Software-Defined Network Moving Target Defense

Project Plan

Team ID: sddec18-07

Client: Argonne National Laboratory, Dr. Benjamin Blakely and Joshua Lyle

Faculty Advisor: Dr. Hongwei Zhang

Team Members:
Andrew Thai - *Project Manager*
Connor Ruggles - *Quality Assurance*
Emily Anderson - *Delivery 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 3
List of Tables 3
List of Definitions and Acronyms 3

1 Introductory Material 4
1.1 Acknowledgement 4
1.2 Problem Statement 4
1.3 Operating Environment 4
1.4 Intended Users and Intended Uses 4
1.5 Assumptions and Limitations 5
1.6 Expected End Product and Other Deliverables 5

2 Proposed Approach and Statement of Work 6
2.1 Objective of the Task 6
2.2 Functional Requirements 6
2.3 Constraints Considerations 6
2.4 Previous Work And Literature 6
2.5 Proposed Design 7
2.6 Technology Considerations 8
2.7 Safety Considerations 9
2.8 Task Approach 9
2.9 Possible Risks And Risk Management 11
2.10 Project Proposed Milestones and Evaluation Criteria 11
2.11 Project Tracking Procedures 12
2.12 Expected Results and Validation 12
2.13 Test Plan 12

3 Project Timeline, Estimated Resources, and Challenges 12
3.1 Project Timeline 12
3.2 Feasibility Assessment 13
3.3 Personnel Effort Requirements 14
3.4 Other Resource Requirements 15
3.5 Financial Requirements 15

4 Closing Materials 15
4.1 Conclusion 15
4.2 References 15
4.3 Appendices 16

List of Figures

Figure 1: Proposed corporate design
Figure 2: Home Network Diagram
Figure 3: VMWare Hypervisor Diagram
Figure 4: Gantt Chart

List of Tables

Table 1: Personnel Effort Requirements

List of Definitions and Acronyms

- SDN: Software-Defined Network
    - Software-defined networking is a concept where the actual routing of data packets is moved to a separate layer and is taken care of programmatically by a network controller, that then sends the packets down to the main network switch to route to the individual servers on the network.
- MTD: Moving Target Defense
    - This is a concept where you detect if a specific machine is being attacked and you have preset rules to mitigate to rotate that machine out of being public facing, and rotate in a “honeypot”, or something that looks like a real machine but it distracts the attacker long enough to block them out.
- NIC: Network Interface Card
    - This is the physical device that connects all of the machines connected to a switch, to the Internet.
- CDC: Cyber Defense Competition
    - A type of competition where teams try and defend a set of servers against a team of attackers in a pre-defined scenario.
- VM: Virtual Machine
    - A software emulation of physical aspects needed to run a full computer operating system.
- Hypervisor
    - A hypervisor is a piece of software that allows a server to run virtual machines on.

1 Introductory Material

1.1 Acknowledgement

Our clients, Dr. Benjamin Blakely and Joshua Lyle from Argonne National Laboratory, 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, Dr. Hongwei Zhang, is going to help us when it comes to the big deliverables later on in the project.

1.2 Problem Statement

As the pace of advancement in information and operational technology systems rapidly increases, cyber-attacks become more sophisticated due to the additional resources available to cyber criminals. They have become harder to detect and more effective at penetrating networks. Cyber criminals might spend weeks and months gathering information on target 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 a software defined network (SDN) controller that dynamically adjusts where incoming packets are directed when they are being transmitted to a server. By doing so, we will be able to route traffic on the fly to migrate, take down, or add new servers to the network without any downtime. We will utilize a SDN as a moving target defense (MTD) system. The controller we develop will dynamically configure the network to detect any packets that are malicious or come from an information gathering reconnaissance tool and direct them to dummy servers, also known as honeypots. This will prevent or delay an attacker from obtaining any reliable information about the network. This could result in many wasted attempts to grab information regarding the constantly changing network, thus allowing the network to be more difficult to attack than a static configuration.

