Exam
Manual

LearnSmart

CISCO (642-812) BCMSN

Smarter Training

This LearnSmart exam manual covers the most important topics you will encounter on the Multilayer Switched Networks exam (BCMSN). By studying this manual, you will become familiar with an array of exam-related topics, including:

- Implementing VLANs
- Implementing Inter-VLAN routing
- Describing and configuring wireless client access
- And more!

Give yourself the competitive edge necessary to further your career as an IT professional and purchase this exam manual today!

Building Converged Cisco Multilayer Switched Networks (BCMSN) LearnSmart Exam Manual

Copyright © 2011 by PrepLogic, LLC
Product ID: 011242
Production Date: July 19, 2011

Warning and Disclaimer

Volume, Corporate, and Educational Sales

Favorable discounts are offered on all products when ordered in quantity. For more information, please contact us directly:

1-800-418-6789
solutions@learnsmartsystems.com

International Contact Information

International: +1 (813) 769-0920

United Kingdom: (0) 20 8816 8036

Table of Contents

Abstract

The Cisco Certified Network Professional is one of the most well respected certifications in the world. By attaining it, students and candidates signify themselves as extremely accomplished and capable Network Professionals. These four exams, created by Cisco Systems, are extremely difficult and not to be taken lightly. They cover a myriad of topics, in particular - the BCMSN (Building Converged Cisco Multilayer Switched Networks) exam covers all the way to the most detailed analysis of routing packets across multiple subnetted networks, including VLANs and other complex network concepts. BCMSN is multiple choice, simulative, and incorporates test strategies such as "drag and drop" and "hot area" questions to verify a candidate's knowledge.

Before taking this exam, you should be very familiar with both Cisco technology and networking. You must have also attained the Cisco CCNA certification by passing either the one or two part path.

Your Product

This BCMSN Exam Manual has been designed from the ground up with you, the student, in mind. It is lean, strong, and specifically targeted toward the candidate. Unlike many other BCMSN products, the LearnSmart BCMSN Exam Manual does not waste time with excessive explanations. Instead, it is packed full of valuable techniques, priceless information, and brief, but precisely worded, explanations. While we do not recommend using only this product to pass the exam, but rather a combination of LearnSmart Audio Training, Practice Exams, and Video Training, we have designed the product so that it and it alone can be used to pass the exam.

About the Author

Andrew Froehlich: CCNP, CCSP, CCDA

Andrew is the President of West Gate Networks, a network and IT consulting firm based in Chicago. He has over 10 years of technology experience with 8 of those years focused on network and IT security related solutions. His expertise in network technologies includes: LAN/WAN architecture, remote access, VPNs, wireless technologies, voice services, and network security. Froehlich's past and present experience includes network design and troubleshooting with major financial institutions, health care providers and airlines.

Implementing VLANs

VLANs in a Hierarchical Network

A virtual LAN (VLAN) is a logical network that allows a group of devices to act as if they are on the same physical network. All devices within a VLAN share the same unicast, broadcast and multicast domain. Networked devices can be connected to different switches in different locations and still be on the same VLAN. This allows for a great amount of flexibility as seen in *Figure 1*. Communication between VLANs must pass through a layer 3 device such as a router.

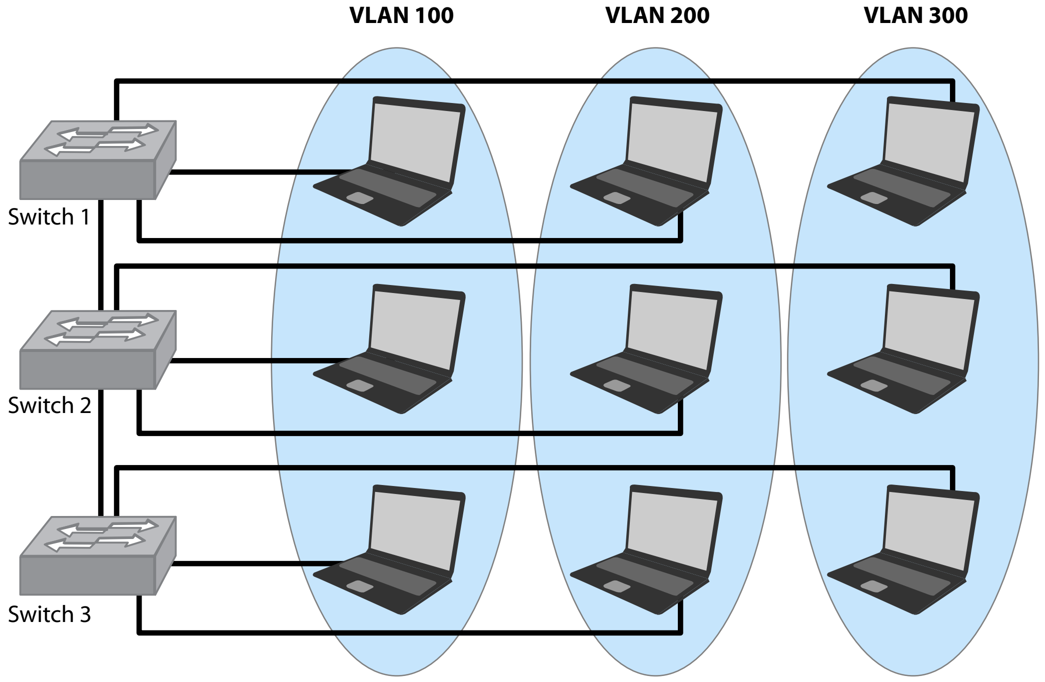


Figure 1

All devices within a particular VLAN are typically in the same IP subnet. For example, all PC's in VLAN 100 have an IP address within the 192.168.100.X/24 range while PC"s in VLAN 200 have an IP address of 192.168.200.X/24. The IP subnet is configured either on a router interface or a Layer 3 switch VLAN interface. These VLAN interfaces are also know as switch virtual interfaces (SVI).

In a typical 3 tier Cisco architecture, the VLAN SVIs are configured at the distribution layer. That effectively creates the layer 3 gateways on the VLAN interfaces. Layer 2 physical interfaces are then configured to connect to the access layer switches. If multiple VLANs are required on a particular switch, a trunk is configured to switch multiple VLANs across the same physical connection.

