Software-Defined Network Moving Target Defense

Project Plan

sddec18-07
Client: Benjamin Blakely and Joshua Lyle (Argonne National Laboratories)
Faculty Advisor: Hongwei Zhang

Team Members and Roles:
Andrew Thai - *Project Manager*
Connor Ruggles - *Quality Assurance*
Emily Anderson - *Deliverables Manager*
Ryan Lawrence - *Communication Manager*

Team email: sddec18-07@iastate.edu
Team Website: https://sddec18-07.sd.ece.iastate.edu/

Table of Contents

List of Figures

List of Tables

List of Definitions and Acronyms

SDN: Software Defined Network
MTD: Moving Target Defense
NIC: Network Interface Card
CDC: Cyber Defense Competition
VM: Virtual Machine

1 Introductory Material

1.1 Acknowledgement

Our clients, Benjamin Blakely and Joshua Lyle from Argonne National Laboratories, have been the biggest contributors to our project so far. They have given us topics to research, tools to use, and an overall idea of what they are looking for in this project. Our advisor, Hongwei Zhang, is going to help us when it comes to the big deliverables later on in the project.

1.2 Problem Statement

As technological advances to technology rapidly increases, we see more sophisticated cyber attacks that become harder to detect and easier to penetrate through less secure networks. Hackers spend weeks and months gathering information on corporate networks to plan out their attack, making sure that they have the right information so that their attacks will work efficiently and effectively.

Our solution consists of creating a software defined network which consists of dynamically programming where packets are directed to when they are being sent to a corporate server. By doing so we will be able to route traffic on the fly so that we can migrate, take down, or add new servers to the network without any downtime. By using a software defined network we will be able to utilize it as a moving target defense system because we can configure the network so that it could transfer any packets that may seem malicious or come from an information gathering reconnaissance and direct it to different dummy servers so that the hackers would not be able to obtain any reliable information about the network. This would result in many wasteful weeks of attempting to grab information of a constantly changing network and allowing corporate networks to be more secure because their network isn’t static anymore.

1.3 Operating Environment

The environment that this design will be used in will be in a location where any public facing services servers are located. In many cases these servers will be located in datacenters but can also be located onsite at a company. Any physical hardware, such as switches and a server to host the controller, that would be put into place would be able to withstand standard networking environments such as networking closets or datacenter cabinets.

1.4 Intended Users and Intended Uses

The intended users for this project is for any company that would have public facing services such as hosting a website or any other service that needs to be accessible over the internet. This design can also be used for government institutions to protect from various information gathering attacks.

The use of this Software Defined Network Moving Target Defense is to provide an extra layer of security to dynamically route traffic to machines allowing for a wide variety of maneuvering to prevent network scanning as well as other types of attacks.

1.5 Assumptions and Limitations

Assumptions:
Physical or virtual switches running must support the OpenFlow protocol.
All switches should have a route to connect to the controller.

Limitations:
Not ideal for a home network.

1.6 Expected End Product and Other Deliverables

The end product will consist of a research paper that we write about implementing the actual Software Defined Network, a controller that has basic routing rules and documentation for creating more specific rules, some sort of installer or directions on how to install the controller onto the network with virtual machines, and the configuration files needed for the system to work as expected.

Research Paper - This is the main deliverable that the client wants, and will lay out the procedures for the entire project.

Installer/Install Directions - These will be either a full-fledged installer that will setup the controller for the user automatically, or directions on how to do it manually. The deliverable at the end will consist of either only the directions, or both the installer and the directions.

Configuration Files - Will be provided if config files are needed for controller setup

Other deliverables include usability and effectiveness of the system which show tested results that describe the impact of using this system as well as if it actually makes a significant difference than just using a regular network.

2 Proposed Approach and Statement of Work

2.1 Objective of the Task

The objective of this project will consist of coming up with a network setup where we implement switches (that run Open vSwitch) in a network with a controller to define how packets are transferred from one place to another. With this we will implement certain rules that will allow us to direct certain packets such as packets from an nmap scan to a dummy server so that it can collect inaccurate information of the system and network.

