

Practical Switch Security

Author: Herbert Haas
Address: herbert AT perihel DOT at
<http://www.perihel.at/dcom>
Revision: 0.3
Date: 2009-06-17
Copyright: Copyright (c) 2007-2009 Herbert Haas.

Abstract

This document summarizes important facts about modern switch security. It is not a tutorial. The reader should already be familiar with security fundamentals. Besides theory, practical issues are exemplified on the basis of Cisco Catalyst products. If you find any mistakes please send me an E-Mail, **thanks!**

Contents

- 1 Port Security
 - 1.1 Basic Idea
 - 1.2 Configuration
 - 1.3 Stop flooding
- 2 VLAN Access Lists
- 3 DHCP Snooping
 - 3.1 Basic Idea
 - 3.2 Configuration
 - 3.3 DHCP Starvation
 - 3.4 DHCP Information Option
- 4 Dynamic ARP Inspection (DAI)
 - 4.1 Basic Idea
 - 4.2 Configuration
- 5 IP Source Guard
- 6 BPDU Guard
- 7 Root Guard
- 8 Storm Control
 - 8.1 Basic Idea

- 8.2 Configuration
- 8.3 Blocking details
- 8.4 Note
- 9 802.1x Authentication
- 10 Protected Ports
 - 10.1 Basic Idea
 - 10.2 Configuration
 - 10.3 Note
- 11 Private VLANs
 - 11.1 Basic Idea
 - 11.2 Configuration
 - 11.3 Private VLAN Attack
- 12 VLAN Hopping Attack
- 13 Port Mirroring
- 14 MAC-Address Notifications

1 Port Security

1.1 Basic Idea

The basic idea of port security is to prevent **CAM-table overflow** attacks. In this attack, an attacker would send thousands of dummy frames with random source MAC addresses until the CAM-table of the switch is full and the switch starts to flood every subsequent frame. Finally the switch started to act like a hub and the attacker could sniff every frame.

There are two important thing to note:

1. CAM flooding only affects the VLAN the attacker is connected to. Other VLANs are not affected!
2. However, all switches configured with the attacked VLAN are affected.

1.2 Configuration

When port security is configured on an interface, only a limited number of MAC addresses are allowed. In detail:

1. Port security only works on **access ports** but also on non-negotiating trunks! Therefore start with the command:

```
(config-if)# switchport mode access
```

2. **Enable** port security:

```
(config-if)# switchport port-security
```

3. Optionally configure one or more **secure addresses** (i. e. allowed):

```
(config-if)# switchport port-security <mac-addr>
```

4. When other-than-secure MAC addresses are seen on the secure port or a secure address is seen on another port, a **port violation** occurred.

The default violation action is **shutdown**. Optionally change the violation action:

```
(config-if)# switchport port-security violation shutdown|protect|restrict
```

- Upon a violation, the **shutdown** mode sends an SNMP-trap plus a Syslog message, increments a violation counter, and sets the port into an **errdisabled** state until the administrator either enters a shut/no shut sequence or the command:

```
(config)# errdisable recovery cause shutdown
```

- The **protect** mode simply drops all frames from unknown source addresses. The port remains up.
- The **restrict** mode also simply drops all frames from unknown source addresses but increments a violation counter and sends an SNMP trap plus a Syslog message. The port remains up.

Note that a too high frame rate could overwhelm the CPU. Therefore the protect and restrict mode should be used together with a **rate limiting** configuration:

```
(config)# mls rate-limit layer2 port-security <packet-per-sec> [burst-size]
```

This command works before and after the violation. The range for <packet-per-sec> is {1...106 }, there is no default. The range for [burst-size] is {1...255}, the default is 10.

5. Optionally define the **number of secure addresses** allowed on this port:

```
(config-if)# switchport port-security maximum <max>
```

The switch will learn new secure addresses until the specified maximum is reached. If you already had defined some secure addresses manually, new addresses can only be learned if the specified maximum is greater than the number of manually configured addresses.

The range of the <max> parameter is {1...132}, the default is 1. When the link goes down, all dynamically learned secure addresses are removed.

