Critical Analysis of Layer 2 Network Security in Virtualized Environments

Ronny L. Bull and Jeanna N. Matthews

Wallace H. Coulter School of Engineering,
Clarkson University,
Potsdam, NY, 13676, USA
E-mail: bullrl@clarkson.edu
E-mail: jnm@clarkson.edu

Abstract: Cloud service providers offer their customers the ability to deploy virtual machines in a multi-tenant environment. These virtual machines are typically connected to the physical network via a virtualized network configuration. This could be as simple as a bridged interface to each virtual machine or as complicated as a virtual switch providing more robust networking features such as Virtual LANs *(VLANs)*, Quality of Service *(QoS)*, and monitoring. In this article, we explore whether Layer 2 network attacks that work on physical switches apply to their virtualized counterparts by performing a systematic study across four major hypervisor environments - Open vSwitch, Citrix XenServer, Microsoft Hyper-V Server and VMware vSphere - in seven different virtual networking configurations. First, we use a malicious virtual machine to run a MAC flooding attack and evaluate the impact on co-resident virtual machines. We find that network performance is degraded on all platforms and that it is possible to eavesdrop on other client traffic passing over the same virtual network for Open vSwitch and Citrix XenServer. Second, we use a malicious virtual machine to run a rogue DHCP server and then run multiple DHCP attack scenarios. On all four platforms, co-resident virtual machines can be manipulated by providing them with incorrect or malicious network information.

Keywords: Virtualization; Networking; Network Security; Cloud Security; Virtual Switches; Layer 2 Attacks; DHCP; DNS; MAC Flooding;

Reference to this paper should be made as follows: Bull, R.L., Matthews, J.N. (2016) 'Critical Analysis of Layer 2 Network Security in Virtualized Environments', *Int. J. Communication Networks and Distributed Systems*, Vol. 17, No. 3, pp. 315-333

Biographical notes: Mr. Bull is a Computer Science Ph.D. candidate at Clarkson University focusing on Layer 2 network security in virtualized environments. Ronny earned an A.A.S. degree in Computer Networking at Herkimer College in 2006 and completed both a B.S. and M.S. in Computer Science at SUNYIT in 2011. Mr. Bull serves as an Assistant Professor of Computer Science at Utica College with a focus in computer networking and cybersecurity. He also co-founded and is one of the primary organizers of the Central New York Intercollegiate Hackathon event which brings together cybersecurity students from regional colleges to compete against each other in offensive and defensive cybersecurity activities.

Dr. Matthews is an Associate Professor of Computer Science at Clarkson University. Her research interests include virtualization, cloud computing, computer security, computer networks and operating systems. Jeanna received her Ph.D. in Computer Science from the University of California at Berkeley in 1999. She is currently the co-editor of ACM Operating System Review and a member of the Executive Committee of US-ACM, the U.S. Public Policy Committee of ACM. She is a former chair of the ACM Special Interest Group on Operating Systems (SIGOPS). She has written several popular books including *Running Xen: A Hands-On Guide to the Art of Virtualization* and *Computer Networking: Internet Protocols In Action*.

This work is part of the primary author's Ph.D. dissertation research at Clarkson University.

1 Introduction

With the growing popularity of Internet-based cloud service providers, many businesses are turning to these services to host their mission critical data and applications. Cloud customers often deploy virtual machines to shared, remote, physical computing resources. Virtual machines running in cloud capacity are connected to the physical network via a virtualized network within the host environment. Typically, virtualized hosting environments will utilize either a bridged network interface or a virtualized switch such as Open vSwitch (Pettit et al., 2010; Pfaff et al., 2009) for Xen and Kernel-based Virtual Machine *(KVM)* environments, or the Cisco Nexus 1000V Series virtual switch for VMware vSphere environments (Cisco Systems Inc., 2014). These virtual switches are designed to emulate their physical counterparts, however the majority of them do not provide any of the Layer 2 protection mechanisms found in modern enterprise grade hardware switches. It is important for users of multi-tenant cloud services to understand how to secure their network traffic is from other users of the same cloud services, especially given that virtual machines from many customers share the same physical resources. If another tenant can launch a Layer 2 network attack and capture all the network traffic flowing from and to their virtual machines, this poses a substantial security risk. By understanding which virtual switches are vulnerable to which attacks, users can evaluate the workloads they run in the cloud, consider additional security mechanisms such as increased encryption and/or increased monitoring and detection of Layer 2 attacks.

In this paper, we present the results of a systematic study to evaluate the effects of MAC flooding and Dynamic Host Configuration Protocol *(DHCP)* attacks across four major hypervisor environments with seven different virtual network configurations. First, we provide some background information on the general network configuration options available to virtualized environments. We then introduce the test environment, and present our attack methodology using Media Access Control *(MAC)* and Dynamic Host Configuration Protocol *(DHCP)* attack scenarios. We conclude the paper by discussing related work and summarizing our results.

2 Network Configuration Options

There are two types of networking configurations that are typically used in virtualized environments: bridging and switching. In this section we describe both options and discuss how each one is applied within a virtualized network.

2.1 Bridging

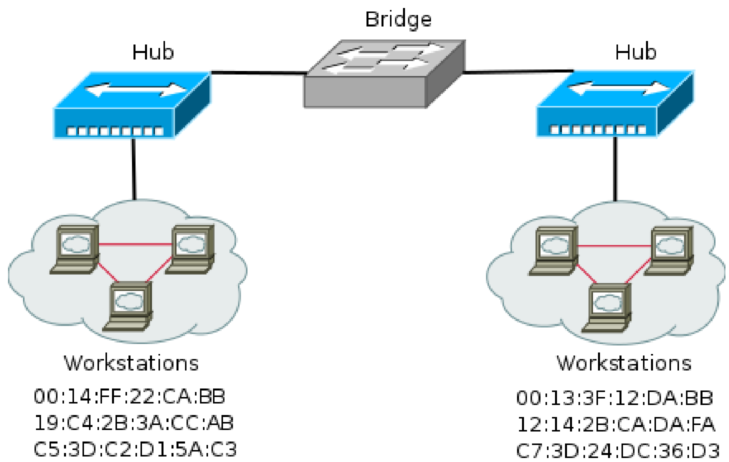
Bridged mode is the simplest configuration providing an interface dedicated to virtual machine use. A bridge connects two or more network segments at Layer 2 in order to extend a broadcast domain and separate each of the segments into their own individual collision domains (Seifert and Edwards, 2008). A forwarding table as described by LAN MAN Standards Committee (2004); Seifert and Edwards (2008) is used to list the MAC addresses associated with devices located on each network segment connected to the bridge *(Figure 1)*. Requests are forwarded based upon contents of this table and the destination MAC address located in the Ethernet frame. A frame is forwarded across the bridge only if the MAC address in the destination block of the frame is reachable from a different segment attached to the bridge. Otherwise, the frame is directed to a destination address located on the same segment as the transmitting device or dropped.

