Catalyst 3550 Common

Jason Healy, Director of Networks and Systems

Last Updated Mar 18, 2008

Contents

Chapter 1

Catalyst 3550 Common

Last updated 2008/03/18

1.1 Introduction

Suffield Academy currently has a core network built primarily from equipment built by Cisco Systems. Many of our edge switches (such as those found in dormatories, classroom buildings, and other wiring closets) are the Catalyst 3550 model. These switches have gigabit uplink ports, and 10/100 ethernet switch ports for clients.

Because we have so many of this particular model, much of the configuration is the same from switch to switch. The purpose of this document is to describe the common configuration options for these switches in a single location. For switch-specific information (such as port naming, VLAN assignment, or other unique information), please see the documentation for that particular switch.

This document discusses site-wide configuration policy, such as authentication, DNS settings, NTP servers, and edge filtering policy.

1.1.1 Intended Audience

We assume that readers of this document understand basic command-line interaction with Cisco equipment. You should already be familiar with entering enable mode, editing configuration parameters, and saving them to flash memory. If you are not familiar (or comfortable) with these procedures, do **not** proceed.

1.1.2 Shortcut to Configs

This document describes the theory behind all of the configuration settings. If you just want to jump straight to our config files, you can find them here:

Catalyst 3550 Common Config Repository

1.2 Banner (MOTD and Login)

When users connect to the switch for management purposes, they can be shown a **message of the day** (MOTD) and/or login banner. Frequently, this area is used to display a warning about unauthorized access to the system, and possibly other useful information.

1.2.1 Suffield Config Files

The following sections discuss the theory behind the configuration setting we use at Suffield. If you're just interested in a copy of the actual config file sections, you can find them here:

Catalyst 3550 Common Banner Settings

1.2.2 Setting the MOTD Banner

The message of the day banner is shown to the user as soon as they connect to the switch's management interface.

To set the motd banner:

1. Enter enable mode:

   ```
   > enable
   ```

2. Enter configuration mode:

   ```
   # configure terminal
   ```

3. Announce that you wish to set the banner:

   ```
   (config)# banner motd #
   ```

The "**#**" mark is the **delimiter** used to mark the end of the banner. The switch will continue reading input until it sees this character again. If your MOTD has a "#" in it, use a different delimiter.

4. You will be prompted to enter your banner:

```
Enter TEXT message.  End with the character '#'.

This is my motd.
#
```

5. (Optional: exit config mode and save your changes.)

1.2.3 Setting the Login Banner

The message of the day banner is shown after the MOTD banner, and before the user authenticates to the switch.

To set the login banner:

1. Enter enable mode:

```
> enable
```

2. Enter configuration mode:

```
# configure terminal
```

3. Announce that you wish to set the banner:

```
(config)# banner login #
```

The "**#**" mark is the **delimiter** used to mark the end of the banner. The switch will continue reading input until it sees this character again. If your login banner has a "#" in it, use a different delimiter.

4. You will be prompted to enter your banner:

```
Enter TEXT message.  End with the character '#'.

This is my login banner.
#
```

5. (Optional: exit config mode and save your changes.)

1.2.4 Testing It Out

The next time you connect to the switch in command-line mode, you should see the MOTD and Login banners.

1.3 DNS

The 3550 is capable of performing its switching duties without being able to resolve Internet domain names. However, without name-to-address resolution, certain tasks become more difficult (such as configuring Kerberos). Therefore, all switches should have a proper DNS configuration.

1.3.1 Suffield Config Files

The following sections discuss the theory behind the configuration setting we use at Suffield. If you're just interested in a copy of the actual config file sections, you can find them here:

Catalyst 3550 Common DNS Settings

1.3.2 Setting the Hostname

All machines on the network should have a hostname, and this name should agree with the name provided by the DNS system.

To set the host name of the switch:

1. Enter enable mode:

   ```
   > enable
   ```

2. Enter configuration mode:

   ```
   # configure terminal
   ```

3. Set the host name: (replace `myname` with desired hostname)

   ```
   (config)# hostname myname
   ```

4. (Optional: exit config mode and save your changes.)