1.3 Operating Environment

This design will be used in a location where public-facing servers are located. For example, many institutions use a demilitarized zone (DMZ) network segment for web servers or email services which require incoming requests to be served. Such servers could be located in datacenters or on-premises at a facility belonging to the owner. Any physical hardware, such as switches or 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 the developed product are any company with services that use multiple virtual or physical servers, whether internal or not, such as hosting a website or any other service that uses some sort of a network connection between multiple other servers. This

design can also be used for government or military institutions to protect from various information gathering attacks.

This SDN MTD product will provide an extra layer of security by dynamically routing traffic to an array of systems thus allowing for a wide variety of maneuvering to impede network scanning.

1.5 Assumptions and Limitations

Assumptions:

1. Physical or virtual switches used must support the OpenFlow protocol.
2. All switches must have a route to connect to the SDN controller.
3. An implementation of SDN can delay an attacker enough so we can mitigate the attack.
4. There are companies/customers willing to implement SDN on their own network.

Limitations:

1. Not ideal for a home network.
    a. The resources required and scope of the whole system would be inefficient for the size of a typical home network.

1.6 Expected End Product and Other Deliverables

The end product will consist of :

- A research paper describing:
    - Background on SDN, the protocols selected for our implementation (e.g., OpenFlow);
    - Gaps in existing implementations similar to what is being developed;
    - The threat model defining the scope of attackers the product is designed to defend against;
    - Details of the implementation of the SDN MTD product (including diagrams, where appropriate);
    - The evaluation methodology used to assess the performance of the product and degree to which it counters the in-scope attacks;
    - Results of the assessment;
    - Recommendations for future work;
- Source code or configurations for a SDN controller with basic routing rules (and documentation for creating more specific rules);
- An executable and/or process to aid the end user in deploying the controller onto a virtual machine network; and
- Any other configuration files needed for the system to work as expected.

Research Paper - This is the primary deliverable and will lay out the procedures for the entire project so that the methodology can be assessed, replicated, and extended.

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 so manually. This will consist of either the directions only, 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:

- The OpenFlow protocol to not be hindered by hardware or physical setup

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 in our research into SDN MTD there were no examples of real implementation, but a lot of theories are out there. The advantages of an SDN MTD solution, as identified in our literature review, demonstrate the potential effectiveness compared to standard practices. These papers discuss implementation similar to that of a dogfight, where specific maneuvers are used during an attack to increase chances of survivability, rather than simply doing nothing.

There are also multiple 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 these have extensive documentation[9]. OpenDaylight fits in well with the design of an SDN using XenServer as the hypervisor because it uses a REST API to talk to the switches, which would make it easy to implement on a closed network. Previous examples have also tried doing prototypes with similar systems using things such as Cisco onePK.

2.5 Proposed Design

Our proposed design consist of adding a hypervisor into the production environment of a company with the following machines: Floodlight Controller, Security Onion, and a honeypot. In addition to the added hypervisor, companies will also need OpenFlow compatible switches that will be able to interface with the Floodlight Controller. This setup will allow for any packets that route through the switches to ask the controller for what the next hop will be based on defined rules that the company has implemented. Along with the controller, there will be implementation with Security Onion to monitor network traffic as an intrusion detection system to allow for alerts of attacks. With the use of Security Onion, companies can create rules based on the results and alerts gathered from Security Onion to mitigate attacks and damages that may affect performance or usability.

Standards include:
IEEE standards - Ethernet packets
OpenFlow standards - Switch protocols

Image of proposed design:

