Media Access Control Address Spoofing Attacks against Port Security

Andrew Buhr, Dale Lindskog, Pavol Zavarsky, Ron Ruhl
Concordia University College of Alberta
andr3wbuhr@gmail.com,
{dale.lindskog,pavol.zavarsky,ron.ruhl}@concordia.ab.ca

***Abstract*— In this paper we describe three separate Media Access Control (MAC) address spoofing attacks that, when deployed in specific yet common layer 2 network topologies, circumvent Cisco's port security. We show first that, with full knowledge of the network, the vendor recommended implementation of port security is both ineffective at preventing all three of these attacks, and actually decreases the difficulty of performing two of them. Next, we re-examine the attacks under less ideal conditions and demonstrate that they are feasible. Finally, we describe mitigation strategies that reduce the likelihood of success, but we argue that the use of port security as a preventative measure is difficult and may require tradeoffs between security and performance, flexibility, administrative cost, and ease of use.**

Keywords-port security; spoofing attacks; mitigation stratigies

I. INTRODUCTION

As implemented by Cisco[1], port security is a restrictive control applied directly to one or more edge interfaces [2]. Port security was originally intended as a control to mitigate Content Addressable Memory (CAM) overflow attacks, and has since been recommended as a control that mitigates MAC address spoofing attacks [10]. Port security enhances security in an IEEE 802.1D controlled Ethernet broadcast domain by restricting input to interfaces [2]. Input to an interface is restricted by comparing source MAC addresses with other learned or configured MAC addresses in the switch address table.

For interfaces *not* configured with port-security, the address table is populated by means of a switch learning process that allows switches to efficiently associate network nodes with interfaces, by observing the source MAC address ingress on those interfaces. This in turn creates a quickly aging entry in the address table with a 1-N (interface-MAC) relationship where N can be null. For ease of contrast, we will often refer to these as *non-secure* interfaces, the *non-secure* switch learning process, *non-secure* MAC addresses, and *non-secure* address table entries. When not qualified, the reader may assume that interfaces, MAC addresses, the learning process, and table entries are non-secure.

When port security *is* enabled on an interface, the switch considers that particular interface to be a *secure* interface and associates it with *secure* MAC addresses. This alters the behaviour of the learning process. This *secure* learning process creates a (usually non-aging) 1-M (interface-MAC relationship) *secure* entry where M is the maximum number of secure MAC addresses. Secure MAC address entries in the address table take precedence over non-secure MAC address entries.

Regardless of whether port security is enabled, when a switch forwards a frame, that frame is destined either to a *directly* connected network node, or to an *indirectly* connected network node. For the former, no intermediary switches exist, while for the latter, there is at least one intermediary switch between this switch and the network node. Even though the address table contains all information necessary for a switch to forward frames to network nodes, switches are not aware of whether the network node they are forwarding frames to is directly or indirectly connected. They simply rely on intermediary switches (if there are any) to forward a frame until it reaches the destination network node.

The switch address table is local to an individual switch, even when multiple switches are interconnected. In order to maintain a loop free, tree like broadcast domain, switches will never duplicate MAC addresses on a *local* address table entry when an identical source MAC address is observed on different ingress interfaces [1]. Instead, for non-secure MAC address entries, switches will replace MAC addresses in the address table on a last observed basis, in order to permit topology reconfiguration and the moving of network nodes, while secure MAC address entries will be removed only when the network cable is unplugged.

According to Cisco, there are two separate circumstances where port security will restrict input to a secure interface (hereafter referred to as *violation conditions*). The first violation condition is satisfied when

(1) "The maximum number of secure MAC addresses have been added to the address table, and a station whose MAC address is not in the address table attempts to access the [secure] interface" [2]. Two caveats are in order concerning violation condition (1). Firstly, it does not require that the first secure MAC address added to the address table be the legitimate MAC address of the station. Secondly, there exists no mechanism to ensure that the legitimately connected node registers their MAC address immediately. These two caveats will take on importance in Section 2, and will be further discussed in 4.2.

(2) The second violation condition of port security is satisfied when "An address learned or configured on one secure interface is seen on another secure interface in the same VLAN" [2]. Note that violation condition (2) applies only when *both* interfaces are secure interfaces. Even though it is supported by Cisco, in an Enterprise environment, interfaces (including VLAN trunks) which interconnect multiple switches within a single broadcast domain should

[1] Although we concentrate in this paper on Cisco's implementation of port security, much of our discussion applies to other vendors' implementations.

not be secure interfaces [3][4], and therefore are susceptible to MAC address spoofing of indirectly connected network nodes. These considerations will also take on importance in Section 2.