This changes the name of the switch. Note that this **does not** change the name in DNS! This is only the switch's local notion of its hostname; if you wish to change the DNS name you must do so on the DNS server itself.

1.3.3 Setting the DNS Name Servers

In order to resolve hostnames, the switch must be given a list of servers to contact for DNS information. The DNS servers must be given by their IP address.

To set the name servers for the switch:

1. Enter enable mode:

   ```
   > enable
   ```

2. Enter configuration mode:

   ```
   # configure terminal
   ```

3. Add a name server:

   ```
   (config)# ip name-server 172.30.0.2
   ```

4. Optionally, add more name servers if your site uses more than one:

   ```
   (config)# ip name-server 172.30.0.3
   ```

5. (Optional: exit config mode and save your changes.)

To remove old or outdated name server entries, use the "`no`" version of the command above.

1.3.4 Setting the Domain Suffix Search

Normally, DNS lookups are performed on fully-qualified domain names (FQDNs). These names have a full host and domain specification (*e.g.*, `gandalf.suffieldacademy.org`). However, many systems allow the user to omit the domain part and have it "guess" the correct suffix when none is supplied.

If your site has a flat domain structure, you will only have one suffix to use for unqualified names. For more complicated domains, you will need to supply a list of possible suffixes to try.

To set the list of domain suffixes for the switch:

1. Enter enable mode:

   ```
   > enable
   ```

2. Enter configuration mode:

```
# configure terminal
```

3. Add the first domain suffix to use for unqualified names:

```
(config)# ip domain-name suffieldacademy.org
```

4. Optionally, if you wish to use a list of domain suffixes, use multiple versions of this line (you must include the default domain, even if you added it with `domain-name` above):

```
(config)# ip domain-list suffieldacademy.org
(config)# ip domain-list net.suffieldacademy.org
(config)# ip domain-list gear.suffieldacademy.org
```

5. (Optional: exit config mode and save your changes.)

1.3.5 Testing It Out

With proper DNS configuration settings, you should be able to `ping` and `telnet` to other servers on the network, both by their FQDN or by just their hostname. You should see the hostname query going out and being resolved; it will look something like this:

```
Translating "gandalf"...domain server (172.30.0.2) [OK]
```

When you see that, you know DNS is working properly on the switch.

1.4 NTP

The 3550 has an internal system clock, which it uses to mark log entries, keep statistics, and perform other time-related functions. Additionally, Kerberos functionality requires that the switch's notion of time agree with that of other machines on the network. Therefore, the clock must both be very accurate, and kept in sync with the other clocks on the network.

The Network Time Protocol (NTP) allows hosts to communicate with a trusted server and get a reliable source of time information. The 3550 supports NTP natively, so syncronizing with a time server is relatively simple.

Note: this section assumes that you already have one or more stable NTP servers configured on your network. If you need to set up an NTP server, you

will need to refer to the documentation for the server operating system of your choice. Many machines can act as NTP servers (including some Cisco equipment); determining which system to use and setting it up is beyond the scope of this document.

1.4.1 Suffield Config Files

The following sections discuss the theory behind the configuration setting we use at Suffield. If you're just interested in a copy of the actual config file sections, you can find them here:

Catalyst 3550 Common NTP Settings

1.4.2 Setting the Clock Options

Before enabling NTP, you should ensure that the system clock options are correct for your site. This includes (optionally) setting the time zone, or leaving it as UTC.

To set the local clock options for the switch:

1. Enter enable mode:

   ```
   > enable
   ```

2. Enter configuration mode:

   ```
   # configure terminal
   ```

3. Set the "standard" time zone name and offset from UTC

   ```
   (config)# clock timezone EST -5
   ```

4. Set the "summer" (daylight savings) time zone name and offset. Additionally, you may optionally specify the dates and times when it goes into effect (this will become more important in 2007 when the US switches to a new set of DST rules):

   ```
   (config)# clock summer-time EDT recurring 1 Sunday April 2:00 last Sunday October 2:00 60
   ```

5. (Optional: exit config mode and save your changes.)