6. Dynamically learned MAC addresses may **age out**. The aging process can be tuned via:

```
(config-if)# switchport port-security aging type absolute|inactivity
(config-if)# switchport port-security aging time <minutes>
```

where the parameter <minutes> may be in the range {1...1440}, the default is 0 (i. e. infinity).

Dynamically learned addresses can also be removed via the command:

```
# clear port-security dynamic
```

7. Some platforms allow **sticky learning**, that is the addresses are kept in the RAM and could even be permanently stored when the administrator enters **write mem**.

1.3 Stop flooding

Additionally it should be noted, that the switch can be told to avoid flooding in case the destination MAC address is unknown. This non-flood behavior can be enabled via the command:

```
(config-if)# switchport block multicast|unicast
```

2 VLAN Access Lists

VACLs or VLAN Maps are applied on frames entering VLANs. Here is an example:

```
switch(config)# vlan access-map MyVACL 10
switch(config-access-map)# match ip address 101
switch(config-access-map)# action forward
switch(config-access-map)# exit
switch(config)# vlan filter test vlan-list 40-48
```

All IP packets matching ACL 101 are forwarded, all others dropped by the implicit deny rule. This VACL is applied on the VLAN range 40-48.

3 DHCP Snooping

3.1 Basic Idea

By configuring DHCP Snooping, the switch analyzes DHCP messages and learns IP to MAC address bindings. This information can be verified using the command:

```
# show ip dhcp snooping binding
```

The main idea is to let the switch act like a firewall between **trusted** DHCP server ports and **untrusted** host-ports.

DHCP Snooping generates an alert when a **server-type DHCP message** arrives at an **untrusted** port. **DHCP request messages** are only forwarded to **trusted** ports.

3.2 Configuration

The configuration goes like this:

1. Enable DHCP Snooping globally:

```
(config)# ip dhcp snooping
```

2. Then enable DHCP Snooping for specific VLANs:

```
(config)# ip dhcp snooping vlan <vlan-id> [,...]
```

3. Official DHCP servers are connected to trusted ports:

```
(config-if)# ip dhcp snooping trust
```

On trusted ports, DHCP Snooping is not performed. Per default all other ports are untrusted.

4. Optionally configure a rate limit for DHCP messages (messages per second):

```
(config-if)# ip dhcp snooping limit rate <rate>
```

Per default any rate is allowed.

3.3 DHCP Starvation

Attacker tools like *gobbler* send a huge amount of DHCP requests, hereby grabbing all IP addresses from the DHCP server. Eventually this results in a DoS situation; other hosts cannot receive any IP address anymore.

Additionally, when the official DHCP server is out of order, the attacker could set up a 'rogue' DHCP server which announces its own address as default gateway or default DNS server.

If DHCP Snooping is already configured, the switch can also mitigate such attacks.

3.4 DHCP Information Option

Note that switches with DHCP Snooping enabled could send a **DHCP information option**, notifying the DHCP server which host MAC address is attached to which switch port. Many DHCP Servers have problems with that and therefore this feature should be disabled:

```
(config)# no ip dhcp snooping information option
```

4 Dynamic ARP Inspection (DAI)

4.1 Basic Idea

A classical **man-in-the-middle (MITM) attack** is based on manipulating the ARP caches of the local hosts and routers.

This is done when the attacker sends **Gratitious ARPs (GARPs)**, i. e. unsolicited ARP replies that bind the IP addresses of some target hosts (or routers) to the attacker's MAC address. Subsequently all interesting packets will be forwarded to the attacker.

If the switch is configured for Dynamic ARP Inspection, the switch checks whether the IP address and the MAC address in ARP reply messages actually belong together.

4.2 Configuration

For DAI the switch relies on two possible information sources:

1. An address table created by the **DHCP snooping** feature (see next section). If DHCP Snooping is configured, the following command enables DAI:

```
(config)# ip arp inspection vlan <vlan-id>
```

2. A manually configured **ARP Access-Lists**, using the following command:

```
(config)# arp access-  
list MyHosts permit ip host 10.3.3.42 mac host 0011.2233.4455  
(config)# ip arp inspection filter MyHosts vlan 1
```

Optionally, some ports (e. g. between switches) can be configured as **trusted ports**, that is no DAI is performed here:

```
(config-if)# ip arp inspection trust
```

