Proceedings of The National Conference
On Undergraduate Research (NCUR) 2019
Kennesaw State University
Kennesaw, Georgia
April 11-13, 2019

A Hybrid Software Defined Network Platform for Undergraduate Research and Education

Steven Cilenti
Information Technology
The United States Military Academy
West Point, New York 10997 USA

Faculty Adviser: MAJ Eric Sturzinger

Abstract

Software Defined Networks (SDNs) are leading the evolution toward network programmability and open architectures. While many corporations, nonprofits, and individuals have developed training on SDNs, the industry has a significant gap with the robustness of entrenched traditional network educational models, such as Cisco's Networking Academy. The Department of Defense (DoD) will likely adopt some form of SDN into its global transport network at various tiers and authority boundaries. It is imperative for 21st century leaders to understand how and why the manner in which DoD provides Information Technology (IT) services to its customers is changing with such rapidity. Therefore, we developed three basic SDN course lessons as a base of knowledge and support and integrated a hybrid physical SDN research platform into existing laboratory infrastructure for faculty research and capstone projects for senior cadets. This was accomplished by leveraging existing SDN-related tutorials and resources and integrating them within a virtualized SDN simulation environment. The three lessons were developed for integration into our core networking course that describes fundamental networking concepts in the context of an SDN - with a centralized control plane, while ensuring lesson learning objectives were achievable by non-technical majors yet sufficiently comprehensive across the fundamental operations of an SDN. The hybrid research platform consists of a number of Virtual Machines (VMs) running Mininet[1] - an SDN simulation environment - and hosted on a VMware vSphere cluster with direct connectivity to twelve physical openflow-capable switches. This will allow students in the networking course to plan, design, implement, and test a basic SDN topology in either a virtual, physical, or hybrid environment. In addition, it will provide topological and experimental flexibility to student and faculty researchers and senior capstone project teams alike.

Keywords: Software Defined Networking, Department of Defense, Undergraduate Technical Education

1. Introduction

The Department of Defense has been on the leading edge of IT innovation since ARPANET. One of the characteristics of IT technologies is that they necessitate continuous change. This new networking paradigm known as Software Defined Networking (SDN) is growing rapidly within the tech industry. With a strong majority of companies turning toward virtualization and software defined data centers, the Department of Defense has fallen behind. It is inevitable that the DoD adopts some type of SDN paradigm, rendering it essential that our education curriculum keep pace with these new advances. The transition to software defined networks is necessary for the DOD to keep its technical infrastructure relevant. If the military is going to adapt to this change, then its future leaders need to be familiar with these developments. Currently, the United States Military Academy has no hardware or software components of a software-defined network capable of supporting SDN research. The Academy's two primary networking courses, CY350 and CS484, do not provide hands-on platforms for students to design and implement a working software-

defined network. The goal of this study was to close this capability gap for future research and establish SDNs as part of the permanent IT/CS curriculum. Since SDNs are becoming increasingly relevant we want students to be introduced and familiarized with their general operation and implementation. Section 3 of this paper discusses how we created three lesson plans that can be integrated into an undergraduate networking course. These lessons are also accompanied by supporting resources and labs. Our next priority was to develop a hybrid physical/virtual SDN research platform for students and faculty, which is discussed in section 4. This platform will act as a resource that will be available for projects and research by both students and faculty in order to further push the bounds of our SDN knowledge and utilization.

2. Related Work

Software defined networking is a relatively new term, but it has its roots in multiple areas of research and development. "The Road to SDN: An Intellectual History of Programmable Networks" explains the history of software defined networking and how it was developed[2]. SDN solves many of the challenges of traditional networks by separating the control plane from the data plane, allowing for easier network management. "Software-Defined Networking: A Comprehensive Survey" largely explains how software defined networks differ from traditional networks[3]. Traditional networks are hard to manage due to operators needing to configure each individual device separately. The difference with the SDN paradigm is reflected in Figure 1. "Opportunities and Research Challenges of Hybrid Software Defined Networks" presents research on hybrid networking models, integrating SDNs with traditional network infrastructure, and providing analysis on different hybrid models[4].

