Detection of a Rogue Switch in a Local Area Network

Vijay Bhuse, Andrew Kalafut and Lisa Dohn
School of Computing and Information Systems
Grand Valley State University
Allendale, MI, USA
email: {bhusevij, kalafuta and dohnl}@gvsu.edu

***Abstract*—There are many solutions available for detecting a rogue wireless access point, but none exists for detecting a rogue switch or a router in a wired environment. This is at least in part due to the implicit assumption that a wired environment is safer. In this paper, we present three solutions to detect a rogue Ethernet switch. Each of these solutions has its advantages and may be used independently of the others. We prove that our approach is practical and feasible by simulation and analysis of our solutions.**

Keywords—rogue switch; detection; network; security; LAN.

I. INTRODUCTION

The concern of potentially compromised switches and routers being installed in a network is significant enough that the United States House of Representatives Intelligence Committee investigated network hardware suppliers over the possibility that their routers could have backdoors allowing them to be compromised [1]. This risk is not only limited to specific suppliers. A recent vulnerability was found in Cisco routers that could allow an attacker to execute malicious code on the vulnerable network equipment [2].

A rogue switch is a switch that is connected to the network without the authorization of the network owner or a network administrator. Rogue switches are a threat to the security of a network. They create a very serious vulnerability, which can be exploited by an attacker to spy on the business, government or military installations. Existing solutions for preventing rogue devices on networks include the IEEE 802.1X [3] protocol (which provides an authentication system for devices requesting to connect to the network) and port security (which helps limit the devices that can connect to a switch and transmit frames). However, prevention methods may not always be successful due to a variety of reasons including lack of availability or misconfiguration. To provide defense in depth, it is extremely important to add a second line of defense by designing solutions that detect rogue switches. New techniques are needed in order to address this problem.

A rogue switch could be introduced to the network in one of the following ways.

- A compromised employee may connect their own *rogue* switch to the network and connect rogue hosts to it and can then use these hosts to launch attacks on the network.
- "Bring your own device" (BYOD) policies are also making it easier for employees to inadvertently introduce a rogue device into the network. A non-malicious user (such as, an employee at a business who prefers to use his own devices) might connect a switch or a host to the existing network, which may not have preventative security measures implemented. This switch or hosts might be insecure and may provide an avenue for hackers to access the network resources.

A rogue switch can be added only by connecting an Ethernet cable from it to an existing port of the valid switch. Rogue switches can threaten the confidentiality of messages on the network, degrade the performance of the network, or even allow unauthorized access to the network.

Adding a rogue router is much more difficult for the attacker than adding a rogue switch, as adding the router requires changes to the configuration of the valid router and need to support one or more Internet Protocol (IP) address subnets. We focus only on the problem of detecting a rogue switch in this paper, not a rogue router.

There are many solutions currently available for detecting a rogue wireless access point, but none exists for detecting a rogue switch or a rogue router in a wired environment. This is because of the *implicit* assumption that a wired environment is safer. However, if left undetected, a rogue switch can cause a number of security issues on the network.

As networks grow in both size and complexity, the detection of these rogue devices becomes even more difficult. The evidence of compromise is hard to detect because it is buried inside the traffic and complexity of these large networks. We discuss three specific solutions in this paper to detect a rogue Ethernet switch.

The rest of the paper is organized as follows. Section 2 covers related work. Section 3 covers problem definition. Section 4 discusses and analyzes our detection solutions in detail, and we conclude in Section 5.

II. RELATED WORK

There are many existing solutions to detect a rogue wireless access point [4]. Wireless traffic analysis monitors all traffic on the network to determine if any packets are suspicious. Site survey software is also available to assist with security analysis. Software tools, such as NetSpot [5] assist with the detection of unauthorized devices, traffic interference, and rogue wireless access points to the network. These solutions are not applicable to wired networks.

There is a slight possibility that the rogue switch can be detected through an IP sweep tool (such as network mapper

 ISBN: 978-1-61208-729-0

tool Nmap [6]), if information is captured about a host connected to a rogue switch (wired). These tools investigate the entire network and alert of any abnormality, at which point the task of analyzing the results and locating the physical whereabouts of the switch falls on the network administrator. However, this is no simple task, and it grows exponentially more complex as the network size increases.

The IP address of the host that is connected to the rogue switch is not singled out using the IP sweeping tool, so an investigative work is required in order for the network administrator to begin to track down the rogue switch.