5 IP Source Guard

Using the IP-to-MAC binding table created by DHCP Snooping, the switch can also check whether an observed source IP address is really located at that port. Thus IP address spoofing can be detected.

```
(config)# ip dhcp snooping
(config)# ip dhcp snooping vlan 10

(config)# interface f0/1
(config-if)# ip verify source
!
(config)# interface f0/2
(config-if)# ip verify source
```

It also works if DHCP is not available and a static IP-to-MAC binding has been configured. Note that even in this case **DHCP Snooping must be configured** on each switchport where the IP source guard is desired. Without the DHCP Snooping feature it simply would not inspect the packets.

```
(config)# ip source binding 0000.cafe.babe vlan 10 10.1.1.1 interface f0/1
```

```
# show ip source binding
```

| MacAddress | IpAddress | Lease(sec) | Type | VLAN | Interface |
|-------------------|-----------|------------|--------|------|----------------------|
| 00:00:CA:FE:BA:BE | 10.1.1.1 | infinite | static | 10 | FastEther- net0/1 |
| 00:00:DE:AD:BE:EF | 10.1.1.2 | infinite | static | 10 | FastEther- net0/2 |

Total number of bindings: 2

```
# sh ip verify source
```

| Interface | Filter-type | Filter-mode | IP-address | Mac-address | Vlan |
|-----------|-------------|-------------|------------|-------------|------|
| Fa0/1 | ip | active | 10.1.1.1 | | 10 |
| Fa0/2 | ip | active | 10.1.1.2 | | 10 |

6 BPDU Guard

This is a Cisco invention, available since IOS 12.1. The basic idea is to shut down an access port when a BPDU arrives.

In detail:

1. First recall that even 'PortFast' interfaces may receive BPDU and participate in spanning tree calculations.
2. But if `spanning-tree portfast bpduguard` has been enabled (globally!), and a BPDU is received at a PortFast-interface this interface transits in the **errdisable** mode.
3. Per default, an errdisabled interface remains in this state forever. Optionally the interfaces can be re-enabled automatically via:

```
(config)# errdisable recovery cause bpduguard
(config)# errdisable recovery interval 300 !!! seconds
```

7 Root Guard

Especially on trunks enable the Root Guard feature. This is also a Cisco invention, available since IOS 12.0. The basic idea is to shut down a port when a **better** BPDU arrives.

In detail:

1. Root Guard does not allow an interface to become an STP root port. The configuration is only:

```
(config-if)# spanning-tree guard root
```

2. Therefore, Root Guard should be enabled on all ports where the root bridge should **not** appear!
3. If a better BPDU arrives, the port transits into the **root-inconsistent** state, i. e. it is blocked until no more superior BPDUs are seen (internally the port transits to the listening state and after two forward-delay periods the port comes up again).

Also enable this feature on access ports where only clients should reside.

8 Storm Control

8.1 Basic Idea

When configured on a port the traffic is blocked when it exceeds a configured maximum rate. Traffic is always measured as packets per **1 second** measurement interval.

8.2 Configuration

Two different types of thresholds can be configured:

1. As percentage of the total port bandwidth
2. Traffic rate in packets per seconds (IOS 12.1(22)EA1 or later)

Optionally a **falling threshold** can also be configured. That is, traffic is blocked when the rising threshold is exceeded and the blocking stops when the rate falls below the falling threshold.

```
(config-if)# storm-
control broadcast level 80 70 ! rising and falling level as percentage
(config-if)# storm-control unicast level 85
(config-if)# storm-control multicast level pps 10000 8000 ! levels in packets per sec-
ond
```

8.3 Blocking details

- If the multicast traffic threshold is exceeded **all** incoming traffic (also broadcast and unicast) is blocked until the multicast traffic drops below the threshold.
- If unicast or broadcast traffic thresholds are exceeded only the specific traffic type (unicast or broadcast) is blocked.
- BPDUs are never blocked

8.4 Note

Storm control does not work if the ports are part of an Etherchannel (although it can be configured).

9 802.1x Authentication

Windows XP supports EAP-MD5 and Protected EAP (PEAP). PEAP is much more secure. Also use at least Service Pack 2 for Windows XP otherwise you could only use the login user name for 802.1x.