Publically available Cisco documentation on security best practices related to network controls make the following recommendations: for a "dynamic environment, such as an access edge, where a port may have port security enabled with the maximum number of [secure] MAC addresses set to one, enable only one [secure] MAC address to be dynamically learnt at any one time" [3][4]. In the attacks described in this paper, we assume port security is enabled with this configuration.

Enabling port security using the vendor recommended configuration appears reasonable for a dynamic environment. However, we will show that this configuration of port security actually decreases the difficulty of performing certain attacks. This is because port security locks a secure MAC address to a specific secure interface, which in turn removes certain race conditions troublesome to an attacker. On the other hand, we argue that, when port security is configured against these vendor recommendations, there are significant additional costs or tradeoffs incurred.

In the next section we describe three specific attacks that, although derived from previous research [5][6][7], are enhanced to be effective against circumventing the vendor recommended configuration of port security in specific, yet common, network topologies. In Section 3, we expand on the practical limitations of these attacks. Finally, in Section 4, we propose mitigation strategies and elaborate on the consequences and tradeoffs resulting from alternative non-recommended configuration of port security.

II. ATTACKS

In this section we describe three specific attacks which circumvent port security, when those attacks are executed in a multiswitch broadcast domain network topology. We begin with a set of assumptions which we believe are reflective of most enterprise environments. After each attack is described in detail, we compare attack difficulty with and without port security enabled, and show that port security actually makes MAC address spoofing easier for the attacker. Finally, for each attack, we describe how it is limited by the number of network nodes within the broadcast domain.

A. Assumptions

As we describe these three attacks, we make two separate assumptions closely related, respectively, to Cisco's two violation conditions of port security (see Section 1).

(1) We assume that an attacker either has not already registered a secure MAC address in the address table, or else is able to remove that already registered secure MAC address. In the case where an attacker has already registered a secure MAC address, there are two possible methods to remove that secure MAC address from the address table. The first method is to disconnect, and then reconnect the network cable. The second method relies on the secure aging timer of the address table, if an aging timer has been configured. Although both methods can be controlled through the use of the sticky feature, this is not one of Cisco's recommendations, and we believe drawbacks for this control exist. We will assume in this section that the sticky feature is not enabled and discuss these drawbacks in Section 4.

(2) We assume that port security has not been enabled on switch interconnecting interfaces. One of the main reasons why these interconnecting interfaces should not enable port security is because edge switches cannot inform intermediary switches that a secure MAC address has left the broadcast domain, i.e., there is no suitable secure inter-switch deregistration mechanism. If a network node moves to a secure interface on a different switch, this will satisfy violation condition (2) because the secure MAC address is retained within intermediary switches in the broadcast domain. We will further discuss the implications of implementing port security on interconnecting interfaces in Section 4.

One final, minor assumption is also necessary. Most modern operating systems require elevated privileges, either to send a specially crafted Ethernet frame, to change the MAC address on the Ethernet card, or to put the Ethernet card into promiscuous listening mode. We assume the attacker already has full administrative access to the operating system.

B. Attack 1 – Impersonation with two edge switches

Suppose we have two edge switches interconnected with a distribution switch (see Figure 1). An attacker, *A*, is connected to the first edge switch, *E1*, along with a primary victim, *V1*. There is also a secondary victim, *V2*, located on the second edge switch *E2*. As noted, we assume that port security is enabled only on the two edge switches (see Subsection 2.1 above)[2]. Interface numbers are incremented from left to right, where edge interfaces begin with *Fa* and interconnecting interfaces begin with *Gi*. (For example, *A* is on *Fa0/2* of *E1* and *D1* is connected to *E2* on *Gi0/2*.)

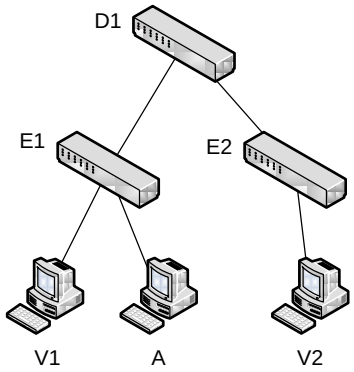

Figure 1. Attacks 1 & 2.