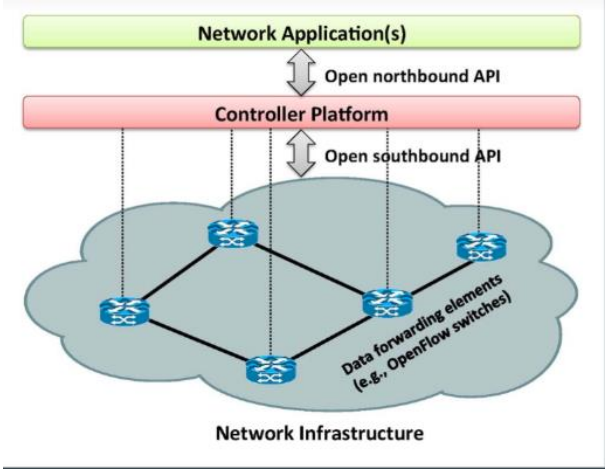

Figure 1. The separation of the control and data planes in SDN[5].

In order to create and run tests on virtual SDNs, we decided to utilize Mininet. Not only can Mininet be used to rapidly prototype large networks, but it also supports SDN functionality. These details were originally presented in "A Network in a Laptop: Rapid Prototyping for Software-Defined Networks"[6]. Mininet is great for both research and educational purposes. "Teaching Software Defined Networking: It's not just coding" talks about integrating SDNs into academics and presents research on SDNs being introduced to students in New Zealand[7]. The school used both physical network equipment as well as a virtual network using Mininet. The paper expresses how important it will be for institutions to prepare students for this expanding technology. As for virtualizing these networks, "Modeling Software Defined Networks using Mininet" describes how Mininet can be used to test SDN features[8]. The software allows for an efficient and rapid deployment of a virtual SDN technology. Mininet is used by schools such as James Madison University to teach SDN concepts, as shown in the paper "Hands-on Labs and Tools for Teaching Software Defined Network (SDN) to Undergraduates"[9]. This paper just discusses a virtualized platform, while our research is on a hybrid, physical and virtual, research platform.

The applicability of SDNs to the DOD is explained in "Software-Defined Networking and Network Programmability: Use Cases for Defense and Intelligence Communities"[10]. Not only does the paper expand on why it is crucial the DOD adopts SDNs, but it also explains how SDNs have the ability to solve many of the DODs networking and security issues. "Employing SDN to Control Video Streaming Applications in Military Mobile Networks" shows

one such use case of Software Defined Networking[11]. SDNs can be used to allow more efficient and dynamic management of our military's data.

3. Educational Approach

One of our goals was to provide students with an introduction to Software Defined Networking. Every cadet takes basic IT classes which don't delve deep into networking. Cadets also choose an Engineering sequence of three classes to take, one of which of which is the Cyber Engineering Sequence. The very first class in this sequence is CY350, Network Engineering and Management. The traditional networking paradigm has been the only networking paradigm taught in that class. USMA had no installed hardware or software capable of supporting SDN research and there was a lack of general familiarity with the SDN paradigm. To do this we developed an educational lesson plan that could be integrated into one of our existing networking courses. To accompany those lessons, we developed some hands-on lab material that would assist in learning and provide a baseline knowledge level for implementing SDNs. We also developed a hybrid virtual-physical research platform for students and faculty to use for education and research.

3.1 Lesson 1: Intro to Software Defined Networking

Lesson Objectives:

- Understand how SDNs differ from traditional networks
- Mininet Demo: Basic switch controller interaction
- Understand how to use MiniEdit
- Be able to create and run a Topology

The first lesson focuses on the basics of SDN and how it differs from traditional networks. It enumerates the pros and cons of each and their different use cases. This lesson is also used to introduce students to Mininet, which allows for the creation and testing of virtual networks, including SDNs. Mainly, they are shown how to start Mininet and set up their network. An example of a basic network in the MiniEdit interface is shown in Figure 1.