In virtualized environments, guest machines utilize user-space virtual network interfaces that simulate a Layer 2 network device in order to connect to a virtual bridge. Typically, the virtual bridge is configured and bound to a physical interface on the host machine that is dedicated solely to virtual machine traffic.

2.2 Switching

Physical switches have the capability of operating at Layer 2 or higher of the OSI model. Switches can be thought of as multi-port bridges (Seifert and Edwards, 2008) where each port of the switch is considered as its own isolated collision domain. Instead of a forwarding table, switches employ

Figure 1 A basic bridge using a forwarding table to pass requests between two network segments.

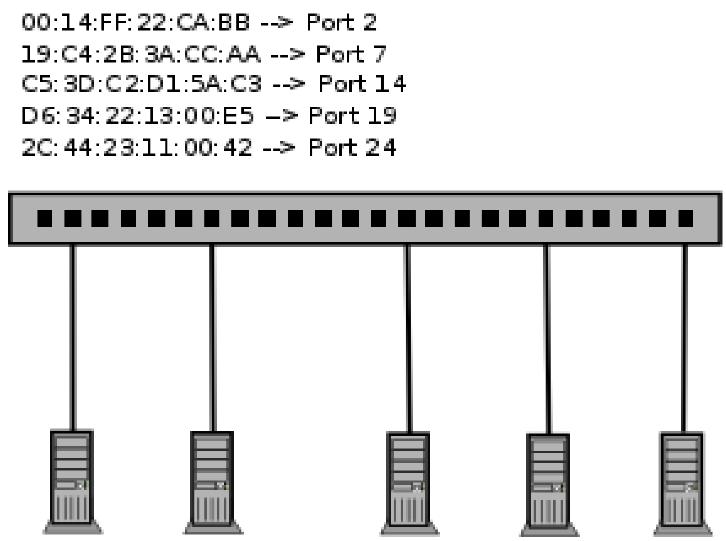


a CAM (content addressable memory) table (Seifert and Edwards, 2008) . Content addressable memory is specialized memory hardware located within a switch that allows for the retention of a dynamic table or buffer that is used to map MAC addresses of devices to the ports they are connected to *(Figure 2)*. When a packet arrives on a port from a new device, an entry mapping that device to that port is added to the table. This allows a switch to intelligently send traffic directly to any connected device without broadcasting frames to every port on the switch. The switch reads the frame header for the destination MAC address of the target device, matches the address against its CAM table, then forwards the frame to the correct device. If a MAC address is not found in the CAM table, a packet destined for it will be sent to all interfaces.

The use of a CAM table and the separation of collision domains are key factors in preventing eavesdropping of network traffic between devices connected to the switch. However, a physical switch is an embedded device and has a finite amount of memory available to its CAM table; once this memory is full, the switch must discard existing entries in order to add new entries. The majority of physical switches in use today employ CAM chips that are capable of holding up to 32,000 addresses (Seifert and Edwards, 2008) which can easily be saturated by a single MAC flooding attack in a very short amount of time.

Virtual switches emulate their physical counterparts and are capable of providing features such as VLAN traffic separation, performance and traffic monitoring, as well as quality of service *(QoS)* solutions. Virtual machines are connected to a virtual switch by the way of virtual network interfaces *(VIF)* that are similar to the Layer 2 network devices used in conjunction with virtual bridges.

Figure 2 A switch and its CAM table.



3 Test Environment

In this section, we provide details about our test environment. It consisted of seven server class systems all located on a test network that was isolated from local production networks to avoid impacting them. We deployed an optimized installation of Gentoo Linux and the Xen 4.3 hypervisor to three Dell PowerEdge 860 servers each equipped with a dual core Intel Xeon 3050 2.13GHz processor, 4 GB of memory, and a 500 GB hard drive. Each system contained dual Broadcom NetXtreme BCM5721 Gigabit Ethernet PCI Express network interface cards integrated into the motherboard. The first network interface was dedicated to the privileged control domain on each server for administrative functions, and the second configured to be utilized by guest virtual machines. Each server's 500 GB hard disk was divided into four partitions; a 100MB ext3 /boot, a 10GB ext3 /, a 2GB swap, with the remainder allocated to Logical Volume Management *(LVM)* storage for virtual machine deployment.

Four additional servers were configured with enterprise level hypervisor solutions: Citrix XenServer 6.2, Microsoft Windows Server 2008 R2 with the Hyper-V hypervisor, Microsoft Hyper-V 2008 *(free edition)*, and VMware vSphere *(ESXi)* 5.5 *(free edition)*. The hardware utilized for the Citrix XenServer 6.2 system was identical to the three Gentoo systems, however the Microsoft Hyper-V and the VMware vSphere hypervisors were configured on systems with different hardware configurations due to a lack of additional

Dell PowerEdge 860 systems. Both Microsoft Windows Server 2008 R2 along with the Hyper-V hypervisor as well as the free version of Hyper-V 2008 were installed to identical Dell PowerEdge 2950 server systems containing dual quad core Intel Xeon 5140 processors at 2.33GHz, 32GB of memory, and a 145GB SATA hard drive. VMware vSphere *(ESXi)* 5.5 *(free edition)* was deployed to a custom built server using a Supermicro X9SCL server motherboard, a quad core Intel Xeon E3-1240 processor at 3.30GHz, 24GB of memory, and a 500GB SATA hard drive. The Hyper-V and vShpere systems were each outfitted with two network adapters in order to provide separate dedicated interfaces for administrative purposes and virtual machine use. Though there are notably some variations in the hardware configurations summarized in *Table 1*, it is important to note that these differences had no impact on the results of the experiments that were performed.

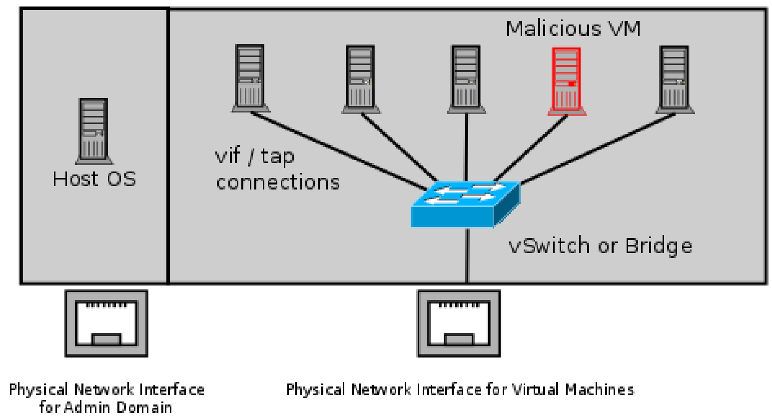
Table 1 Summary of test environment hardware.

