THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Mobility Management in Wireless Software-Defined Networks

KENTON HERTZOG
SPRING 2023

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Computer Science
with honors in Computer Science

Reviewed and approved* by the following:

Thomas La Porta
Evan Pugh Professor, William E. Leonhard Endowed Chair and Director of EECS
Thesis Supervisor

Danfeng Zheng
Associate Professor of Computer Science and Engineering
Honors Adviser

* Electronic approvals are on file.

ABSTRACT

Multilevel security serves a unique and vital function within a network: restricting access of users where necessary and protecting the flow of information between nodes. Such software-defined networks are marked by their central controller, which deploys rules on switches across the network. The advent of software-defined networks opened an avenue for a multilevel security algorithm managed by the central controller; said algorithm was defined as MLSnet by researchers at Penn State and boasts impressive improvements over similar algorithms. However, MLSnet has been primarily studied in wired networks with end nodes that do not move. In this work we expand the algorithm by considering wireless networks with end nodes that can move between access points, characterizing the overhead of MLSnet and its impact on performance with mobile users. Finally, we discuss methods for improving this performance.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I want to start by thanking Dr. Thomas La Porta for his willingness to take on this project with me, his reliability, and his flexibility as new challenges arose. As well, I would like to thank graduate students Rahul George and Mingli Yu who's familiarity with the technologies involved and willingness to help with hurdle after hurdle made this thesis possible. Lastly, I want to thank my girlfriend Laura, whose support over this entire process helped me through many moments of frustration and worry.

Chapter 1

Introduction

In this chapter we define our problem and give a background on the types of technology involved in our system. As well, we provide real-world examples and cite related works around our problem. Finally, we introduce a solution to be explored further in the next chapter.

1.1 Problem Description

Wireless tactical networks (also known as mobile ad hoc networks or MANets) are important. To begin, MANets are multi-hop and infrastructure-less networks. Multi-hop means that a given host requires several relays of data to reach its destination, while infrastructure-less means no base stations exist to route traffic. Another vital technology alongside these MANets are wireless mesh networks (WMNs), which share the multi-hop nature of MANets but are altered due to their full connectivity of internal nodes and the existence of a base station [2].

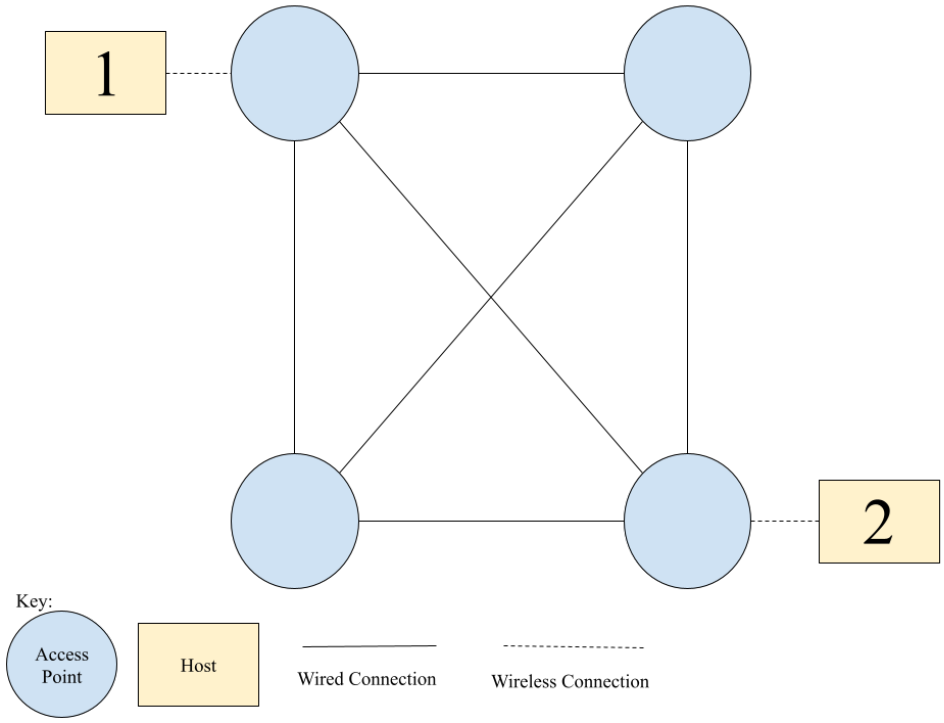

Figure 1: A simple n=4 Mesh Network

Their flexibility and quick deployment allow for utilization in a myriad of situations; these include setting up MANets in military environments or disaster response areas (said examples will be explored in a later section). In these situations, the networks will be host to a variety of different users - e.g., police, soldiers, first responders, media, civilians. These different users have unique needs when it comes to the network, and all have needs for restricted information.

However, despite these valuable and variable use cases, MANets come with some noticeable drawbacks. Notably the ad hoc nature of the networks creates three key issues: issues of security, access control, and information priority. The issues of security are that of network integrity; the allowance of any node to communicate in all areas of the network develops vulnerabilities such as spoofing and modification per [10]. The same source illuminates the area of confidentiality, to say the restriction of information to certain users. The problems of access

control overlap with that of general security but magnify the necessity of data safety and the idea that, "… [all] nodes are cooperative and non-malicious." Information priority is especially important in MANets due to the limited availability of resources and the equality of all users. An example of such would be the equality of a civilian making a call in an emergency zone versus an emergency medical service worker consulting experts on treating wounded. Providing specific flows of information not only protection but priority on a network is essential. Together, these three areas of concern show that despite their obvious boons, MANets' issues are glaring and must be addressed for further use.

1.2 Examples of MANets

As noted, MANets (and WMNs) find their best use cases in situations where flexibility is vital, resources are slim, and setup time is limited. Situations including disaster recovery and military operations meet all these conditions and are hallmarks of MANet and WMNs use.

As such an example, [3] discusses the ability for ad hoc mobile networks to enable the use of Emergency Telehealth Services (ETS) in disaster-stricken areas. They note that due to the frequency and plausibility of local communication resources being damaged, ad hoc networks uniquely solve said issue due to their affordability and comparably higher bandwidth capabilities. The enhanced bandwidth is especially vital to allow for the use of video during consulting. When properly executed, first responders and survivors would be able to quickly access nearby personnel and get timely medical information and instructions.

The paper simulates such a network where communication from the primary site to the nearest gateway node needs several hops. Despite a drop in throughput after the first hop, such a

network would enable ETS engagement in such an area. However, the paper specifically notes that, "… prior to provid[ing] service, it is required to solve private information security issues caused by network sharing by unspecified node in MANET." This falls in line with our above noted existing issues.

On the side of military applications, [4] discusses both the use case of MANets and WMNs for military operations but also investigates the intrinsic issues posed in creating such a network.

The paper notes that such MANets and WMNs would connect to the larger Global Information Grid (GIG) as edge networks, referred to as "tactical edges." They discuss the dynamic and shifting nature of said tactical edges, where points that are fixed may relocate and many nodes are moving at any point. They outline the ideal MANet for military operation:

> "In the ideal military MANET, membership is dynamic, but limited only to authorized terminals."
>
> "The military MANET is completely self-forming and self-healing in the sense that authorized units can join, leave, and rejoin the MANET without manual intervention (need-ed currently)."

From the above point (and many more as noted in the original paper), the ideal MANet for military use is far from existence.

In the conclusion of the paper, they review the key areas that are holding back MANets from the ideal use case described. Of largest interest, they note routing, management, and security; these areas once again mirror the above issues of ad hoc and mesh networks.

It becomes increasingly obvious that should such networks become commonplace and see utilization in key areas, improvements need to be made in the areas of security, access control, and information priority.

1.3 Enter MLSnet

MLSnet is a software-defined network (SDN) algorithm that labels and relabels nodes within networks with a security level to enforce multilevel security flows. The algorithm creates security through isolation of flows. It creates access control by restricting the flow of information between nodes of varying level and it creates routing priority for flows of higher security levels [9]. Simply put, MLSnet seems poised to solve many of the issues raised with MANets and WMNs. In exploring MLSnet further in the next section, it brings the possibility to solve the necessary issues. Furthermore, the algorithm's limitations are shown as well.

In this thesis we expand the use case of MLSnet to wireless software-defined networks with end user mobility to ensure the same security and flow maximization. In the expansion, we see that MLSnet maintains its ability to route flows and enforce multilevel security, while providing a level of mobility support not present in standard component controllers. In Chapter 2 we present MLSnet as it was defined prior to the expansion of its use case, show the problems it solves in more traditional networks, and finally outline its issues with wireless networks and user mobility. In Chapter 3 we define the testbench and its associated technologies, namely Mininet (a software that creates virtual SDNs), its extension Mininet-Wifi (which adds wireless network emulation), and the POX SDN controller. In Chapter 4 we outline the suite of testing done and delve into the results, comparing several trials involving MLSnet and stock components. Lastly

in Chapter 5 we discuss the results shown in Chapter 4. We show that MLSnet can support mobility with low overhead. Further we show the relabeling and security enforcement remains consistent in these new environments. Lastly, we discuss the possibility of future works on the experiment.