Configuring VLANs

The Cisco recommended method to create and verify a VLAN is as follows:

Syntax:

```
Switch# configure terminal
Switch(config)# vlan vlan-id
Switch(config-vlan)# name vlan-name
Switch(config-vlan)# end
Switch# show vlan {name vlan-name | id vlan-id}
```

So let's say that we want to create VLAN 200 on our switch and name it "Accounting". Here's how it would be configured:

Example:

```
Switch# configure terminal
Switch(config)# vlan 200
Switch(config-vlan)# name Accounting
Switch(config-vlan)# end
Switch# show vlan Accounting (or show vlan 200)
```

A second method that can be used on a Cisco switch is by entering the VLAN database in privileged EXEC mode:

Syntax:

```
Switch# vlan database
Switch(vlan)# vlan vlan-id name vlan-name
Switch(vlan)# exit
Switch# show vlan {name vlan-name | id vlan-id}
```

So to create a VLAN 200 with the name Accounting using the VLAN database method, it would look like this:

```
Switch# vlan database
Switch(vlan)# vlan 200 name Accounting
Switch(vlan)# exit
Switch# show vlan Accounting (or show vlan 200)
```

Note: You cannot add extended-range VLANs in VLAN database configuration mode Normal-range vlans are 1-1005 with 1002-1005 being reserved for token-ring VLANs. Extended-range VLANs (1006-4094) are configured using the config-vlan method.

Now that we know how to build a VLAN, we can configure specific switch interfaces so end devices can become part of the VLAN. By default, all switch ports are placed in VLAN 1. There are multiple ways to assign VLAN membership to a port. The first and most widely used method is static assignment. Below is the syntax to statically assign a port to a VLAN and verify the configuration:

Syntax:

```
Switch# configure terminal
Switch(config)# interface interface-id
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan vlan-id
Switch(config-if)# end
Switch# show interfaces interface-id switchport
```

Using our example above, let's statically assign switch port fa0/10 to VLAN 200:

Example:

```
Switch# configure terminal
Switch(config)# interface fastethernet0/10
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 200
Switch(config-if)# end
```

VLAN Trunking

When connecting multiple switches together, a trunk link can be used to transport traffic from multiple VLANs across a single point-to-point link. This is called a trunk link. There are two ways to configure a trunk on Cisco devices. Cisco has their own proprietary trunking protocol called Inter-switch link (ISL). The other method is 802.1q which is a non-proprietary IEEE standard.

Configuring ISL Trunks

The Inter-switch link protocol is a method of tagging a switch frame with the VLAN identification number so connected switches can identify the VLAN numbering correctly and to transfer end device communication from one switch to another on the same VLAN. ISL prepends a 26 byte header, including a 10 bit VLAN ID, to the frame header. It also appends a 4 byte CRC to the end of the frame. This effectively "encapsulates" the VLAN tagged frame. There are three steps to configuring an ISL trunk link on each side of the switch. The steps are:

1. Set the trunk encapsulation
2. Set the trunk mode
3. Set the default VLAN

The trunk encapsulation method must be the same on both sides of the trunk. A trunk port can be configured to negotiate the encapsulation method using the negotiation command.

The mode can be hard-coded to trunk or use the dynamic desirable/ dynamic auto methods to dynamically transition to a trunking state. If both sides of the link are set to dynamic desirable, a trunk will form. If both sides are set to auto, the trunk will not form. If one side of the trunk is auto and the other is desirable, the trunk will form. And if one side is set to trunk while the other is either desirable or auto, the trunk will form. The default mode for switch interfaces is dynamic desirable.

The default VLAN command is optional but is a good idea to do. It dictates what VLAN is used if the interface stops trunking. This is important if your trunk links are set to dynamic desirable or dynamic auto.

Figure 2 will serve as the example to configuring an ISL trunk between 2 switches. VLANs 100 and 200 are configured on both of the switches and we want the VLANs to be trunked using ISL for seamless communication of end devices on the same VLAN.

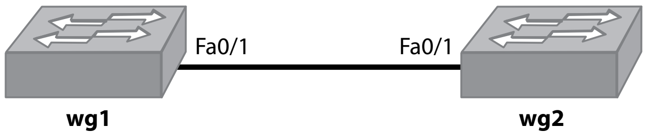


Figure 2

Syntax:

```
Switch# configure terminal
Switch(config)# interface interface-id
Switch(config-if)# switchport trunk encapsulation {isl | dot1q | negotiate}
Switch(config-if)# switchport mode {dynamic {auto | desirable} | trunk}
Switch(config-if)# switchport access vlan vlan-id
```

The mode can be hard-coded to trunk or use the dynamic desirable/dynamic auto methods to dynamically transition to a trunking state. If both sides of the link are set to dynamic desirable, a trunk will form. If both sides are set to auto, the trunk will not form. If one side of the trunk is auto and the other is desirable, the trunk will form. And if one side is set to trunk while the other is either desirable or auto, the trunk will form. The default mode for switch interfaces is dynamic desirable.

The default VLAN command is optional but is a good idea to do. It dictates what VLAN is used if the interface stops trunking. This is important if your trunk links are set to dynamic desirable or dynamic auto.

Example:

Because we know we want an ISL trunk link between wg1 and wg2 in *Figure 2*, the encapsulation will be set to ISL and the mode will be hard-coded to trunk on both sides. Even though the port will be hard-coded the default VLAN will be set to 100.

Switch wg1:

```
wg1# configure terminal
wg1(config)# interface fa0/1
wg1(config-if)#switchport trunk encapsulation isl
wg1(config-if)#switchport mode trunk
wg1(config-if)#switchport access vlan 100
```

Switch wg2:

```
wg2# configure terminal
wg2(config)# interface fa0/1
wg2(config-if)#switchport trunk encapsulation isl
wg2(config-if)#switchport mode trunk
wg2(config-if)#switchport access vlan 100
```

To verify our trunk we can do a show interfaces trunk:

```
wg1#show interfaces trunk

Port      Mode      Encapsulation      Status      Native vlan
Fa0/1      on              isl            trunking          1

Port     Vlans allowed on trunk
Fa0/1      1-4094

Port      Vlans allowed and active in management domain
Fa0/1      1,100,200

Port      Vlans in spanning tree forwarding state and not pruned
Fa0/1      1,100,200
```

The output shows that the trunk status is turnking with ISL encapsulation. All VLANs that are configured on the switches are allowed to traverse the trunk but only VLAN 1, 100 and 200 are currently active on the switch.