There are a number of issues that complicate the process of finding a solution for rogue switch detection on wired networks. Some of these include:

- Unmanaged Layer 2 switches are not commonly traceable, because they do not have IP addresses [7].
- Neighbor discovery protocols are not typically supported on unmanaged switches. Therefore, connectivity information of neighboring switches will not be reported [8].
- Detection of a rogue switch does not necessarily reveal its physical location, which, in turn, is a completely separate and a difficult task.

All of the above issues make the task of detecting a rogue switch extremely difficult for a network administrator.

III. PROBLEM DEFINITION

The network being analyzed and tested will be similar to one which may be used by a small-to-medium size business, where all computers (used by employees) are part of the intranet Local Area Network (LAN) and are connected using Ethernet switches.

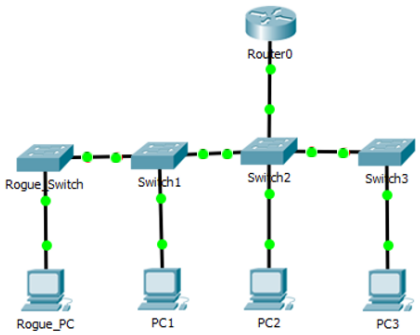

Fig. 1. Rogue switch connected to a normal network.

Figure 1 shows an example topology where a rogue switch is connected to a network. In this example, there are three switches (namely Switch1, Switch2 and Switch3), each with a single PC device connected, and a router (Router0) connected to one of the three valid switches. The rogue switch, Rogue_Switch, is connected to an open slot on one of the valid network switches (Switch 1), and there is a single rogue host (Rogue_PC) connected to the rogue switch. In this example, Rogue_Switch and Rogue_PC would not have permission to be connected to the rest of the network, though are connected anyway.

In all rogue switch cases, the rogue device is physically connected to a valid network switch without permission. This is a problem since network traffic can possibly be intercepted by hackers, who may use that information to gain access to the system and potentially use this access to steal sensitive information. Scenarios involving rogue switches can range from hackers attempting to gain access to the network, to non-malicious employees hoping to use their own devices either to gain more convenient network access or to simply use their own hardware. These devices may not be as secure and may provide an easy vulnerability for hackers to exploit. If left undetected, rogue switches or routers can cause many security issues on a network.

IV. DETECTION SOLUTIONS

We propose three unique, realistic, and independent solutions for detecting a rogue switch that is already on the network. Our solutions are based on using collected evidence for detecting *anomalies* using (a) Dynamic Host Control Protocol (DHCP) request messages, (b) Simple Network Management Protocol (SNMP) and Address Resolution Protocol (ARP) broadcast traffic detection and (c) Media Access Control (MAC) address whitelisting. These solutions act as a second line of defense that can be implemented alongside typical prevention measures. We validated our solutions by using Cisco Packet Tracker [9], a simulator that supports an extremely realistic simulation of IP networks using routers, switches and hosts. Our solutions work for routers and switches from any manufacturer or hosts with any operating system.

A. DHCP Request Message Detection

In many networks IP addresses are assigned DHCP. We can use the observed behavior of DHCP when hosts become disconnected and reconnected as a way of detecting rogue switches. This solution requires the ability to temporarily shut down a switch port connecting a switch to a host to determine if a rogue switch exists in between the two. This solution will not work on networks that assign addresses statically or otherwise avoid DHCP, but DHCP use is quite common.

Figure 2 displays an example of a "normal" network (one with no rogue devices) whereas Figure 3 shows a similar network where a rogue switch is present. We configured LAN interface Gi0/1 of Router0 and configured a DHCP server to run on Router0. PC0 receives its IP address, subnet mask and default gateway configuration from the DHCP server running on Router0.

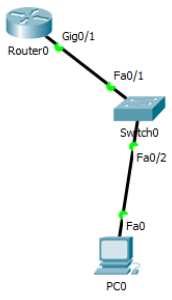

Fig. 2. Normal Network (No Rogue Switch).

In the network from Figure 2, the Fa0/2 port on Switch0 could be temporarily shut down and then brought back up, while we monitor the Fa0 interface on PC0. This process can either be automated or manual. When the switch port shuts down and comes back up, the host logs should indicate a change in the state of interface of the PC. If the interface state did not change, this indicates that the link between the host and the switch also includes one or more devices in between the two. This is because the host did not lose connection to the device it is connected to (a rogue switch, for example), so the interface retains the state.