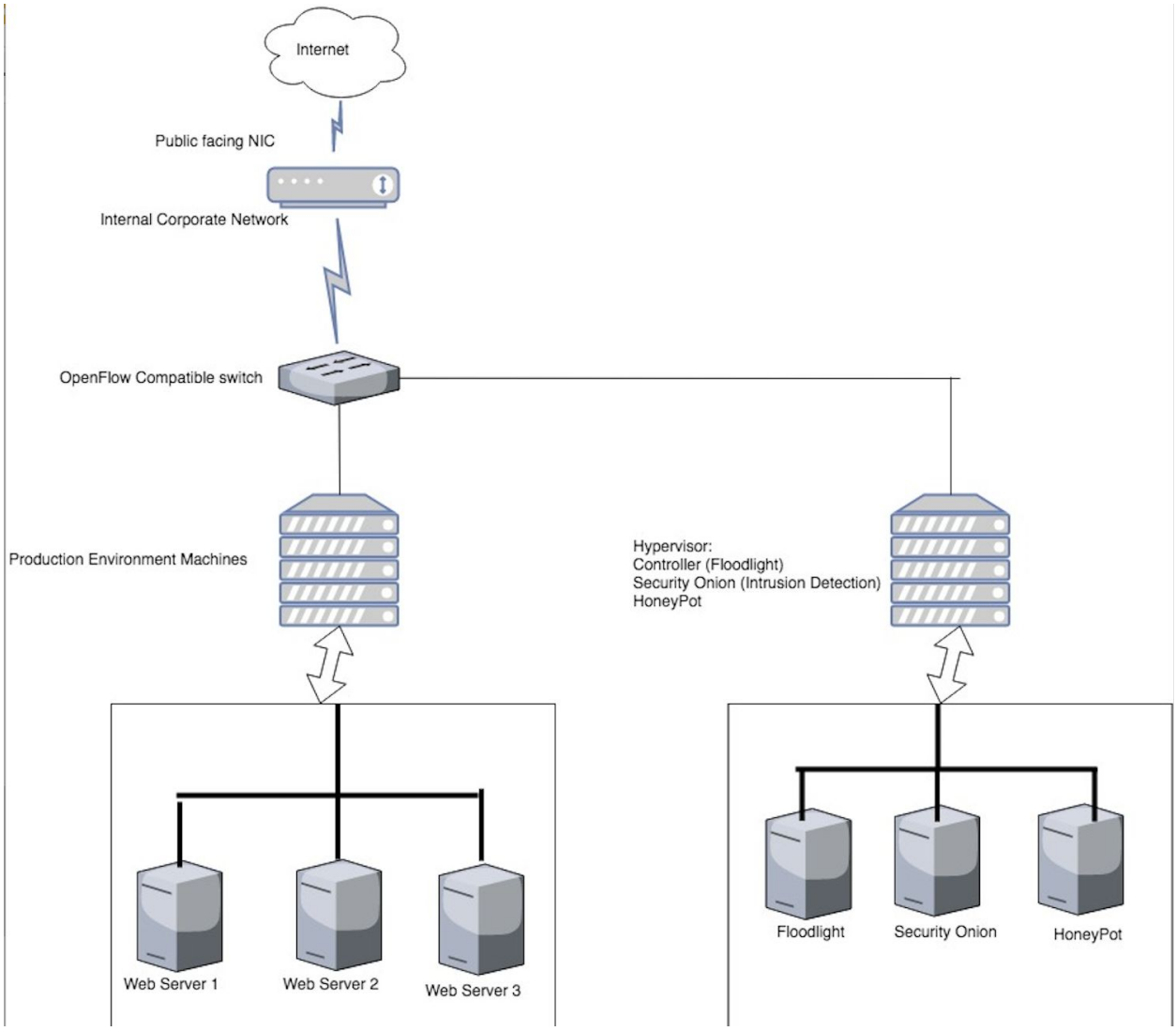

Figure 1: Proposed corporate design

2.6 Technology Considerations

Our current test design to mimic a company's production environment 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.

A strength of XenServer 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.

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

We will first create a test network infrastructure, including a VMWare hypervisor, for testing multiple virtual machines. To do so, we decided to do this on a home network with the network diagram shown in Figure 1.