To limit the VLANs that are used on the trunk, the **switchport trunk allowed vlan** *vlan-id* command could be used. In our example, we want to limit the VLANs trunked to 100 and 200.

```
wg1(config-if)#switchport trunk allowed vlan 100,200
```

Now, looking at the show interfaces trunk command, we see that only the two VLANs are allowed.

```
wg1#show interfaces trunk

Port      Mode      Encapsulation  Status      Native vlan
Fa0/1      on           isl          trunking       1

Port     Vlans allowed on trunk
Fa0/1      100,200

Port      Vlans allowed and active in management domain
Fa0/1      100,200

Port      Vlans in spanning tree forwarding state and not pruned
Fa0/1      100,200
```

Another useful command is show VLAN brief. This command shows what switchports the ports are configured for:

```
wg1#sh vlan brief

VLAN Name                        Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                     active
100  VLAN0100                    active    Fa0/2, Fa0/3, Fa0/4, Fa0/5
                                           Fa0/6, Fa0/7, Fa0/8, Fa0/9
                                           Fa0/10, Fa0/11, Fa0/12, Fa0/13
                                           Fa0/14, Fa0/15, Fa0/16, Fa0/17
                                           Fa0/18, Fa0/19, Fa0/20, Gi0/1
200  VLAN0200                    active    Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                           Fa0/25, Fa0/26, Fa0/27, Fa0/28
                                           Fa0/29, Fa0/30, Fa0/31, Fa0/32
                                           Fa0/33, Fa0/34, Fa0/35, Fa0/36
                                           Fa0/37, Fa0/38, Fa0/39, Fa0/40
                                           Fa0/41, Fa0/42, Fa0/43, Fa0/44
                                           Fa0/45, Fa0/46, Fa0/47, Fa0/48
                                           Gi0/2
1002 fddi-default                act/unsup
1003 token-ring-default          act/unsup
1004 fddinet-default             act/unsup
1005 trnet-default               act/unsup
```

You'll notice that all the switchports are configured to belong in either vlan 100 or 200. But our Fa0/1 trunk port is missing. This is because the trunk will only belong to a single VLAN if the port stops trunking.

Configuring 802.1q Trunks

Configuring 802.1q trunks is very similar. The 802.1q trunking standard does not add a VLAN tag to the frame. Instead, it inserts a 4 byte tag into the frame itself. *Figure 3* will be our example used for configuring 802.1q trunking. The 4 steps for configuring 802.1q trunking are:

1. Set the trunk encapsulation
2. Set the trunk mode
3. Set the default VLAN
4. Set the native VLAN

The first 3 steps are identical to the ISL trunk except you set the encapsulation to dot1q. The native VLAN assigned to the 802.1q trunk link must be identical on both sides.

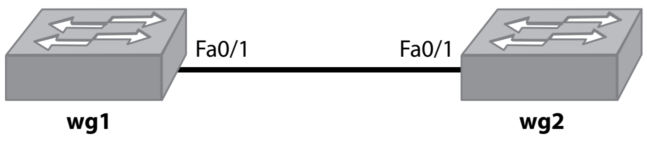


Figure 3

Syntax:

```
Switch# configure terminal
Switch(config)# interface interface-id
Switch(config-if)# switchport trunk encapsulation {isl | dot1q | negotiate}
Switch(config-if)# switchport mode {dynamic {auto | desirable} | trunk}
Switch(config-if)# switchport access vlan vlan-id
Switch(config-if)# switchport trunk native vlan vlan-id
```

Example:

This time, we'll configure both sides to encapsulate using dot1q. The modes on both sides will be set to hard-code with trunk. And finally, the native VLAN will be set to 100.

Switch wg1:

```
wg1# configure terminal
wg1(config)# interface fa0/1
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#switchport access vlan 100
wg1(config-if)#switchport trunk native vlan 100
```

Switch wg2:

```
wg2# configure terminal
wg2(config)# interface fa0/1
wg2(config-if)#switchport trunk encapsulation dot1q
wg2(config-if)#switchport mode trunk
wg2(config-if)#switchport access vlan 100
wg2(config-if)#switchport trunk native vlan 100
```

To verify our trunk we can do a show interfaces trunk:

```
wg1#show interfaces trunk

Port     Mode      Encapsulation  Status      Native vlan
Fa0/1     on          802.1q       trunking     100

Port    Vlans allowed on trunk
Fa0/1     1-4094

Port     Vlans allowed and active in management domain
Fa0/1     1,100,200

Port     Vlans in spanning tree forwarding state and not pruned
Fa0/1     1,100,200
```

You can see that the 802.1q trunk is up with the native vlan of 100. If you want to limit what VLANs are allowed on the trunk, you can use the **switchport trunk allowed vlan** *vlan-id* command just like the ISL trunk.

VLAN Trunking Protocol (VTP)

VLAN trunking protocol (VTP) is a way to simplify management of VLANs on a network. The setup allows an engineer to add, delete or edit VLAN's in a single location and have that information propagated to all switches within a VTP domain.

There are three ways to configure VTP on Cisco devices. The switch can be defined as a server, client or transparent.

- **VTP Server:** This mode allows an engineer to add, delete and modify VLANs. This is the default VTP mode for switches.
- **VTP Client:** This mode listens to VTP server messages and copies the settings of the switch that is configure with the server VTP mode. No changes can be made on a switch that is configured as a client.
- **VTP Transparent:** These switches do not listen to VTP messages at all. They can add, delete and change VLAN information on the local switch but it does not propagate the information to any other switch like the server would.

VTP advertisements are sent across ISL and 802.1q trunks to other switches in the same VTP domain. If a switch is listening to VTP advertisements and receives a message with a revision number that is lower than the one it is currently on, it will ignore the message all together. The VTP revision number increments each time a VLAN change is made on the network. This helps to prevent a new switch being added onto the network that is set for VTP mode server. If the new switch is not configured properly with the current VLAN information on the network, it can wipe out all VTP information. It is important to remember that when adding a new switch set for VTP server mode to the network that it has been rebooted prior to connecting it to a production network as the revision number might be higher and will delete all of your VLANs!

When configuring VTP for the first time on a network, it is important to remember a few considerations. The first consideration is to note that trunk negotiation does not work across VTP domains. All switches need to be within a VTP domain to participate. The VTP version must also be identical across a VTP domain. The default VTP version is 2. And finally, the VTP servers in the domain should all have the same revision number and they should be the highest number of all the switches.