1.4.3 Setting the NTP Servers

You must provide at least one (and preferably more) NTP servers to use as a stable source of time information.

To set the time servers for the switch:

1. Enter enable mode:

   ```
   > enable
   ```

2. Enter configuration mode:

   ```
   # configure terminal
   ```

3. Add a time server:

   ```
   (config)# ntp server 172.30.0.2
   ```

4. Optionally, add more time servers if your site uses more than one:

   ```
   (config)# ntp server 172.30.0.3
   ```

5. (Optional: exit config mode and save your changes.)

To remove old or outdated time server entries, use the "`no`" version of the command above.

1.4.4 Testing It Out

Once you've entered the time server information, you can query the switch for its status:

```
# show ntp status
```

The output should show that the clock is synchronized, and list one of your servers as its primary reference.

Additionally, you can view the list of servers and see what the syncronization status is for each of them:

```
# show ntp associations
```

1.5 Kerberos

(**FIXME**: not yet written.)

1.5.1 Suffield Config Files

The following sections discuss the theory behind the configuration setting we use at Suffield. If you're just interested in a copy of the actual config file sections, you can find them here:

Catalyst 3550 Common Kerberos Settings

1.6 Syslog

The 3550 has the capability to send log information to a remote `syslog` server. This allows for centralized collection of log messages, and also allows for log analysis long after the switch's internal logging buffer may have cleared.

This document assumes you have already set up a centralized syslog server that is ready to receive messages from the switch.

1.6.1 Suffield Config Files

The following sections discuss the theory behind the configuration setting we use at Suffield. If you're just interested in a copy of the actual config file sections, you can find them here:

Catalyst 3550 Common Syslog Settings

1.6.2 Basic Logging Setup

Begin by defining the global settings for logging that you would like to use. This means turning on logging and defining the format that log messages should take.

To set the logging options on the switch:

1. Enter enable mode:

   ```
   > enable
   ```

2. Enter configuration mode:

```
# configure terminal
```

3. Enable logging:

```
(config)# logging on
```

4. Set the timestamp format to include any information you'd like (time, zone, etc):

```
(config)# service timestamps log datetime msec localtime show-timezone
```

5. Disable sequence numbers in the log entries if you don't want them (we use millisecond stamps in the timestamp format above, and so don't need sequence numbers):

```
(config)# no service sequence-numbers
```

6. (Optional: exit config mode and save your changes.)

1.6.3 Local Logging Options

The switch retains some logging information that can be viewed with the `show logging` command. You can determine how much information to retain and its format.

To set the local logging options for the switch:

1. Enter enable mode:

```
> enable
```

2. Enter configuration mode:

```
# configure terminal
```

3. Set the local log buffer to 8Kb (default is 4Kb):

```
(config)# logging buffered 8192
```

4. (Optional: exit config mode and save your changes.)

1.6.4 Remote Syslog Options

The best use of the logging system is to send the messages to a remote host for collection, archiving, and possible analysis. You may specify more than one server, if desired.

To set the remote syslog server options for the switch:

1. Enter enable mode:

   ```
   > enable
   ```

2. Enter configuration mode:

   ```
   # configure terminal
   ```

3. Specify the remote server to send log information to (you may specify more than one line for multiple servers):

   ```
   (config)# logging 172.30.0.10
   ```

4. Set the maximum syslog level of messages that will be sent to the server. Levels range from 7 (debugging) to 0 (emergencies), with each level including all levels below it. Note that "debugging" may generate a very large amount of messages!

   ```
   (config)# logging trap informational
   ```

5. Set the syslog facility to use when sending messages. The remote server may use this facility to sort or group messages. In general, you should select one of the "local" facilities (numbered 0 - 7):

   ```
   (config)# logging facility local7
   ```

6. (Optional: exit config mode and save your changes.)

1.6.5 Testing It Out

As soon as the logging settings have been established, your remote logging host should begin to receive messages from the switch. Additionallly, running `show logging` on the switch should show a list of the messages in the local cache. They should have the new timestamp format you defined.