[2] This first attack is not a completely new idea, but the context is. The original attack was proposed by Wilkins et al, with a working proof of concept. In his research, he mentioned that his "invisible traffic redirection" would work on port security "only on reboot or cable disconnect" [5]. This is not entirely true, as the configuration of port security and the topology of the network plays a very important role. Perhaps he was already aware of this, but he did not explain. We believe this is worth the explanation.

Suppose *A* is promiscuously listening on its Ethernet card and suppose that *V1* communicates with *V2* with some frequency. Provided our attacker is moderately patient, there will probably be some time where *A* sees *V1* send a local broadcast frame containing an ARP-Request:

V1 says, who has V2 IP Address, tell V1 (Broadcast)
V2 says, V2 IP Address is at V2 MAC Address (unicast)

The ARP-Reply that *V2* sends to *V1* will cause table updates in each forwarding switch. (The same process has already occurred in the other direction because of the ARP-Request). The address table of *E2* will be updated with a secure entry for *V2* on *Fa0/1*, if it has not already been updated. Also, the address table of *D1* will be updated with a non-secure entry for *V2* on *Gi0/2*. Finally, the address table of *E1* will be updated with a non-secure entry for *V2* on *Gi0/1* (but crucially, and as we will see below, the address table of *E1* will *not* be updated with a secure entry for *V2*). Table 1 shows the address table of *E1* at this point.

TABLE I. INITIAL ADDRESS TABLE OF E1

VLAN	MAC Addr	Type	Ports	Secure
1	*V1*	DYNAMIC	*Fa0/1*	Yes
1	*V2*	DYNAMIC	*Gi0/1*	No

When *A* sees the ARP-Request from *V1*, *A* responds with the identical ARP-Reply message, encapsulated in the identical frame that *V2* responds with. (The order in which these frames are received is not important.) *V1* will be unaware, when it receives two copies of the same ARP-Reply, that one originated from *A*. The result of *A* sending this frame sourced from *V2* is a new entry in the table of *E1* mapping *V2* to *Fa0/2*.

The reason why *A* can send a frame sourced from *V2* and not throw a port security violation is because violation condition (2) requires that *both* interfaces concerned be secure interfaces. The interconnecting interfaces of *D1* cause the interface mapping of the secure MAC address of *V2* to apply only to *E2*. Before *A* poisons the address table with the ARP-Reply, *E1* will not have a secure address table mapping to *V2*. And even though *E1* already had a non-secure entry mapping *V2* to *Gi0/1*, that non-secure entry will be overridden by the secure entry created by *A*'s ARP-Reply. It will be overridden because secure entries take precedence over non-secure entries[3].

TABLE II. RESULTING ADDRESS TABLE OF E1

VLAN	MAC Addr	Type	Ports	Secure
1	*V1*	DYNAMIC	*Fa0/1*	Yes
1	*V2*	DYNAMIC	*Fa0/2*	Yes

[3] In our experiments we found that some older IOS versions implemented this logic slightly differently. The secure entries in the address table would still take precedence, but the address table would not insert a new entry if a non-secure entry already existed in the table. This required a minor modification of the attack. We frequently poisoned the address table until the entry timed out through either an STP topology change (cable unplug) or because of the default aging timer of 5 minutes.

Table 2 shows the address table of *E1* after *A* sends a forged frame. At this point, all traffic sent from any node on *E1* (including *V1)* to *V2* will arrive at *A* on Fa0/2, which of course can be leveraged for various purposes. For example, suppose that the first frame *V1* sends unknowingly to *A* contains a TCP SYN packet. The attacker could respond with a TCP SYN+ACK packet, and establish a connection with *V1*. The end result is server impersonation of *V2* from the perspective of *V1*. (This could be achieved simply by changing the MAC address and IP address on the Ethernet card of *A* so as to impersonate *V2*.)

Port security actually makes this attack easier. To show this, let us now suppose that port security is not enabled on any switch. The attack is still possible but more difficult because a race condition is introduced. When *A* sends a frame sourced from *V2*, it will update the address table of *E1* with a non-secure entry for *V2* on *Fa0/2*. However, after the frame is sent this begins the race. The race is won by *A* when the address table entry on *E1* for *V2* is *Fa0/2*. Then frames destined to *V2* from *V1* will be forwarded to *A*. The race is lost when *V2* sends a frame which reaches *E1* and updates the address table. As a result, the race constantly restarts and *A* will need to continually send frames sourced from *V2*. Moreover, if *V2* sends a frame to *V1* and the race is previously lost, that frame will be delivered to *V1*, which may cause communication problems between *V1* and *A*.