By default, VTP is disabled. To enable VTP, you have to configure a VTP domain name. A VTP password can also be used and is recommended. All participating switches in the VTP domain must have identical passwords. VTP pruning can also be turned on to allow for fewer VTP advertisements to be sent. If a VTP message comes through about a VLAN that is not configured on any ports of the downstream switch, VTP pruning figures this out and does not send the message to that particular switch. Note that only VLANs 2-1001 are prune eligible. Pruning is disabled by default and VLANs must be configured to be prune eligible.

Configuring VTP

In our *Figure 4* example, we'll configure switch wg1 as the VTP server that will propagate all VLAN information to the two switches that are configured as VTP Clients. Assume the switches are already configured for 802.1q trunking and have VLAN 100 and 200 already configured. The native VLAN is set as 100.

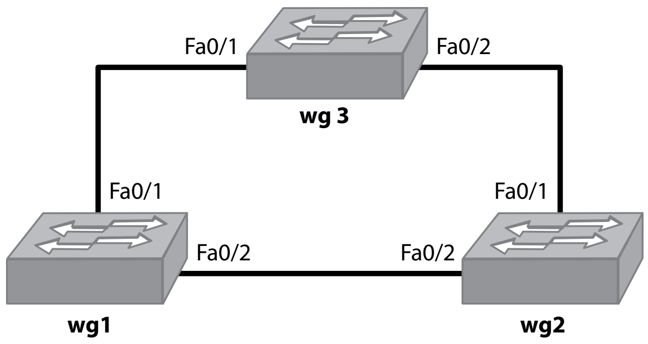


Figure 4

Prior to any VTP configurations being made, this is the status of our VTP:

```
wg3#show vtp status
VTP Version                        : 2
Configuration Revision             : 4
Maximum VLANs supported locally    : 1005
Number of existing VLANs           : 7
VTP Operating Mode                 : Server
VTP Domain Name                    :
VTP Pruning Mode                   : Disabled
VTP V2 Mode                        : Disabled
VTP Traps Generation               : Disabled
MD5 digest                         : 0x6C 0x40 0x6E 0x9B 0x73 0x77 0xB9 0xD9
```

As you can see, there is no VTP domain listed so VTP is currently disabled.

Syntax:

Switch# **configure terminal**
Switch(config)# **vtp mode** {**server** | **client** | **transparent**}
Switch(config)# **vtp domain** *domain-id*
Switch(config)# **vtp password** *password*
Switch(config)# **vtp version** {**1** | **2**}

In our example we'll use "preplogic" as the domain ID and bcmsn as the password. The configuration of the three switches is as follows:

Example:

Switch wg1:

```
wg1#configure terminal
wg1(config)#vtp mode client
wg1(config)#vtp domain preplogic
wg1(config)#vtp password bcmsn
```

Switch wg2:

```
wg2#configure terminal
wg2(config)#vtp mode client
wg2(config)#vtp domain preplogic
wg2(config)#vtp password bcmsn
```

Switch wg3:

```
wg3#configure terminal
wg3(config)#vtp mode server
wg3(config)#vtp domain preplogic
wg3(config)#vtp password bcmsn
wg3(config)#version 2
```

Note that since the two client switches have the VTP domain and password configured, all other VTP configuration changes can be made on wg3 and propagated to the clients. That's why the version is only enabled on the server, you'll find that once it is entered on the server, the VTP version 2 is also enabled on the client switches. You can also see that the VTP Revision number incremented because of the change.

Now, the VTP status for our respective switches in the preplogic domain looks like this:

```
wg3#sh vtp status
VTP Version                        : 2
Configuration Revision             : 5
Maximum VLANs supported locally    : 1005
Number of existing VLANs           : 7
VTP Operating Mode                 : Server
VTP Domain Name                    : preplogic
VTP Pruning Mode                   : Disabled
VTP V2 Mode                        : Enabled
VTP Traps Generation               : Disabled
MD5 digest                         : 0x8F 0x6B 0x23 0x32 0x18 0x0B 0x2D 0x73
```

```
wg2#sh vtp status
VTP Version                        : 2
Configuration Revision             : 5
Maximum VLANs supported locally    : 68
Number of existing VLANs           : 7
VTP Operating Mode                 : Client
VTP Domain Name                    : preplogic
VTP Pruning Mode                   : Disabled
VTP V2 Mode                        : Enabled
VTP Traps Generation               : Disabled
MD5 digest                         : 0x8F 0x6B 0x23 0x32 0x18 0x0B 0x2D 0x73
```

```
wg1#sh vtp status
VTP Version                          : 2
Configuration Revision               : 5
Maximum VLANs supported locally      : 68
Number of existing VLANs             : 7
VTP Operating Mode                   : Client
VTP Domain Name                      : preplogic
VTP Pruning Mode                     : Disabled
VTP V2 Mode                          : Enabled
VTP Traps Generation                 : Disabled
MD5 digest                           : 0x8F 0x6B 0x23 0x32 0x18 0x0B 0x2D 0x73
```

To verify the VTP password information, execute the **show vtp password** command:

```
wg3#show vtp password
VTP Password: bcmsn
```

Now we'll go back to our VTP server and add VLAN 300.

```
wg3#configure terminal
wg3(config)#vlan 300
```

Moving over to switch wg2, you'll see that the new VLAN has been propagated automatically!

```
wg2#sh vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/3, Fa0/4, Fa0/5, Fa0/6,
                                                Fa0/7, Fa0/8, Fa0/9, Fa0/10,
                                                Fa0/11, Fa0/12, Fa0/13, Fa0/14,
                                                Fa0/15, Fa0/16, Fa0/17, Fa0/18,
                                                Fa0/19, Fa0/20, Fa0/21, Fa0/22,
                                                Fa0/23, Fa0/24
100  VLAN0100                         active
200  VLAN0200                         active
300  VLAN0300                         active
1002 fddi-default                     active
1003 trcrf-default                    active
1004 fddinet-default                  active
1005 trbrf-default                    active
```

VLAN Pruning

As stated before, VTP pruning reduces network traffic by reducing unnecessary flooded traffic, such as broadcast, multicast, and flooded unicast packets. VTP pruning restricts flooded traffic to those trunk links that have the VLAN configured on a port. Using our previous example, we'll enable VTP pruning and set the prune eligible VLAN on the trunk to switch wg2. We will prune VLAN 200 for the example below.

