DDoS Attacks on Plant/Controller Networks

Kyle Marlin
Computer Information Systems
Youngstown State University
kpmarlin@student.ysu.edu

Mentors:
Suzanna Schmeelk
Blaine Nelson
Saurabh Amin

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

ABSTRACT

In this paper, I attempt to create an inside attack on some critical infrastructure components by emulating a scaled version of the backbone structure of the Abilene network, and the traffic associated with it using the DETERlab Testbed. I will be utilizing a power plant/controller infrastructure that is capable of creating traffic and still being a scalable proportion of data. For an experiment of this size, teams of scientists were formed. The infrastructure development group designed the network topology based on the Abilene network, as well as the routing structure and background traffic for the network. The software development group set up the software to be placed on the nodes, as well as monitoring software to track the progress of the experiment. The final group, the detection analysis and attack development group, developed the attacks and set up ways of monitoring the progress of the attacks. The attacks are based on a Distributed Denial of Service attack (DDoS) from the inside of the topology. Once all of these groups were successful with this phase of the experiment, attacks overloaded the communication between plant and controller using a DDoS attack [1] [2] [3]. In the end, I wanted to develop an algorithm that will improve these systems so that they are less susceptible to attack.

INTRODUCTION

Control systems are a critical part of the nation's infrastructure. They also have a major effect on society. Control systems tell critical systems, such as nuclear reactors, water plants, sewage plants, electrical grids, and more, how to react in normal day-to-day processes, as well as in an emergency situation. In most situations, the control systems are located apart from the critical piece of infrastructure that they control. There is occasionally an on onsite control system as a backup or for use in the event of an emergency. Because these control systems are almost always offsite, they need a way, a medium, to communicate. The chosen method is the internet, which was originally constructed in a manner that was inherently unsecure.

It is very possible for a hacker to gain control of, or to disrupt the communications between the plants and the control systems. Some methods are outlined later in this document. But it is imperative that everyone realize the impact that this could have. If a hacker were to take control of a nuclear reactor, for example, that hacker could create unsafe conditions, and if performed in extremes, destroy the reactor. It may also be possible to create electrical surges and outages on the nation's power grids if a hacker were to take control of it, and a hacker could merge water and sewage if they are located in the same plant, as well as other catastrophic damage if they were located separately and hacked. The possibilities are endless, indeed.

The task at hand was to recreate a plant and controller scenario and then to attack it using multiple variations of attacks. In the end, the goal was to at least show the impacts that these attacks have, but I also wanted to recommend ways of creating a more secure plant/controller setup, as well as if I would recommend any changes to current plant/controller communication

methods. It seemed like there were three well-defined goals, so the research was split up into three subgroups, and each group of researchers set out to accomplish one of the following goals:

- **Goal 1:** Show the dynamics of a major attack on the controllers by recreating a model of the Abilene Network, which is "an Internet2 high-performance backbone network that enables the development of advanced Internet applications and the deployment of leading-edge network services by Internet2 [4] universities and research laboratories across the country." [5]
- **Goal 2:** Get traffic monitoring software up on all the nodes and routers, as well as any additional software. Create the plant and controller software.
- **Goal 3:** Create a scenario where controllers are attacked, the communication between the plant and the controller is flooded, and create a scenario where the communication between the plant and the controller is completely severed.

For the group's software and hardware needs, the project utilized the DETERlab Testbed (cyber-**DE**fense **T**echnology **E**xperimental **R**esearch **lab**oratory Testbed), which is a cluster of about three hundred nodes, all designed for conducting these types of scalable and repeatable experiments. [6] [7]

EXPERIMENTATION

While the infrastructure group was creating the Abilene Topology, setting up static traffic routes, and creating background web traffic based off of current Abilene models, and while the software development group was writing the software required for the plants and controllers, as well as making sure that all the nodes could run this software without error, the attack development group began the process of determining what types of attacks were feasible given their constraints. The constraints were that I had to create attacks that utilized tools that were at my disposal, did not rely too heavily on writing code, and were safe to operate within the testbed environment. This phase of my research had specific time constraints, so it was very important that the project had simple, yet powerful attacks that could be used to achieve a successful network disruption between the plant and controller.