	Hardware Specs			
Platform	**CPU Type**	**Memory Size**	**Hard Disk**	**NICs**
OS Xen w/ Linux Bridging	Xeon 3040	4 GB	500 GB	2
OS Xen w/ Open vSwitch 1.11.0	Xeon 3040	4 GB	500 GB	2
OS Xen w/ Open vSwitch 2.0.0	Xeon 3040	4 GB	500 GB	2
Citrix XenServer 6.2	Xeon 3040	4 GB	500 GB	2
MS Server 2008 R2 w/Hyper-V	Xeon 5140	32 GB	145 GB	2
MS Hyper-V 2008 Free	Xeon 5140	32 GB	145 GB	2
VMware vSphere (ESXi) 5.5	Xeon E3-1240	24 GB	500 GB	2

For the MAC flooding scenario, two virtual machines were deployed to each virtualization platform: one malicious virtual machine attempting to eavesdrop on the traffic of any other tenant virtual machines and one victim virtual machine *(Figure 3)*. The Kali Linux security distribution was selected due to the plethora of network security auditing tools that come pre-installed and configured. Two complete installations of Kali were installed to each server on 20GB LVM partitions as hardware virtual machine *(HVM)* guests. The systems were then allocated static IP addresses that positioned them on the same isolated subnet as the servers and were completely updated.

The DHCP attack testing required a more elaborate setup. Specifically, we created four virtual machines within each hypervisor platform. Each of these virtual machines used a minimal installation of CentOS 6.5 and was

Figure 3 A malicious virtual machine located on a multi-tenant virtual network.



configured for a specific purpose *(Table 2)*. First, a virtual machine acting as a rogue DHCP server was setup and configured using DNSMasq, a lightweight DHCP and DNS server. Second, a simple router using iptables on a separate virtual machine was used to forward traffic between two broadcast domains using NAT and two network interfaces. Third, a basic Apache web server was setup to act as a malicious web server. Fourth, the final virtual machine was configured as a minimal client that was left unpatched and vulnerable to shellshock (National Vulnerability Database, 2014).

Table 2 New virtual machines added to each hypervisor platform for Layer 2 DHCP attack testing.

Operating System	Completely Updated	System Purpose	Virtual Interfaces
CentOS 6.5	Yes	DHCP/DNS Server	1
CentOS 6.5	Yes	Simple Router	2
CentOS 6.5	Yes	HTTP Server	1
CentOS 6.5	No	Left Vulnerable to ShellShock	1

4 Attacks Performed

In this section, we describe the two types of Layer 2 networking attacks we conducted: MAC flooding and DHCP attacks. Each attack was performed identically on all platforms in order to analyze the differences between the environments when subjected to the different attack scenarios.

4.1 MAC Flooding

The most common Layer 2 Media Access Control attack is a MAC flooding attack in which the attacker generates many packets with random MAC addresses in an attempt to overflow the CAM buffer within a switch and thus force the switch into a mode in which it broadcasts packets on all interfaces. This happens because the legitimate MAC addresses are evicted from the CAM table in favor of the many random MAC addresses generated by the attacker. This is referred to as hub mode and when a switch is operating in hub mode, the inherent separation of collision domains is broken and all frames passing through the switch are forwarded to all connected devices. This allows for passive eavesdropping of all traffic passing through the device. MAC flooding can be mitigated by enforcing port security on physical switches which imposes a limit on the amount of MAC addresses that can send traffic to a specific port (Cisco Systems Inc., 2015). This feature is not implemented within the majority of the virtual switches available today rendering them vulnerable to MAC flooding attacks.

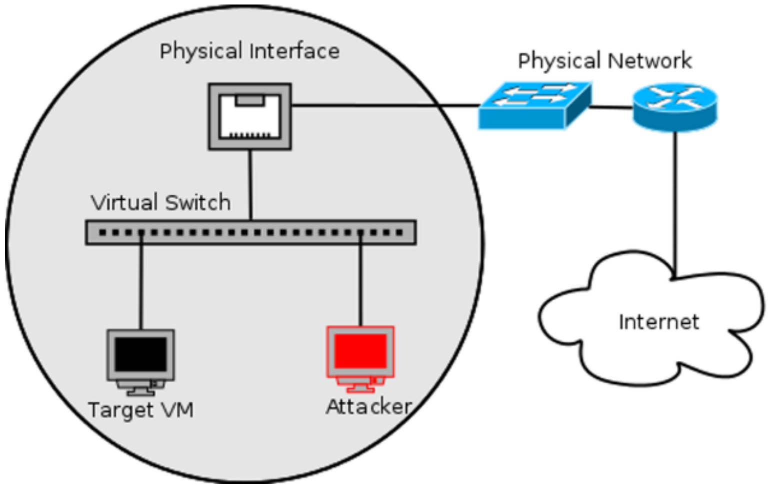
The program *macof* from the *dsniff* package (Yeung et al., 2008) was used on a Kali virtual machine to perform a MAC flooding attack on the virtual network within each test environment. This type of attack when performed on a physical switch typically causes the CAM table on the switch to fill up forcing the device to go into a fail safe or hub mode which in turn causes all packets on the network to be broadcast to every node connected to the switch. Wireshark was used to determine if the attack was successful by monitoring the network for HTTP traffic which should not be interceptable by other hosts on the virtual network.

All tests were conducted in the same manner. Each server had two Kali Linux virtual machines deployed on them and connected to the internal virtual network as illustrated in *Figure 4*. During each experiment both virtual machines were brought online. On the first virtual machine *(Kali1) macof* was started up using the command:

```
macof -i eth0
```

and left to run. Then Wireshark was started on the same virtual machine and an HTTP filter was applied to only display sniffed HTTP traffic. The second Kali virtual machine *(Kali2)* was then used to surf the web. If the attack proved to be successful then the HTTP traffic from *Kali2* should be viewable in Wireshark on *Kali1*.

Figure 4 Network diagram for MAC flooding attack scenarios.



