauricio de Souza, and Thelma Virginia Rodrigues. OpenPLC: An open source alternative to automation. In *IEEE Global Humanitarian Technology Conference (GHTC 2014)*, pages 585–589, October 2014.

[26] Hajar Homayouni, Sudipto Ghosh, and Indrakshi Ray. ADQuaTe: An Automated Data Quality Test Approach for Constraint Discovery and Fault Detection. In *2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI)*, pages 61–68, Los Angeles, CA, USA, July 2019. IEEE.