Example:

```
wg3#configure terminal
wg3(config)#vtp pruning
wg3(config)#interface fa0/2
wg3(config-if)#switchport trunk pruning vlan 200
```

If switch wg2 does not have any switchports configured in VLAN 200, the broadcast traffic will not be sent across the trunk and into the switch.

Spanning Tree Protocol (STP)

Spanning Tree Protocols in a Hierarchical Network

Spanning tree protocol (STP) is an IEEE standard based on the 802.1D algorithm. The purpose of STP is to prevent layer 2 loops on a network. It uses bridge protocol data units (BPDU) to talk to other switches to remove loops by putting them into a blocking state. The exchanges of BPDUs also can turn blocked ports back into speaking ports when a link failure is detected. This allows engineers to configure redundant connections between switches while guaranteeing that a loop will not be formed.

Every port on a switch has a unique identifier which is its MAC address. When two switches are connected, the STP algorithm assigns a default path cost to the link based on the type and speed of the link. This path cost can be manipulated as needed.

The spanning tree algorithm (STA) is recalculated each time a network topology change occurs such as an adding or removal of a new link or trunk, or router, or switch. This is determined when a switch port begins or stops receiving BPDUs. BPDUs contain a wealth of information that assist the switch in figuring which path to forward on and which path to block. The default STP information that is contained in the BPDU is as follows:

Setting:	Default:
Switch Priority	32768
Port Cost	1000 Mbps: 4
	100 Mbps: 19
	10 Mbps: 100
Port Priority	128
Hello interval	2 seconds
Forward delay	15 seconds
Max age	20 seconds

The switch priority plays a determining role in which switch becomes the root bridge. The switch with the lowest priority becomes the root. The other factor that determines the root bridge is the port MAC address. The lower MAC address becomes the root if the switch priority is the same on two or more switches. The path costs and port priority help to compute which links are placed into forwarding or blocking state. The hello timer is the time between BPDUs being sent. The forward delay timer determines the amount of time the listening and learning states last prior to being moved to a forwarding state. The max age timer determines the amount of time the switch stores STP information received on a port.

There are several types of STP that can be configured on Cisco devices.

- Common Spanning Tree (CST): This STP protocol provides a single instance for the entire layer 2 network. All VLANs within the switched network share the same instance.
- Per VLAN Spanning Tree (PVST): This protocol maintains a separate STP instance for every VLAN configured on the switched network. One of the advantages of PVST is that it can load balance and separate traffic across trunk links because one instance can block a port on a trunk inside a particular VLAN while the other instance will forward their traffic out the same trunk.
- Multiple Instance Spanning Tree (MISTP): This is an IEEE 802.1S protocol that allows engineers to map multiple VLANs into an instance of STP.
- Rapid Spanning Tree (RSTP): RSTP is an IEEE 802.1w standard that is the evolution of STP over time. The protocol implements many new features to allow for faster convergence after a topology change has occurred. The features are the equivalent to PortFast, BackboneFast and UplinkFast which can be configured with regular STP but are Cisco proprietary. RSTP can converge in less than a second compared to up to 50 seconds with standard STP.
- Per VLAN Rapid Spanning Tree (PVRST): A combination of PVST+ and Rapid spanning tree that allows for use of the rapid spanning tree features on a per VLAN basis.

For the BCMSN exam, you will need to know how to configure PVRST and MISTP.

Configuring PVRST

Figure 5 will be used for our STP configuration examples. Swtches MLS-1 and MLS-2 are the distribution layer switches. VLANs 100, 200, 300 and 400 are already pre-configured on the MLS switches. 802.1q trunk links will be configured between the MLS switches and the wg1 switch at the access layer. We want to take advantage of both links to wg1 so we'll load balance the VLANs by making MLS-1 the root bridge for VLAN 100 and 200 and MLS-2 will be the bridge for VLAN 300 and 400. That means that on wg1, all VLAN 100 and 200 traffic will go in and out of port fa0/1 and VLAN 300/400 traffic on port fa0/2.

Syntax:

```
Switch# configure terminal
Switch(config)#spanning-tree mode {pvst | rapid-pvst} | mst}
Switch(config-if)# spanning-tree vlan vlan-id priority priority
```

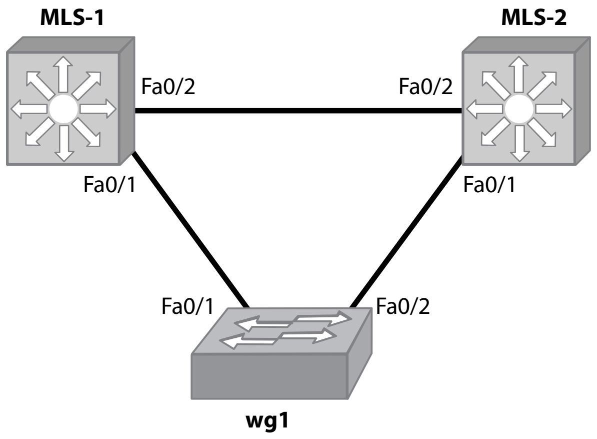


Figure 5

Example:

```
MLS-1#configure terminal
MLS-1(config)#spanning-tree mode rapid-pvst
MLS-1(config)#spanning-tree backbonefast
MLS-1(config)#spanning-tree vlan 100,200 priority 8192
MLS-1(config)#spanning-tree vlan 300,400 priority 16384
MLS-1(config)#interface FastEthernet0/1
MLS-1(config-if)# switchport trunk encapsulation dot1q
MLS-1(config-if)# switchport mode trunk
MLS-1(config-if)# switchport trunk allowed vlan 100,200,300,400
MLS-1(config)#interface FastEthernet0/2
MLS-1(config-if)# switchport trunk encapsulation dot1q
MLS-1(config-if)# switchport mode trunk
MLS-1(config-if)# switchport trunk allowed vlan 100,200,300,400
```

```
MLS-2#configure terminal
MLS-2(config)#spanning-tree mode rapid-pvst
MLS-2(config)#spanning-tree backbonefast
MLS-2(config)#spanning-tree vlan 300,400 priority 8192
MLS-2(config)#spanning-tree vlan 100,200 priority 16384
MLS-2config)#interface FastEthernet0/1
MLS-2(config-if)# switchport trunk encapsulation dot1q
MLS-2(config-if)# switchport mode trunk
MLS-2(config-if)# switchport trunk allowed vlan 100,200,300,400
MLS-2config)#interface FastEthernet0/2
MLS-2(config-if)# switchport trunk encapsulation dot1q
MLS-2(config-if)# switchport mode trunk
MLS-2(config-if)# switchport trunk allowed vlan 100,200,300,400
```

```
wg1#configure terminal
wg1(config)#spanning-tree mode rapid-pvst
wg1(config)#spanning-tree portfast bpduguard default
wg1(config)#interface FastEthernet0/1
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#switchport trunk allowed vlan 100,200,300,400
wg1(config)#interface FastEthernet0/2
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#switchport trunk allowed vlan 100,200,300,400
```