1.7 Edge Filtering

We use various methods for filtering traffic on our internal LAN. Some of this filtering is done for policy reasons (to prevent access to certain network segments or hosts), and some is done for performance or protocol reasons.

Most of the security-related configuration is performed on our core switch, where traffic is switched between VLANs. Additionally, the core performs policy routing to send packets to particular destinations (for example, routing them over a different internet connection, or forcing them to use a web proxy).

However, low-level policy filtering needs to be applied at the VLAN level (rather than only when packets travel between VLANs). These types of rules prevent unwanted packets from ever making it on to the network, and as such they must be supplied at the edge switches.

1.7.1 Suffield Config Files

The following sections discuss the theory behind the configuration setting we use at Suffield. If you're just interested in a copy of the actual config file sections, you can find them here:

Catalyst 3550 Common Filter Settings

1.7.2 Types of Unwanted Traffic

At Suffield, we run a relatively open network. However, we do prevent some troublesome protocols from being used on the network in an effort to make the network more robust.

Currently, Suffield uses edge filtering to **prevent** the following from being used on the network:

1. AppleTalk ethernet frames (AFP via IP is now preferred over AppleTalk, and most printers now support IPP or LPD for print services). AppleTalk is very "chatty", and also difficult to filter for purposes of access control.
2. "Rogue" DHCP broadcasts. Users who connect 3rd-party routers, or who enable "Internet connection sharing" may accidentally broadcast DHCP responses onto the network. This can cause other clients on the network to use incorrect configurations and be unable to access the network properly. We block all DHCP server packets from "untrusted" machines.
3. "Rogue" IPP broadcasts. Users frequently enable "Printer sharing" on their computers, advertising themselves as a route to a printer that is

better reached directly. By discarding these advertisements, we ensure that clients always connect directly to printers rather than selecting an untrusted machine.

4. Multicast DNS advertisements. We run a well-configured network, including a static DNS Service-Discovery configuration to allow users to find printers and other resources. To prevent users from registering their own machines (or duplicates of our services), we quash ad-hoc advertisements.

5. Spoofed addresses. Our internal LAN uses a specific segment of RFC 1918 private IP ddresses, so we automatically drop any packets that are not sourced or destined to this block of addresses.

1.7.3 Quick Background: ACLs and VLAN Access Maps

The **only way** to filter traffic **on** a VLAN (as opposed to **between** VLANs) is to use a combination of Access Control Lists (ACLs) and VLAN access-maps.

First, we build several ACLs which identify packets that we wish to match. We may wish to match packets to explicitly deny, or to specifically allow, so the ACLs do not actually decide what to do with the packets; they simply identify the packets we're interested in.

Next, we create VLAN access maps, which decide what to do with packets that match a particular ACL. **Note:** access maps apply to all traffic on a VLAN, **regardless of direction**, so your ACLs must match traffic in both directions (ingress and egress).

Finally, we apply the VLAN access maps to one or more VLANs on the edge switch. If the map is generic enough, it can be applied to several VLANs, making administration simpler.

1.7.4 ACLs

To identify packets, we must create ACLs that match these packets. Two types of ACL are available: "`ip`" access lists, and "`mac`" access lists. IP access lists only match IP protocol information such as IP address, protocol (TCP/UDP), and port number. MAC access lists can match physical Ethernet addresses (48-bit hexadecimal), as well as Ethernet protocol families (AppleTalk, DECNET, IPX, VINES).

Because of the way VLAN access maps are applied, you **must** create an ACL for each type of traffic you wish to **allow** (access maps deny packets by default whenever an ACL is present). Also keep in mind that ACLs are applied in order, with the first matching ACL being the one that gets applied. You may need to create several ACLs to properly match certain types of traffic.

Finally, all ACLs are comprised of rules that either "permit" or "deny" packets. In the context of VLAN access maps, "permit" means that the packet matches the current access map entry, and "deny" means that it does not. **The VLAN access map decides what to do with a matching packet**, not the ACL! Therefore, it is possible to have a packet "permitted" by an ACL, but have the action in the map set to "drop". Similarly, just because you "deny" a packet in an ACL does not mean it is dropped; it simply means that the packet is no longer considered for that VLAN access map entry.