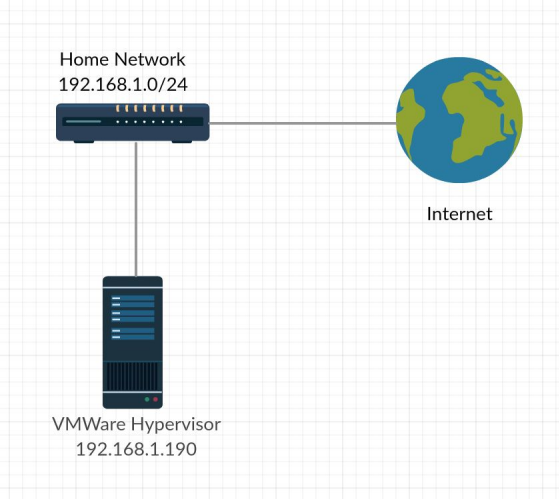

Figure 2: Test Network Diagram

We will then create a XenServer Hypervisor within the VMWare Hypervisor. We need both the XenServer hypervisor and VMWare hypervisor because that will allow us to create multiple VM's on the same "level" of the network, all of them being inside VMWare and then only the switch itself inside XenServer. We will configure the management network interface for the XenServer to be mapped to the physical NIC of the VMWare hypervisor, so that we can access it from the home network. Lastly, we will deploy a Kali Linux (a LiveCD-based Linux distribution used by penetration testers) virtual machine to allow us to perform numerous types of network scanning. This topology is shown in Figure 2.