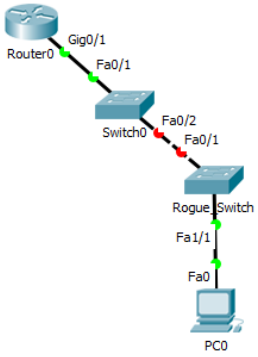

Fig. 3. Network With Rogue Switch.

Once the link is re-established, a DHCP request is triggered by PC0 if there is no rogue switch between the PC and a valid switch. There is no DHCP request if a rogue switch is present. In Figure 3, there is a rogue switch (titled Rogue_Switch) between Switch0 and PC0. When Fa0/2 port of Switch0 is shut down and brought up, the link between PC0 and Rogue_Switch retains the state. When the link between Switch0 and Rogue_Switch is disconnected, the interface Fa0 of PC0 will remain up the entire time and will not trigger DHCP request from PC0. Upon re-establishment of the link between Switch0 and Rogue_Switch, PC0 has no need to send another DHCP message. The missing DHCP message is an indicator of the presence of a rogue switch.

This detection mechanism can be circumvented if the rogue switch has ability to generate DHCP request messages and make it look like it came from PC0. In this case the switch can send the request when it becomes reconnected, as PC0 would have it the switch was not present. It can also be circumvented if the rogue switch disconnects PC0 whenever the rogue switch becomes disconnected from Switch0. However, both of these circumventions rely on non-standard behavior from the rogue switch and therefore increase the effort required from the attacker to evade detection.

In a large network, this process can be automated. The system will cycle through each switch directly connected to a host, and one by one, disconnect each link, wait, then reconnect the link, all while monitoring the interface status of hosts. If the status did not change as expected, the system will record the specific information of the link being tested, and flag the link as potentially part of a connection to a rogue device.

Link interruption can be scheduled to be completed at a convenient time (such as, at 3:00 a.m. when network usage might be minimum). This solution can also be automated, which removes much of the physical burden from information technology (IT) administrators.

DHCP is extremely time consuming, especially for large networks, since each link needs to be individually disconnected and reconnected one at a time. The larger the network, the more links that need to be tested, and the more time required to get through them all. This in turn affects the frequency of testing each link. Overall, the larger the network, the less frequently each link can be tested.

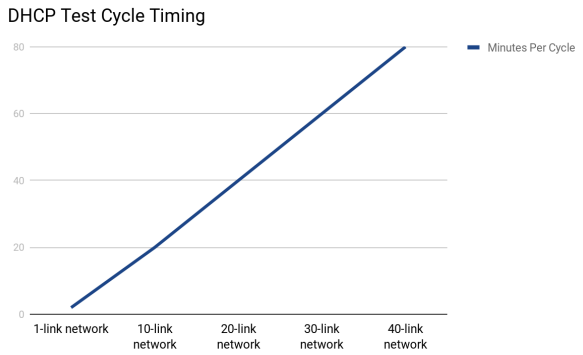

Fig. 4. Time it takes to test all switches in networks of various lengths (1-40 links).

Figure 4 displays the approximate time it takes (in minutes) to test each link in networks of various sizes. As seen in each graph, time increases linearly as the size of the network grows. This graph assumes that one testing cycle (to test all links in the network exactly once each) requires approximately two minutes to complete based on the following experimental measurements we did.

- 30 seconds time where the link is disconnected.
- 30 seconds initialization/power on time.
- 45 seconds time to monitor/verify the interface status and DHCP traffic.
- 10 seconds time to verify network connectivity re-established.
- 5 seconds time to transition to next link and tolerance.

The larger the network, the fewer options that exist for when the links can be tested. A 10-link network that takes approximately 20 minutes assuming all links are shut down and brought back one after another, so only one host is disconnected at a time. Similarly, a 1,000-link network (or larger) requires much more of a time commitment (2,000 minutes, or 33.3 hours). You can reduce the downtime significantly by shutting down multiple links at a time.

This solution will not work for hosts that have a critical need to be connected to the network at all times with no interruptions, since it requires a temporary disconnection of the link between the switch port and host. Additionally, it requires the network to be using DHCP. However, DHCP use is common, and many hosts can tolerate a very brief temporary loss of connectivity.