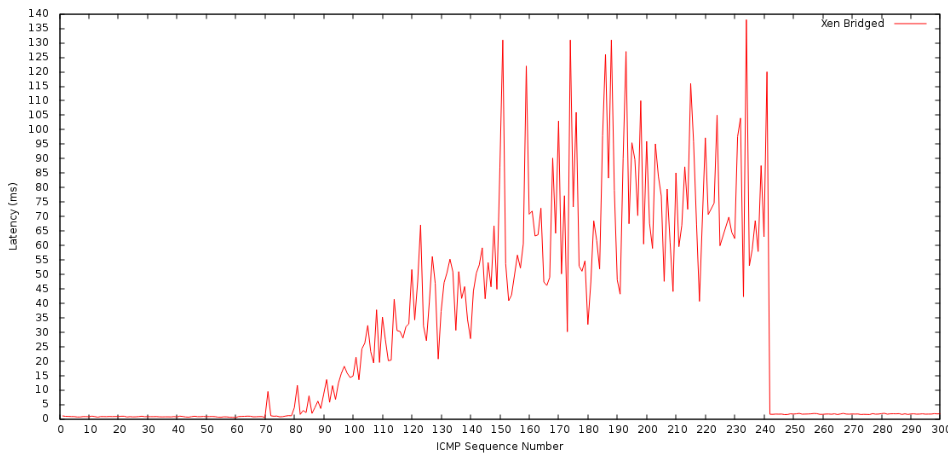
4.1.1 Linux 802.1d Bridged Interface

Running the attack within the bridged virtual network test environment resulted in a significant performance degradation that impacted the usability of the tenant virtual machines, essentially creating a denial of service *(DoS)* type of attack. This effect was observed as a large increase in latency when attempting to interact with any of the virtual machines on the system either through Secure Shell *(SSH)* or Virtual Network Computing *(VNC)*. While the MAC flooding attack was occurring remote connections to the virtual machines became unstable due to the saturation of the virtual network with spoofed frames. This effect was quantified by using the ping utility on the second virtual machine to measure the transmission latency to a server located on the physical network while the attack was occurring *(Figure 5)*. The attack however did not result in the ability to sniff other virtual machine traffic passing over the interface. This most likely comes from the fact that the standard bridge interface is missing the CAM table that typically is found on switches mapping known MAC addresses to switch ports, an essential element of the attack.

4.1.2 Open vSwitch 1.11.0 Interface

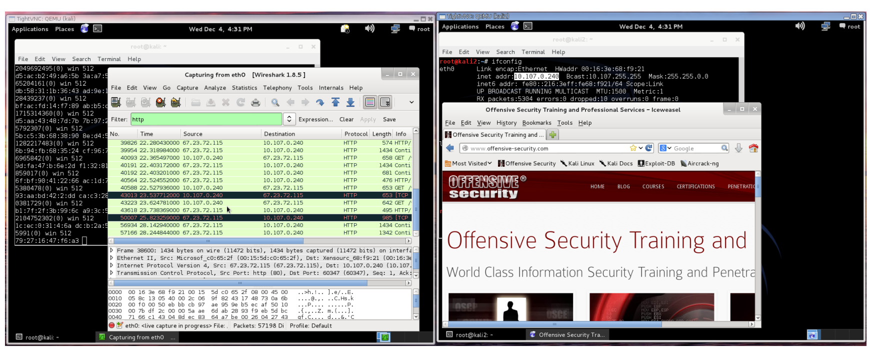
When running the attack on the Open vSwitch 1.11.0 virtual network test environment network performance degradation was also observed, and the attacking machine could also successfully sniff traffic from another tenant machine. *Figure 6* depicts the results of the successful attack and provides

Figure 5 Latency measured using the ping utility on a bridged virtual network during a MAC flooding attack. The attack was launched at ICMP request 61 and terminated at ICMP request 241.



substance to the claim that virtual switches are vulnerable to some of the same Layer 2 attacks as physical switches.

Figure 6 A malicious virtual machine running macof on an Open vSwitch virtual network and successfully sniffing HTTP traffic with Wireshark from another tenant virtual machine.



4.1.3 Open vSwitch 2.0.0 Interface

Running the attack on the latest version of Open vSwitch available at the time of this research revealed that the vulnerability still existed and had not been addressed. The system responded in the same way as the previous two attempts and the other tenant's HTTP traffic was viewable in Wireshark.

It should be noted that in February of 2015 we notified the Open vSwitch security team of our discovery. They confirmed the vulnerability and immediately responded with a patch (Pfaff et al., 2015; Pfaff, 2015*a*) to resolve the issue. Since then the patch has been merged into every major branch of Open vSwitch from 2.0.0 on (Pfaff, 2015*b*). It is our recommendation that any environment running any version of Open vSwitch prior to the patched version of the 2.0.0 branch should be upgraded immediately, since both the vulnerability and exploitation technique have been made public.

4.1.4 Citrix XenServer 6.2

Citrix XenServer 6.2 utilizes an older version of Open vSwitch (version 1.4.6) to provide virtual switching services to its client machines. When the MAC flooding test was attempted in the XenServer environment, it was also discovered that the flooding was able to escape the virtual environment which caused all upstream physical switches to go into hub mode as well. Not only did this allow the malicious virtual machine running Wireshark to sniff traffic from other tenant virtual machines, it also was able to eavesdrop on traffic from physical machines located within the same broadcast domain to which the physical Ethernet adapter was connected. Since this test was performed on a university network well over one hundred upstream switches were affected and put into hub mode which also had the side effect of disabling all VLAN separation within the affected devices. This made it possible to view the majority of the traffic flowing over the campus network from the attacking virtual machine using Wireshark. After disclosing the issue to the IT staff they promptly took action to enforce port security on all of their physical switches limiting each port to learning at most fifty MAC addresses at a time to avoid the possibility of CAM table overflow.

4.1.5 Microsoft Hyper-V Server 2008 R2

Testing under the Microsoft Hyper-V environment was performed both with and without the Windows Firewall service enabled to identify if there was any effect on the results. Both scenarios proved to be unsuccessful due to the fact that Microsoft Windows Server 2008 R2 provides some minimal protection for virtualized network traffic, this includes protection against MAC address spoofing (Microsoft, 2013).

Further testing was performed on the free version of Microsoft Hyper-V to see if the protection offered by Server 2008 R2 is also built into the bare metal product. As with the previous environment testing was performed both with and without the Windows Firewall service enabled. It was concluded that under both conditions the free version of Microsoft Hyper-V 2008 was also unaffected by the MAC flooding attack since it is built upon a minimal version of Microsoft Windows Server 2008 R2 entitled Server Core. The

Core version of Microsoft Server 2008 R2 still provides the same level of network protection as the full version, but only allows for the installation of specific server roles to the operating system (Microsoft, 2008), in this case the Hyper-V hypervisor.