2.2 Functional Requirements

Functional requirements for this project will consist of a working demo of the design. It will consist of the system being able to analyze a packet and determine the route the packet to take depending on the type of packet that is being sent. Such as if a packet was sent from a nmap scan we would have the network switches redirect the packet to a “dummy” server so that the person who ran the scan would not be able to get any reliable information from the scan. Ideally there would be more than one dummy server, however that would be up to the customer to decide the exact implementation.

2.3 Constraints Considerations

Non-functional requirements would include that whichever type of service that we are implementing this network in would be that the accessibility and usability is not hindered in anyway whatsoever.

We will be using the OpenFlow Protocol Standard for this project. This standard is used to define how network controllers are implemented so that it can determine a path for a network packet. As with this standard, our switches will be able to connect to our controller on port 6633.

2.4 Previous Work And Literature

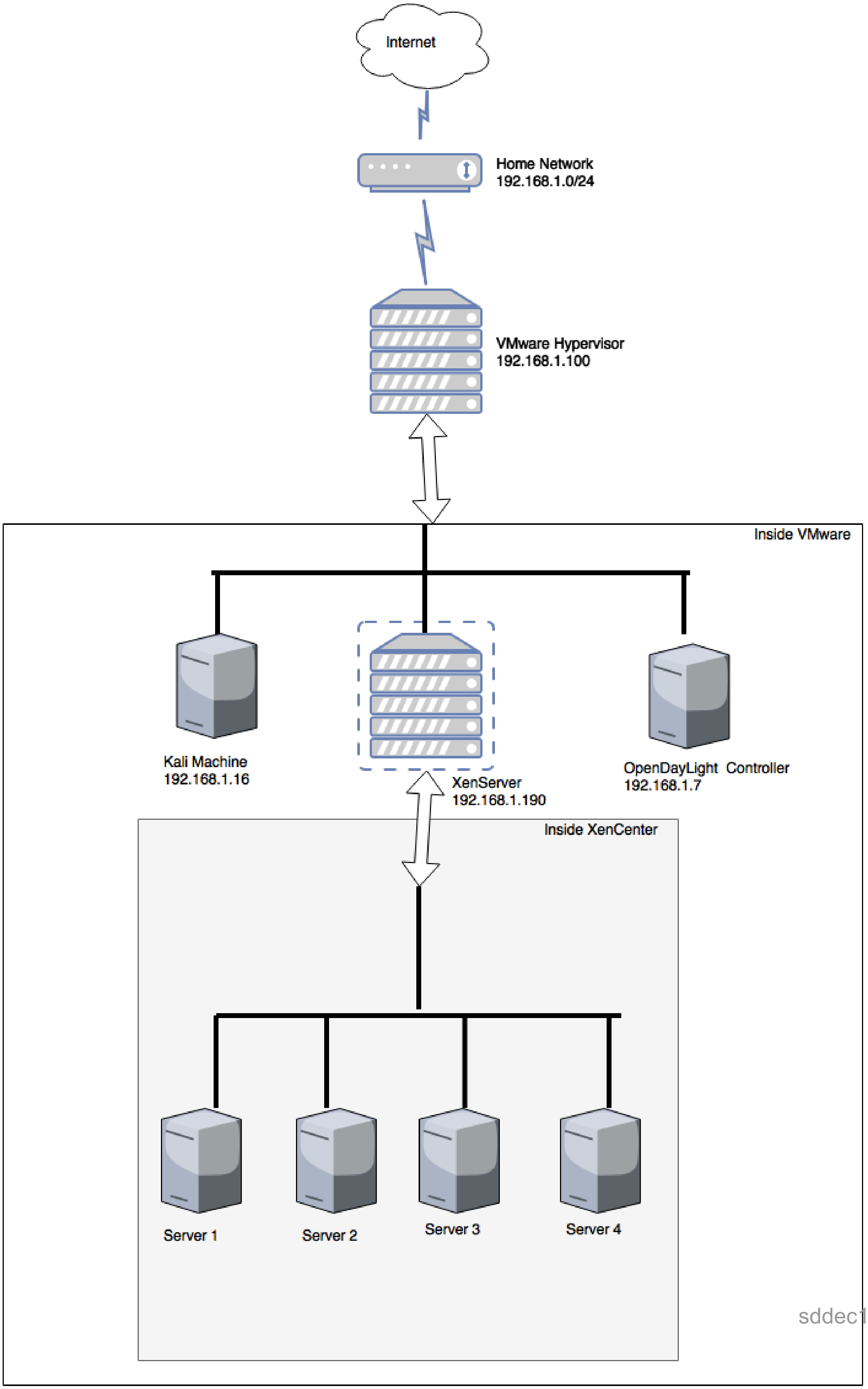
There is a lot of research that has been done into this topic before, but it is still not implemented by anyone, at least publicly. The advantages found from the research shows that an SDN MTD form of implementation can be extremely effective compared to standard practices. There are also many tools that can make up and control different aspects of the project, such as Citrix XenServer for the hypervisor that will contain the whole network and OpenDaylight as the flow controller that uses the OpenFlow protocol, all of which have extensive documentation.