I decided on a combination of DDoS attacks: flood, cut-link, and ramp-up. Each is explained briefly below.

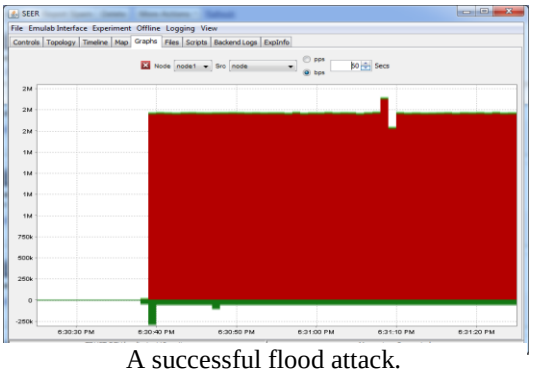
- Flood: Flood is an attack where the communication between controller and plant is overloaded with nonsense traffic using a constant bit rate that is usually higher than the network router or direct link can handle. For example, if a router can handle 100 Mb/s, the attacker would send 300 Mb/s, which would cause queuing on the router, until the information buffer would start dropping packets, thus dropping the communication between plant and controller, as well as some of the nonsense attack traffic.

- Cut-link: Cut-link is an attack where the communication between plant and controller is completely severed, generally by way of killing the network interface that the plant or controller is communicating on or by killing a node between the plant and controller. This will immediately guarantee that no communication packets make it to the plant, and that the plant will become rapidly more unstable.
- Ramp-up: Ramp-up is similar to a flood attack, in that the communication between controller and plant is slowly interrupted with nonsense traffic. In this method, I am able to set a value to increase packets per second. When the experiment starts, I can utilize this to constantly increase the bit rate flowing to the router incrementally. This would at first delay the arrival of the communication between plant and controller, and in the end, it would have similar results to the flood, as it would overwhelm the router and router buffers, and then packets would be lost.

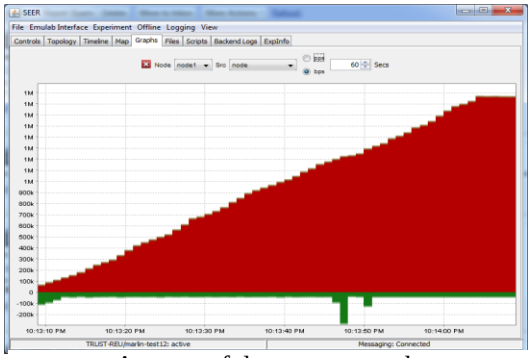
Two of the above mentioned attacks, flood and cut-link, are coded within the experiment's startup script, which is coded in TCL. These attacks are hard-coded to start at a specific time within the experiment. Ramp-up on the other hand, is coded in The Security Experimentation EnviRonment (SEER) and is started at the experimenter's discretion. SEER "is a set of tools and agents for helping an experimenter setup, script and perform experiments in the DETER environment. It includes agents for traffic generation, attack generation, traffic collection and analysis." [8]

RESULTS

After performing many experiments, it turns out that the attacks were consistent in causing a plant failure every time. On the right, there are screen captures of SEER, proving that the attack rates completely surpassed the scale of the experiment. The attack and ramp-up attacks both showed that out of a 100Mb/s router traffic limit, the network was overloaded with 300Mb/s. The plant failed and eventually indicated a status of destroyed, simply based on the fact that it was not successfully communicating with the controller. While the results were consistent across the board, it is important to note that these attacks were not conducted on a physically existing plant, and that they were on software that was created without the specific intention of using in an actual plant/controller scenario, using ordinary differential equations to monitor plant stability. A plant or piece of critical infrastructure would have many failsafes.



A successful flood attack.



A successful ramp-up attack.

But if not protected securely from attack within a secure network, the results could prove deadly.