Chapter 2 MLSnet

MLSnet is a security-focused network relabeling algorithm that places emphasis on internal nodes of a software-defined network with multilevel security capabilities. Here we review the basic services and operation of MLSnet. Interested readers can review [7][9] for full details. Originally written and defined in paper [9], the algorithm utilizes a Bell La-Padula model [12] (or its relaxed variant) to build and route flows. In its trials, it displayed an impressive 87-89% success rate in maximizing fully secured flows and minimizing security cost to remaining flows; these solutions involved a two-pronged approach: maximizing the number of functional flows, and prioritizing flows with the given security rates [9]. As earlier stated, MLSnet acts as a prime candidate for fixing the issues that plague MANets and WMNs.

2.1 Multilevel Security

To understand just how MLSnet can solve said challenges, an explanation of multilevel security is required. Multilevel security (MLS) is a style of network security in which both active entities (known as subjects) and passive entities (known as objects) are labeled with a security level and security area. Levels are hierarchical in nature and rank lowest to highest, while different areas serve to categorize the objects and define which objects a subject has authorization over, per [11]. For instance, MLSnet defines a four-level hierarchy: one being public, two being confidential, three being secret, and four being top-secret. Therefore, a subject with level one security in a category only has access to objects with level one security in that category. Meanwhile a subject with level four security in a category can access all levels of objects in that category.

MLS-styled policies are often defined on a read-write basis with network wide file control in mind. However (as noted in the original MLSnet experiment) this can be difficult to enforce in a network where information may have hops across several nodes to be retrieved. When these nodes fail to meet the security clearance of the data, said data or even the network itself can be susceptible to attack. On the other hand, when the internal nodes are constantly and inflexibly striving to meet the nodes of the highest security subjects, the network loses substantial amounts of efficiency for its lower-level subjects. Even worse, lower-level flows are often blocked, meaning no information flow can occur. This was the driving issue behind the development of MLSnet, where testing showed they were able to route 90+% of flows under normal conditions [7].

2.2 Problems Tackled

The policies innately defined and utilized by MLSnet tackle the core problems behind the specified networks configurations. As discussed in the original paper, MLSnet defends against a variety of common network attacks; packet spoofing, lateral movement, and man-in-the-middle attacks are topics of the original paper, which displays that not only can it defend an SDN from said attacks but can do so much more effectively than its peers. This layer of security provided is a necessity for having secure communication on network. The extension of such security into the wireless situations defined above would readily solve part of the security issue encountered.

Furthermore, they discuss the exact constraint model used for access control within the system. By following the guidelines described, an implementation of MLSnet in a wireless ad

hoc or mesh network would be able to achieve such access control, thus solving yet another of the core problems defined.

Finally, they discuss creating policy compliant flows, and in doing so describe that flows of elevated level (for instance level four) are given priority in being routed over lower-level flows. This stands to solve the issues of information priority. In the earlier case of a civilian and an EMS worker each making a call, the higher priority held by the EMS worker would ensure their call would be routed.

Paper [7] also introduces two variables for MLSnet's switch relabeling: M and B. Both M and B impact MLSnet's relaxed routing mode. Relaxed routing seeks to relax the strict rules to increase throughput. M represents the percentage of overall switches that can be relabeled in each relabeling period, set to 0.1 in the experiments described. Researchers introduced M to reduce the loss of throughput due to the need to reboot switches. On the other hand, B limits the "route down" values of flows. Route down refers to the ability for a higher-level flow to be run across switches of a lower level. B here seeks to allow for routing down to increase the throughput of higher-level nodes. At the same time, B seeks to limit the amount of routing down to minimize security risk.

Based on the description given of MLSnet in its original paper (and the enhancements defined in the follow up paper), MLSnet is a key candidate for enabling the use of MANets and WMNs. However, MLSnet does not come without drawbacks. While the base paper tackles its use in well-defined networks, the mobile ad hoc and mesh networks described are far from well-defined. Although the follow up paper tackles the use of MLSnet in unstable network conditions, the core factor of user mobility and wireless connectivity has yet to be fully explored.

2.3 Issues with Mobility

Even though MLSnet has been evaluated thoroughly on static and wired networks, the issues of user mobility in a wireless environment need to be addressed. Since user mobility creates a dynamic network environment, the state of the network loses predictability. The lack of definitive states opens the door for new issues that must be addressed for MLSnet's goal of multilevel security. The predominant issues with mobility come in two main forms: issues with relabeling and issues with connectivity.

One problem to tackle is when to relabel. If a lower security node moves to a higher security access point, should it relabel immediately? If that node relabels immediately, it might block existing active flows; if not the node will lack connectivity until the next relabeling occurs. This becomes more nuanced when considering a node of higher-level security. To use an earlier example, say an EMS worker is consulting a doctor for advice, and moves to another access point; should that access point be relabeled immediately to prioritize the integrity of a network connection with high importance? The fundamentally dynamic nature of mobility mixed with the nature of the relabeling periods creates a core issue.

Connectivity issues are an extension of the relabeling issues. When a node moves access points, it will relabel said access point to its security level. However, cases exist where this disadvantages existing nodes. If a maximum-security level node moves, it stands to throw off nodes of existing priority. These nodes may be left without connection, leaving their flows unable to complete. One solution would have them connect to other nearby access points after being thrown off, but this would require a higher density of access points than may be available. The last observation proves to be true in the cases described prior, where resources would already be scarce in an emergency or tactical military setting.

Chapter 3 Testbed

In this chapter we define our experimental testbed. We review the concepts surrounding a software-defined network and the SDN emulator Mininet plus its expansion Mininet-Wifi. Lastly, we review the POX controller technology, its components, and changes made to MLSnet for the experiment.

3.1 Software-Defined Network

SDN's are a recent progression in networking technology that centralizes the forwarding logic into a controller. The following description of SDNs is adapted from [2]. The controller used in this experiment will be discussed in detail later. The controller acts as the brain of the software-defined network, building an understanding of the network it observes and defining flows between users across the network. By having a wholesale understanding of the network (its individual pieces and their condition), the controller generates rules perpetuating or halting necessary data flows. These flexible rules are what makes SDN technology appealing, as they pertain to the lower level of the network hierarchy.

SDNs' utilize a unique dynamic between the controller and the switches; switches contain a flow table storing routing commands for packets. These commands base their route on header data within the packet. A benefit of SDN switches is their ability to route based on information from the link, network, and transport layers, earning themselves the name "packet switches" (per [2]). The flow table is the defining characteristic of a packet switch. Said table stores three key pieces of information: header info (as noted above), counters, and commands. Header information allows the switch to compare stored data with incoming packet headers.

Depending on the match, switches take a series of commands managing the forwarding of packets. These commands are known as flow rules. The switch also stores a set of information per flow; these include the number of packets routed by a flow rule and the last time said rule ran. Finally, the switch stores rules for a given flow. These commands can range from forwarding the packet to another connection via a port to dropping the packet entirely. As well, these flow rules can be dropped, altered, and timeout if they sit unused.

However, the relationship between the controller and the switches highlights the true function of the SDN. Refer again to the wireless mesh network showcased in figure 1.1. If host one wants to send information to host two, said information must travel through at least two switches. As host one sends packets out, they first travel through the first switch in the top left. When the switch receives the first packet, it checks the header section of its flow table (which is empty). Since the switch lacks any flows for the packet, it forwards the information to the controller. The controller takes in the packet information which uses understanding of the network to generate and return a rule to the switch. For example, a rule could be to forward packets within the flow to the bottom left switch. When the rule reaches the original switch, it updates its flow table and upon receiving the next set of packets, forwards them out of the correct port to the proper switch. When the bottom left switch receives these packets, it contains no flow rules. As such it repeats the steps of contacting the controller, eventually receiving the rule to forward the packets to host two.

Although the above describes a simple example of how an SDN works, it outlines the pivotal parts of said technology's value. The centralized nature of the controller allows for it to optimize and install dynamic packet forwarding rules. Said rulesets prove useful in the equally dynamic network scenarios of wireless networking and this experiment.

3.2 Mininet

To simulate the network for experimentation, enter Mininet: the self-dubbed “instant virtual network on your laptop.” More than just creating a virtualized network, Mininet allows for the definition of a Software-Defined Network (SDN), where the rules of traffic control can be installed, altered, and removed from any switch as described in section 3.1[5].