Default ACLs

In order to allow traffic on the VLAN, its helpful to create a few default ACLs to permit all (or most) traffic. Because we run a fairly open network, we have selected to allow all traffic that we do not explicitly deny. You may wish to change this to a more restrictive policy.

VLAN access maps explicitly deny unmatched packets whenever any matching rule for that packet type exists. In other words, if you specify at least one IP ACL, then any IP packets **not** matching that ACL are dropped. Similarly, if you specifiy at least one MAC ACL, any non-IP packets that do **not** match that ACL are dropped. Therefore, if you plan to use both IP and MAC ACLs, at least one of each type must be used to permit traffic, or you'll end up denying everything by default.

Also note that Cisco gear treats fragmented packets differently from normal (header) packets. If you don't explicitly match fragments, they may end up getting denied because fragments do not include full port information.

Here is a simple IP ACL which matches all IP fragments:

```
ip access-list extended vlan_filter_ip_fragments
 permit ip any any fragments
```

Here is a simple IP ACL which matches all (non fragmented!) IP-based traffic:

```
ip access-list extended vlan_filter_ip_any
 permit ip any any
```

And here is a simple MAC ACL which matches all non-IP (Ethernet) traffic:

```
mac access-list extended vlan_filter_mac_any
 permit any any
```

Note that these rules say nothing about blocking or passing traffic; they simply match packets. We will use these lists later in our VLAN access maps to make policy decisions.

AppleTalk

AppleTalk is a protocol developed by Apple Computer. It provides a broadcast-based system for advertising services and hosts. Because of this, the protocol is very "chatty". Additionally, it requires an entirely separate set of controls (*i.e.*, "zones"), as it is not IP-based.

The primary uses of AppleTalk (file sharing and printing) have been ported to IP-based protocols (AFP over TCP for file sharing, and IPP or LPD for printing). Therefore, AppleTalk is unecessary in a modern network. However, many devices (especially printers) still ship with AppleTalk turned on by default, so filtering helps prevent misconfigured devices from flooding the network.

Because AppleTalk is a non-IP protocol, we must use a MAC ACL. Here is an ACL that matches AppleTalk packets:

```
mac access-list extended vlan_filter_appletalk
 permit any any aarp
 permit any any appletalk
```

AARP is the AppleTalk Address Resolution Protocol, and is the most vital to block (as it prevents services from finding each other). It serves the same purpose as ARP packets on an IP network. Without it, address mappings cannot take place, and communication is prohibited.

Rogue DHCP Servers

Many sites use DHCP for automatic assignment of IP addresses, as well as other configuration information (subnet mask, gateway addresses, name servers, etc).

Because the purpose of DHCP is to provide configuration to clients with no knowledge of the network topology, the service model is very insecure. Clients simply broadcast a query on the network and hope for a response. Meanwhile, any listening server broadcasts back a response for the client to use.

The system works well, provided that the only DHCP servers on the network are the ones you set up. Unfortunately, numerous consumer devices (wireless access points, routers, or even computers with "Internet sharing" enabled) are configured to act as DHCP servers as well. If these devices are connected to the network, they will attempt to answer DHCP queries. Because the client does not know who to trust, it simply chooses the first response as its source of authoritative information.

We can remedy this situation by preventing the broadcast of DHCP server packets from all computers except those that we trust. We use an ACL which

matches all **bad** servers, and ignores thost that we trust, so that we can take specific action on the untrusted packets.

Note that you must list any servers which provide DHCP service, as well as any DHCP relays on your network (if you use relays to forward the broadcast packets accross VLANs).