On the Catalyst the 802.1x configuration goes like this:

```
Switch(config)# aaa new-model
Switch(config)# radius-server host 10.5.5.42 auth-port 1812 key GeHeIm
Switch(config)# aaa authentication dot1x default group radius
Switch(config)# dot1x system-auth-control    !!! globally enable 802.1x !!!
Switch(config)# interface fa1/7
Switch(config-if)#dot1x port-control auto
```

A very **bad** feature is `dot1x multiple-host` which requires only one authenticated host on that port to allow all other hosts to utilize the switch port. Always demand each host to authenticate!

If you use an ACS:

- Global Authentication Setup: Enable PEAP
- Interface Configuration > RADIUS (Cisco IOS/PIX 6.x): Scroll down to the bottom and click on **Enable Authenticated Port cisco-av-pair**

On your Windows Client enable 802.1X:

Under Vista first start `services.msc` enable tab “Standard” click on “Wired Autoconfig”. Then you have an additional tab in your Windows network settings where you can configure 802.1X parameters.

10 Protected Ports

10.1 Basic Idea

This is the one-switch simplified PVLAN solution. There is no L2 traffic between **protected ports** except control traffic which is forwarded by the switch CPU such as PIM packets. Only traffic between non-protected ports and protected ports is possible (of course traffic between non-protected ports is always possible).

Typical application: Attach WLAN Access Points (APs) to protected ports and configure **Public Secure Packet Forwarding (PSPF)** on the APs. Then the APs only bridge traffic between wireless and wired ports, so WLAN clients cannot attack each other (ideal for WWW-access only at airports or in hotels for example). Because of the protected port feature the clients cannot attack each other even when they are associated to different APs.

Other applications (besides WLAN) are:

- Multiple customers/divisions share a common Internet access connection
- Multiple divisions need access to some servers but the divisions should not be able to communicate with each other.

10.2 Configuration

Very simple:

```
(config)# interface fastethernet 0/1
(config-if)# switchport protected
```

10.3 Note

- Also 802.1Q trunk ports can be configured as protected ports.
- If WLAN clients can be connected to APs that are attached to other switches then extend this concept over multiple switches.
- If the switch receives a frame from a non-protected port and does not know the destination MAC address the switch will flood this frame (as usual) through all ports, including the protected ports. If this is a security problem you can block flooding on protected ports via the commands:

```
(config-if)# switchport block multicast
(config-if)# switchport block unicast
```

11 Private VLANs

11.1 Basic Idea

The basic idea of Private VLANs is to prevent communication of hosts within the **same subnet** except with dedicated destinations such as firewalls or default gateways.

Now the VLAN notion is used differently as usual. We distinguish between **Primary** and **Secondary** VLANs. Devices connected to ports in Secondary VLANs can only reach devices connected to a port in a Primary VLAN. Devices in the Primary VLAN can reach every device in a Secondary VLAN, but there are some exceptions. There are two major types of Secondary VLANs:

- Isolated
- Community

For isolated ports the assertions above are true. Devices connected to isolated ports cannot communicate with each other. Their traffic can only be forwarded to the primary VLAN port they are associated to. For example servers should not talk to each other but should be reachable by the rest of the world. Therefore it is a good idea to attach these servers to isolated ports and the router or firewall to the primary VLAN port, also known as **promiscuous port**.

Devices on community ports can communicate with each other if they are in the same community VLAN. So devices in different community VLANs cannot talk to each other but may share the same promiscuous port.

Note that not all Cisco Catalyst switches support Private VLANs, especially the older and smaller ones.

11.2 Configuration

1. Define VLANs:

```
Switch(config)# vlan 501
Switch(config-vlan)# private-vlan isolated

Switch(config)# vlan 502
```

```
Switch(config-vlan)# private-vlan community

Switch(config)# vlan 503
Switch(config-vlan)# private-vlan community

Switch(config)# vlan 20
Switch(config-vlan)# private-vlan primary
Switch(config-vlan)# private-vlan association 501-503
```

2. Define host ports:

```
Switch(config)# interface fastethernet1/0/22
Switch(config-if)# switchport mode private-vlan host
Switch(config-if)# switchport private-vlan host-association 20 501    !!! pri-
mary=20, secondary=501
```

3. Define promiscuous ports:

```
Switch(config)# interface fastethernet 1/0/2
Switch(config-if)# switchport mode private-vlan promiscuous
Switch(config-if)# switchport private-vlan mapping 20 501-502
```

11.3 Private VLAN Attack

An attacker who resides on an **isolated port** could indeed send frames to another VLAN if his/her frames are forged such that the destination IP address is the IP address of the target station and the destination MAC address is the MAC address of the local default gateway.