 ISBN: 978-1-61208-729-0

False positives (detecting a rogue switch when none is present) will be very unlikely with this mechanism. A false positive would require the DHCP request to not be received even though the rogue switch is not present. All DHCP clients must however send a DHCP request when disconnected and reconnected in case they have been reconnected to a different network. A DHCP client without this behavior would not function in very common real-world situations. Therefore, the only reasonable way the DHCP request could be missing is packet loss. However, a properly implemented DHCP client will retransmit this request several times. Therefore, a false positive will only occur if the hosts on the network have improperly implemented DHCP clients.

B. SNMP & ARP Broadcast Traffic Detection

This solution detects a potential rogue device through a multi-step process. We assume that an SNMP agent is configured on every valid switch and the switch monitors the traffic. We implemented the topology in Figure 5 within the simulator. We configured the LAN interface of Router0, PC0 and PC1. We assume that Rogue_Switch and Rogue_PC are added by an adversary.

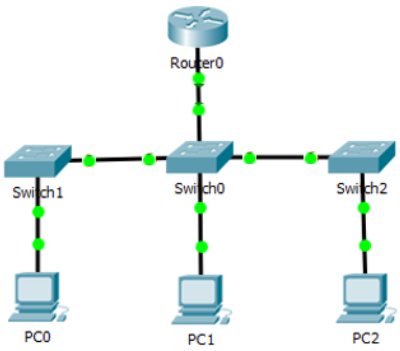

Fig. 5. Rogue_Switch broadcasts traffic. When either Switch0 or Switch1 detect the broadcasted traffic, a trap alert will be sent.

As seen in Figure 5, when an adversary adds Rogue_Switch and Rogue_PC to the network, traffic sent from Rogue_PC is broadcast by the Rogue_Switch (because the MAC Address table of Rogue_Switch would be empty). In this case, frames are broadcast to both Switch0 and Switch1. All valid switches in the network will detect the broadcast traffic and will alert the network administrator because a sudden broadcast traffic is an anomaly. There would also be ARP traffic at least at the beginning when new rogue devices are added. Once an alert is sent, the network administrator filters the ARP traffic that was generated around the time that the alert message was received. Broadcast frames and ARP traffic indicate the possible presence of a new device. The network administrator will be able to either confirm that a rogue switch or rogue device is connected or can clear the alert if the administrator is confident that the flagged traffic is permitted.

The rogue switch could be configured to prevent the detected behavior. This is only possible if the rogue switch MAC address table is already populated with correct entries. This would require an attacker to put lot of efforts and it may not even be possible to learn MAC addresses of neighboring switches or PCs. We can automate the detection of broadcast traffic using SNMP, as explained next. SNMP is designed to monitor events, and can be configured to send trap alerts whenever a specified event occurs. Almost any network monitoring software will work for this experiment, as long as broadcast traffic can be detected.

TABLE I. OIDS USED FOR DETECTION

Object ID	Name	Description
.1.3.6.1.2.1.2.2.1.12	ifInNUcastPkts	Number of inbound broadcast/multicast packets
.1.3.6.1.2.1.2.2.1.17	ifOutUcastPkts	Number of outbound broadcast/multicast packets

Table 1 shows a list of some OIDs that would be useful in this solution [10] [11]. These OIDs can be used to monitor broadcast traffic. Trap alerts should be sent when a valid network switch detects that another switch has sent broadcast traffic, indicating that the switch does not know where to forward the packet. This can help uncover a rogue switch that was recently connected.

The benefits of this solution are that the rogue switch will be detected fairly quickly after it is connected, because the forwarding table will be empty and the switch is forced to broadcast frames that it does not know where to forward. Also, before any corrective action is taken against the rogue switch, the network administrator is able to investigate whether the alert was caused by an actual rogue switch or if it is a just a false alarm. This minimizes potential disruptions to the network. The effectiveness of this solution does not depend on the size of the network.

The drawback of this solution is that it takes a significant amount of time and effort to find the physical location of the rogue switch once the alert is sent (and determined to be a legitimate issue). The switch will remain in place until a network administrator can spend the time to investigate, locate the switch, and remove it. This leaves the network potentially vulnerable during that time. While much of this solution can be automated, some steps will require human interaction, leaving open the possibility of mistakes caused by human error.

A business using this solution would experience minimal to no interruption in day-to-day tasks. Unlike the DHCP solution, this solution does not cause network interruptions. Each switch would monitor the defined OIDs in addition to forwarding packets as normal. This solution does create extra tasks for the IT administrator, such as analyzing the trap messages to determine whether an alert leads to a rogue switch or if it was a false alarm. IT administrators would also have the responsibility of finding the physical location of the switch and removing it from the network.

