Networked Control System Emulation: Analysis of Controller Design

John Rivera
Computer Science
Youngstown State University
jrivera01@student.ysu.edu

Mentors:
Blaine Nelson
Saurabh Amin
Suzanna Schmeelk

July 31, 2010

TRUST Research Experiences for Undergraduates (TRUST-REU)
in Cyber Security and Trustworthy Systems 2010

Department of Electrical Engineering and Computer Sciences
College of Engineering
University of California, Berkeley

1. Abstract

Control systems are an integral aspect of modern industrial architecture. These systems often regulate critical infrastructures, such as power plants, traffic control systems, security systems, flight control systems, and a wide range of other potentially mission critical systems. Using the DETER testbed, a networked control system (NCS) was emulated over the Abilene network topology. The behavior was simulated using an ordinary differential equation (ODE), and experiments were run to determine the influence of the control signal on plant stability. Results show that the aggressiveness of the control signal is paramount in determining the relative stability of the control system.

2. Introduction

Control systems are an integral aspect of modern industrial architecture. These systems often regulate critical infrastructures, such as power plants, traffic control systems, security systems, flight control systems, and a wide range of other potentially mission critical systems. The functionality and integrity of these systems is paramount, as a miscalculation by one of the controllers in the system could potentially shut down a power grid, disable security systems, or even cause a meltdown at a nuclear reactor.

The goal of this project is to investigate the effects that attacks have on control systems. These systems communicate over an emulated public communication network. The goal is broken down into three objectives. The first objective was to setup the infrastructure. The second objective was to deploy the control system software onto the infrastructure. The third objective was to attack and defend the control system. This paper focuses on the deployment of control system software onto the DETER testbed.

The focus of the controls software area of research was to simulate the behavior of an emulated control system over a physical network. The Abilene network was emulated on the DETER testbed. Once completed, control system software was installed on top of the Abilene network topology. With the control system emulated, the behavior could be simulated and the attacks performed. The behavior was modeled using an ordinary differential equation (ODE). This paper focuses on controller design with respect to the gain and aggressiveness of the control signal.

To understand the significance of the problem, the concepts involved must first be defined. A control system is defined as a device or set of devices that regulates the behavior of other devices or systems. These systems behave with a plant/controller relationship where the controller regulates the plant, and the plant's sensors relay state information back to the controller. A networked control system (NCS) expands upon the idea of a control system by allowing a network to be the infrastructure on which the plant and controller communicate. A good example of a control system is a nuclear reactor. A nuclear reactor acts as a plant by sending temperature readings to the controller over a network. The controller determines the expected state of the reactor, and it sends control signals back to the plant that regulate the temperature and keep the plant in a stable state. The entire system is vulnerable if the controller is vulnerable, and oftentimes the controller is vulnerable.

The controller of an NCS is a computer [1]. NCS's often use the internet or other public communication medium to route messages, which means the controller is vulnerable to any host that has access to that medium. For example, a host could use a denial of service (DOS) attack to flood the controller with an unserviceable amount of traffic. The inordinate amount of traffic causes latency to increase, which interferes with the message relay between the plant and controller and may lead to a critical plant failure.

The emulated NCS uses the DETER testbed as a physical infrastructure. The DETER

testbed is a dedicated network specifically designed for cybersecurity research [3]. Based on emulab, the DETER testbed consists of clusters of computers located at the University of Southern California Information Sciences Institute and the University of California, Berkeley [4] [5]. The cluster provides support for several hundred experimental nodes [4]. The experimental nodes of the DETER testbed are cut off from other public networks to prevent the leakage of malicious code to areas outside the testbed.

While the DETER testbed is the underlying hardware used for the project, a realistic topology was emulated on top of the testbed to create a platform that emulates a real life network. The Abilene Network is a high performance backbone network that was developed to connect the Internet2 community, and it was the topology chosen for this project [6].

This paper delves into control system message passing, and it focuses specifically on how the aggressiveness of the control signal impacts plant stability.