The switch would forward this packet to the router which resides on the **promiscuous port** and the router would route it back with another VLAN label. This is called the 'Private VLAN Attack'.

However there is no return traffic to the attacker unless the target host performs the same trick. The only solution is to configure appropriate ACLs on the router.

12 VLAN Hopping Attack

There are several possibilities for an attacker to send packets in another VLAN. First the attacker could act as a switch, sending e. g. DTP packets and establish a VLAN trunk to an official switch. Therefore it is recommended to disable DTP via the `switchport mode access` command.

Alternatively, an attacker could perform **double-tagging**. That is, every frame sent into the switch is already tagged by the attacker. The switch accepts such frame only if the outer tag matches the VLAN of that access port. In this case the switch will also remove the outer tag (but leave a second tag unchanged). This only works if the VLAN of that access port is the *native VLAN* on the subsequent trunk port.

Example:

1. Assume Switch_1 and Switch_2 are interconnected via an 802.1Q trunk. Let's say on that trunk the native VLAN (=untagged) is VLAN 5.
2. Let's assume the target host is on VLAN 7, connected to Switch_2.
3. Now assume the host of the attacker is attached to a port on Switch_1 which resides in VLAN 5 (the native VLAN).
4. When the attacker sends a packet with two 802.1Q tags, '5' (outer) and '7' (inner), then Switch_1 will accept the frame and remove the outer tag.
5. Furthermore Switch_1 forwards the frame on the trunk with intact VLAN tag '7'

6. The peer switch, that is Switch_2, accepts the frame, interprets the VLAN tag '7' and forwards the frame to an appropriate VLAN-7 port. If it was a broadcast then it would forward it to **all** VLAN-7 ports. Perfect if the attacker tries a DoS attack for VLAN 7!

Therefore:

1. Assign an unused VLAN number to the native VLAN on your trunks!
2. Put all unused ports in an unused VLAN! (This follows the 'least privileges' principle)
3. Never put hosts in VLAN 1 because it is the default native VLAN on switches.
4. Consider to disable untagged VLANs, or equivalently tag every VLAN:

```
(config)# vlan dot1q tag native
```

5. Avoid broadcast storms using the `storm-control` command.

13 Port Mirroring

This technique, also known as Switch Port Analyzer (SPAN), is important to attach IDS devices to the switch.

First turn off any existing SPAN session:

```
Switch(config)# no monitor session 1
```

Then specify which traffic (source port) should be mirrored to what SPAN port (destination):

```
Switch(config)# monitor session 1 source interface gigabitethernet0/5
Switch(config)# monitor session 1 destination interface gigabitethernet0/1 en-
capsulation replicate
```

The `replicate` keyword specifies to leave the encapsulation as it was.

14 MAC-Address Notifications

When a MAC address is **added or deleted** from the CAM table a SNMP trap is sent. This only works if all of the following commands are used:

```
(config)# mac address-
table notification          !!! note the missing hyphen '-
' since 12.1(19)EA1
(config-if)# snmp trap mac-
notification added|removed !!! trap is sent when MAC ad-
dress is added or removed
(config)# snmp-server enable traps mac-notification
```

As verification of the settings use:

```
# show mac address-table notification interface
```

Of course your also must configure a SNMP trap receiver:

```
(config)# snmp-server host 10.1.1.1 traps [version 1 | 2c | 3 {auth|noauth|priv}] My-
Community [mac-notification]
```

Using the last option `mac-notification` only mac-notification traps are sent (no other traps).

Similarly, on the **Cat6500** a Syslog can be generated when a host (i. e. its MAC address) moves between switch ports:

```
Cat6500(config)# mac-address-table notification mac-move
```

Note: Only moves are reported (not deletions or initial entries).