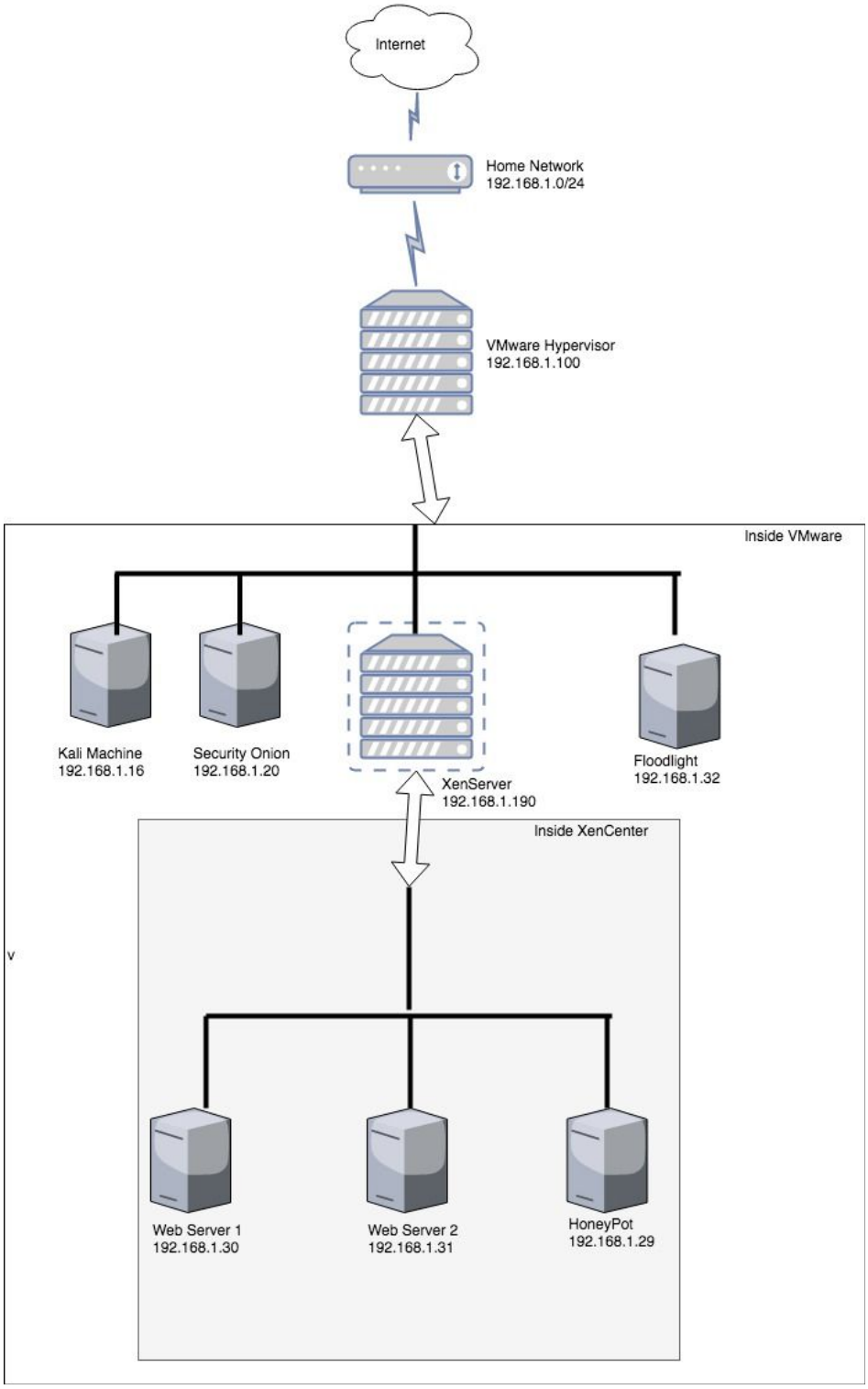

Figure 3: VMWare Hypervisor Diagram

2.9 Possible Risks And Risk Management

- Point of failure if the controller is compromised by a hack
    - Put the controller in a separate segregated network such that it can't be used as a pivot point for corporate machines
- Unstable/unavailable connection to the controller could result in downtime for the overall network.
    - Allow for multiple controllers and default switch configurations in case of controller failure or maintenance without downtime
- Incorrectly defined rules can result in block services
    - Testing in a quality assurance network to ensure fully functional services and connectivity.
    - Rules can be easily deleted if accidentally created incorrectly

2.10 Project Proposed Milestones and Evaluation Criteria

Milestones will include:

1. Setting up the proposed testing network. - week 4
2. Configuring Open vSwitch Controller to be able to route packets. - week 16
3. Real life implementation test in a Cyber Defense Competition - week 21

The weeks are according to the two-semester schedule for senior design and can be seen in the Gantt chart attached at the end of the document.

Tests will include:

1. Perform nmap scan with packets being correctly routed.
2. 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 the issue tracking capabilities on 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 (this will most likely be a home network of one of our team members').
- 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 test network.

The Gantt chart shown in Section 4.3 covers both the spring and fall semesters. By the end of the first semester we will have a rough prototype completed with a design plan detailing how to scale up and test the prototype to production standards. By the end of the second semester, we hope to have a fully documented, thoroughly descriptive research paper and maybe a real world example or two to deliver to the client.

A tentative side-plan for second semester is 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 deliver a research paper that will fully document how to implement an SDN MTD in a production environment. It may provide an example or two to help show what is described in the document.

3.3 Personnel Effort Requirements

Task	Andrew	Connor	Emily	Ryan
Project Plan v1	25%	25%	25%	25%
Design Plan v1	25%	25%	25%	25%
Project Plan v2	25%	25%	25%	25%
Project Plan Final	5%	85%	5%	5%
Design Plan Final	5%	5%	85%	5%
Team Website	5%	5%	5%	85%
SDN Research	20%	30%	30%	20%
MTD Research	20%	30%	30%	20%
Test Network Setup	100%	0%	0%	0%
Nmap Scanning/Wireshark	20%	20%	20%	40%
OpenDaylight Research	30%	30%	20%	20%
OpenDaylight Flow Control	30%	20%	20%	30%
Setup OpenDaylight, XenServer OpenFlow switch, and test machines	80%	0%	0%	20%
Setup Floodlight controllers	100%	0%	0%	0%
Floodlight Research	30%	30%	20%	20%
Floodlight Flow Control	30%	20%	20%	30%
Research on different intrusion detection systems	40%	20%	20%	20%
Final Presentation	45%	5%	5%	45%

Table 1: Personnel Effort Requirements

3.4 Other Resource Requirements

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. Financial requirements to put this design in a company's production environment would include the cost of setting up server with multiple virtual machines as well as buying and replacing switches with OpenFlow compatible switches.