Even with port security enabled, there are limitations to this attack. Most importantly, the attacker cannot impersonate a network node which is directly connected to the same switch without throwing a violation (violation condition (2)). Moreover, the attacker cannot intercept communication between two indirectly connected network nodes. Therefore, the scope of this attack is limited to the number of indirectly connected network nodes multiplied by the number of directly connected network nodes. For example, Table 3 shows all possible outcomes if the attacker is placed on *E1* in a two edge switch topology, assuming an equal distribution of network nodes. This results in approximately half of the total network nodes being susceptible to impersonation, but yields only a quarter of the total communication streams because impersonation occurs in a unidirectional fashion.

TABLE III. TWO EDGE SWITCH MATRIX

A	*V1*	*V2*	Result
E1	*E1*	*E1*	Port security violation
E1	*E1*	*E2*	Impersonate *V2* (*V1* perspective)
E1	*E2*	*E1*	Impersonate *V1* (*V2* perspective)
E1	*E2*	*E2*	No port security violation

Another limitation specific to our server impersonation example is that there may also be other network nodes on the same switch as the attacker that will attempt to communicate with the impersonated server, some of which may be in the middle of previously initiated communication. Furthermore, the impersonated server may need to impersonate multiple

listening services, since it may be unable to anticipate which services a client will attempt to connect to.

C. Attack 2 – Full MITM with two edge switches

Attack 2 is arranged exactly like Attack 1. The only difference is that the attacker possesses interface access to *E2*. Such access may be possible if, e.g., the attacker has access to a wiring closet, server room, or even an adjacent office cubical. In the previous attack, the attacker was constrained to forging MAC addresses of victims not currently connected to the same switch. If the attacker wanted to relay frames sourced from *V1* to *V2*, a violation would occur (violation condition (2)). The additional interface access in Attack 2 allows the attacker to perform a full Man in the Middle (MITM) attack by relaying traffic using two interfaces as the transport medium. [4]

This attack begins exactly like Attack 1. The attacker patiently waits for an ARP-Request from *V1* to *V2* and poisons the address table of *E1* using the same method. However, *V2* will likely not send an ARP-Request for *V1* because it has already received and cached the MAC address of *V1* from the ARP-Request *V1* sent. To accommodate this, the attacker can replay the ARP-Request it received from *V1* out the interface connected to *E2*. This replayed ARP-Request will result in the opposite poisoning of the address table for *E2*. That is to say, *E2* will have a secure address table mapping for *V1* on the interface connecting to *A*. The end result is a full MITM attack, because frames sent from *V1* to *V2* will be forwarded to *A*, who may then forward them to *V2*. Conversely, frames sent from *V2* to *V1* will be delivered to *A*, who may then forward them to *V1*. Of course, the attacker may also choose to observe or modify frames in transit.

This attack may be detected because the replayed ARP-Request that the attacker sends to *V2* will generate an unsolicited ARP-Reply to *V1*. Alternatively, the attacker could send a UDP frame sourced from *V1* destined to *V2* on *E2*, but this may also generate an unsolicited reply to *V1*. If the attacker sent a frame with an unknown destination MAC address, all network nodes, including *V1*, would see the frame. There appears to be no simple mechanism to avoid detection. However, in our experiments on an older Cisco Catalyst 3550, we found that sending jumbo Ethernet frames would update secure address table entries without forwarding the frame, i.e., the switch receives and processes the frame but does not send it to any other network nodes. (This effect could not be reproduced on a newer Catalyst 2960.) This method could be used to avoid detection.

As with Attack 1, this attack is easier with port security enabled. When port security is not enabled on any switch, two race conditions are introduced. Both race conditions are structurally identical to the one described in the previous attack, but each race condition applies only to one of the two sides of the communication channel. When one race on either side is lost, the communication on that channel would resume normal behaviour and likely result in an erroneous communication stream. Therefore, because an additional race condition is introduced, maintaining a full MITM communication stream for a longer period of time becomes correspondingly more difficult.

Compared with Attack 1, the potential victims in Attack 2 are doubled, again assuming an equal distribution of network nodes. The attacker has the ability to impersonate all of the network nodes but only half of the total communication streams, because directly connected network nodes communicating with other directly connected network nodes still cannot be impersonated without throwing a violation (violation condition (2)).