OpenDaylight fits in well with the design of an SDN using XenServer as the hypervisor.

2.5 Proposed Design

Our current design will consist of creating a virtual hypervisor, Citrix XenServer, because we noticed that the hypervisor supported the use of Open vSwitches. Through this we will create a backbone of machines within XenServer to test our the virtual switch configuration on the controller and determine the packet’s path.

Standards include:
IEEE standards - Ethernet packets
OpenFlow standards - Switch protocols
Image of proposed design:


Internet
Home Network
192.168.1.0/24
VMware Hypervisor
192.168.1.100
Inside VMware
Kali Machine
192.168.1.16
XenServer
192.168.1.190
OpenDayLight Controller
192.168.1.7
Inside XenCenter
Server 1
Server 2
Server 3
Server 4

2.6 Technology Considerations

Our current design will consist of creating a virtual hypervisor, Citrix XenServer, because we noticed that the hypervisor supported the use of Open vSwitches. Through this we will create a backbone of machines within XenServer to test our the virtual switch controller configuration and determine the packet’s path with multiple virtual machines running.

The strengths of that software is that you can set up rules for incoming packets to be routed to specific VMs, based on the type of packet and/or the contents of the packet. This is done with an implementation of Open vSwitch that is included with XenServer.

Another way to do something similar would be to use a different hypervisor software and then run Open vSwitch manually, setting it up as the only route through the physical NIC for the VMs, and setting up rules or routes for packets in the same way as XenServer.

2.7 Safety Considerations

There are no safety concerns that need to be addressed at this time. There would be the concern of data security on the customer's end after implementation, however that is not within the scope of the project.

2.8 Task Approach

Our solution is to first create a network with a VMWare hypervisor in place so that we could do testing with multiple live virtual machines. To do so we decided to do this on a home network with the network diagram shown as:

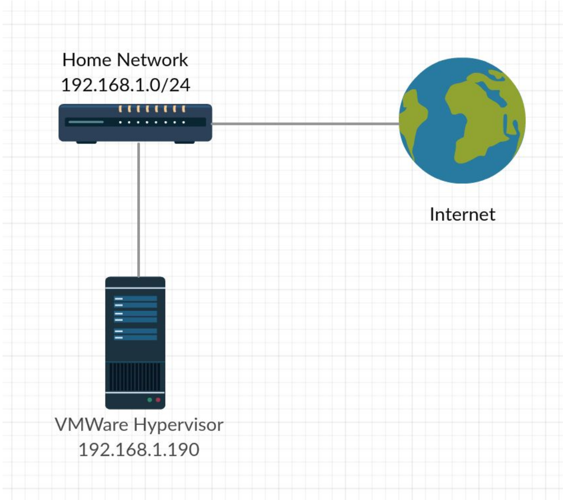


Figure 1: Home Network Diagram

We will then need to create a XenServer Hypervisor within the VMWare Hypervisor. In addition we will set the NIC for the XenServer to be on the same physical NIC of the VMWare hypervisor so that we can access it in the home network. As well as setting up a Kali machine to allow us to perform numerous types of penetration testing as well as scanning. A network diagram within

VMWare Hypervisor:

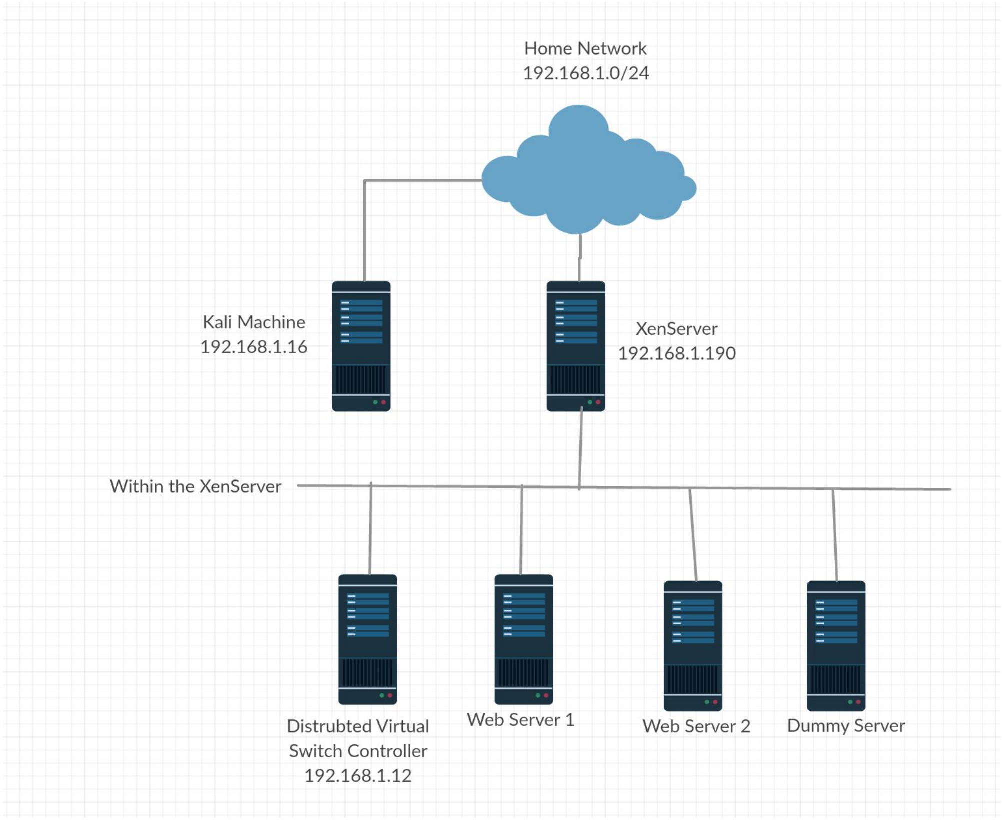


Figure 2: VMWare Hypervisor Diagram

2.9 Possible Risks And Risk Management

With this project, there is a lot of new knowledge that will need to be researched and take time to fully understand how to each part of the system works, this may slow down our plan as we adjust to new information that we discover and understand.

2.10 Project Proposed Milestones and Evaluation Criteria

Milestones will include:
Setting up the proposed testing network.
Configuring Open vSwitch Controller to be able to route packets.
Real life implementation test in a Cyber Defense Competition

Tests will include:
Perform nmap scan with packets being correctly routed.
Make sure machines that are not supposed to be seen outside the network are not able to be seen outside the network.

2.11 Project Tracking Procedures

Our group will be utilizing GitLab to track our progress throughout this course and any issues that we run into.

2.12 Expected Results and Validation

The desired outcome for this project is to create a software defined network that can dynamically route traffic to act like a moving target defense system.

We will confirm that our solution works by performing functional tests using a Kali Linux box.

2.13 Test Plan

Functional tests will include but are not limited to:

- Accessing a web server that will direct to two or three different servers.
- Nmap scan from a Kali Linux box and seeing that the packets route to the correct server.
- Make sure the controller can be inserted into an existing network.
- Relieving DDOS pressure by blocking connection from an ip if an overload of packets is sensed

3 Project Timeline, Estimated Resources, and Challenges

3.1 Project Timeline

The beginning stages of our project have mainly been research focused. We need to have a strong foundation so we can fully understand what is going on as we get further into the project. We have been researching software defined networks and moving target defense systems, potential ideas on how to implement our project, and more. Now that we have decided on using Citrix XenServer, moving forward we can start to test routing traffic on a dummy network. We will work on implementing as much functionality as possible so that by late March we can test our system in the ISU Cyber Defense Competition (CDC). From there, we will use our test results to make adjustments and improvements.