Now if we do a **show spanning-tree active** command on switch wg1 we see that there is indeed a separate instance of RSTP on each VLAN. VLANs 100 and 200 are forwarding traffic out Fa0/1 and blocking on Fa0/2. VLANs 300 and 400 are blocking on Fa0/1 and forwarding on Fa0/2:

```
wg1#show spanning-tree active

VLAN0100
 Spanning tree enabled protocol rstp
 Root ID   Priority      8192
           Address       000e.392c.68bc
           Cost          19
           Port          1 (FastEthernet0/1)
           Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

 Bridge ID  Priority       32768
            Address        000e.39d9.40a4
            Hello Time     2 sec  Max Age 20 sec  Forward Delay 15 sec
            Aging Time    300

Interface       Role Sts Cost     Prio.Nbr   Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1          Root FWD 4       128.1       P2p
Fa0/2          Altn BLK 4        128.2      P2p
```

```
VLAN0200
 Spanning tree enabled protocol rstp
 Root ID   Priority      8192
           Address       000e.392c.48c4
           Cost          19
           Port          1 (FastEthernet0/1)
           Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

 Bridge ID  Priority       32768
            Address        000e.39d9.40a4
            Hello Time     2 sec  Max Age 20 sec  Forward Delay 15 sec
            Aging Time    300

Interface       Role    Sts    Cost      Prio.Nbr     Type
----------------  ----      --- --------- -------- --------------------------------
Fa0/1          Root    FWD      4        128.1        P2p
Fa0/2          Altn    BLK      4        128.2        P2p
```

```
VLAN0300
 Spanning tree enabled protocol rstp
 Root ID   Priority      8192
           Address       000e.392c.43b4
           Cost          19
           Port          1 (FastEthernet0/1)
           Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

 Bridge ID  Priority     32768
            Address      000e.39d9.30a1
            Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec
            Aging Time   300

Interface      Role    Sts    Cost     Prio.Nbr  Type
---------------- ----    --- --------- -------- --------------------------------
Fa0/1          Root    FWD   19        128.1     P2p
Fa0/2          Altn    BLK   19        128.2     P2p
```

```
VLAN0400
 Spanning tree enabled protocol rstp
 Root ID   Priority      8192
           Address       000e.392c.18b6
           Cost          19
           Port          1 (FastEthernet0/1)
           Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

 Bridge ID  Priority     32768
            Address      000e.39d9.20c9
            Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec
            Aging Time   300

Interface      Role    Sts    Cost     Prio.Nbr  Type
---------------- ----   ---   --------- -------- --------------------------------
Fa0/1          Altn    BLK   19        128.1     P2p
Fa0/2          Root    FWD   19        128.2     P2p
```

Next, we'll move to the MLS-1 switch and look at the VLAN 100 RSTP instance to verify the proper STP setup. Here, we can see that the Root ID MAC address and it's own Bridge ID MAC address are the same. It also states that this bridge is the root for VLAN 100. Also note that both Fa0/1 and Fa0/2 have a role of Desg and a status of forwarding (FWD).

MLS-1#show spanning-tree active

```
VLAN0100
  Spanning tree enabled protocol rstp
  Root ID    Priority     8192
             Address      000e.392c.68bc
             This bridge is the root
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority     8192
             Address      000e.392c.68bc
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time   300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1            Desg FWD 19        128.1    P2p
Fa0/2            Desg FWD 19        128.1    P2p
```

Now we can compare what switch MLS-2 looks like for VLAN 100. The bridge priority of VLAN 100 for this switch is 16384. Since MLS-1 has a lower priority, it is the root switch. The Root ID address is MAC address of the Bridge Address on switch MLS-1. Finally, interface Fa0/1 is Desg and forwarding (FWD) and Fa0/2 is Root and FWD.

MLS-2#show spanning-tree active

```
VLAN0100
  Spanning tree enabled protocol rstp
  Root ID    Priority     8192
             Address      000e.392c.68bc
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority     16384
             Address      000e.392c.6863
             Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
             Aging Time   300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/1            Desg FWD 19        128.1    P2p
Fa0/2            Root FWD 19        128.261  P2p
```

Configuring MISTP

The goal of MISTP is to group multiple VLANs into an instance of rapid-spanning tree. Grouping multiple VLANs into one rapid-spanning tree instance can reduce CPU and bandwidth resources by reducing BPDU traffic. All VLANs are first placed into a single instance which is IST0 (internal spanning tree). You can then create new instances from 1 to 4094. These manual instances are called MST's. Keep in mind however that Cisco switches can only support a maximum of 16 to 65 instances depending on the hardware. IST is the only instance that can send and receive BPDUs in the MST network. ISTs in different regions are interconnected through common spanning tree (CST). Also keep in mind that MISTP is fully backward compatible with PVST and 802.1D STP.