D. Attack 3 – Full MITM with three edge switches

Suppose we have 3 edge switches all connected to one distribution switch in a hub and spoke fashion (see Figure 2). The first victim, *V1*, is located on the first edge switch, *E1*. The attacker, *A*, is located on the second edge switch, *E2*. Finally, the second victim, *V2*, is located on the third edge switch, *E3*. Again, we assume that port security is enabled on all three edge switches as per Cisco's recommendations (see Section 1 above). Interface numbers are incremented from left to right where edge interfaces begin with *Fa* and interconnecting interfaces begin with *Gi*.

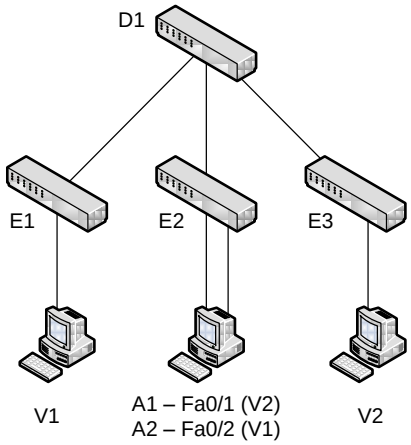

Figure 2. Attack 3.

For this attack, the attacker does not need to be in as privileged of a position as in the previous MITM attack (Attack 2). In particular, the attacker does not require interface access on more than one switch. The only requirement for this attack is that the attacker either has access to two interfaces on *E2*, or else port security is misconfigured to allow two or more secure MAC addresses. (We will henceforth ignore this second alternative, since it is inconsistent with Cisco's recommendations.) The reasoning behind this is that, unlike the previous attacks, the attacker is on a different switch than either of the two victims, and therefore can forge frames with source MAC addresses from either victim without port security throwing a violation. In order to clearly identify which interface the attacker is

[4] Wilkins et al. also notes that a potential MITM attack with a dual-homed attacker is possible. We are not sure whether he believed this MITM is possible with port security enabled, because he does not elaborate any further [5].

sending frames from, we will refer to *Fa0/1* and *Fa0/2* on *E2* as *A1* and *A2* respectively (see Figure 2).

In this attack, unlike others, the attacker communicates with the distribution switch. This presents a problem for performing a full MITM attack. As frames are relayed between the attacker and the victims, the non-secure entries of the address table for the distribution switch will be in constant flux. This flux introduces race conditions which were not present in Attacks 1 and 2 (at least when port security was enabled). There are some interesting methods we can use to manipulate the table on demand, and the manipulations themselves in turn cause race conditions. Removing port security from all switches in this scenario has no effect on the difficulty of the attack. The race conditions involved do not change and no new race condition is introduced on *E2*, because both victims are located on separate indirectly connected switches. We will describe this attack in a stepwise fashion.

1) *A* sees an ARP-Request originating from *V1*, and asking for the MAC address of *V2*.

V1 says, who has V2 IP address, tell V1 (Broadcast)

This initiates Race Condition 1 (*RC1*), which begins in an unfavourable state for *A*. Let us say that *A* is winning *RC1* when an entry for *V2* on *D1* is *Gi0/2*, and losing whenever *V2* communicates with *D1*, causing this entry to be overwritten.

2) *A1* responds with the identical (unicast) ARP-Reply frame that *V2* responds with. Unlike the previous two attacks (with port security enabled), this frame must be received by *D1 after* the ARP-Reply has been received from *V2*. Therefore, *A1* may send this frame multiple times to ensure *A* is winning *RC1*. Moreover, subsequently *A* may not be able to determine whether they are losing *RC1* due to *V2*'s involvement in unicast communication over *D1*. Therefore, *A1* may continually update the address table of *D1* if it is believed that *V2* will be frequently communicating with other network nodes through *D1*.

3) *V1* sends a frame destined to *V2* which, assuming *A* is winning *RC1*, is forwarded to *A1*. Once *A* receives the frame, *RC1* ends for now (assuming *V1* has only one frame to send to *V2*).

4) *A2* rebroadcasts the same ARP-Request through *Fa0/2*:

A2 (V1) says, who has V2 IP address, tell V1 (Broadcast)

This is required to ensure *D1's* address table entry for *V2* points to *E3*; otherwise the frame would not be delivered. This begins Race Condition 2 (*RC2)*, where *A* is winning *RC2* when *Gi0/2* on *D1* maps to *V1*'s MAC address, and losing when *V1* communicates with *D1*. *A2* can learn that *A* is winning *RC2* if it receives the ARP-Reply.