The Gantt chart we created, as shown in Section 4.3, is for our first semester of work only. By the end of the first semester we hope to have a rough prototype completed with a design plan how to scale up and test the prototype to production standards. Our tentative plan for second semester is to to use our new version of the project in another CDC for further testing. This CDC will be more useful, we think, as our system will be more refined than it was the first time and we will be able to more accurately pinpoint what we need to change and improve.

3.2 Feasibility Assessment

This project will end up being a real-world implementation of a software-defined network moving target defense system, whether that be created only by 3rd party tools, or we create some software to handle some aspects of the system. The challenges will be figuring out how to utilize multiple tools together into a realistic network system, and where the slack will be to pick up (if any).

3.3 Personnel Effort Requirements

Task	Andrew	Connor	Emily	Ryan
Design Plan v1	25%	25%	25%	25%
Project Plan v2	25%	25%	25%	25%
Project Plan Final	25%	25%	25%	25%
Design Plan Final	25%	25%	25%	25%
Website Final	10%	10%	10%	70%
OpenDaylight Research	30%	30%	20%	20%
SDN Research	20%	30%	30%	20%
SDN Controller Setup	30%	20%	20%	30%
Nmap Scanning	20%	20%	20%	40%
Test Network Setup	100%	0%	0%	0%
MTD Research	25%	20%	25%	30%

Table 1: Personnel Effort Requirements

3.4 Other Resource Requirement

Currently there are no other resource requirements in relation to this project. For real implementation of our project, the user would need either an existing network or the financial and physical resources to create one.

3.5 Financial Requirements

Currently there are no financial requirements in relation to this project.

4 Closure Materials

4.1 Conclusion

With the amount of security risks that static networks can face in today's world, a solution to provide extra layers of security to the network is needed. Our goal of creating a Software Defined Network Moving Target Defense (SDNMTD), will help to alleviate this risk. By creating this we will be able to monitor, control, and analyze packets that go through a network and minimize the risk of information gathering and manipulate the flow of traffic to protect the network as a whole.

4.2 References

Jafarian, J. H., Niakanlahiji, A., Al-Shaer, E., & Duan, Q. (2016). Multi-dimensional Host Identity Anonymization for Defeating Skilled Attackers. Proceedings of the 2016 ACM Workshop on Moving Target Defense - MTD16. doi:10.1145/2995272.2995278

Kampanakis, P., Perros, H., & Beyene, T. (2014). SDN-based solutions for Moving Target Defense network protection. Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014. doi:10.1109/wowmom.2014.6918979

Mahler, D. (2014). Netfool Networking. Retrieved from https://www.youtube.com/user/mahler711

Okhravi, H., Rabe, M. A., Mayberry, T. J., Leonard, W. G., Hobson, T. R., Bigelow, D., & Streilein, W. W. (2013). Survey of Cyber Moving Target Techniques. doi:10.21236/ada591804