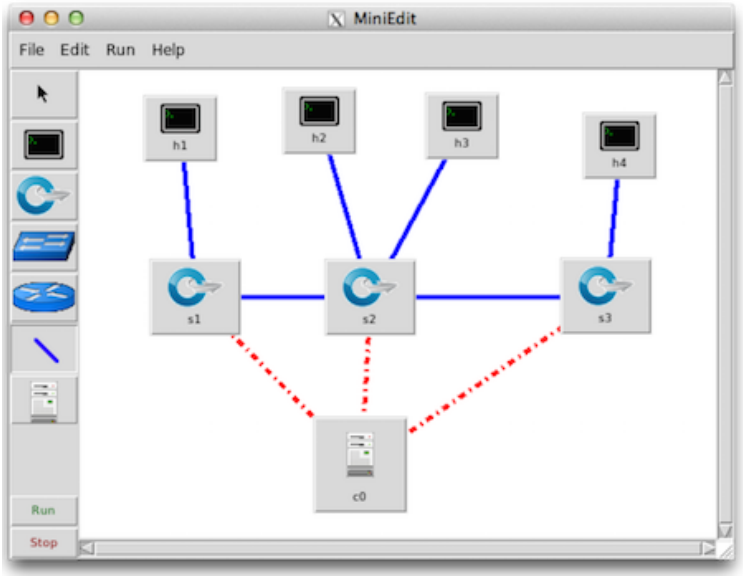

Figure 1. MiniEdit: a graphical interface for building Mininet networks

The second part of the lesson shows students how to link a minimal controller to the network and run a quick test to show functionality.

3.2 Lesson 2: The Control Plane

Lesson Objectives:

- Describe how the control plane works
- Develop rules for routing
- Program a controller application, using Ryu (python)

The second lesson is meant to really get into how the control plane works and how SDNs utilize controllers. Students are shown how to set up and develop rules for a controller which would be the equivalent of what they had already learned with traditional networks. The controller is written using Ryu, which is a python based module. Students are not expected to build controllers from scratch, but instead manipulate prebuilt templates to adapt to various scenarios. The controller instance is written using a python Class. The controller template being used in this lesson comes from the Ryu documentation and the beginning of which is shown in figure 2.

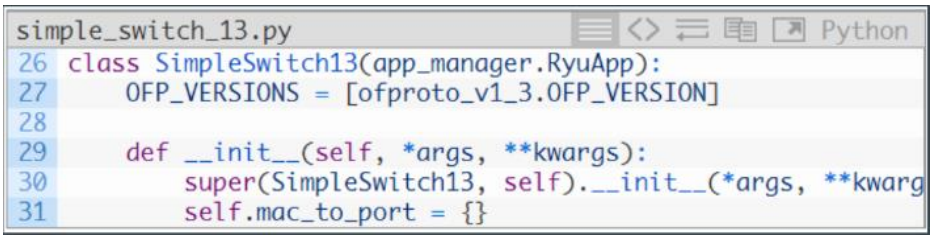

Figure 2. Controller Code Snippet[12]

Students will have the opportunity to test out their controller using the Mininet network they had created from the previous lesson.

3.3 Lesson 3: SDN Implementation

Lesson Objectives

- Describe how SDNs are used commercially
- Describe SDN application to tactical environment
- Transfer virtual network to physical network

The third lesson is meant to transition the students from a virtualized network to a physical network. The concepts emphasized during this lesson include some of the use cases for SDNS. This includes how they are used commercially by corporations like Google, as well as how they can be used in a tactical environment. This lesson includes a demonstration which shows how the students can utilize the Zodiac GX switches at their computer stations and connect them to the controllers they set up during the previous lesson. The layout of the physical network is defined in section 4.2. They are introduced to configuring the switches using the web interface shown in Figure 3. Students will then work together to direct and manage traffic on the lab network to accommodate given scenarios.