Mininet served to create the backbone of the original MLSnet experiment, allowing for the creation of a network testbed that could dynamically relabel switches with new security levels at regular intervals. However, for all its functionality, Mininet is constrained to wired networks. To expand the functionality of MLSnet into a wireless environment that supports mobility (and emulates all the issues faced in said environments), an entire additional layer of software on top of the base Mininet library is required. Enter Mininet Wi-fi.

3.3 Mininet Wi-fi

Mininet Wi-fi is an extension to the original Mininet emulator developed by Ramon Fontes; it maintains all the original functionality but adds wireless access points and stations as well as a slew of models for mobility, signal propagation, and more. These add-ons provide a logical area of extension in MLSnet where stations take the place of hosts and access points take the place of switches [8].

The use case of Mininet-Wifi is two-fold; on one hand Mininet-Wifi acts as the perfect software to implement and test the usability of MLSnet in a wireless enabled network, while on the other hand its baked in support for relative positioning, node mobility, and distance-based signal propagation means it also enables the emulation of user movement across several fields.

Alongside Mininet-Wifi, the parameters of network creation must be discussed as well. The network (independent of the topology) utilizes Open vSwitch styled access points running OpenFlow 1.0, a POX controller (which will be discussed in a later section) and a strongest signal first heuristic to determine which access point a host should connect to first. The propagation model used is a logarithmic distance model.

3.4 POX Controller

As discussed earlier, the core aspect of an SDN lies in the controller. The original MLSnet experiment takes advantage of the centrality of the controller, installing dynamic rules across the network as part of its flow optimization scheme [9]. While Mininet and its extension Mininet-Wifi both provide standardized controllers for basic network routing, they also define remote controllers. Remote controllers provide the capability of utilizing self-defined components for routing and tracking flows across the network. MLSnet hinges on such a controller, POX.

POX is a Python implementation of the Nox controller. Both are OpenFlow based controllers which provide and load subcomponent pieces that handle a range of network events. These include packet flow, connection creation, and connection destruction to name those important to MLSnet [6]. What is OpenFlow? [2] defines OpenFlow as, “a highly visible standard that has pioneered the notion of the match-plus-action forwarding abstraction and controllers, as well as the SDN revolution more generally.” Its key contribution is the flow table, discussed above. The matching of packets to their destination depends on the controllers view of

the network and several fields including the MAC address and port information of the packet source and destination [2].

Pox handily defines numerous stock components of which MLSnet augments and utilizes three: the OpenFlow spanning tree, the OpenFlow discovery model, and the forwarding l2 pairs module. The spanning tree implements the controllers view of the network topology and helps it eliminate packet loops (a vital contribution). The discovery model uses Link Layer Discovery Protocol Packets to traverse the underlying network topology and discover the links and connections between hosts and switches. Both previous components work with the controller directly on building an understanding of the networks and its nodes and links. L2 pairs works on switches to route information based on data link layer information (such as the earlier noted MAC address). L2 pairs does this in, "… just about the simplest possible way to do it correctly … install[ing] rules based purely on MAC addresses" [6]. MLSnet augments the latter two components (discovery and l2 pairs) in two keyways: discovery builds an internal representation of the topology that tracks security levels amongst other data, l2 pairs utilizes the underlying representation and invokes the relabeling and routing heuristics defined in the original MLSnet.

3.5 Changes to MLSnet

MLSnet's original implementation utilized an l2 pairs module as discussed in the previous section. It is important to note that POX's base l2 pairs lacks flow timeout rules and does not routinely clear out flow tables, as is done in a general SDN network. For this reason, l2 pairs mobility support lacks, which is shown further in the results section and the following example. The purely MAC address-based nature of the l2 pairs routing means Mininet-Wifi

encounters issues trying to resolve mobile nodes. For example, consider a simple topology of two hosts and three access points in a triangular configuration.

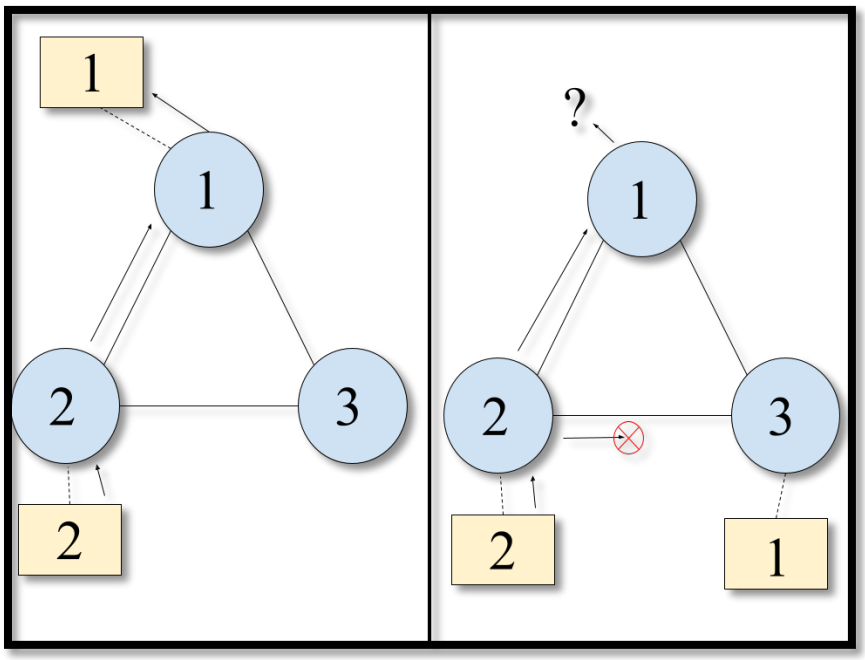

Figure 2: Example Mobility Issues

Say host one connected to access point one and talked to host two via access point two. Then l2 pairs would install a flow forwarding incoming packets on access point two from host two to switch one. However, if host one was to move from access point one to access point three, the previously defined flow would not change as host one has a constant MAC address. Access point one would continue to receive the packets and (with nothing else to do) drop them. This poses a significant issue in terms of expanding MLSnet to a mobile environment.

As such, one change made to MLSnet to support such mobility was the swap from l2 pairs logic to l2 learning logic during the warmup phase. The l2 learning component works like its l2 pairs sibling. “While this component learns L2 addresses, the flows it installs are exact-matches on as many fields as possible. For example, different TCP connections will result in different flows being installed” [6]. The increased depth on the matching of fields means that

when such mobility as earlier described occurs, the controller and access points consider the packets as part of an entirely new flow. In the case above, three steps would occur at access point 2: it would recognize the change in the location of host one, request a new flow from the controller, and begin properly forwarding packets to switch three. Although a minor change from the logical implementation side of the experiment, the code changes proved to be more in-depth but did indeed yield the necessary flexibility during the warmup phase of MLSnet.

Chapter 4

Results

In this chapter, we discuss the results of our various trials. Namely, we split our trials over three variables: with or without MLSnet, with or without node mobility, and with or without varied security levels. For each trial, we review one or more throughput graphs, highlighting how different events impact network performance.

4.1 Test Cases

To gather a variety of results, a series of trials were recorded with three changing variables: what controller components were used, what security levels are present in the topology, and whether mobility is enabled. Regardless of the trial, the underlying topology is always the same and can be seen in figure 3.

Another important aspect is the warmup period required for MLSnet to begin. After the controller (and MLSnet with it) alongside the network itself are instantiated, MLSnet invokes its discovery component which sends a series of link layer discovery protocol (LLDP) packets [1].

In the warmup period, MLSnet develops its internal representation of the network which is used when routing flows. However, the base pox controller components lack the necessity of this warmup time. As such, MLSnet utilizes an l2 learning style logic prior to warmup to support user mobility and discoverability, at the expense of throughput. After the warmup period is

complete, the flow routing done by MLSnet is an augmented l2 pairs logic, based solely on MAC addresses, where the mobility support is overseen by MLSnet.

Important to note are the essential time periods in which certain events occur. Flow epochs (of which there are five epochs per trial) are sets of data flows between hosts in the network and allow for crucial measurements of connectivity and throughput to be made. Within the code, in each epoch a ping command is run sending ICMP packets across the network. These are generated every 60 seconds and last for a duration of 110 seconds regardless of level. This means epochs overlap with one another with a slight disparity: as the nth flow ends the nth+2 flow starts ten seconds after. This difference leads to noticeable drops in throughput as overall traffic drops on the network. During the trials shown, each host in the network talks to every other host, with the flow level depending on the security level of the hosts involved. Of note, MLSnet runs a version of the Bell La-Padula model described in [12]. This model allows for hosts of a lower level to send and receive data with hosts of a higher level, provided the data sent is accessible to the lower-level host. As such, bidirectional communication exists between all hosts in an epoch.