CONCLUSION

In conclusion, attack experiments conducted on the scaled version of the Abilene Topology reflected my expected results. All of these attacks resulted in a successful destruction of the plant. Without the control data from the controller, the plant will become unstable and fail every time. In the plant/controller setup, when I conducted a DDoS attack on the plant, controller, or the network connecting them, I managed to destroy the plant every time. The project as a whole was able to create plant and controller software, as well as apply that to the Abiline Topology that was created. In addition, background traffic was successfully generated across the network by analyzing several weeks' worth of archive data dumps from existing Abilene network data.

FUTURE WORK AND RECOMMENDATIONS

For the next phase of my work, it is imperative that the plant and controller software is perfected. It lacks precision in its current state. For right now, it only returns a status of working or failed, and while it is running an ordinary differential equation, only those two values are able to determined. Also, I would like to be able to have varying attacks, so attacks can vary in nature. Some proposed attacks would include worms, viruses, trojan horses, and the utilization of other types of malware to infect the controllers in some manner that would make systems malfunction, or would force it to send bad data to the plant. Using this same idea, it can be assumed that one could also could corrupt the plants into feeding the controllers bad data as well, and have it return a state that it was not in. And finally, in the final phase, I would have liked to have developed some sort of algorithm to prevent an attack similar to the ones described above in the first place.

It would be wise to use security software, firewalls, and malware detection systems. This would greatly improve the security of plants and controllers. Have redundant controllers, as well as an onsite controller that is not connected to a network, to all control one plant. This way, there are multiple failsafes, so if a hacker was to compromise the entire network, one controller can still ensure plant integrity. Have multiple network paths to from controller to plant. That way, if a flood attack is underway, and a portion of the network is congested, the majority of the network can still function.

ACKNOWLEDGEMENTS

I would like to thank the Team for Research in Ubiquitous Secure Technology (TRUST), Dr. Kristen Gates, Sally Alcala, Larry Rohrbough, Ted Faber, Jelena Mirkovic, the DETERlab team, The University of California, Berkeley, as well as the National Science Foundation (NSF) for funding the program. Additionally, I would like to thank my mentors: Suzanna Schmeelk, Blaine Nelson, and Saurabh Amin. Special thanks also to my groupmates: Katherine Gabales, Ricardo Estrada, and Darrel Brower, who were a significant help in getting to this final point.

REFERENCES

[1] CERT. (2001, June 4). *Denial of Service Attacks* (3rd ed.) [Online]. Available: http://www.cert.org/tech_tips/denial_of_service.html

[2] CERT. (1996, September 24). *CERT® Advisory CA-1996-21 TCP SYN Flooding and IP Spoofing Attack* (14th ed.) [Online]. Available: http://www.cert.org/advisories/CA-1996-21.html

[3] Jignesh Patel. (2006). *Distributed Denial of Service Attacks* (1st ed.) [Online]. Available: http://www.slideshare.net/jignesh/ddos-attacks

[4] Internet2. (2009, April). *Internet2 Combined Infrastructure Topology* [Online]. Available: http://www.internet2.edu/pubs/200904-Internet2CombinedInfrastructureTopology.pdf

[5] Internet2. (2005, February). *Abilene Network* [Online]. Available: http://www.internet2.edu/pubs/200502-IS-AN.pdf

[6] University of Southern California USC Viterbi School of Engineering Information Sciences Institute. (2010, August 24). *DETERlab Testbed* [Online]. Available: http://www.isi.edu/deter/

[7] T. Benzel, R. Braden, D. Kim, C. Neuman, A. Joseph, K. Sklower, R. Ostrenga, S. Schwab. TridentCom2006. *Experience with DETER: A Testbed for Security Research*. 2nd IEEE Conference on testbeds and Research infrastructures for the Development of Networks and Communities.

[8] The University of Utah and Information Sciences Institute. (2010, June 29). *DETER SEER Wiki* (99th ed.) [Online]. Available: http://seer.isi.deterlab.net/trac