 ISBN: 978-1-61208-729-0

C. MAC Address Whitelisting

This solution requires every switch to download a copy of a whitelist (maintained by the network administrator), which includes every MAC address that has a permission to transmit traffic on the network. As a switch receives a packet, it first checks to see if the sender MAC address is on the forwarding table. If there is no entry for the sender's MAC address, the switch references the whitelist. The MAC address is only added to the forwarding table if it is found on the whitelist; otherwise, the packet is flagged as potentially from a rogue device.

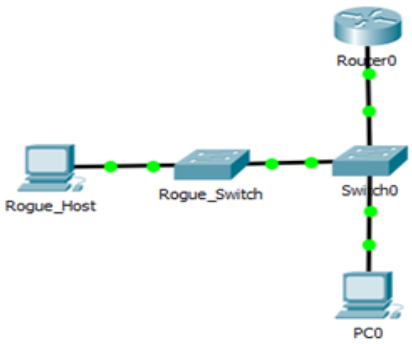

Fig. 6. Whitelisting example.

In Figure 6, Switch0, PC0 and Router0 are valid network devices. When Rogue_Host transmits a packet, Rogue_Switch will forward the packet to Switch0. Upon receiving the packet from Rogue_Switch, Switch0 will reference the whitelist, and will determine that Rogue_Host MAC address is not included. Therefore, Switch0 will drop the packet (or otherwise notify the network administrator of a potential rogue device on the network). It is possible to connect a rogue PC directly but it will still generate ARP broadcast queries, which is what we use for detection.

Given the benefits and drawbacks of this solution, whitelisting would be most ideal for smaller networks or for networks where the frame transmission speed is not critical. Smaller networks would allow for the network administrator to most effectively maintain the whitelist, and would also minimize the time that each switch takes to search the list for the sender's MAC address.

This solution is intended to detect and block rogue PCs (or other communicating devices), as well as protect the network against rogue switches added without permission. This solution does not provide information for the physical location of the rogue switch, though instead provides a layer of security to be used in parallel with other methods. This solution will always be successful in immediately detecting a rogue switch as soon as it begins to transmit data on the network, assuming that the MAC address of the sender is not *spoofed*. Only permitted devices will be able to communicate on the network, and all others will be blocked. The packets being transmitted would not even reach the destination before the rogue switch is detected, which allows for the packet to be intercepted by the IT administrator or dropped by the switch. No additional traffic is generated on the network as a result of the implementation of this solution apart from downloading the whitelist. Whitelisting is extremely effective in protecting the network. Detection of the rogue device using this method is almost immediate after the device begins to communicate, and 100% of the packets from the rogue device would be blocked from being delivered to their intended destination.

Whitelisting does not protect against an attack where the attacker spoofs the MAC address of the rogue device. In order to successfully spoof the MAC address, the attacker would either need in-depth knowledge about the addresses on the whitelist, or would need to repeatedly send frames with different MAC addresses until a valid address is discovered. Whitelisting also requires high maintenance from the IT administrator. Whenever a new device needs to be added (or an old device removed) from the whitelist, the list needs to be updated and a fresh copy is downloaded locally by each switch. This presents a vulnerability where a previously allowed device that has been removed from the whitelist may still be permitted to transmit on the network if the switch has not yet received an updated copy of the whitelist.

This solution is costly in terms of processing time and resources. Depending on the length of the whitelist, the packet forwarding time can also be impacted. The longer a whitelist, the longer time it takes to search through the list to determine if the sender is permitted to transmit on the network. By adding an extra step to a switch's forwarding process, the processing time increases linearly as the size of the whitelist increases.

Figure 7 compares the estimated maximum search time for whitelists of various sizes. The range selected represents a realistic whitelist size for small to medium sized networks. Companies with around 500-800 employees could have around 1,000 devices on the whitelist. If the time a switch takes to search the whitelist for a specified MAC address is around 3 microseconds per entry, then a relatively small whitelist with 10 entries would use up to 30 microseconds in the worst case (assuming linear search). A larger whitelist with around 1,000 entries would take up to 3,000 microseconds (0.003 seconds) to search. This greatly impacts the time from when the packet is initially sent from the source until it is received by the target.