Here is a sample ACL for identifying rogue DHCP servers:

```
ip access-list extended vlan_filter_dhcp_rogue
 remark Allow Suffield servers to broadcast
 deny   udp host 172.30.0.2 eq bootps any eq bootpc
 deny   udp host 172.30.0.3 eq bootps any eq bootpc
 deny   udp host 172.24.48.4 eq bootps any eq bootpc
 remark Allow DHCP helper relay addresses (gateway addresses)
 deny   udp host 172.22.0.1 eq bootps any eq bootpc
 deny   udp host 172.24.0.1 eq bootps any eq bootpc
 deny   udp host 172.28.0.1 eq bootps any eq bootpc
 deny   udp host 172.30.0.1 eq bootps any eq bootpc
 deny   udp host 172.31.0.1 eq bootps any eq bootpc
 remark Prevent all unauthorized clients from broadcasting
 permit udp any eq bootps any eq bootpc
```

Note the use of "deny" for the trusted hosts; this prevents the ACL from matching packets from our known servers. Later, when we apply the VLAN access map, we will use it to drop packets that match this ACL.

IPP Broadcasts

At Suffield, we use the Internet Printing Protocol (IPP) as our preferred means of submitting jobs to networked printers. IPP has many features, including a way to share printers between computers by broadcasting their availability on the network.

Some users turn on this printer sharing, and begin advertising themselves as a route to all the networked printers. Obviously, this is not a desirable state, as other users should be printing directly to the printers, not through other computers.

To prevent this advertisement of printers, we have an ACL that prevents these broadcast messages from being sent:

```
ip access-list extended vlan_filter_ipp
 permit udp any any eq 631
```

Note that if you use print servers on your network, you may need to specifically exempt them from this ACL (using "deny" rules) so that they can still broadcast to the network.

mDNS Advertisements

Mac OS X automatically advertises a computer on the network using ad-hoc DNS Service Discovery (sometimes branded as "Bonjour", "Rendezvous", or "Zeroconf"). This is great for ad-hoc networks with no formal DNS structure, but on a well-run network it becomes a hinderance when user-advertised services conflict with legitimate ones.

Services are advertised via multicast DNS, which uses a specific multicast address and port. Therefore, we can easily deny it by blocking all traffic with this address/port combination:

```
ip access-list extended vlan_filter_mdns
 permit udp any host 224.0.0.251 eq 5353
 permit tcp any host 224.0.0.251 eq 5353
```

Rogue IP Filter

As an added layer of security, we filter all IP traffic to ensure that it originates from, or is destined to, a valid IP address. Our internal network uses a single block of RFC 1918 private IP addresses, so we filter out any packets that do not use these addresses. This prevents misconfigured clients from sending packets on the network.

Note that this approach requires two additional steps to work correctly:

1. Broadcast traffic from unconfigured DHCP clients must be explicitly allowed, regardless of the source address. Unconfigured clients may use 0.0.0.0 as an address, or part of the auto-assigned 169.254.0.0/16 block (or any other address), so we must be careful not to filter these requests for DHCP.

2. Multicast traffic uses a separate special IP block, known as the Class D block, at 224.0.0.0/4. We must also allow addresses to/from this block if we wish to allow multicast IP on our network. Of course, if you don't want multicast on your network, you can leave this out.

Here is our ACL to prevent unwanted IP traffic. Again, the rule is to "deny" packets that we want, as this ACL will be used in an access map rule that drops the matched packets.

```
ip access-list extended vlan_filter_ip_rogue
 remark Allow traffic to/from our internal addresses
 deny   ip 172.16.0.0 0.15.255.255 any
 deny   ip any 172.16.0.0 0.15.255.255
```

```
remark Allow multicast traffic (224.0.0.0/4)
deny   ip 224.0.0.0 15.255.255.255 any
deny   ip any 224.0.0.0 15.255.255.255
remark Allow anyone to make a DHCP request
deny   udp any eq bootpc any eq bootps
remark Prevent non-Suffield IP addresses from getting on the network
permit ip any any
```

Finally, note that if you use other/additional IP ranges on your network, you must change the values in the rule above.

1.7.5 VLAN Access Maps

Once we've written our ACLs, we need to match traffic against them and take specific actions on those matched packets. The way we do this is by using a **VLAN access map**. Just as a reminder: access maps do not differentiate between "ingress" and "egress" filtering (which is different from some other Cisco ACL-based commands). All traffic on the VLAN is matched against these maps.