4.1.6 VMware vSphere (ESXi) 5.5 - free edition

All testing within the VMware vSphere environment was performed identically to the previous trials for completeness. Testing was performed on the free version of ESXi using the default virtual networking configuration. The results show that this particular configuration was not vulnerable to the MAC flooding attack in terms of a malicious user being able to eavesdrop on another tenant's network traffic. With respect to the VMware end user license agreement (VMware Inc., 2012) we attempted to acquire the proper permission from VMWare in order to release the performance results of the latency tests in the ESXi environment. However, we did not receive a response in time and therefore could not include any of the performance related results in this publication.

4.1.7 Summary of MAC Flooding Results

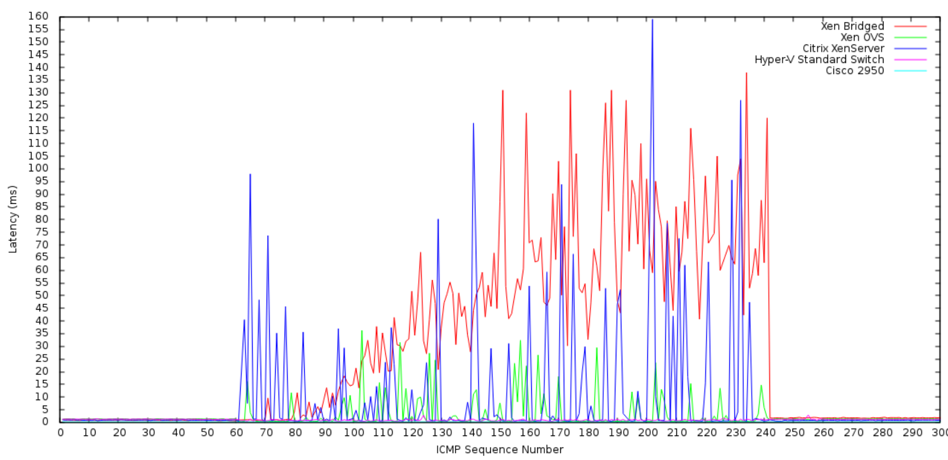
It can clearly be seen from the results summarized in *Table 3* that any virtualized network environment built upon the Open vSwitch virtual switch could be vulnerable to MAC flooding attacks, and has the potential to expose its client traffic to eavesdropping. Therefore, if a virtual machine is transmitting sensitive information over a virtual network that uses Open vSwitch precautions should be taken such as using encryption in order to ensure that the information in transit remains confidential.

Figures 7, 8, and 9 illustrate the effects of the MAC flooding attack on each of the virtual environments utilizing the same ping latency test that was previously described in *Section 4.1.1* and depicted in *Figure 5* for the bridged interface. The following figures also provide a control comparison via a Cisco 2950 hardware switch which underwent the same testing. All of the experiments were performed using Gigabit network interfaces to maintain uniformity in flood rate. As can be seen by the results any environment using a bridged or Open vSwitch based virtual network was heavily affected by the attack in terms of network performance. The Microsoft Hyper-V virtual network seemed to maintain a consistent performance throughout the entire test which was very similar to the performance of the Cisco 2950 hardware switch which remained unaffected during the attack.

Table 3 MAC flooding attack results across seven test environments. ✓indicates the platform was affected. *(* - results omitted due to VMware EULA.)*

Platform	Results of Attack: Eavesdropping Allowed	Results of Attack: Impacted Network Performance
OS Xen w/ Linux Bridging		✓
OS Xen w/ Open vSwitch 1.11.0	✓	✓
OS Xen w/ Open vSwitch 2.0.0	✓	✓
Citrix XenServer 6.2	✓	✓
MS Server 2008 R2 w/Hyper-V		
MS Hyper-V 2008 Free		
VMware vSphere (ESXi) 5.5		*

Figure 7 Latency comparisons against all tested platforms measured using the ping utility during a MAC flooding attack with a Cisco 2950 hardware switch used as a control. The attack was launched at ICMP request 61 and terminated at ICMP request 241. *(Note: VMWare results omitted due to EULA)*



4.2 Mitigation Techniques

As stated previously the traditional way to prevent these attacks on physical switches is to utilized port security to lock down switch ports so they can only learn a limited amount of MAC addresses to avoid CAM table overflow. It is also wise to configure port security to limit network connectivity to authorized MAC addresses only. Since this feature is not possible on the majority of enterprise grade virtual switches in use today other mitigation techniques need to be investigated. The most obvious solution would be

Figure 8 Latency comparisons against all tested platforms measured using the ping utility during a MAC flooding attack with a Cisco 2950 hardware switch used as a control. The attack was launched at ICMP request 61 and terminated at ICMP request 241. *(Note: VMWare results omitted due to EULA)*

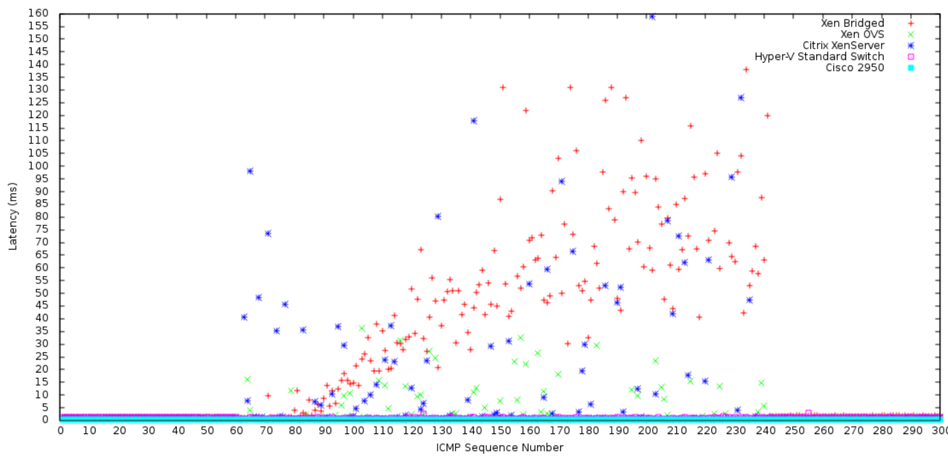
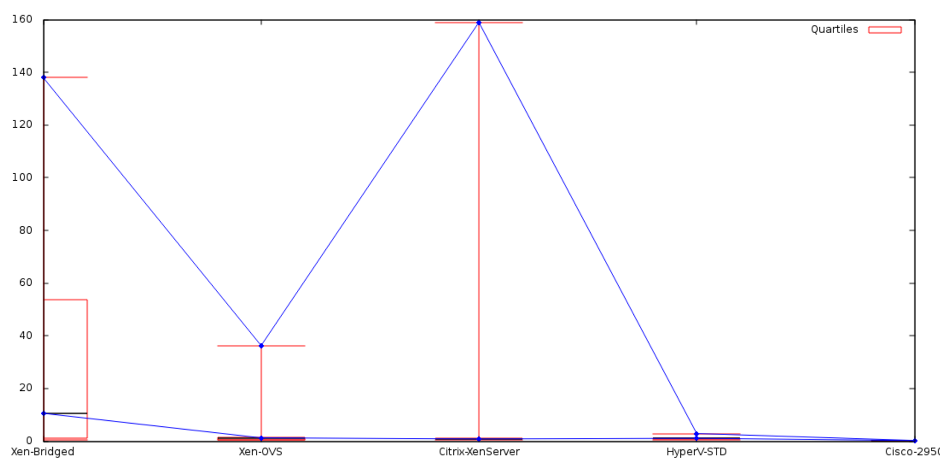