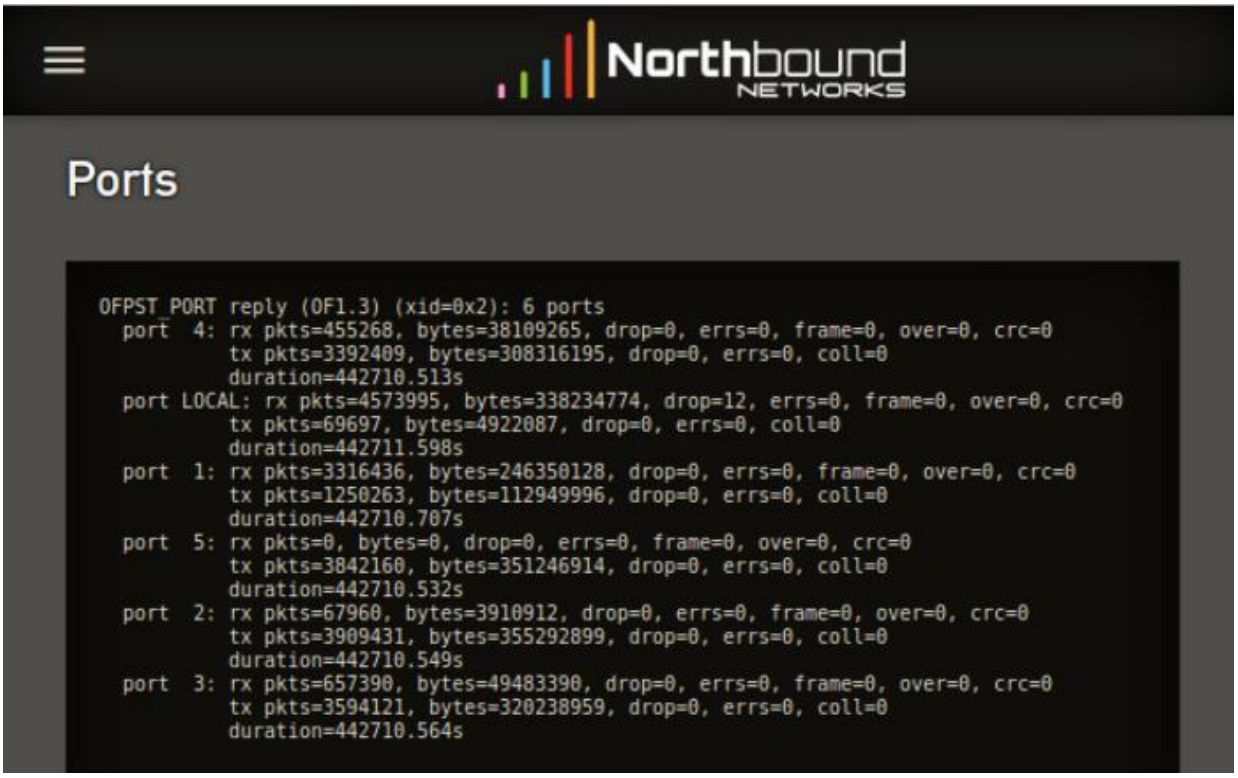

Figure 3. Zodiac GX web interface

3.4 Labs and Support

As a companion to the lessons, we developed two labs which aid in learning how to use Mininet and Ryu. The Mininet lab details how to utilize to design and manipulate an SDN topology. It also includes instructions on connecting the topology to a controller and running it. The Ryu lab breaks down the key functionality of the basic Ryu controller application. It details how to manipulate the code to turn off and on functions like updating the MAC-to-port table on the controller and updating the flow tables on the switches.

Many different sources are used in the development of these lessons, most of which haven't been combined before. By integrating these sources into a lesson plan, students will have a much more comprehensive and efficient learning experience. It is important to note that this material is not completely comprehensive. It is meant to also be understandable to someone who is not a networking related major. These lessons will utilize what students have already learned about traditional networks in order to introduce them to software defined networks.

4. Platform/Infrastructure

4.1 Virtual Platform

In order to work with and teach SDN concepts, we chose to use both virtual and physical platforms. The advantage to having virtual networks includes being able to scale and adapt these networks to suit any classroom or research requirements, without having to purchase and configure additional equipment or infrastructure. For our simulated network, we chose Mininet. Mininet runs in on an Ubuntu VM and allows you to build and run tests on a virtual network. The program also allows you to experiment with OpenFlow switches and controllers. A great thing about working with Mininet is that you can use either a console or a GUI (Graphical User Interface) to build out and test network functionality. The controllers run using Ryu, which is a python API that also supports the OpenFlow protocol. A virtual platform allows for scalability and flexibility, which one doesn't have with physical equipment. It allows students to each design and run their own networks. For research, it greatly speeds up the design and testing process.