Syntax:

```
Switch# configure terminal
Switch(config)#spanning-tree mode {pvst | rapid-pvst} | mst}
Switch(config-if)#spanning-tree mst configuration
Switch(config-mst)#name instance-name
Switch(config-mst)#revision revision-number
Switch(config-mst)#instance instance-number vlan vlan-id
Switch(config-mst)#exit
MLS-1(config)# spanning-tree mst instance-number root {primary | secondary}
```

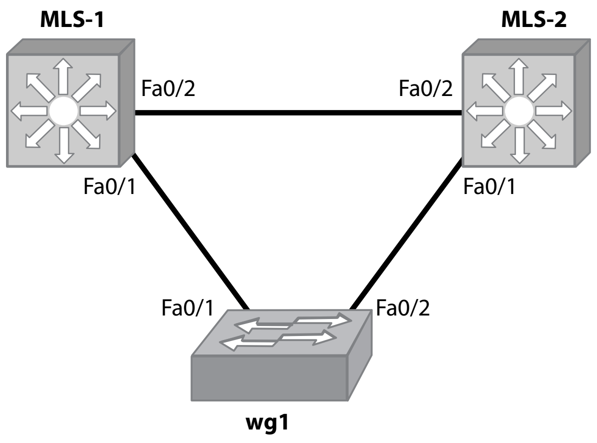


Figure 6

Figure 6 is the example we will be using to configure MISTP. Instance 1 will contain VLANs 100, 200 and 300. Instance 2 contains 10, 20 and 30. IST 0 and MST 1 will be configured to use MLS-1 as the root. MST 2 will use MLS-2 for it's root switch. This effectively load balances the traffic across both trunk links on access switch wg1.

Example:

```
MLS-1(config)# spanning-tree mode mst
MLS-1(config)# spanning-tree mst configuration
MLS-1(config-mst)# name preplogic1
MLS-1(config-mst)# revision 10
MLS-1(config-mst)# instance 1 vlan 100, 200, 300
MLS-1(config-mst)# instance 2 vlan 10, 20, 30
MLS-1(config-mst)# exit
MLS-1(config)# spanning-tree mst 0-1 root primary
MLS-1(config)# spanning-tree mst 2 root secondary
```

```
MLS-2(config)# spanning-tree mode mst
MLS-2(config)# spanning-tree mst configuration
MLS-2(config-mst)# name preplogic1
MLS-2(config-mst)# revision 10
MLS-2(config-mst)# instance 1 vlan 100, 200, 300
MLS-2(config-mst)# instance 2 vlan 10, 20, 30
MLS-2(config-mst)# exit
MLS-2(config)# spanning-tree mst 2 root primary
MLS-2(config)# spanning-tree mst 0-1 root secondary
```

STP Mechanisms

A Cisco switch has several methods that can be implemented to insure faster convergence and so a spanning tree loop does not accidentally occur. These mechanisms include:

- PortFast
- UplinkFast
- BackboneFast
- BPDU guard
- BPDU filtering L
- Loop guard
- Root guard
- Unidirectional link detection (UDLD)

PortFast

When a switchport connects to a single end device such as a PC or server, spanning-tree PortFast can be configured to allow the port to bypass the listening and learning phases of STP and begin forwarding immediately. PortFast is an interface command on the switch.

Syntax:

```
Switch#configure terminal
Switch (config)#interface interface-id
Switch(config-if)#spanning-tree portfast [trunk]
```

```
show spanning-tree interface interface-id
```

Note that you can add PortFast to a trunk port with the trunk command. You need to insure that there are no physical loops prior to removing PortFast on a trunk link.

BPDU Guard

PortFast is a good thing but the problem occurs when a switch gets plugged into one of these ports does not check for other forwarding ports. If this happens, a loop can occur. BPDU Guard fixes this problem by monitoring the port for any signs of BPDU's on the switchport. This command can be configured on any non-trunking access port. These ports should never see BPDU information on them. If the switch does see a BPDU sent across a switchport, that particular port is put into errdisable mode and becomes inoperable until it is brought back up by performing a no shutdown on the interface. BPDU Guard is a global configuration command.

Syntax:

```
Switch# configure terminal
Switch(config)#spanning-tree portfast bpduguard
```

You can also configure BPDU guard on a per-interface basis.

Syntax:

```
Switch#configure terminal
Switch(config)#interface interface-id
Switch(config-if)#spanning-tree bpduguard enable
```

BPDU Filtering

When ports are configured for spanning tree Port Fast, they send out a few BPDU's when the port becomes active. Enabling BPDU filtering at a global level stops all BPDU's from being sent. If a BPDU is detected on a Port Fast enabled port with BPDU filtering, it changes to a standard non-Port Fast port.

Syntax:

```
Switch#configure terminal
Switch(config)#spanning-tree portfast bpdufilter default
Switch(config)#interface interface-id
Switch(config-if)#spanning-tree portfast
```

UDLD

Unidirectional link detection (UDLD) is a protocol that monitors and detects physical problems with fiber optic or copper cabling. Physical failures can cause an up state in a single direction. This can lead to spanning-tree loops. When the UDLD protocol detects a problem, it shuts down the port administratively and sends a log message.

UDLD only works when both end of the links are capable of running the protocol. The protocol sends and receives hello messages that communicate the health of the physical link. The message timers can be modified to send hello's from 1 to 90 seconds.

The two modes of UDLD are normal and aggressive.

- UDLD normal: Detects unidirectional problems on fiber optic cabling due to misconnected fiber.
- UDLD aggressive: Detects one-way traffic on fiber and copper links in addition to misconnected fiber links. For example, if a fiber or copper point-to-point connection is able to send traffic but not receive it. Another example is if one side of the link is up while the opposite side is down.

Syntax:

UDLD can be configured at a global or interface level. The global configuration only affects fiber ports. Copper ports must be configured at an interface level to enable UDLD aggressive mode.

Global Configuration:

```
Switch#configure terminal
Switch(config)#udld {aggressive | enable | message time message-timer-interval}
```

Interface Configuration:

```
Switch#configure terminal
Switch(config)#interface interface-id
Switch(config-if)#udld port [aggressive]
```

To verify the configuration of UDLD use the **show udld** for global configurations and **show udld** interface-id to verify UDLD configured at an interface level.

If an interface has been shut down to a UDLD violation, the **udld reset** command clears resets the interface so it can begin functioning properly again.

Loop Guard

Loop guard is used to prevent non-designated ports from becoming designated ports because of a UDLD link detection. Loop guard only works with P2P links and operates only when PVST+, RPVST+ and MSTP are configured as the spanning-tree protocol.

Syntax:

```
Switch#configure terminal
Switch(config)#spanning-tree loopguard default
```

Root Guard

Root guard prevents a switch from becoming the root bridge. Root guard is performed on a per-port basis and insures that the root guard configured port maintains a designated state.