The basic operation of the access map is simple. The map consists of one or more **tests**, with each test having an **action** associated with it. A packet is checked against each test in order, and the first one that matches causes the action to be taken on it. Packets that do not match any tests are dropped by default (unless there are no tests, but we'll assume that is not the case here). Therefore, you must have at least one test which allows packets, or you'll end up blocking everything.

Since the matching rules are considered in order, you must apply the rules so that packets always match more specific rules first. In our case, we have a set of specific "drop" rules, followed by some permissive "forward" rules. In this sense, we eliminate all traffic we know we don't want, and allow everything else by default.

Here is our access map entry. Note the rule numbers applied to each matching condition. Also, note that the action for each rule; sometimes we explicitly forward packets, and sometimes we drop them. The ACL names given here match those discussed in previous sections.

```
vlan access-map vlan_broadcast_suppress 100
match mac address vlan_filter_appletalk
action drop

vlan access-map vlan_broadcast_suppress 200
match ip address vlan_filter_dhcp_rogue
action drop

vlan access-map vlan_broadcast_suppress 300
match ip address vlan_filter_ipp
```

```
action drop

vlan access-map vlan_broadcast_suppress 400
match ip address vlan_filter_ip_rogue
action drop

vlan access-map vlan_broadcast_suppress 65533
match ip address vlan_filter_ip_any
action forward

vlan access-map vlan_broadcast_suppress 65534
match mac address vlan_filter_mac_any
action forward
```

1.7.6 Applying Access Maps and Testing

Once the VLAN access map has been created, we apply it to the VLANs where the policy should have effect. You may specify each VLAN individually, or provide ranges of VLANs.

Warning: improperly configured access maps may block all traffic on a given VLAN. If you are accessing your switch remotely, make sure to test your configuration on a VLAN other than the one you're on. Otherwise, you may disconnect yourself from the switch and be unable to revert the changes remotely.

Our VLANs all fall within a well-specified range, so we simply apply the range operator to cover all our VLANs:

```
vlan filter vlan_broadcast_suppress vlan-list 500 - 899
```

The command should take effect immediately. Confirm that normal traffic is still being passed, and that unwanted traffic is being blocked. If necessary, you can remove the filter simply by using the "no" version of the command above.

1.8 Switch Installation / Replacement

This section describes how to prepare a "factory-fresh" switch for use on our network. Usually, this will occur when a failed switch is replaced with a new unit.

1.8.1 Hardware Setup

1. Attach all GBICs and external interfaces to the switch

2. Attach the primary cabling to the switch (usually, this will be the fiber optic link from the core)

3. Attach a serial cable to the switch's monitor port, and start a terminal emulation program on your computer

4. Power up the switch

1.8.2 Initial Configuration

1. When asked "Would you like to enter the initial configuration dialog", say **no**.

2. When asked "Would you like to terminate autoinstall", say **yes**.

3. You can now enter a command prompt. Type **en** to enter enable mode so you can begin configuration.

1.8.3 VLAN Configuration

In order to properly configure the switch, it must be connected to the core, and it must download the VLAN database.

1. Start by moving into configuration mode:

```
conf t
```

2. Set the trunk interface to the correct mode (substitute the correct interface):

```
int gi0/1
switchport trunk encapsulation dot1q
switchport mode trunk
no shut
exit
```

3. Next, we need to get the VLAN database from the core switch. We do this by joining the VTP domain, which slaves us to the master switch:

```
vtp domain suffield
vtp password <use actual password here>
vtp version 2
vtp mode client
```

4. Set an IP address for this switch:

```
int vlan901
ip address 172.31.0.XXX 255.255.128.0
no shut
exit
```

5. Finally, exit configuration mode:

   ```
   exit
   ```

   After several seconds, the management VLAN should come up, and you should be able to ping the core switch. At this point, the switch is configured for network use, and ready for the next step.

   Do not proceed unless you have completed a successful ping test.

1.8.4 Upgrading the IOS Software

Depending on the age of the switch you're installing, you may need to replace the software with the version currently in production on our other switches.