To demonstrate how this virtual network functions, we used a simple topology with 2 hosts connected to a switch. Figure 4 shows this network being run along with a Ryu controller. When h1 (host 1) pings h2 (host 2), s1 (switch 1) sends a packet to the controller to know where to send the pings. The controller then sends a flow update to switch 1. s1 then sends the ping to h2. When h2 sends the ping response, switch 2 contacts the controller to locate h1. For every subsequent ping, s1 has both h1 and h2 in its flow table so it no longer needs to contact the controller for this exchange. This is the reason why the first ping took 7.47 MS while every subsequent ping took less then .4 ms.

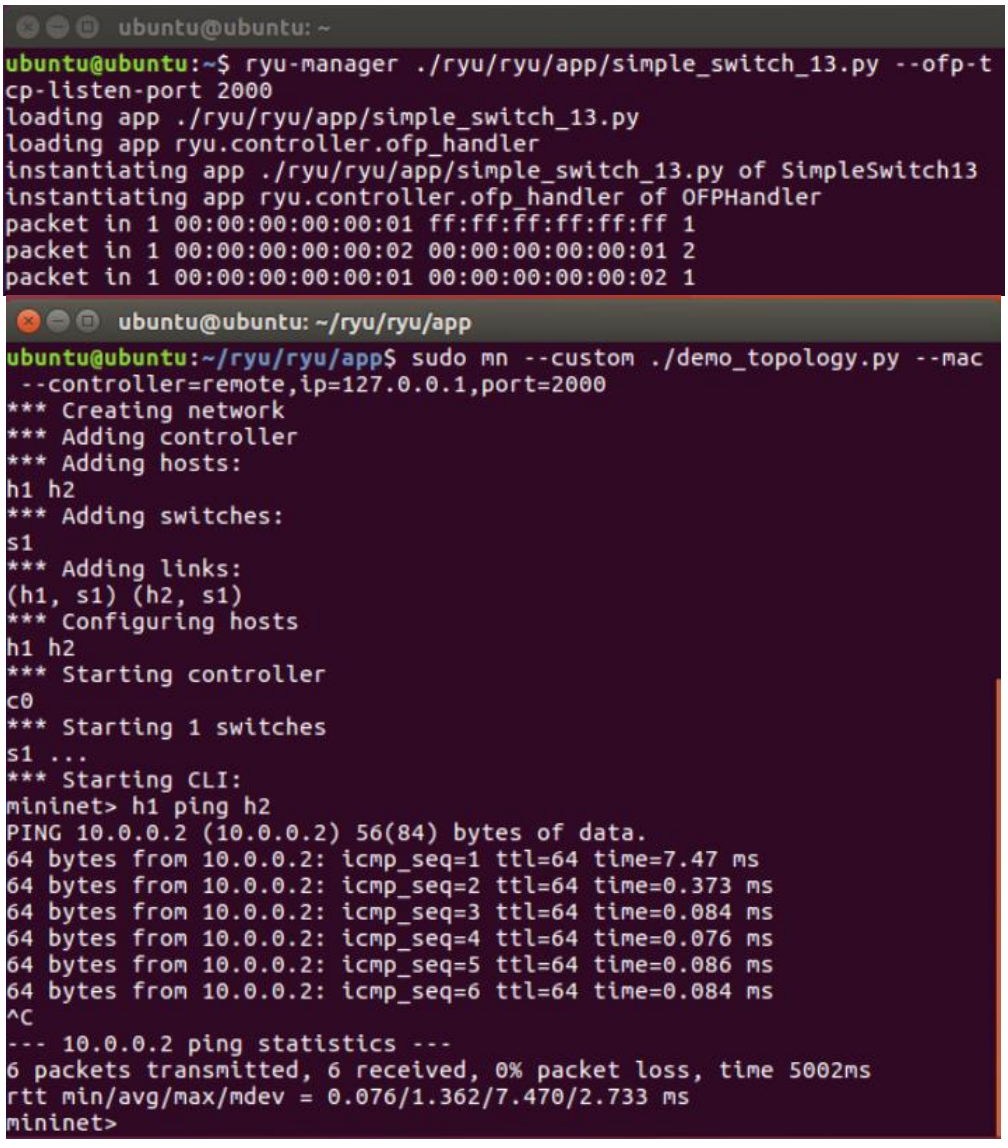

Figure 4. Controller output (left terminal) after pinging from one virtual host to another (right terminal)