Figure 9 Box and whisker plot showing latency variations for each environment while being subjected to the MAC flooding attack. *(Note: VMWare results omitted due to EULA)*



for the virtual switch developers to start incorporating port security features into their software switches as a default feature that is easily configurable. Another simpler solution would be to prevent the switches from going into a hub mode state in the first place by not emulating physical switch CAM table limits. Bare-metal hypervisor platforms are typically deployed on systems with large amounts of memory. Unlike physical switches which have a limitation on available resources, server systems have an abundance of memory that could be allocated to the task of providing dynamic CAM tables to the virtual switching devices that could be configured to adapt to a flooding situation and alert an administrator if the memory usage rises above a certain threshold. This would eliminate the possibility of information leakage due to the virtual switch going into a hub mode state. As stated previously most modern physical switches have a CAM table capacity limit of 32,000 MAC addresses. With each MAC address comprising of 48 bits this would require around 187.5MB of memory in order to store all of the addresses. This memory requirement is insignificant on most modern server systems today which could easily provide a larger more dynamic buffer to the virtual switch.

4.3 DHCP Attacks

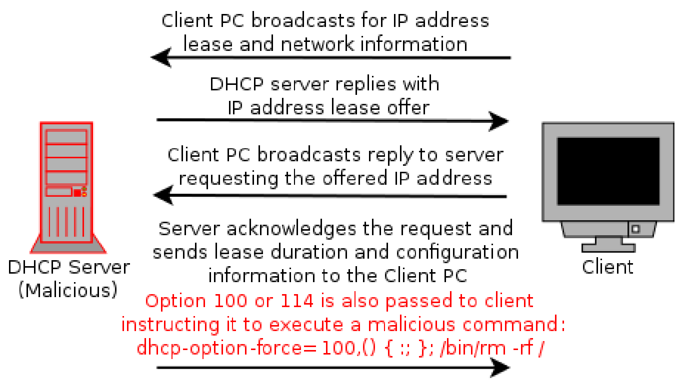
In order to perform a Layer 2 DHCP attack, an attacker must place a rogue DHCP server on a network in hopes that clients in the broadcast domain associate with it rather than the legitimate DHCP server. Once a client receives an IP address lease from a malicious DHCP server under an attacker's control, that client could also be seeded with the IP address of a poisoned DNS server, an incorrect default gateway, or be forced to run malicious code. This type of attack could also cause DoS situations where duplicate addressing occurs on the network causing the resources bound to those addresses to be inaccessible, or allow for the execution of man-in-the-middle attacks where traffic is first sent to an attacker and then onto the original destination. These attacks can be mitigated by enforcing static addressing, or by employing DHCP snooping on physical switches as well as DHCP server authorization within Active Directory environments.

Four different attack scenarios were duplicated across each of the seven test environments in order to evaluate the impact of these Layer 2 DHCP attacks. In the first scenario, the DNSMasq server was setup to pass option 100 to clients *(Figure 10)* which was configured to leverage the shellshock exploit in order to remotely execute the echo command with root privileges on the target machine and place text into a file in */tmp*. The following code was placed into the */etc/dnsmasq.conf* file on the DHCP server as a proof of concept to illustrate the vulnerability without damaging the client system.

```
dhcp-option-force=100,() { :; }; /bin/echo \\
'Testing shellshock vulnerability'>/tmp/shellshock_test
```

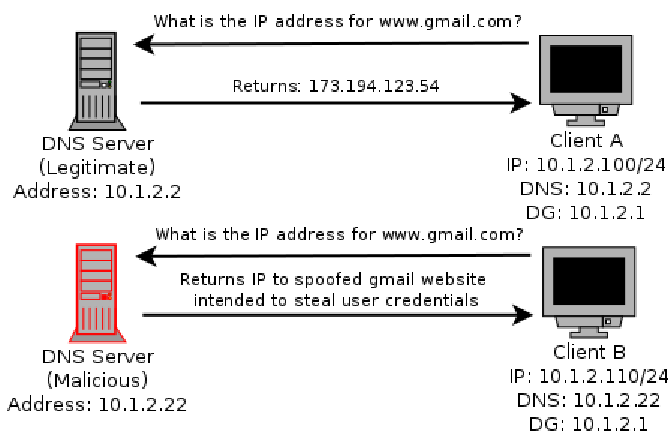
For the second scenario, the DNSMasq server was used to seed the minimal shellshock client with a poisoned DNS server through DHCP. Since DNSMasq also provides DNS server functionality the rogue DHCP server doubled as the poisoned DNS server that was passed to clients

Figure 10 Malicious DHCP lease process leveraging shellshock to issue the *rm -rf /* command using option 100.



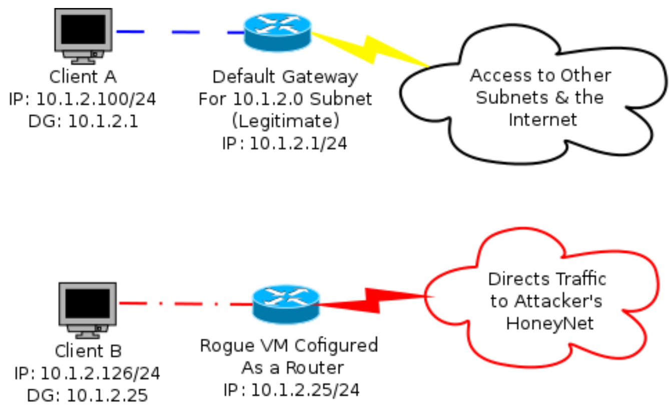
receiving addresses. The DNS server was setup to direct all traffic destined to www.gmail.com to be redirected to the malicious web server *(Figure 11)*. A command line web browser called *elinks* was then used in the shellshock virtual machine to visit www.gmail.com in order to observe the effect.