Preparing to Upgrade

1. Confirm the software version on the switch:

   ```
   show version
   ```

   The first line of output should show the IOS version for the switch. If it is not identical to the line on our production switches, you must upgrade the IOS software.

2. Confirm that there is enough free space on the flash disk to perform an upgrade:

   ```
   dir flash:
   ```

   You'll get an accounting of free space from this command. If there is not enough room for the new image, delete files using the `delete flash:<file name>` command. There should be ample room for both the current and upgraded IOS images.

Setting Up a TFTP Server

The easiest way to get the new software image to the switch is via TFTP. Several TFTP servers exist for UNIX and Windows computers. We use Mac OS X, which comes with a built-in TFTP server. If you are not using Mac OS X 10.4 or better, please find other documentation on setting up a TFTP server.

1. To see if tftpd is running, type the following in the Mac OS X terminal:

   ```
   sudo launchctl list | grep -i tftp
   ```

   If you see `com.apple.tftpd`, then the service is running, and you're ready to go.

2. If you need to start the service, type the following:

   ```
   sudo launchctl load -w /System/Library/LaunchDaemons/tftp.plist
   ```

   Then, run the first command again to confirm that it's started.

   If you wish to stop the TFTP service, simply run:

   ```
   sudo launchctl unload -w /System/Library/LaunchDaemons/tftp.plist
   ```

3. The TFTP server serves files from the `/private/tftpboot` directory. You'll need to copy any software images you'd like to serve to that directory:

   ```
   sudo cp <path to image> /private/tftpboot/
   ```

   The "path to image" should be a **bin** file downloaded from Cisco (or from our repository). The name should be something like `c3550-ipservicesk9-mz.122-25.SEE2.bin`.

4. Finally, make note of the IP address of your computer. You'll need this to connect from the swtich to your computer to download the files.

Upgrading the IOS Software

1. On the switch, in enable mode, copy the IOS image to the flash disk:

   ```
   copy tftp flash:
   ```

   You will be prompted for the address of the remote host. Enter the IP address of the machine with the TFTP server.

You will be prompted for a filename to copy. Enter the name of the IOS image. You should not need to enter any leading path information (such as `/private/tftpboot`).

You will be prompted for a destination filename. Just hit return to accept the default (which matches the source filename).

The switch will show a progress indicator as it downloads the image.

2. Verify that the image was correctly transferred:

```
verify flash:<name of image>
```

3. Set the boot variable on the switch to use the new image:

```
conf t
boot system flash:<name of image>
exit
```

4. Verify that the boot parameter is correct:

```
show boot
```

5. Write the configuration to memory:

```
write memory
```

6. Reload the switch:

```
reload
```

7. Once the switch has rebooted, confirm that the IOS version has changed:

```
show version
```

1.8.5 Restoring the Configuration

If this is a replacement switch, you'll want to load a backed up copy of the configuration onto the new switch.

The best way to restore the configuration is via TFTP. Make sure you have a functioning TFTP server (as described in the previous section).

1. Copy the backup of the switch configuration into the TFTP directory:

```
sudo cp <path to config> /private/tftpboot/
```

2. On the switch, in enable mode, copy the configuration on to the switch:

   ```
   copy tftp running-config
   ```

   You will be prompted for the address of the remote host. Enter the IP address of the machine with the TFTP server.

   You will be prompted for a filename to copy. Enter the name of the configuration file. You should not need to enter any leading path information (such as `/private/tftpboot`).

   You will be prompted for the destination filename. Just press return to accept the default (**running-config**).

   A progress indicator will show the load of the config file. Additional, the configuration file may cause one-time events to fire (such as the generation of PKI SSH keys, NTP queries, or DNS probes).

3. At this point, the running configuration of the switch should be correctly loaded from the backup configuration. Spot-check the configuration (interface names, VLAN assignments, connectivity, DNS resolution). If everything appears to be working properly, save the configuration permanently:

   ```
   write memory
   ```

At this point, the switch has been properly restored, and is ready for active service.