4 Closing 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 (SDN MTD), 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

[1] 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

[2] 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

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

[4] 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

[5] 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

[6] 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

[7] 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

[8] 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

[9] Citrix Systems, Inc., Documentation. https://xenserver.org/overview-xenserver-open-source-virtualization/documentation.html

4.3 Appendices

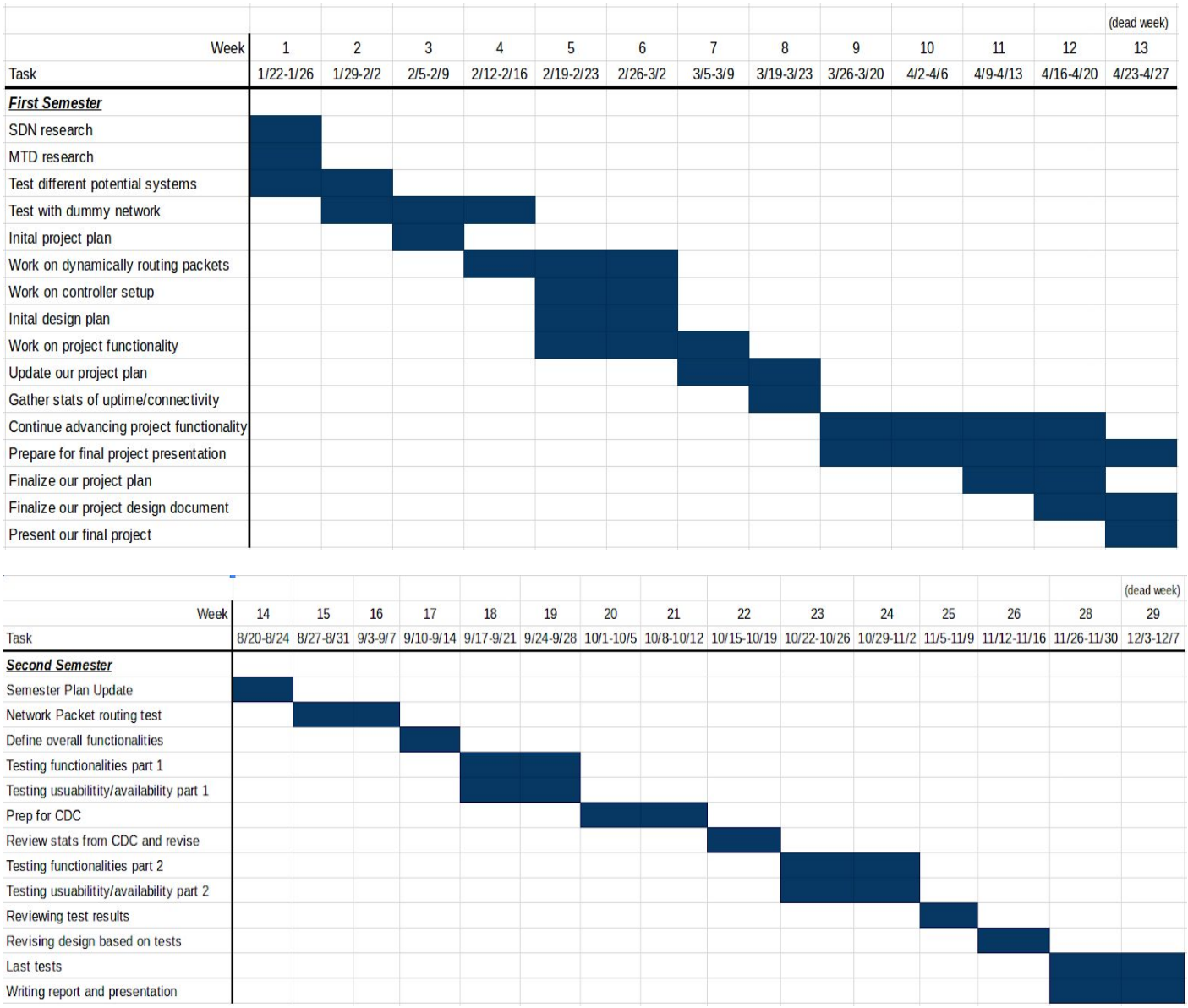

Figure 4: Gantt Chart