Rotating Adaptive Network Defense (RAND)

Active, Anti-Intrusion, Antivirus, Artificial

Design Document

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 Introduction

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 and Project 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 Operational 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 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 Deliverables

The end product will be the configuration files and controller used to direct traffic on the network. The controller will integrate with software defined network to configure switches. This can be seen as the OpenDaylight Controller in the network diagram in section 4.

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. Specifications and Analysis

2.1 Proposed Design

We first tried to implement our design using a piece of software that virtualized a small network called mininet. Testing using this was very limited since the machines were only accessible within that virtualized network so running any sort of scanning or penetration testing was unavailable.

With that in mind we decided to created a small network to implement our design. Our network consist of a Kali machine, OpenDayLight machine, and a XenServer Hypervisor to control our networked machines. We are current in the progress of connecting our XenServer Hypervisor to our OpenDayLight controller to control the packets.

More packet design configurations will be added as we determine what will be the best course of action to be taken depending on the situation that is presented.

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

2.2 Design Analysis

With our current setup we were able to create a more realistic like network that we could perform functional and non-functional requirements and allow us to directly control all machines within the environment. This setup currently seems like the best setup that will allow us to provide the best environment for us to determine what sort of packet control we want given a specific scenario.

The strengths of our proposed solution makes it easier for us to determine all endpoints of where packets are transferred, giving us full accessibility to the machines and the switches.

3 Testing and Implementation

3.1 Interface Specifications

We are using VMWare Hypervisor to interface with all of our machines. The XenServer Hypervisor within our VMWare Hypervisor is used to manage the individual virtual machines that will be used for testing and connecting to the controller.

3.2 Hardware and software

Hardware that we used for this design consisted of a Dell PowerEdge R710 Server. This server is running VMware Hypervisor to allow for creation and configuration of virtual machines. Having our own physical server to work on allows for us to easily create and destroy machines as needed in our design without having to get more hardware.

We will be using Citrix XenServer Hypervisor to host all of our machines that will be tested on because the hypervisor support the OpenFlow Protocol which will allow us to connect its internal switch to our OpenDaylight controller. The controller will then be able to manage the flow of packets within the testing network to the machines.

We will also be using utilities such as Wireshark to analyze packets as well as create determine the effectiveness of our packet control design.

3.3 Functional Testing

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.

3.4 Non-Functional Testing

For security testing we will need to make sure that attackers will not be able to get in or view the actual layout of our network, such as not being able to see the machines that are set up behind the controller/switch, they should only be able to know about the purposely outward facing machines. We also will need to test usability and compatibility so that someone can seamlessly plug the controller we make into their network.

3.5 Process

The process diagram below shows the processes taken for testing each of our methods described in section 2.

3.6 Results

So far NMap scans have been tested on the network. A Kali box was used to perform a typical scan on a machine in the network. Wireshark was used to analyze the packets that were exchanged in during this time so that a controller could recognized this kind of action.

OpenDaylight is being tested as the form of controller for the network. It is currently deployed on the server, but tests are being done on how to configure it to recognize different forms of attacks on the server and tell the switch what it should do with the packets.

4 Closing Material

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

"Defense4All:Tutorial." Wiki OpenDaylight, MediaWiki.org, wiki.opendaylight.org/view/Defense4All:Tutorial.