4.2 Physical Platform

While a physical platform is logistically difficult to acquire and implement, requiring the purchase and use of hardware, it is much more realistic. Our physical platform consists of two different switch models. The first is the Zodiac GX[13], which is a simple 5 port switch that utilizes OpenFlow enabled software. The way we designed the lab layout is that every student has access to their own Zodiac switch linked to their work station. These switches are linked together as seen in figure 5.

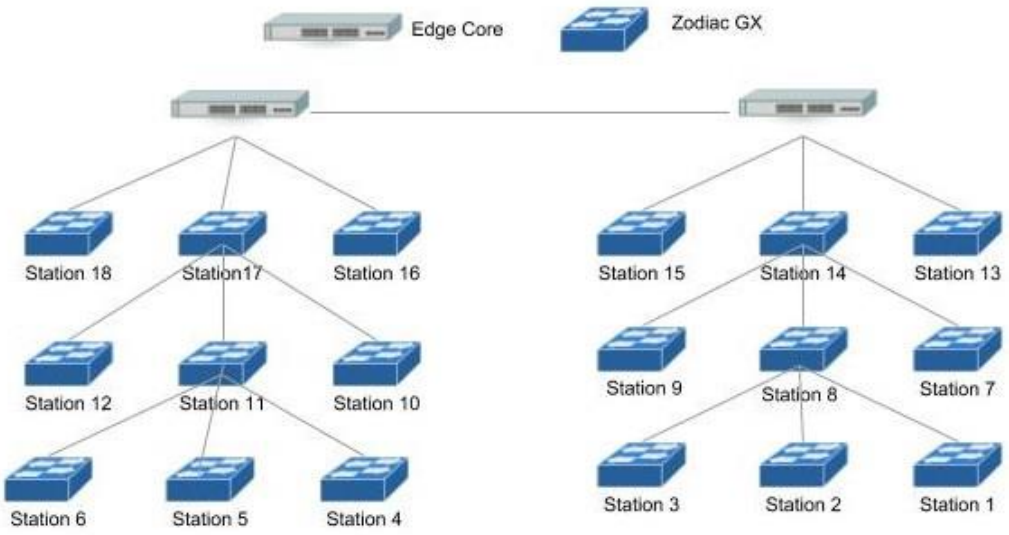

Figure 5. Physical lab topology diagram

We are also utilizing two EdgeCore switches, which are larger 54 port switches without any pre-installed software[14]. This is what connects our external network to the network of Zodiac switches. Because of this layout, every student is able to run their own controller on a different port, which the switches can be set to listen to depending on the scenario.

4.3 Hybrid Infrastructure

Finally, we created a hybrid SDN platform to support a more adaptable and efficient architecture. As part of this layout, the SDN controllers can be run in Ubuntu VMs in our VSphere cluster. This VSphere cluster is connected to our SDN and can run VMs which can be used as both controllers and hosts, as seen in Figure 6.

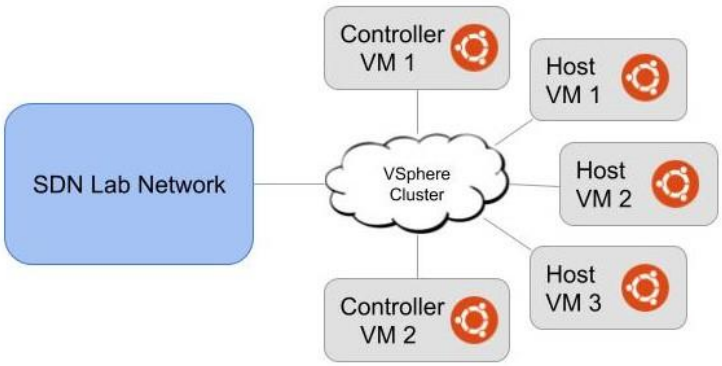

Figure 6. A VSphere cluster example for testing the SDN

The students will be able to use the same controllers that they did with Mininet, easily transitioning between their simulated and physical networks.