Network relabeling occurs in 100 second intervals. At these times, the controller evaluates the state of the network and its flows. If flows are currently unable to be routed (i.e., are blocked), it will seek to reconfigure the network to allow for the maximum number of flows to be routed. As well, if flows enter or exit the network, the controller will seek to accommodate both the new and existing flows. This is the core behavior introduced in MLSnet and a deeper explanation of its mechanisms can be found in [7] and [9].

Host mobility occurs at 90 seconds and 210 seconds. As discussed later in section 4.4, host h0 is connected to access point four and at its initial position x=0, y=25 from time 0 (the

beginning of the first epoch) until time 90 (halfway through the generation period of the third epoch). It is at time 90 that h0 swaps positions and by extension access points. From 90 to 210, h0 is at position x=20, y=20 and connects to access point five. At time 210, h0 again swaps positions and access points, now sitting at x=45, y=20 and connecting to access point six. Due to the implementation of how MLSnet manages mobility, any given host is limited to moving up to one access point away in each relabeling period.

Since MLSnet and the network itself are independent components, the above time periods can be disjoint of one another. Mobility acts in tandem with the network code, and as such always occurs 90 seconds after the first flow is generated. The controller sits separate from the network code and manages relabeling. Relabeling occurs in 110 iterations after the warmup period for MLSnet completes. The completion of the warmup and the beginning of flow generation are meant to start coordinated with one another, but due to some human overhead can

be desynced by several seconds. Due to the nature of that desync, the relationship between all the above events is constant, but data sees a difference of several seconds for certain trials.

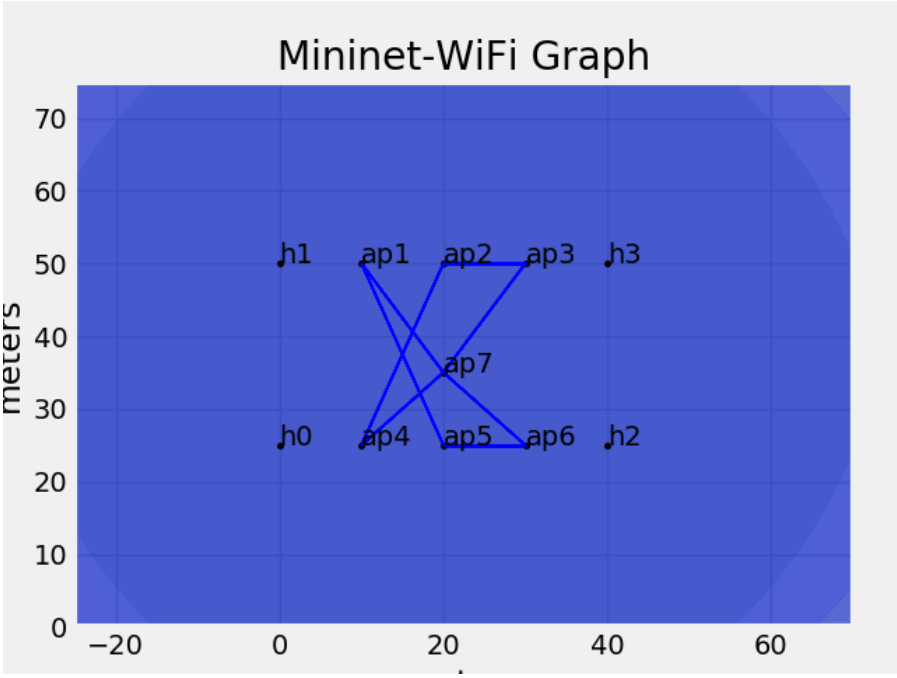

Figure 3: Standard Topology

4.2 Control Cases

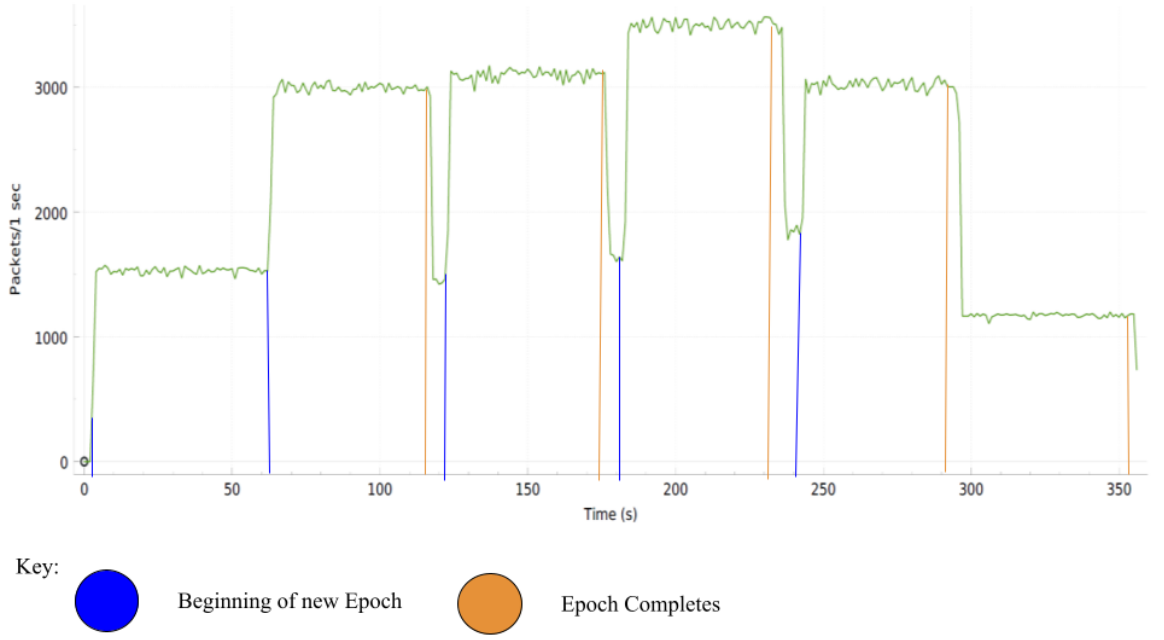

Figure 4: Base L2 Pairs w/o Mobility

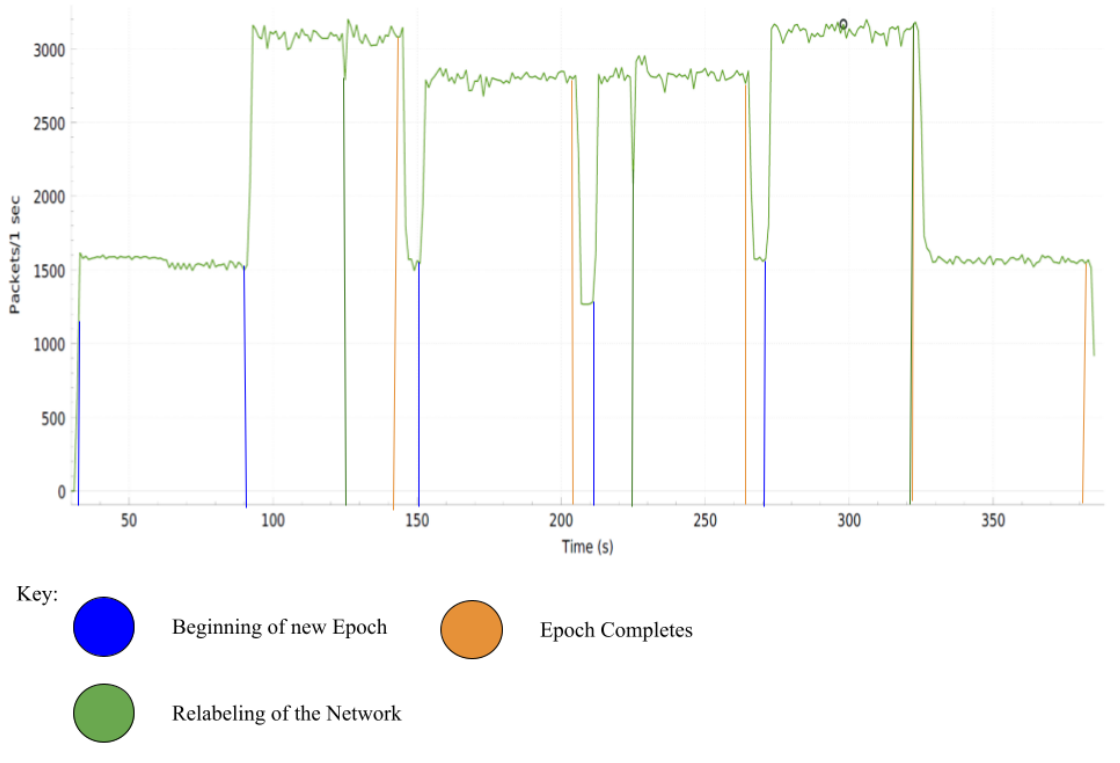

Figure 5: Base MLSnet Level 4 w/o Mobility

In the initial trial (shown in figure 4), the topology shown in figure 3 is instantiated with a POX controller running the L2 pairs component. During the base line MLSnet trial (shown in figure 5), the same topology is instantiated with a POX controller running MLSnet. During the duration of both trials, no hosts move. As marked in figure 4, L2 pairs sees a surge in throughput at times (in seconds) 0, 60, 120, 180, and 240. The dips seen around times 110, 170, 230, 290, and 350 are marked by the end of one flow epoch and the beginning of another (excluding time 350). The gap between the end of one epoch and the start of the next leads to momentary dips in throughput, with a duration of 10 seconds. The behavior of the network throughput is observed and explainable given the parameters of the experiment. Figure 4 acts as an ideal comparison point for trials involving MLSnet that also lack mobility, keeping in mind that the flows in said figure lack the guarantee of the multilevel security ensured by MLSnet.