Figure 11 Presence of a poisoned DNS server on a network whose address is provided to clients associated with a rogue DHCP server.



Lastly, the DHCP server was configured to pass a bad default gateway address to clients that obtained their network configuration from it. First, it was set to pass 1.1.1.1 as the default gateway with the intention of causing a DoS attack for access of subnets outside of the existing broadcast domain. Second, the DHCP server was configured to point clients to the second virtual machine that was setup as a router to direct traffic to a malicious honeynet *(Figure 12)*. This in conjunction with a poisoned DNS server allows the attacker to direct traffic to malicious servers setup within the honeynet. In each case, the previously used web server was placed in the honeynet, and a DNS entry was setup to direct traffic to it through the rogue default gateway.

Figure 12 Malicious virtual machine configured as a router on a network whose address is provided to clients as a default gateway when associated with a rogue DHCP server.



4.3.1 Summary of DHCP Attack Results

Table 4 illustrates the results of all four DHCP attack scenarios that were run within each test environment. In all of the environments we tested, there was no protection provided against the attacks in their default configurations.

Table 4 DHCP attack scenario results across seven test environments. ✓indicates a successful attack.

	Attack Scenarios					
Platform	**MAC Flooding Eavesdropping**	**MAC Flooding Performance**	**Shell Shock**	**Poisoned DNS**	**Invalid Gateway**	**Malicious Gateway**
OS Xen w/ Linux Bridging	✓	✓	✓	✓		✓
OS Xen w/ Open vSwitch	✓	✓	✓	✓	✓	✓
ProxMox	✓	✓	✓	✓		✓
Citrix XenServer	✓	✓	✓	✓	✓	✓
MS Hyper-V Std. Switch	✓	✓	✓	✓		
MS Hyper-V Nexus 1000v	✓	✓	✓	✓		
VMware vSphere (ESXi)	✓	✓	✓	✓		

4.4 Malicious Extension of the ShellShock Proof of Concept Attack

After performing a successful proof of concept using a fairly harmless shellshock exploit via DHCP to write to a file in */tmp* we decided to evaluate each of the platforms against a more dangerous implementation of the attack. In this extension, the rogue DHCP server was configured to use option 100 to download a public key file from a remote web server and then add the key to the root user's *.ssh/authorized_keys* file. If successful this attack would allow the attacker to use the corresponding private key to gain remote root access to the system via SSH without entering a password. The following code was added to */etc/dnsmasq.conf* on the rogue DHCP server in order to implement the attack.

```
dhcp-option-force=100,() { :; }; /usr/bin/curl -s \\
webserver/pubkey >>/root/.ssh/authorized_keys
```

Like the proof of concept, the SSH public key attack was successful in each of the test environments. However, it was discovered that if SELinux was set to enforcing on the vulnerable client the public key could not be written to the root user's *.ssh/authorized_keys* file. This also resulted in a permission denied error being sent to the console on the client system during the DHCP lease process. Further investigation showed that we were able to write to */tmp* with SELinux set to enforcing because traditionally */tmp* is set to world writable so that any service can add or manipulate files that are located there. An additional test was performed to verify that the *dhclient* process executed the shellshock command with root privileges by changing option 100 in */etc/dnsmasq.conf* to run the *id* command:

```
dhcp-option-force=100,() { :; }; /usr/bin/id
```

which resulted in the following output indicating that *dhclient* does run commands fed to it via option 100 as the root user.

```
uid=0(root) gid=0(root) groups=0(root)\\
context=unconfined_u:system_r:dhcp_t:s0-s0:c0.c1023
```

SELinux offers some protection against this attack. In the default configuration of SELinux, a remote network service is prevented from writing to a file that is not configured to be used by that service unless a specific SELinux rule exists to permit access (Red Hat, Inc., 2015). Multiple tests were done on different file and folder combinations with different permission levels. It was found that SELinux only permitted the *dhclient* service to write to files located in directories that were set to world writable with the *chmod 777* command. However, if this feature in SELinux is set to disabled or permissive, the attack was again successful because full root write access to the system from a network service was no longer blocked. Knowing this provides a way of mitigating the remote execution of code via DHCP by enforcing the use of SELinux on all critical systems and learning how to configure it appropriately. It is important to note though that SELinux did not prevent any of the MAC flooding or DHCP attacks described in this paper..

4.5 Mitigation Techniques

The common theme among all of the DHCP attack scenarios is that the attacker must be able to place a rogue DHCP server on the network and beat out the legitimate DHCP server when responding to client requests. Preventing clients from interacting with the malicious DHCP server is the most logical answer to this problem. Static IP addressing is one solution, but it does not scale well and would not be applicable in a cloud environment. Cisco offers a feature in their hardware switches called DHCP snooping which according to them "acts as a firewall between untrusted hosts and trusted DHCP servers" Cisco Systems Inc. (2015). It allows validation of DHCP messages and rate-limiting of DHCP traffic to prevent rogue DHCP servers and DHCP starvation attacks from being effective on a network. Other utilities such as DHCP authorization can also be used in order to force clients to use a legitimate DHCP server, and ignore or drop responses from any other system trying to act as a malicious DHCP server on the network. Currently neither of these features are offered in the virtual switches that were used in this research leaving them all wide open to these types of attacks. If functionality similar to DHCP snooping on Cisco switches and DHCP authorization in Microsoft Active Directory environments were added to the virtual switching environments as standard features then data center operators could take proactive measures to safeguard their environments. The majority of these switches are capable of interacting with software defined networking *(SDN)* controllers which opens up the possibility of creating a solution that leverages SDN in order to provide a generic solution that would work across all virtual switching environments. One possible solution would be to create a SDN application that monitors the network for DHCP activity and verifies that that activity is legitimate and not malicious. This could be accomplished by monitoring each DHCP lease request and response to verify that the response is coming from a certified DHCP server on the network. If the response is found to be malicious then the traffic is dropped and an alert is sent to the administrator so that prompt action can be taken to identify the source of the attack quickly.