5) *A2* relays the captured frame to *V2* forged from *V1*. *V2* processes the frame and replies to *A2*, assuming that *A* is winning *RC2*.

6) *A2* receives the frame, thus terminating *RC2* for now, and sends an ARP-Request to *V1* from *A1*:

A1 (V2) says, who has V1 IP address, tell V2 (Broadcast)

The ARP-Request has the same purpose as step 4 above (so that *V1* can receive frames), but it also serves to remap *V2* on *D1* to *Gi0/2* and thus restarts RC1.

7) *A1* sends the frame to *V1*. This frame restarts the process from step 3 onward and, incidentally, does not require *A1* to send an extra ARP-Request, because *A* is now winning *RC1* again.

The number of potential victims in Attack 3 are reduced compared with Attack 2 (Section 2.3) but still higher than Attack 1 (Section 2.2), assuming an equal distribution of network nodes. As shown in Table 4, two thirds of total potential victims can be impersonated. If an attacker has access to an additional interface on the same switch or a different switch, 22% of total potential victims can have a MITM attack performed against them.

TABLE IV. THREE EDGE SWITCH MATRIX

A	***V1***	***V2***	**Result**
E1	*E1*	*E1*	Port security violation
E1	*E1*	*E2*	Impersonate *V2* (*V1* perspective)
E1	*E1*	*E3*	Impersonate *V2* (*V1* perspective)
E1	*E2*	*E1*	Impersonate *V1* (*V2* perspective)
E1	*E3*	*E1*	Impersonate *V1* (*V2* perspective)
E1	*E2*	*E2*	No port security violation
E1	*E3*	*E3*	No port security violation
E1	*E2*	*E3*	Impersonate *V1** (*V2* perspective)
E1	*E3*	*E2*	Impersonate *V2** (*V1* perspective)

III. ATTACK LIMITATIONS

In this section we discuss the practical feasibility of our attacks, depending on the amount of knowledge an attacker possesses about the target network. Obtaining knowledge of the broadcast domain can be performed by either technical or non-technical means, and we focus on the former.

Suppose an attacker is placed in a topology like Figure 1 (Section 2.2), randomly and without any knowledge of which network nodes are directly or indirectly connected to either edge switch. Assuming that both switches have an equal number of network nodes (excluding the attacker, to present rounded numbers), there is a 50% probability that an attacker's randomly forged MAC address will be that of a directly connected node, and will therefore trigger a violation (violation condition (2)). However, as we increase the number of edge switches, like we do in Figure 3 (Section 2.4), the probability of randomly forging a MAC address of a directly connected node decreases, and therefore so does the probability of a violation. Assuming equally placed network nodes, the probability of violation is decreased to 33% for Figure 3. The chance of an attacker triggering a port security violation decreases as the number of edge switches increases.

There are techniques which may be used to further reduce the chance of violation. Although obtaining information about the location of network nodes in a broadcast domain is difficult, this difficulty is eased when a network has similar devices and hardware, or by using

operating system fingerprinting. For example, an attacker can use ICMP Echo-Request probes to determine the average round trip time over a large sample of a broadcast domain. Although operating system kernels tend to prioritize ICMP Echo-Requests differently, an attacker can use various operating system fingerprinting techniques, such as observing the IP TTL of ICMP Echo-Replies, to group similar operating systems. After grouping similar operating systems together, an attacker can compare the average round trip time of all network nodes.

Strictly speaking, it is not possible to be certain about whether the average varying delay is caused from a long network cable or from latency of a switch forwarding frames. However, as a general rule, an attacker can assume that each intermediary switch will add some latency. So although this technique does not provide any definitive means for determining where the network node is located, an attacker can use this technique to decrease the probability of triggering a violation, by choosing network nodes where communication has a higher than average round trip latency.

There are other techniques an attacker may use to help reduce the amount of error introduced when performing reconnaissance on a broadcast domain. For example, an attacker may observe the IP ID within ICMP Echo-Replies to look for inconsistent increments, which suggest the network node is involved in frequent network communications with other network nodes, which in turn may cause ICMP Echo-Reply delay [8].