Turning attention to figure 5, the introduction of MLSnet introduces a new event: node relabeling. As explained in Chapter 2, MLSnet adds labels to each node in the network corresponding to the security clearance assigned. By managing these nodes, their labels, and the flows between, MLSnet assures multilevel security of its users and the flows they send. By relabeling the graph (a data structure tracking the state of the network), MLSnet searches for labeling schemas that minimizes blocked flows. This relabeling takes an understandable amount of overhead, and even in static topologies (where nodes are locked in place and security levels are constant among all nodes) MLSnet verifies the state of the network during the relabeling.

Adjusting for the first flow being introduced at time 30 seconds, figure 5 once again sees surges in throughput at times 30, 90, 150, 210, and 270, as well as dips in throughput at times 140, 200, 260, 320, and 380. This is consistent with the data in figure 4. Also marked are instances of relabeling, occurring at 120, 220, and 320, respectively. Seen at these instances are

small dips immediately followed by climbs as MLSnet clears the flows, before revalidating said flows and continuing to forward packets as before.

4.3 Split Level Case

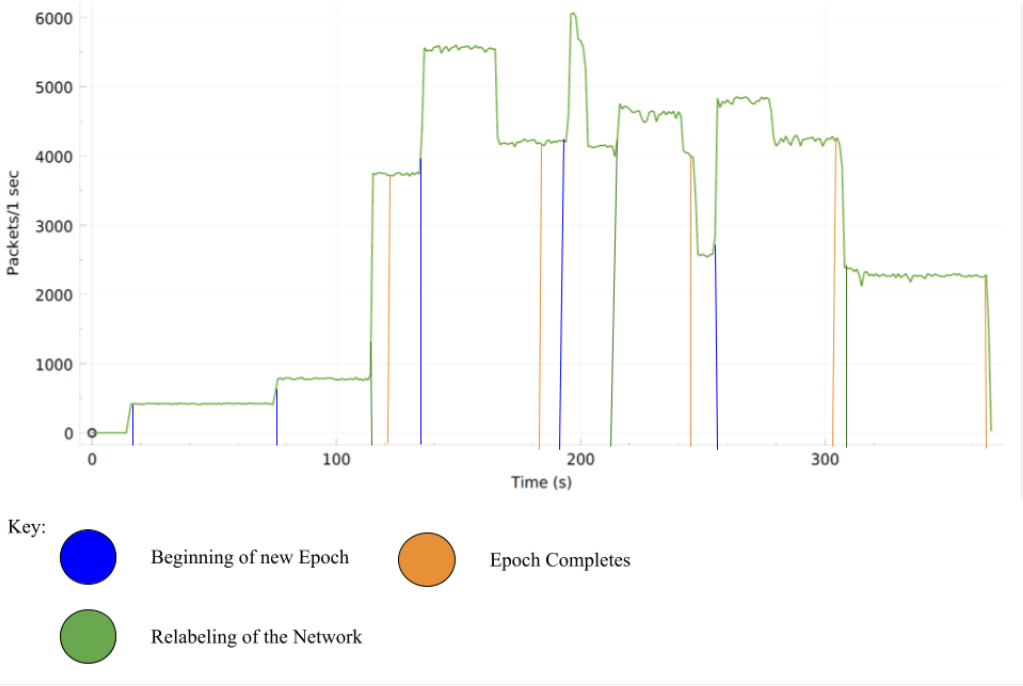

Figure 6: H0 Split Level 1

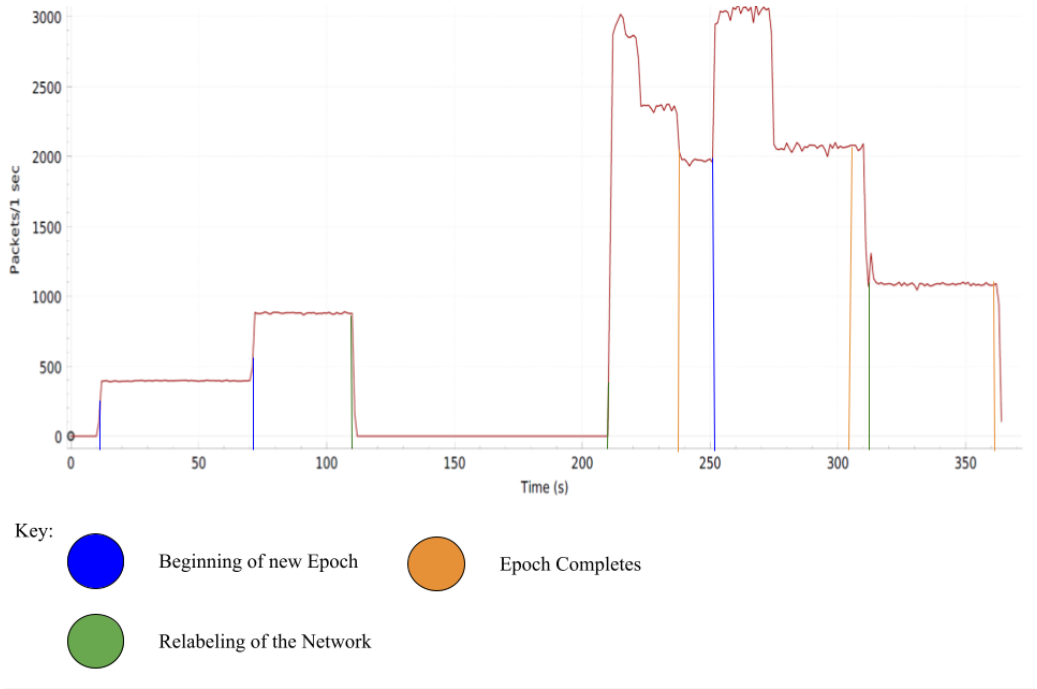

Figure 7: H2 Split Level 4

Figures 6 and 7 introduce a more robust case; the hosts h0 and h1 are labeled with level four security (the maximum level defined within our experiment) while the internal switches and other two hosts (h2 and h3) are labeled with level one security (the minimum level). The other two concepts brought in from MLSnet are blocked flows and waiting flows. In the previous examples, due to the homogenous nature of the topology and security levels, no flows were unable to be routed. As flows are unblocked, the both the receiving and sending hosts experience a surge as packets previously unable to be routed are sent in succession. In this experiment the values M and B discussed prior in 2.2 are set to 0.5 and 5. With seven switches in our topology and an M value of 0.5, this means three switches can be relabeled per period. To delve further into the B value for our test, it means the cumulative difference between the security level of a flow's host and the security level of each node the flow traverses must be no greater than B. For instance, prior to any relabeling, the flow h0 to h3 would traverse access points 4, 7, and 3,

which all have a security level of one. Therefore, the total difference for this flow is nine and cannot be routed as a result.

With these additional concepts in mind, we note that upon the initial period six flows are active and six flows are blocked. The active flows are the two flows between h2 and h3, and the outgoing flows from h2 and h3 to h0 and h1. The blocked flows are all outgoing flows on both h0 and h1. Given the understanding of the topology, this outcome proves expected; the level four flows between h0 and h1 are unable to be routed over the level one switches in the topology and (as described above) flows going from h0 and h1 to h2 and h3 cannot be routed down without exceeding B. Looking at the first period (10-110 seconds) we see dramatically reduced throughput compared to earlier sections.