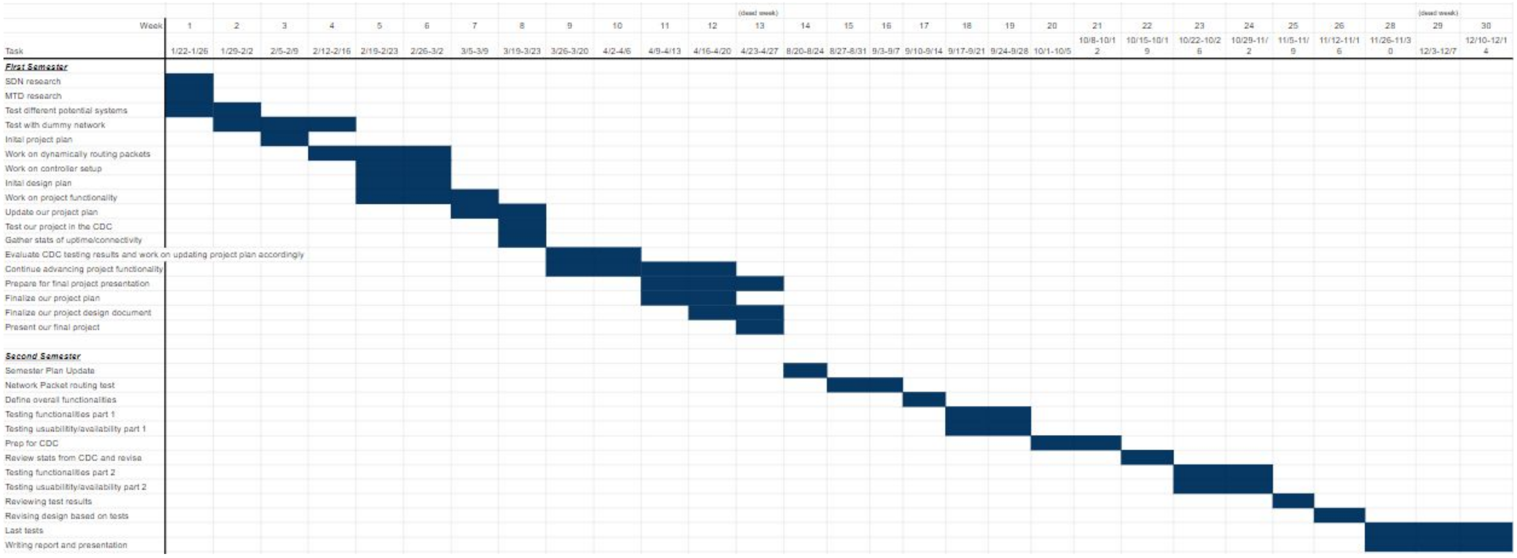
Skowyra, R., Bauer, K., Dedhia, V., & Okhravi, H. (2016). Have No PHEAR. Proceedings of the 2016 ACM Workshop on Moving Target Defense - MTD16. doi:10.1145/2995272.2995276

Stakhanova, Natalia; Basu, Samik; and Wong, Johnny S., "A Taxonomy of Intrusion Response Systems" (2006). Computer Science Technical Reports. Paper 210. http://lib.dr.iastate.edu/cs_techreports/210

Zhuang, R., Bardas, A. G., Deloach, S. A., & Ou, X. (2015). A Theory of Cyber Attacks. Proceedings of the Second ACM Workshop on Moving Target Defense - MTD 15. doi:10.1145/2808475.2808478

Zhuang, R., S. A., & Ou, X. (2015). Towards a Theory of Moving Target Defense. Proceedings of the First ACM Workshop on Moving Target Defense - MTD 14. doi:10.1145/2663474.2663479

4.3 Appendices



													(dead week)															(dead week)	
Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	28	29	30
Task	1/22-1/26	1/29-2/2	2/5-2/9	2/12-2/16	2/19-2/23	2/26-3/2	3/5-3/9	3/19-3/23	3/26-3/20	4/2-4/6	4/9-4/13	4/16-4/20	4/23-4/27	8/20-8/24	8/27-8/31	9/3-9/7	9/10-9/14	9/17-9/21	9/24-9/28	10/1-10/5	10/8-10/12	10/15-10/19	10/22-10/26	10/29-11/2	11/5-11/9	11/12-11/16	11/26-11/30	12/3-12/7	12/10-12/14
First Semester																													
SDN research	■																												
MTD research	■																												
Test different potential systems	■	■																											
Test with dummy network		■	■	■																									
Inital project plan			■																										
Work on dynamically routing packets				■	■	■																							
Work on controller setup					■	■																							
Inital design plan					■	■																							
Work on project functionality					■	■	■																						
Update our project plan							■	■																					
Test our project in the CDC								■																					
Gather stats of uptime/connectivity								■																					
Evaluate CDC testing results and work on updating project plan accordingly									■	■																			
Continue advancing project functionality									■	■	■	■																	
Prepare for final project presentation											■	■	■																
Finalize our project plan											■	■																	
Finalize our project design document												■	■																
Present our final project													■																
Second Semester																													
Semester Plan Update														■															
Network Packet routing test															■	■													
Define overall functionalities																	■												
Testing functionalities part 1																		■	■										
Testing usuabilitity/availability part 1																		■	■										
Prep for CDC																				■	■								
Review stats from CDC and revise																						■							
Testing functionalities part 2																							■	■					
Testing usuabilitity/availability part 2																							■	■					
Reviewing test results																									■				
Revising design based on tests																										■			
Last tests																											■	■	■
Writing report and presentation																											■	■	■

Figure 3: Gantt Chart