The IP ID technique is actually doubly useful because it may also help determine which network nodes are clients and which network nodes are servers. Generally speaking, servers will have higher IP ID increments than clients. Another technique an attacker may use to gather this information is to promiscuously observe ARP-Requests. Generally speaking, clients are more likely to send ARP-Requests to servers. It is often important for an attacker to determine which network nodes are clients and which are servers. Impersonating a server as opposed to a client will usually have a much greater impact, since the attacker will probably intercept network traffic from more network nodes.

IV. Defences and Countermeasures

In this section we look at three separate, practical methods for mitigating these attacks. We consider both the effectiveness of these methods, and their consequences for performance, flexibility, administrative cost, and ease of use.

A. Interconnecting Switch Port Security

For most enterprise networks, we do not recommend enabling port security on either end of interconnecting interfaces. Doing so does, however, prevent our attacks, assuming the control is applied to all switch interconnections within a broadcast domain. This is because the secure entries in the address table will be shared amongst all switches within the domain. There are four reasons why it may be difficult to properly enable port security on switch interconnecting interfaces:

1) Port security does not support Etherchannel [2][9]. This is likely the result of an interoperability issue caused by MAC addresses being load balanced across multiple interfaces. Etherchannel is commonly used to combine multiple Ethernet interfaces and interconnect multiple switches with high aggregate bandwidth and primitive load balancing. Removing Etherchannel in order to support interconnecting switch port security also means removing the benefits networks may receive from the overall low cost of providing high aggregate bandwidth and rudimentary redundancy between interconnecting interfaces.

2) Spanning Tree Protocol (STP) is not interoperable with port security on interconnecting interfaces [9]. When a topology change beyond the edge occurs within the distribution layer, the end result is a violation (violation condition (2)) on non-root bridge interfaces previously in the blocking state. In order to prevent a violation, STP would need to provide a method to inform the switch to reset secure MAC addresses during a topology change. This reset would be difficult to do without potentially allowing for an attack during the topology reconfiguration process.

3) Enabling port security on interconnecting interfaces causes network node relocation problems. Dynamically configured port security can inform only the directly connected switch about an interface removal. To prevent a violation when a secure MAC address is relocated within the distribution layer, the address table entry for that relocated secure MAC address must either time out within the rest of the infrastructure or must be manually cleared. Enabling aging timers on the interconnecting interfaces actually makes the situation worse because an attacker can take over an adjacent victim after the timeout occurs. The attacker could then impersonate the adjacent victim or perform a Denial of Service (DoS) attack against the interconnecting interface itself. A potential solution to this problem would be to introduce a deregistration mechanism into port security. To ensure deregistration is authentic, this would require utilizing or creating protocols where messages can be sent or received only from trusted network switches. Two possible ways to do this is to either utilize a private link, or digitally sign the deregistration message.

4) Interconnecting switch port security increases risk to the infrastructure. Port security was designed as a control to be applied as closely to the threat as possible, on edge interfaces. Implementing port security on interconnecting switch ports increases the severity of accidental violations. On edge interfaces, the impact of a violation is limited to the individual interface, but when port security is implemented on interconnecting interfaces, a single violation could result in blocking large sections of the broadcast domain.

B. Port Security Sticky

When port security sticky is enabled, a secure MAC address is *not* removed from the address table after a network node has disconnected the network cable from the switch. This is significant because we previously assumed in our attacks that, if an attacker already registered a secure MAC address, then they were capable of removing that secure

MAC address. One method of removing that secure MAC address is to unplug the network cable. With port security sticky enabled, this method is no longer available to the attacker. It is still possible, however, that an attacker has not already registered the maximum amount of allowable secure MAC addresses in the address table. Therefore, although port security sticky does help mitigate the attacks we describe, it will not completely prevent these attacks.

Moreover, enabling port security sticky in an enterprise environment is usually not cost effective. This feature requires administrative intervention every time a network node moves to a new interface, e.g., by means of a manual change control procedure. This may help further mitigate our attacks, by helping to ensure that the legitimately connected node registers their MAC address immediately. But it does so at a greater administrative cost, because the secure MAC address locking mechanism of port security sticky, we feel, undermines the cost effective nature of the secure learning process. Another possibility is that a separate authentication protocol could be used in conjunction with port security sticky, to ensure a secure MAC address is immediately registered; this option falls outside the scope of our research.