The following scenario is meant to demonstrate how this network works. Switch 8 wants to send data to switch 10, but does not know where that switch is, so the mac address has not been added to its flow table. Switch 8 will take the first packet and send it to its controller in the VSphere cluster. If Switch 10 is in the controller's MAC-to-port table, then it will send a flow rule back to switch 8 to update its flow table. Now switch 8 will be able to send the data to Switch 10 without needing to contact the controller because its flow table has been updated. On the other hand, if the

controller did not have switch 10 in its MAC-to-port table, it would have sent a FLOOD command back to the switch in order to locate switch 10. This will tell the switch to send out the frame on all ports. Once switch 10 responds, the controller will update its MAC-to-port table and send flow table updates to the switches.

5. Experimentation

To date, we have only tested the basic connectivity of the network with the scripts which were developed for the lessons. We have ensured the functionality of the virtual, physical, and hybrid networks using both physical and virtual controllers. Further testing and research will be continued through capstone projects led by seniors at the Academy. Our goal is to develop a novel trust model to secure future SDN topologies in a tactical military network. The purpose of the hybrid topology is to simulate the integration of the tactical network with the national-level cloud-based data centers.

6. Conclusion

Through our research, we filled a capability gap at West Point, which we feel is important due to the growing importance of Software Defined Networks. We were able to consolidate existing resources to build an easy to understand, yet effective educational resource. This will allow students with limited knowledge of networks to learn and understand how to implement a software defined network. It is imperative to educate our future leaders to on this emerging paradigm as it will serve to better our ability to maneuver in cyberspace while denying the same to our adversaries.

7. References

1. Bob Lantz, Brandon Heller, and Nick Mckeown, "A Network in a Laptop," Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks - Hotnets 10 (2010).
2. Nick Feamster, Jennifer Rexford, and Ellen Zegura, "The Road to SDN," Queue 11, no. 12 (2013): 20–40.
3. Diego Kreutz, Fernando Ramos, Paul Verissimo, Christian Rothenberg, Siamak Azodolmolky, and Steve Uhlig, "Software-Defined Networking: A Comprehensive Survey," Proceedings of the IEEE, vol. 103, no. 1 (2015): 1476.
4. Diego Kreutz, "Software-Defined Networking: A Comprehensive Survey." (2015): 1476.
5. Stefano Vissicchio, Laurent Vanbever, and Olivier Bonaventure, "Opportunities and Research Challenges of Hybrid Software Defined Networks," ACM SIGCOMM Computer Communication Review 44, no. 2 (2014): 70–75.
6. Bob Lantz, "A Network in a Laptop," (2010).
7. Cosgrove, Steve, "Teaching Software Defined Networking: Its Not Just Coding," 2016 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE) (2016).
8. Casimer DeCusatis, Aparicio Carranza, and Jean Delgado-Caceres, "Modeling Software Defined Networks using Mininet," Proceedings of the 2nd International Conference on Computer and Information Science and Technology (CIST16) (May 2016).
9. Emil Salib, and John Lester, "Hands-on Labs and Tools for Teaching Software Defined Network (SDN) to Undergraduates," Cisco (2014).
10. Mark Mitchiner and Reema Prasad, "Software-Defined Networking and Network Programmability: Use Cases for Defense and Intelligence Communities," Cisco White Paper (2014).
11. Iulisloi Zacarias, Janaina Schwarzrock, Luciano P. Gaspary, Anderson Kohl, Ricardo Q. A. Fernandes, Jorgito M. Stocchero, and Edison P. De Freitas, "Employing SDN to Control Video Streaming Applications in Military Mobile Networks," 2017 IEEE 16th International Symposium on Network Computing and Applications (NCA) (2017).
12. "Ryu Documentation," Release 4.28 (2018).
13. "Zodiac GX," Northbound Networks, https://northboundnetworks.com/products/zodiac-gx.
14. "AS4610-54T," Edgecore Networks, https://www.edge-core.com/productsInfo.php?cls=1&cls2=9&cls3=46&id=21.