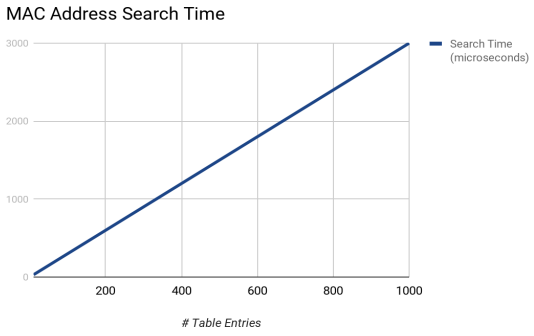

Fig. 7. Estimated Whitelist search time (based on linear search algorithm).

The larger the whitelist, the longer the potential search time for each switch. A whitelist with 100 entries would take

significantly less time to search than a whitelist with 100,000 entries.

Additional time may be spent sorting through false positives (or permitted MAC addresses that are initially incorrectly flagged as potentially rogue devices). This solution is designed to prefer false positives as opposed to false negatives (devices not permitted on the network that are incorrectly allowed network access), however, this is at the expense of processing time and inconvenience to the device attempting to access the network. False positives may be caused by a switch not having the latest copy of the whitelist. If a device is added to the whitelist after the switch updates its local copy, any traffic sent by the device will be blocked until the switch updates its copy again to include the new device.

MAC addresses can be easily spoofed to an address that is on the whitelist. If the whitelist addresses are not known to the attacker, a larger network would be easier for an attacker to randomly guess an allowable address. Based on these drawbacks, this solution would be ideal for smaller to medium networks.

V. CONCLUSIONS

In this paper, we proposed three solutions to detect a rogue switch. All three solutions can be implemented independently of each other, and all three solutions can be implemented at the same time. Each solution uses different sets of anomalies to detect a rogue switch. None of the three solutions rely on each other in order to be successful, which strengthens the network exponentially if all three solutions are implemented at the same time.

The whitelisting-based solution prevents devices that are not permitted from communicating on the network altogether. The DHCP-based detection solution assists in first finding if a rogue device exists, and second, finding the location of the device. The SNMP-based detection solution monitors the type of traffic being transmitted to determine whether a rogue device exists.

All three solutions will also work on any type of network, though each one has a specific network type where its maximum benefit can be reached. The whitelisting-based detection solution works best on smaller to medium size networks. The SNMP-based detection solution works best on networks that do not frequently have new devices connected. The DHCP-based detection solution works best on smaller to medium size networks where some downtime for hosts is acceptable.

To the best of our knowledge, this is the first such attempt to detect a rogue Ethernet switch (regardless of the vendor). We prove the feasibility of our solutions by carrying out realistic simulations and in-depth analysis. False positives are very unlikely because of the way our solutions are designed.

ACKNOWLEDGEMENTS

We would like to thank Adrian Mora for his initial contributions.

REFERENCES

[1] M. Rogers and D. Ruppersberger, “Investigative Report on the U.S. National Security Issues Posed by Chinese Telecommunications Companies Huawei and ZTE”, U.S. House of Representatives 112thCongress, 2012.

[2] “Cisco Adaptive Security Appliance Remote Code Execution and Denial of Service Vulnerability.”, Available at https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20180129-asa, 2018.

[3] P. Congdon *et al*, “IEEE 802.1X Remote Authentication Dial In User Service (RADIUS)”, Available at https://tools.ietf.org/html/draft-congdon-radius-8021x-10, 2001.

[4] 802.1AB-REV - Station and Media Access Control Connectivity Discovery, Available at http://www.ieee802.org/1/pages/802.1AB-rev.html, 2018.

[5] “NetSpot, FREE WiFi Site Survey Software for MAC OS X & Windows”, Available at: https://www.netspotapp.com/, 2018.

[6] “Nmap: the Network Mapper”, Available at https://nmap.org/, 2018.

[7] J. Kurose and K. Ross, “Computer Networking: A Top-Down Approach”, 6th ed. Upper Saddle River, New Jersey: Pearson Education, Inc., pp.27, 353-355, 480-482, 2013.

[8] “Configuring Spanning Tree Protocol”, Available at: https://www.cisco.com/c/en/us/td/docs/wireless/access_point/1300/12-2_15_JA/configuration/guide/o13span.html, 2018

[9] “Cisco Packet Tracer”, Available at https://www.netacad.com/courses/packet-tracer, 2018.

[10] ifInNUcastPkts(12), Available at http://oid-info.com/get/1.3.6.1.2.1.2.2.1.12, 2018.

[11] ifOutUcastPkts(17), Available at: http://oid-info.com/get/1.3.6.1.2.1.2.2.1.17, 2018

 ISBN: 978-1-61208-729-0