C. *Segregation of Multiswitch Broadcast Domains Mitigation Strategy*

Our preferred method for mitigating the attacks described in this paper is to eliminate multiswitch broadcast domains. This will prevent the attacks we describe, with one unlikely exception, where an attacker impersonates network nodes that have been removed from the network.

However, multiswitch broadcast domains are often desirable, because they allow enterprises the flexibility to logically group network nodes based on similar policy requirements regardless of physical location. Enterprises that use port security must be willing to give up this flexibility in order to prevent our attacks. Since such a radical solution is often undesirable, we propose a strategy for enterprises which, although it does not eliminate multiswitch broadcast domains, it does help further mitigate attacks against port security. The strategy we propose is to segment network nodes based on trust and role. This strategy is based on three interrelated tactics:

1) Segregate trusted from untrusted network nodes. Untrusted network nodes, e.g., mobile or temporary users, should be placed into separate broadcast domains, away from trusted network nodes. Untrusted network nodes can, therefore, attack only other untrusted nodes.

2) Segregate untrusted network nodes into single switch broadcast domains. It may be practical for an enterprise to segregate *only* the untrusted nodes into single switch broadcast domains. This approach works well if multiswitch broadcast domains are used only for layer 2 redundancy, and not for affording mobile workers the flexibility to move around the office and yet remain within the same broadcast domain.

3) Segregate trusted network nodes based on user or server roles. Server network nodes should be placed into single switch broadcast domains separate from user network nodes, because they are an attractive target for attackers. When servers are separated from user network nodes, but cannot be placed into separate single switch broadcast domains, they should be further segregated into separate multiswitch broadcast domains based on similar service roles. This approach limits the broadcast domain attack exposure if a server were to be compromised. Another approach is to implement static or sticky port security for critical servers. This approach is consistent with Cisco's recommendations [3][4], but may limit redundancy or failover solutions.

In general, segregating multiswitch broadcast domains has an additional benefit because segregation causes the size of the broadcast domain to be reduced. Reducing the size of a broadcast domain will in turn reduce the number of network nodes which can be attacked. Wherever possible, multiswitch broadcast domains should be reduced in size as much as possible to limit the attack surface.

The segregation of broadcast domains is unfortunately a compromise, because the extra routing introduced normally increases the cost, and limits both flexibility and performance. However, this strategy is still favoured over port security sticky (Section 4.2) because it does not impact the efficiency of the switch learning process.

V. CONCLUSION

Port security fails to mitigate the three MAC address spoofing attacks described in this paper, when those attacks are deployed in multiswitch broadcast domains. For two of those attacks, the way in which port security alters the switch learning process removes race conditions that would otherwise exist, and this makes these attacks easier with port security enabled than without. Victims of all three attacks must be indirectly connected network nodes, and attempts to attack directly connected network nodes will be detected by port security. We described various technical means an attacker may use to determine whether a potential victim is directly or indirectly connected.

Finally, we evaluated three methods for mitigating these attacks in enterprise environments. No method we considered was perfect: each method required significant sacrifice of performance, flexibility, administrative cost, or ease of use. Of these methods, we recommended the segregation of network nodes based on trust and role.

REFERENCES

[1] IEEE 802.1D Std, "IEEE standard for local and metropolitan area networks. Media Access Control (MAC) bridges," 2004.

[2] Cisco Systems, "Configuring port-based traffic control. Catalyst 3550 multilayer switch software configuration guide," ch 22, 2007.

[3] Cisco Systems, " Cisco SAFE reference guide," 2009.

[4] Cisco Systems, "Network security baseline," 2008.

[5] J. Wilkins and J.Taranis, "Invisible traffic redirection on Ethernet switches,"Proc. BlackHat USA Conference 2002, 2002.

[6] A. Ornaghi and M. Valleri, "Man in the middle attacks," Proc. BlackHat USA Conference 2003, 2003.

[7] Ettercap. Port stealing, Available: http://ettercap.sourceforge.net/

[8] H.D.Moore and V. Smith, "Tactical exploitation - The other way to pen-test," Proc. BlackHat USA Conference 2007. Available:

http://www.metasploit.com/data/confs/blackhat2007/tactical_blackhat2007.pdf

[9] HP Procurve, “Configuring and monitoring port security,” ch 9, 2005.

[10] Cisco Systems. “Cisco Catalyst 3750 series switches. Layer 2 security features on Cisco Catalyst layer 3 fixed configuration switches configuration”, 2007.