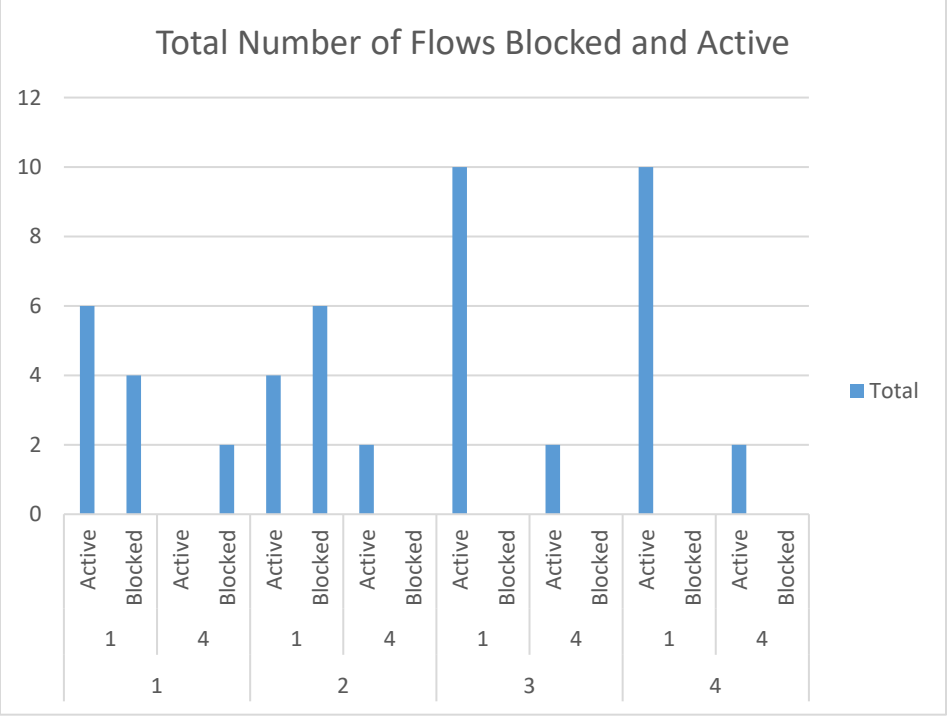

Figure 8: Flow Status by Security By Time Period over the Experiment

Figure 8 displays the number of active and blocked flows based on their security level during a distinct relabeling period. As seen, in the shift from period one (which takes place prior

to the first relabeling call) to period two, the level 4 flows are unblocked. Figure 6 reflects this as well, with throughput climbing rapidly after the first instance of relabeling. On the other side, level one flows see an uptick in blockage, reflected in the drop of throughput showcased in figure 7 at the same point. H2 continues to see zero throughput over the course of period two as its flows are blocked. Once the shift from period two to period three occurs (and relabeling along with it), h2 sees a surge in throughput as its flows become unblocked. Following the rapid climbs and declines in the second period, both graphs stabilize (outside of momentary dips due to flow generation and completion) and return to a form like those showcased in figures 4 and 5. This is once again supported by the above graph, which showcases that after the third relabeling occurs, all the flows reached an active state. The period and stability that marks period three also continues into period four and unto the completion of the experimental trial.

4.4 Base Mobility Cases

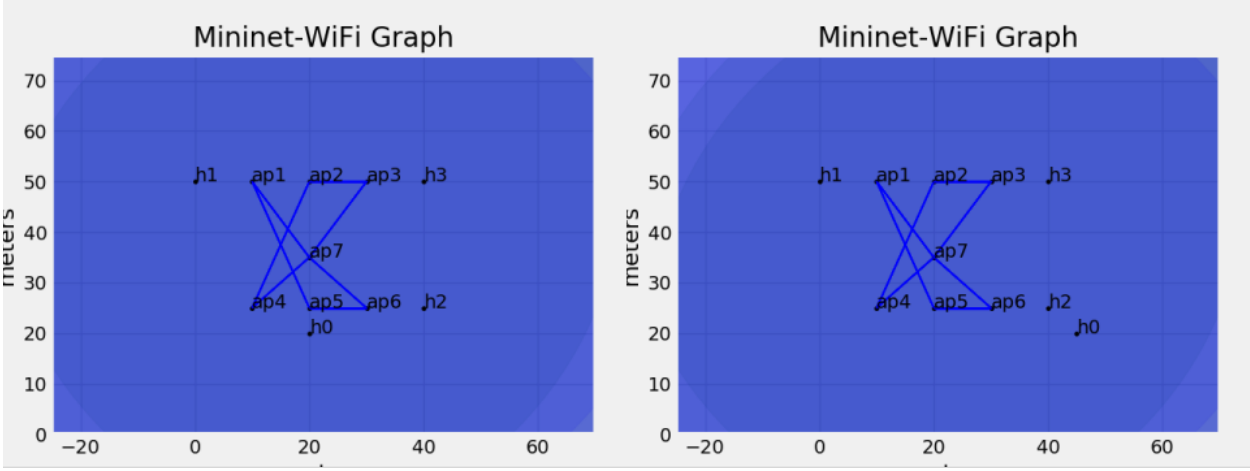

Figure 9: Intermediate Position of h0 (left) and Final Position of h0 (right)

The final cases tackle trials with user mobility, a fundamental part of wireless networking, as the node is handed off between access points. This leads to the introduction of a

new event that affects throughput: node mobility. As discussed in the example in section 3.5, node mobility can cause serious interruptions within a network. For all trials discussed, the node h0 (as pictured above in figure 9) moves discretely to positions x = 20, y= 20 and x=45, y= 20 at times 90 and 210, respectively.

Figure 10 outlines the base issues with mobility in a network without MLSnet. The below trial used the base pox defined l2 pairs module. One detriment of this module is its flows are not regularly cleared. Like the example given in section 3.5, the flows prior to node mobility are still intact after h0 moves. Since the routes are based solely on h0's mac address and the other switches have no way of knowing that h0 moved, they continue to route information from h0 and to h0 as if it was connected to ap4. As seen, throughput plummets when h0 initially moves and values remain at zero. When movement occurs again at 210, throughput climbs back up, but without the consistent high values shown prior to the mobility event. Since h0 and h2 are both connected to access point six, the data between the two is routed while flows between h0, h1 and h3 continue to fail, hence the increase. Over the remainder of the trial, throughput slowly tapers off as expected.

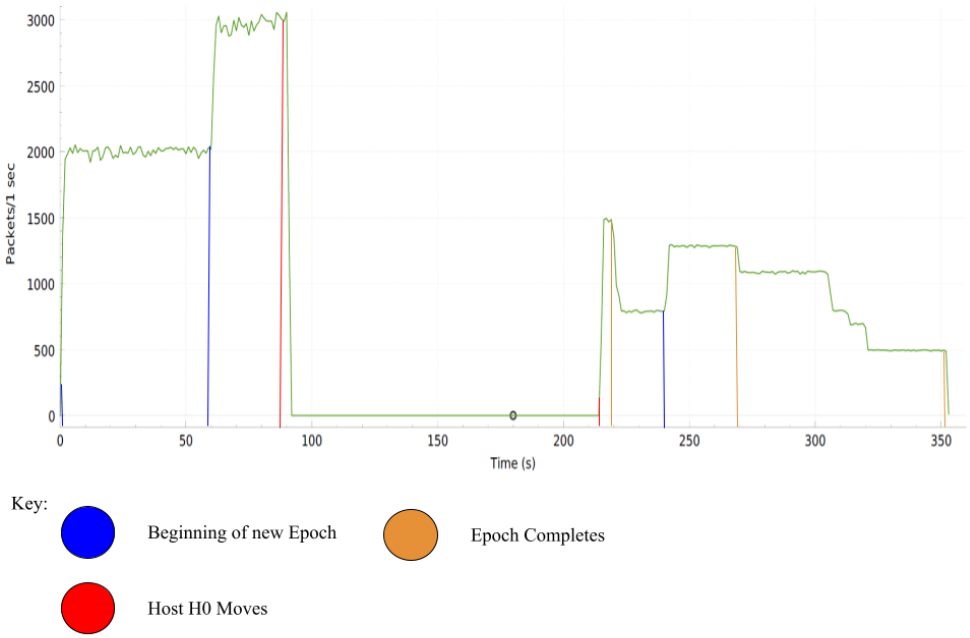

Figure 10: Base L2 Pairs w/ mobility

As evidenced by figure 10, MLSnet must solve a genuine issue with node mobility, on top of continuing its reliability and assurance of multilevel security. Therefore, the adaptation to l2 learning was made for the warm-up phase as discussed in section 3.5. Its deeper checks on flow rules mean the switches identify a flow coming from h0 via port three on ap4 as fundamentally different than one from h0 via port 2 on ap5 and route them accordingly. MLSnet solves this issue by clearing flows regularly (at every relabeling period) and establishing a controller wide understanding of node location through its data structure. By updating the said data structure and refreshing flows at each relabeling period, it maintains the connectivity of host.