5 Related Work

There has already been a substantial amount of work studying the vulnerability of physical networks to Layer 2 attacks (Yeung et al., 2008; Altunbasak et al., 2005; Cisco Systems Inc., 2002; Lauerman and King, 2010), but the impact on virtual networks has not received as much attention. This is beneficial in the fact that published research previously performed on physical networks can serve as a model for testing in virtual environments and comparisons can be made based upon the physical baselines. For instance, Yeung et al. (2008) provided an overview of the most popular Layer 2 networking attacks as well as descriptions of the tools used to perform them. This work was very helpful in identifying possible attack vectors that could be emulated within a virtualized environment. Altunbasak et al. (2005) also described various attacks that can be performed on local and metropolitan area networks, as well as the authors' idea of adding a security tag to the Ethernet frame for additional protection. Cisco Systems Inc. (2002) also published a white paper regarding VLAN security in their Catalyst series of switches. The paper disclosed testing that was performed on the switches in August of 2002 by an outside security research firm @stake which was acquired by Symantec in 2004. In the white paper, they discussed many of the same attacks that were mentioned by Yeung et al. (2008), however the authors also went into detail about best practices and mitigation techniques that could be implemented on the physical switches in order to prevent the attacks from being successful.

6 Future Work

Going forward, we intend to evaluate other Layer 2 networking attacks within these environments as well as develop mitigation techniques and hardening strategies that will contribute to an increased level of network security in virtualized environments. We also are especially interested in working with cloud service providers to assess the vulnerability of their platforms to these attacks. Understandably, it is unacceptable to run such experiments without the permission and cooperation of the cloud service provider. We hope that these results highlight that users should have the right to ask cloud service providers to document what additional defenses - either prevention or detection - if any they are providing to protect users from these types of attacks on their systems.

7 Conclusion

This study demonstrates the degree to which virtual switches are vulnerable to Layer 2 network attacks. The Layer 2 vulnerabilities described in this article are directed towards the virtual networking devices and not the hypervisor and without additional mitigation or preventive measures, could be performed on any host running a virtual switch including in a multi-tenant environment. Further study is necessary in order to perform a full Layer 2 security assessment on the state of virtual networking devices. The information could then be used to develop hardening and mitigation techniques focused on securing virtual networks against common Layer 2 networking threats. In their current state, virtual switches pose the same liability as their physical counterparts in terms of network security. However, the lack of sophisticated Layer 2 security controls like those present on enterprise grade physical switches increase the level of difficulty in securing these environments. One malicious virtual machine performing a MAC flooding or DHCP attack against the virtual network could be able to manipulate or eavesdrop on all traffic passing over the virtual switch, potentially compromising the confidentiality, integrity, and availability of co-located clients.

References

Altunbasak, H., Krasser, S., Owen, H. L., Grimminger, J., Huth, H.-P. and Sokol, J. (2005), Securing layer 2 in local area networks, *in* 'ICN'05 Proceedings of the 4th international conference on Networking - Volume Part II', pp. 699–706.

Cisco Systems Inc. (2002), 'Vlan security white paper [cisco catalyst 6500 series switches]'. Retrieved 22 August 2014.
URL: *http://www.cisco.com/warp/public/cc/pd/si/casi/ca6000/tech/stake_wp.pdf*

Cisco Systems Inc. (2014), 'Cisco nexus 1000v series switches for vmware vsphere data sheet'. Retrieved 22 August 2014.
URL: *http://www.cisco.com/en/US/prod/collateral/switches/ps9441/ps9902/data_sheet_c78-492971.html*

Cisco Systems Inc. (2015), 'Catalyst 6500 release 12.2sx software configuration guide'. Retrieved 20 June 2015.
URL: *http://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst6500/ios/12-2SX/configuration/guide/book/pref.html*

LAN MAN Standards Committee (2004), *IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges*, The Institute of Electrical and Electronics Engineers, Inc., New York, NY.

Lauerman, K. and King, J. (2010), 'Stp mitm attack and l2 mitigation techniques on the cisco catalyst 6500'. Retrieved 22 August 2014.
URL: `http://www.cisco.com/c/en/us/products/collateral/switches/catalyst-6500-series-switches/white_paper_c11_605972.pdf/`

Microsoft (2008), 'What is server core?'. Retrieved 22 August 2014.
URL: `http://msdn.microsoft.com/en-us/library/dd184075.aspx`

Microsoft (2013), 'Hyper-v virtual switch overview'. Retrieved 22 August 2014.
URL: `http://technet.microsoft.com/en-us/library/hh831823.aspx`

National Vulnerability Database (2014), 'Vulnerability summary for cve-2014-6271'. Retrieved 20 April 2015.
URL: `http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2014-6271`

Pettit, J., Gross, J., Pfaff, B., Casado, M. and Crosby, S. (2010), Virtual switching in an era of advanced edges, *in* 'ITC 22 2nd Workshop on Data Center - Converged and Virtual Ethernet Switching (DC-CAVES)'.

Pfaff, B. (2015*a*), '[ovs-dev][patch] mac-learning: Implement per-port mac learning fairness.'. Retrieved 11 February 2015.
URL: `http://openvswitch.org/pipermail/dev/2015-February/051201.html`

Pfaff, B. (2015*b*), '[ovs-dev][patch] mac-learning: Implement per-port mac learning fairness.'. Retrieved 11 February 2015.
URL: `http://openvswitch.org/pipermail/dev/2015-February/051228.html`

Pfaff, B., Bull, R. and Jackson, E. (2015), 'mac-learning: Implement per-port mac learning fairness, openvswitch/ovs - github'. Retrieved 11 February 2015.
URL: `https://github.com/openvswitch/ovs/commit/2577b9346b9b77feb94b34398b54b8f19fcff4bd`

Pfaff, B., Pettit, J., Koponen, T., Amidon, K., Casado, M. and Shenker, S. (2009), Extending networking into the virtualization layer, *in* 'HotNets-VIII'.

Red Hat, Inc. (2015), 'Selinux user's and administrator's guide'. Retrieved June 26th 2015.
URL: `https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/SELinux_Users_and_Administrators_Guide/chap-Security-Enhanced_Linux-Introduction.html`

Seifert, R. and Edwards, J. (2008), *The All-New Switch Book*, Wiley Publishing, Inc., Indianapolis, Indiana.

VMware Inc. (2012), 'Vmware vsphere end user license agreement'. Retrieved 22 August 2014.
URL: `http://www.vmware.com/download/eula/esxi50_eula.html`

Yeung, K.-H., Fung, D. and Wong, K.-Y. (2008), Tools for attacking layer 2 network infrastructure, *in* 'IMECS '08 Proceedings of the International MultiConference of Engineers and Computer Scientists', pp. 1143–1148.