3. Procedure

The first step of the project was to set up the network topology. This task involved the creation of the Abilene network on the DETER testbed, and the specifics are beyond the scope of this paper.

The next step of the project was to setup the control system software on the network. This task involved setting up and deploying the control system software and modeling the behavior. The simulation of plant/controller behavior was carried out in the following three steps: first a basic topology, such as the star topology as shown in figure 1, was set up on the DETER testbed, next the control system was emulated by installing client/server software on different nodes of the topology, and finally the behavior was modeled and simulated.

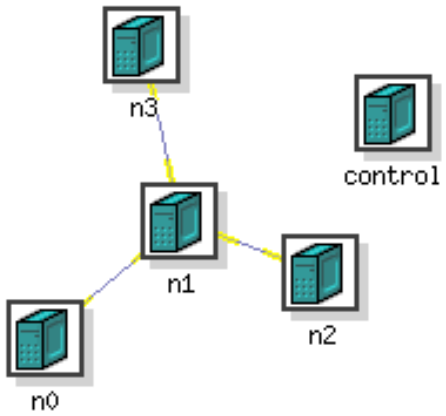

Figure 1. Star topology with control node

First, the DETER testbed was used to set up a basic topology. The development of the Abilene network was performed simultaneously with the development of the control software, so the Abilene network was unavailable for testing at the start of the project. The star topology was used for the software testing and deployment because of its simplicity and its ability to emulate a basic client/server relationship. A basic client/server approach is all that is required to achieve the emulation of an NCS. With separate nodes acting as clients and servers, the emulation of an NCS was complete.

With a test network setup, control software deployment could begin. The plant/controller software was provided during the research, however this software was found to be inadequate for needs of the project. The software given to us was used in previous control system experiments using Network Simulator (NS) [7]. The software was limited to only one node, and this was inadequate for the goals of the project. To solve the problem, much of the script was converted to Python. The Python script allowed for even more control than the original code. The Python script used sockets to connect the nodes, so the node limitation was no longer a factor.

Once the control system software was up

and running, the behavior of the plants and controllers could be modeled. The software provided modeled the behavior using an ordinary differential equation. The same ODE, shown in figure 2, was used to simulate the behavior of the control system. The reason why the behavior was simulated rather than emulated is due to resources. Plants could have been setup using sensor nodes. A controller could then have been used to regulate the readings coming from those sensor nodes. Instead, the behavior was simulated using the ODE.

$$x' = ax + u, with\ x(t_0) = x_0$$

$$x_1 = (x_o + \frac{u}{a})e^{(a)(\Delta t)} - \frac{u}{a}$$

Figure 2. Ordinary differential equation used for modeling behavior

Understanding the variables in the ODE is crucial to understanding the behavior of the control system. The constant *a* is a spring constant. The spring constant is set before running an experiment, and it determines the stability of each plant. The higher the magnitude of *a*, the more unstable the plant becomes. T is the time variable. T stores the time when the state sampling last occurred. Sampling occurs when the plant or controller uses the ODE to determine a state. Δt is the change in time from the previous sampling to the current sampling. *X* is the state of the plant. *X* is analogous to the temperature of the water in a water heater. Once the temperature gets past a certain threshold, the heating element is shut off by the controller, and if the heating element is not shut off for any reason, then the water temperature may rise to a dangerous level causing an explosion. The constant *u* is the focus of this paper, and it's the plant input.

The plant input, *u*, is determined by the controller, and the value of *u* is sent back to the plant as input. This input is the adjustment that the controller is making to keep the plant in a stable state. The controller determines *u* by using the ODE to calculate the predicted future state of the plant, i.e. X_1. The controller uses the plant's state, X_0, to calculate X_1. The future state, X_1, is multiplied by a gain to determine *u*. The gain determines the aggressiveness of the controller's regulation. The sign of the plant input is dependent upon the sign of the predicted future state. If the predicted future state is negative, then the plant input is positive. If the predicted future state is positive, then the plant input is negative.

The final step of the project was to attack and defend the control system, and the specifics are beyond the scope of this paper.