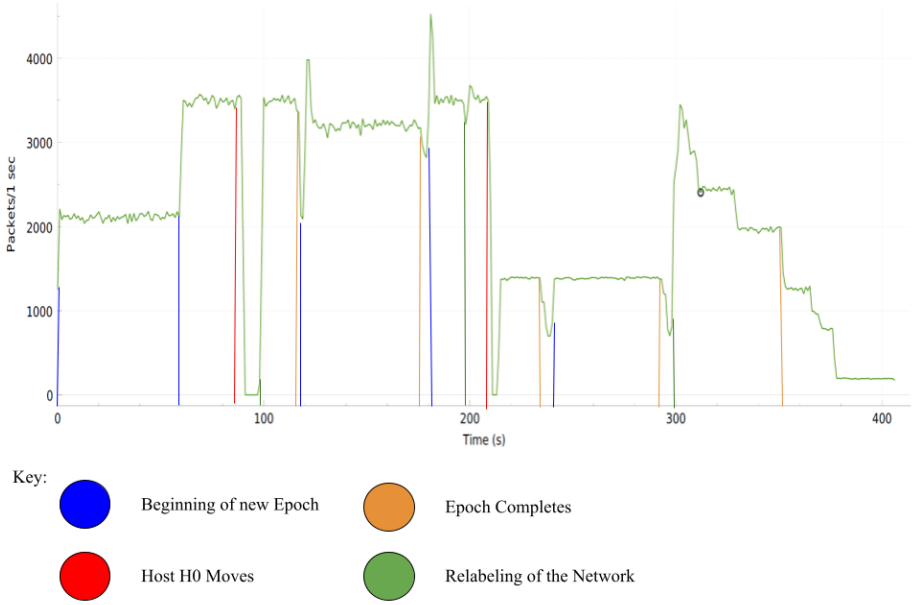

Figure 11: Base MLSnet w/ mobility Level 1

Figure 11 displays the control case of MLSnet described in section 4.2 but with the addition of node mobility. Up until the initial instance of node mobility, the graph mirrors figure 5. However, like the output shown in figure 10, throughput plummets as h0 moves from access point four to access point five. Unlike the L2 pairs trial, throughput returns quickly when MLSnet relabels the topology. One of the additions that was made in bringing MLSnet to a wireless environment was the flagging of mobile nodes being handed off between access points. When these handoffs occur, MLSnet flags the node and stores the new data, updating the underlying topology in tandem with relabeling the data structure. As result, the throughput of h0 sees the earlier noted rapid return. Over the period from time 100 until 210, the throughput graph mirrors its non-mobile counterpart in figure 5. At 210, h0 moves again and once again throughput plummets. Like figure 10, throughput steadily climbs back as packets between h0 and h2 can be routed in a single hop across their shared node. It is not until relabeling occurs again at time 300 that h0 sees a full return in throughput, before tapering off as epochs complete. It is

important to note that should h0 have not shared an access point with h2 in the period between times 210 and 300 (and instead were isolated on its own access point), the dip seen would be akin to the period 100 to 210 in figure 10.

4.5 Split Level Mobility

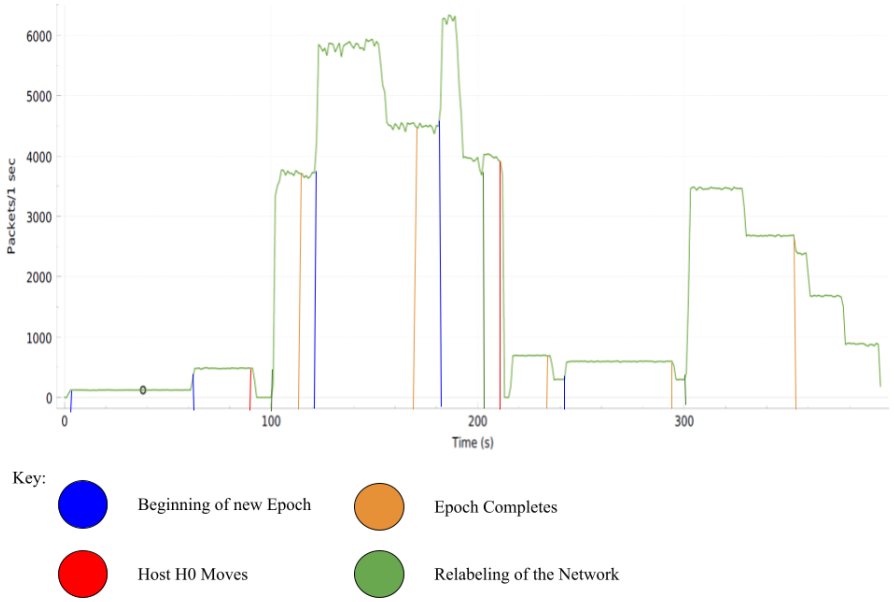

Figure 12: H0 Split Level w/ Mobility Level 4

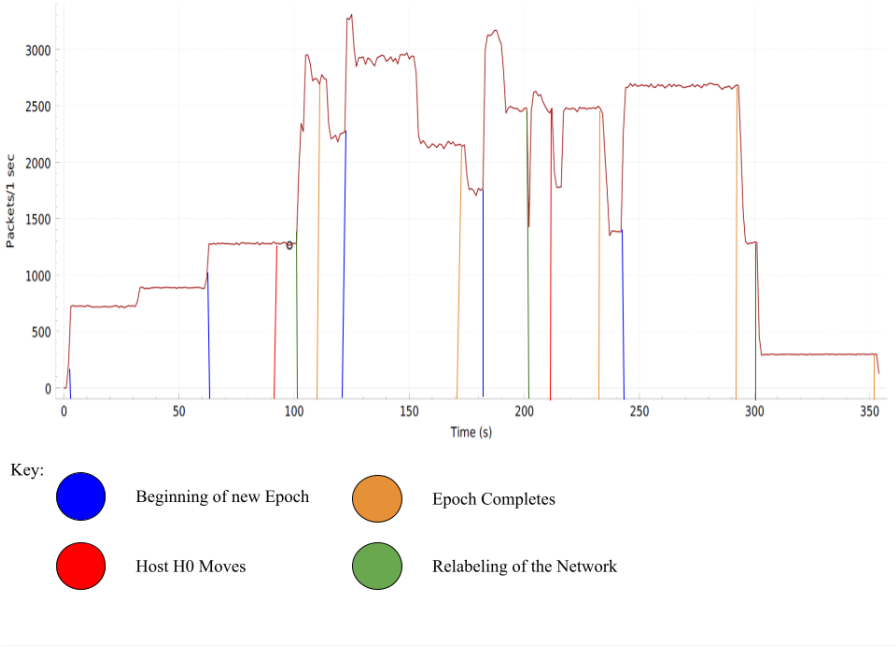

Figure 13: H2 Split Level w/ Mobility Level 1

Figures 12 and 13 show the scenario described in section 4.3 with the added layer of node mobility described in section 4.4. When combined with the blocked and active flow information in figure 14, the figures show a much different outcome than their counterparts in figures 6 and 7. Their beginnings (prior to node movement and relabeling) are like that of the previous trial, where h0 is initially blocked and sees reduced throughput as a result.