4.3 Appendices

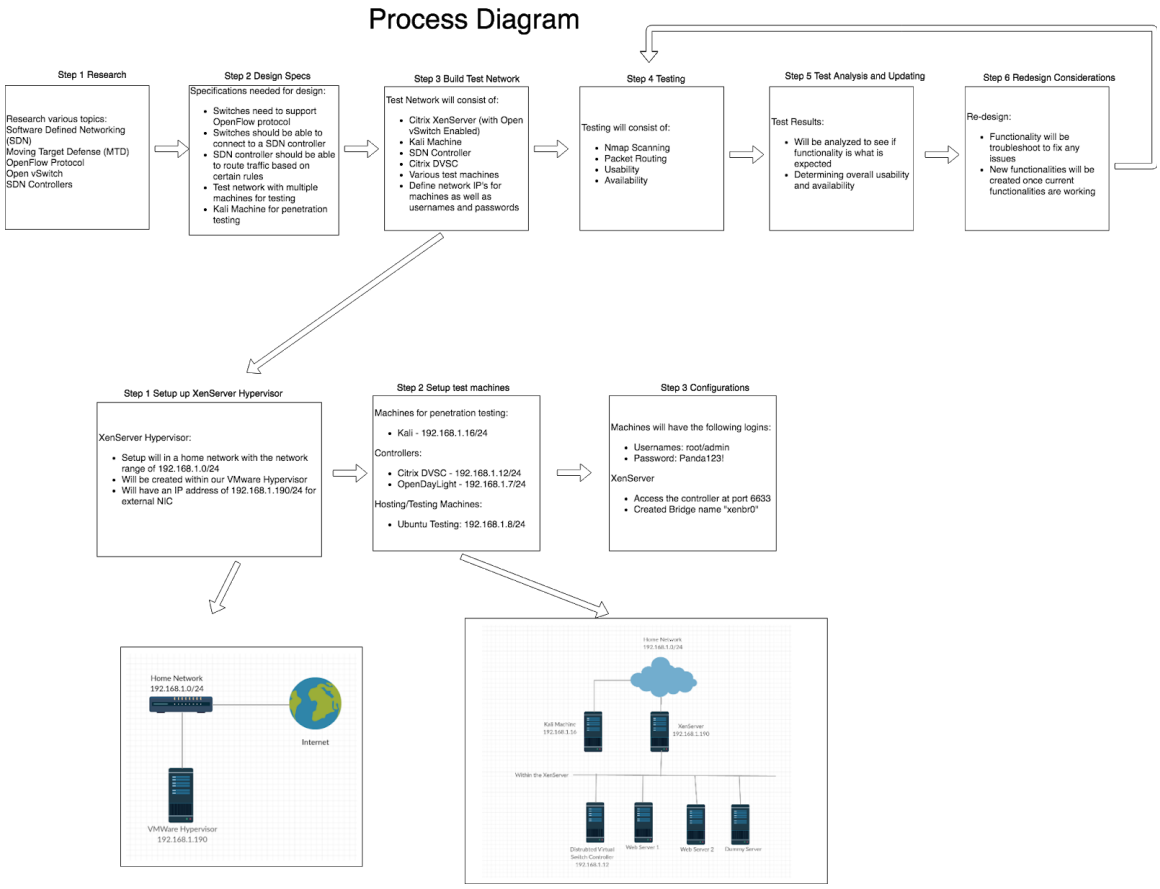

Figure 1: Process Diagram

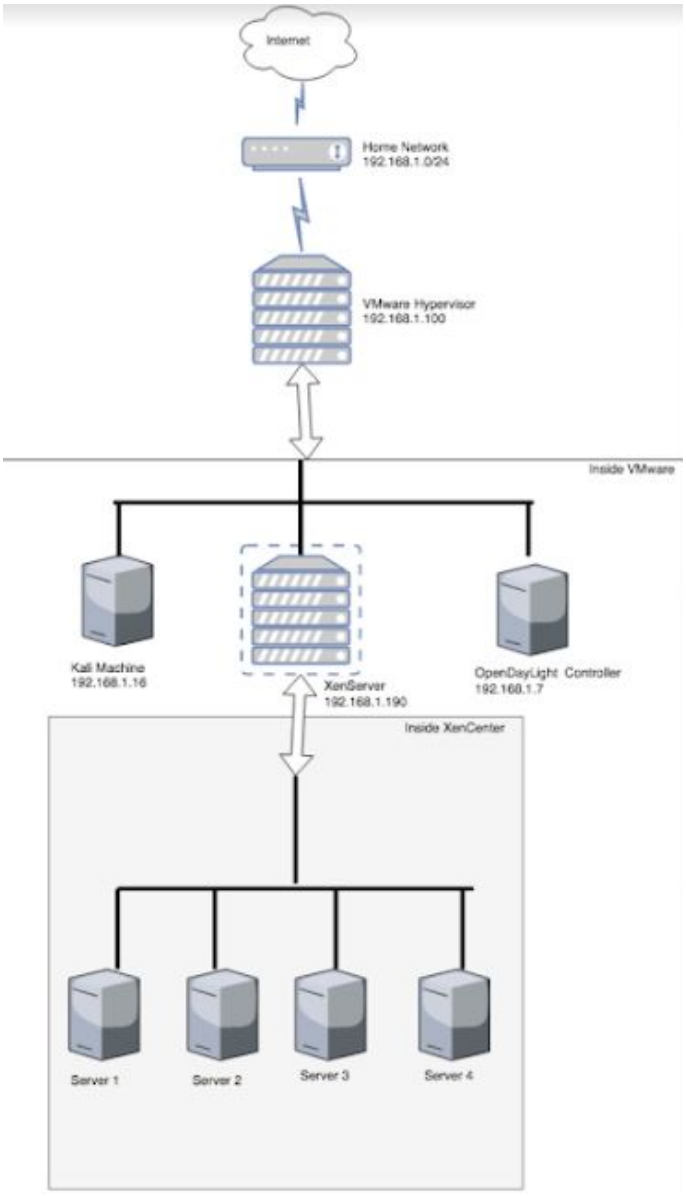

Figure 2: Representation of a software defined network with the deliverable controller installed