4. Experiments

Once the control system software was up and running and the behavior modeled, experiments could be run. Several experiments were run to investigate the *u* value that is passed from the controller to the plant. The *u* value consists of three elements. The first element is X_1. X_1 is the predicted value of the plants future state as determined by the controller's calculations. A gain is multiplied to this X_1 value, and then the value is multiplied by negative 1. The sign of the *u* value must always be opposite the sign of the future state to reverse the direction of the plants state. The controller is always trying to get the plants state to converge to zero, which means the plant is stable. The gain determines the aggressiveness of the controller's regulation. The higher the magnitude of the gain, the more aggressive the control system regulation becomes.

A hypothesis was that the controller's gain could be optimized to better regulate unstable plants. Several experiments were run that tested controller aggressiveness by altering the controller's gain and varying the stability of the plant. The stability of the plants were altered by increasing or decreasing the *a* value.

Results showed that the gain plays a significant role in plant stabilization. The state of a highly unstable plant converges to zero when the controller uses a high gain. Convergence to

zero shows state stabilization. The same unstable plant shows different behavior with a low gain. The low gain of the controller does not adjust the state of the plant enough to stabilize it. The result of an unstable plant with a low gain is plant failure. The state of a stable plant converges to zero when the controller uses a small gain. Large adjustments are not needed to keep the plant stable. However, a stable plant with an overly aggressive controller fails because the state divergently oscillates out of control.

5. Conclusion

The results of the experiments show that the gain of the control signal is significant to the proper regulation of the control system. A suitable gain can correct almost any amount of instability within a plant. The opposite is true, as well. An otherwise stable system can be made unstable by an overly aggressive control signal. A higher magnitude gain is suitable for unstable systems. A lower magnitude gain is suitable for plants that are stable, since small adjustments keep the plant stable.

This work can be expanded in several ways. First, realistic background traffic can be modeled onto the network. Another extension would be to model a specific kind of plant or control system, rather than the action of control systems in general. A third extension could be to emulate the entire project. This is achieved by using a sensor as a plant. The sensor node sends state information over a network to a controller, and the controller regulates the state of the sensor.

6. Acknowledgements

I would like to thank the University of California, Berkeley and TRUST for allowing me to conduct this research. I specifically mention Dr. Kristen Gates, Larry Rohrbough, Ted Faber, and Jelena Mirkovic for their assistance, as well as the NSF for funding the program. I would like to thank my advisor and faculty mentor from Youngstown State University, Dr. Graciela Perera, for her hard work and dedication. Also, I would like to thank my mentors: Dr. Suzanna Schmeelk, Blaine Nelson, and Saurabh Amin for all their help and support.

7. References

[1] A. A. Cárdenas, S. Amin, and S. Sastry, "Research Challenges for the Security of Control Systems," Technical Report.

[2] A. T. Al-Hammouri, M. S. Branicky, and V. Liberatore, "Co-simulation Tools for Networked Control Systems," Technical report, 2008.

[3] T. Benzel, B. Braden, T. Faber, J. Mirkovic, S. Schwab, K, Sollins, J. Wroclawski, "Current Developments in DETER Cybersecurity Testbed Technology," Technical report, USC Information Sciences Institute, Sparta, Inc., and MIT CSAIL, 2004.

[4] T. Benzel, R. Braden, D. Kim, C. Neuman, A. Joseph, K. Sklower, R. Ostrenga, and S. Schwab, "Experience With DETER: A Testbed For Security Research," Technical report, Information Sciences Institute University of Southern California and University of California, Berkeley, 2006.

[5] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, A. Joglekar, "An Integrated Experimental Environment for Distributed Systems and Networks," Technical report.

[6] Internet2. (2009, April). Internet2 Combined Infrastructure Topology [Online]. Available: http://www.internet2.edu/pubs/200904-Internet2CombinedInfrastructure Topology.pdf

[7] V. Liberatore, “Network Control Systems,” Technical report, Case Western Reserve University, 2002.