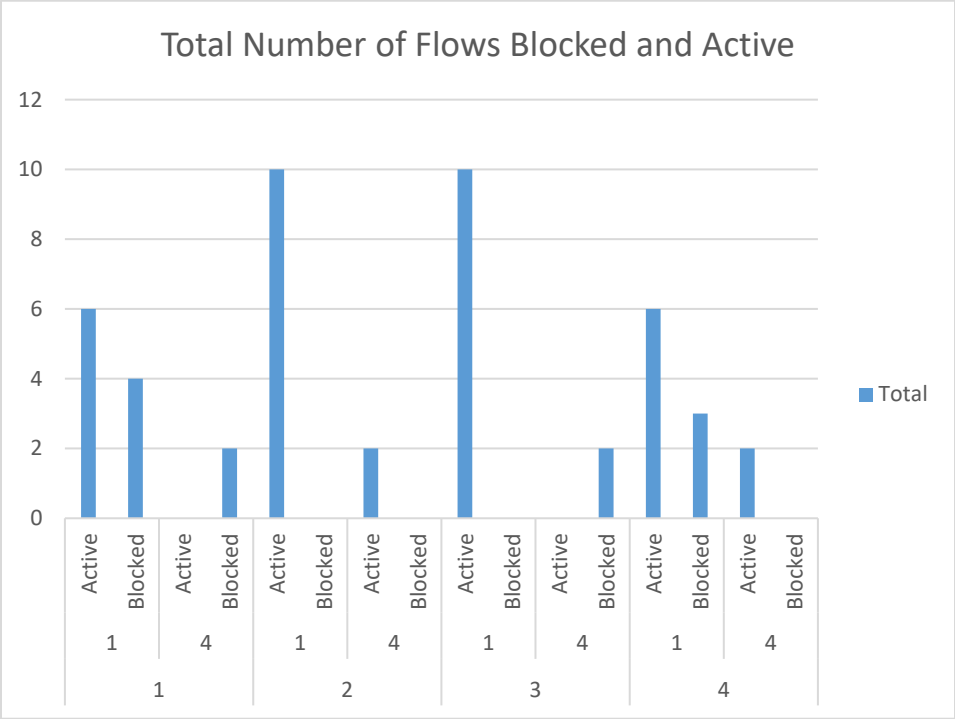

Figure 14: Flow Status by Security by Time Period over the Mobility Experiment

Looking at period two (from time 100 to 200), the flows are all active as MLSnet can more easily route the level four flows given h0's new position attached at access point five. Both graphs show their increased throughput with the usual dips and surges for epoch generation and completion, as well as relabeling causing overhead. H0 sees a particular series of surges as flows that were previously blocked due to its mobility are now rerouted instead. Period 3 (from time 200 to 300) sees the two level four flows between h0 and h1 again blocked, resulting in reduced throughput on h0. The throughput was already reduced due to the underlying data structure having not yet been updated. However, once the third instance of relabeling occurred, interesting behavior can be seen in both h0 and h2. The controller once again routed the level four flows, and the mobility update in the controller sees the expected surge in throughput on h0. This change in place causes an amount of blocking in the level one flows. The blocked flows currently were flows from hosts h0, h1, and h3 to h2. The movement of h0 onto access point six

caused the access point to elevate to security level four. Because of this, the other nodes in the system are unable to communicate with h2. As a result, the throughput of h2 after this event plummets, with the remaining throughput stemming from tiny amounts of connection with h0 through the shared access point.

This case was discussed earlier in chapter 2.3 with the possibility of lower-level nodes being kicked off an access point trying to accommodate a higher-level peer. Interestingly, the MLSnet controller stops tracking the flow from h2 to h0, and the fourth period only notes eleven total flows. Since the underlying experiment code is using a ping command to generate flows, packets are still routed by the access point. This is due to how pings are managed under the hood. As result, the results of this trial are expected.

Chapter 5

Conclusion

In this chapter we review the results in chapter 4 and draw conclusions on the outcome. As well, we discuss possibilities for future works while recognizing limitations in our own experiment.

5.1 Conclusion

In conclusion, the expansion of MLSnet into wireless networks sees it maintain its assurance of multilevel security and optimal flow routing in a new environment. At the same time, MLSnet builds in mobile routing that lacked in stock components. The results shown in the previous section show that MLSnet continued to employ its policies effectively along wireless connections. During node movement throughput took a serious hit, until MLSnet was able to provide and install new flows as it relabeled the graph. The consistency of the throughput across all cases shows that MLSnet can compete with the straightforward throughput managing methods while adding complexity at the cost of small performance hits. As well, figures 8 and 14 show that MLSnet reduces the blockage across networks effectively while providing priority to higher security flows.

5.2 Future Works and Limitations

Although this paper acts as a starting point for adapting the multilevel security enforcement protocol behind MLSnet to a wireless network, it undoubtedly met several limitations. Changes to the experimental parameters are the source of many opportunities for future works. One such change would be testing the wireless algorithm on larger topologies, as all trials were performed on a small-scale network. Another opportunity would be to trial continuous mobility models such as random walk. The model used in the trials saw discrete cases of movement rather than continuous movement. Finally, having multiple mobile end user nodes move, in specific those of varied levels, would yield itself to another research project.

Opportunities to improve performance are also areas for future works. For instance, while this experiment continued to trigger relabeling at timed instances only, invoking said algorithm in tandem with mobility would give the opportunity for nodes to communicate at the cost of other hits to performance. An adaptation where low-level nodes forced off an access point by higher level nodes (like h2 in the split-level mobility trial) are forcibly handed off to nearby lower-level access points could also see improvements in overall network performance. One final avenue would be to increase the number of paths between nodes (such as in a full mesh network) to prevent blocking due to mobility.

BIBLIOGRAPHY

1. "IEEE Standard for Local and metropolitan area networks - Station and Media Access Control Connectivity Discovery," in IEEE Std 802.1AB-2016 (Revision of IEEE Std 802.1AB-2009), vol., no., pp.1-146, 11 March 2016, doi: 10.1109/IEEESTD.2016.7433915.
2. J. F. Kurose and K. W. Ross, *Computer networking: A top-down approach*. Harlow, United Kingdom: Pearson, 2022.
3. J. Kim et al., "Development of Mobile Ad Hoc Network for Emergency Telemedicine Service in Disaster Areas," 2009 International Conference on New Trends in Information and Service Science, Beijing, China, 2009, pp. 1291-1296, doi: 10.1109/NISS.2009.242.
4. J. L. Burbank, P. F. Chimento, B. K. Haberman and W. T. Kasch, "Key Challenges of Military Tactical Networking and the Elusive Promise of MANET Technology," in IEEE Communications Magazine, vol. 44, no. 11, pp. 39-45, November 2006, doi: 10.1109/COM-M.2006.248156.
5. M. P. Contributors, *Mininet*. [Online]. Available: http://mininet.org/. [Accessed: 19-Mar-2023].
6. “Pox - The POX Network Software Platform.” The POX Network Software Platform, Gar, Noxrepo, https://noxrepo.github.io/pox-doc/html/. Accessed 19 Mar. 2023.
7. Q. Burke et al., "Enforcing Multilevel Security Policies in Unstable Networks," in IEEE Transactions on Network and Service Management, vol. 19, no. 3, pp. 2349-2365, Sept. 2022, doi: 10.1109/TNSM.2022.3176820.
8. Ramonfontes, “Ramonfontes/MN-WIFI-ebook,” *GitHub*. [Online]. Available: https://github.com/ramonfontes/mn-wifi-ebook. [Accessed: 19-Mar-2023].
9. S. Achleitner, Q. Burke, P. McDaniel, T. Jaeger, T. L. Porta and S. Krishnamurthy, "MLSNet: A Policy Complying Multilevel Security Framework for Software Defined Networking," in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 729-744, March 2021, doi: 10.1109/TNSM.2020.3045998.
10. S. Sarika, A. Pravin, A. Vijayakumar, and K. Selvamani, “Security issues in mobile ad hoc networks,” *Procedia Computer Science*, vol. 92, pp. 329–335, Aug. 2016.

11. W. -P. Lu and M. K. Sundareshan, "A model for multilevel security in computer networks," in IEEE Transactions on Software Engineering, vol. 16, no. 6, pp. 647-659, June 1990, doi: 10.1109/32.55093.
12. D. E. Bell and L. J. LaPadula, “Secure computer system: Unified exposition and Multics interpretation,” Tech. Rep. ESD-TR-75-306, 1976.

ACADEMIC VITA
Kenton Hertzog

EDUCATION

The Pennsylvania State University – Schreyer Honors College — **University Park, PA**
College of Engineering| B.S. in Computer Science, Minor in German Language — *Class of 2023*
Honors: Eta Kappa Nu (HKN) IEEE Honors Society

Warwick High School | Class of 2019
Lititz, PA

RELEVANT COURSEWORK

Introduction to Systems Programming	Object Oriented Programming with Web-Based Applications
Programming Language Concepts	Discrete Mathematics for Computer Science
Operating Systems	Data Structures and Algorithms
Database Management Systems	Intro to Theory of Computation
Fundamental Computer Graphics	Applications Programming
Honors English: Technical Writing	Artificial Intelligence

RELEVANT SKILLS

Java	C/C++
Python	SQL and Databases
JavaScript and jQuery	HTML
Linux Experience	Git
Swift and SwiftUI	Microsoft Azure
Racket/Scheme	EC2 in AWS
Leadership/Public Speaking	Agile Sprint
German Language Proficiency	Technical Writing

RELEVANT EXPERIENCE

Minitab, LLC — **State College, PA**
Software Engineering Intern — *May 2022—September 2022*

- Developed new features for Minitab Statistical Software
- Reviewed personal and peer code and pull requests
- Attended and contributed to daily standups
- Tested, debugged, and resolved issues within code additions

PSU Esports — **University Park, PA**
Penn State's Esports-oriented Student Run Organization — *October 2021—Present*
Website Manager

- Utilize JavaScript libraries, most notable Bootstrap.js
- Update website to reflect current events and sponsors
- Develop new sections, including sections for two newly integrated divisions
- Inherited existing codebase and adapted as needed for updated content

LEADERSHIP EXPERIENCE

The GLOBE — **University Park, PA**
Close-knit community of globally minded individuals who desire to foster global awareness and an international perspective — *April 2020—April 2021*
President

- Planned programming for over fifty members of club with four peers
- Fostered engagement, global perspective, community, and mutual respect
- Adapted unique events given covid restrictions in Club's only year without in-person events
  Communicated clearly via agendas, emails, and constitutional revisions to provide clarity and simplicity