Syntax:

```
Switch#configure terminal
Switch(config)#interface interface-id
Switch(config-if)#spanning-tree guard root
```

Link Aggregation (EtherChannel)

EtherChannel provides a way to aggregate multiple Ethernet connections into one logical link. It's a simple way to increase bandwidth between your switches where you find bottlenecks. Once a group of links are bonded with EtherChannel, it has the ability to recover when a link failure occurs within the aggregate channel. Up to 8 fast Ethernet or gigabit Ethernet links can be channeled to provide for up to 800 and 8000 Mbps full duplex bandwidth respectively. All the links in the EtherChannel must be of the same speed and they need to be configured as layer 2 or layer 3 interfaces.

EtherChannel is configured in one of three ways on a Cisco switch. EtherChannel On mode statically sets up the links into a channel. Link Aggregation Control Protocol (LACP) and Port Aggregation Control Protocol (PAgP) negotiates channelized ports with the other side. No matter which method you choose, both sides of the channel link must be configured the same.

Load balancing on EtherChannels happens at the MAC address level. By default, source-address load balancing is configured. What this means is that when a frame enters an EtherChannel, it gets randomly assigned a link to use based on the source MAC address. This works great if you have multiple hosts attached to a switch that is configured for EtherChannel. Each PC or end device has a unique MAC address and load-balancing will occur across the links. When traffic has to go through a layer 3 device that connects to an EtherChannel link, source based MAC address balancing does not work because all traffic will come from the MAC address assigned to the layer 3 interface. In this case, it is better to use destination-based Load balancing which randomly assigns frames to ports within the EtherChannel based on the destination MAC address.

Layer 2 EtherChannel

Syntax:

```
Switch#configure terminal
Switch(config)#interface interface-id
Switch(config-if)#switchport mode {access | trunk}
channel-group channel-group-number mode {{auto [non-silent] | desirable [non-silent] | on} |
{active | passive}}
```

Layer 3 EtherChannel

```
Switch#configure terminal
Switch(config)#interface port-channel port-channel-number
Switch(config-if)#no switchport
Switch(config-if)#ip address ip-address mask
Switch(config-if)#interface interface-id
Switch(config-if)#no switchport
Switch(config-if)#no ip address
Switch(config-if)#channel-group channel-group-number mode {auto [non-silent] | desirable
[non-silent] | on | active | passive}
```

EtherChannel Load Balancing

```
Switch#configure terminal
Switch(config)#port-channel load-balance {src-mac | dst-mac}
```

EtherChannel on Configuration

EtherChannel On mode does not negotiate channeling. It forces a port to join an EtherChannel and is useful when one side of a switch does not support either PAgP or LACP. If a port is configured that is not compatible (i.e. configuring 2 GigE ports and one FE port) that port will be placed into a suspended state. Our example will configure a layer 2 EtherChannel mode on between wg1 and wg2. The EtherChannel will be an 802.1q trunk that transports VLANs 100 and 200. Use *Figure 7* as the physical setup.

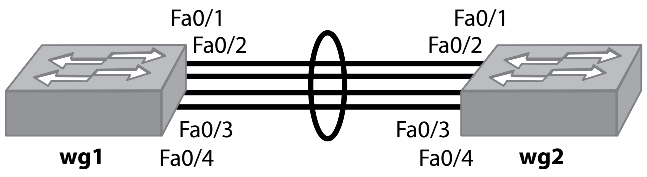


Figure 7

Example:

Switch wg1:

```
wg1#configure terminal
wg1(config)#interface fa0/1
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#channel-group 1 mode on
wg1(config)#interface fa0/2
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#channel-group 1 mode on
wg1(config)#interface fa0/3
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#channel-group 1 mode on
wg1(config)#interface fa0/4
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#channel-group 1 mode on
```

Switch wg2:

```
wg2#configure terminal
wg2(config)#interface fa0/1
wg2(config-if)#switchport trunk encapsulation dot1q
wg2(config-if)#switchport mode trunk
wg2(config-if)#channel-group 1 mode on
wg2(config)#interface fa0/2
wg2(config-if)#switchport trunk encapsulation dot1q
wg2(config-if)#switchport mode trunk
wg2(config-if)#channel-group 1 mode on
wg2(config)#interface fa0/3
wg2(config-if)#switchport trunk encapsulation dot1q
wg2(config-if)#switchport mode trunk
wg2(config-if)#channel-group 1 mode on
wg2(config)#interface fa0/4
wg2(config-if)#switchport trunk encapsulation dot1q
wg2(config-if)#switchport mode trunk
wg2(config-if)#channel-group 1 mode on
```

You can check the status of your EtherChannel by using the following commands:

```
wg2#show etherchannel summary
Flags:  D - down        P - in port-channel
      I - stand-alone s - suspended
      H - Hot-standby (LACP only)
      R - Layer3      S - Layer2
      u - unsuitable for bundling
      U - in use      f - failed to allocate aggregator
      d - default port

Number of channel-groups in use: 1
Number of aggregators:          1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SU)         -       Fa0/1(P)   Fa0/2(P)    Fa0/3(P)
                              Fa0/4(P)

wg2#show etherchannel protocol
            Channel-group listing:
            ----------------------

Group: 1
----------
Protocol:  -  (Mode ON)

wg2#show etherchannel load-balance
Source MAC address
```

PAgP Configuration

PAgP is a Cisco proprietary protocol that facilitates the creation of EtherChannels automatically. There are two ways to setup the exchange of PAgP negotiation messages between two Cisco devices:

- Desirable: Actively sends out PAgP negotiation messages to establish the EtherChannel.
- Auto: A passive listening state where the interface will respond to PAgP messages but will not initiate communication. It helps to reduce negotiation messages.

Note: If both sides of the channel are configured for auto, the channel will never establish.

In our example, we will configure a Layer 2 PAgP EtherChannel to channelize 4 Fe 802.1q trunk ports. One side of the PAgP setup will be configured as desirable and the other will be setup as auto. We also want to configure destination-based load balancing. *Figure 8* is the diagram for the physical setup in this example.

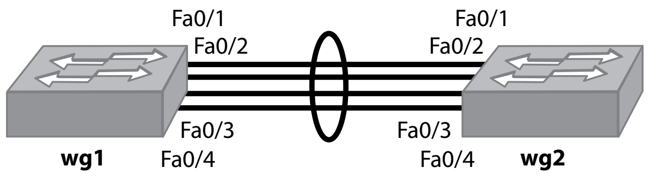


Figure 8

Example:

Switch wg1:

```
wg1#configure terminal
wg1(config)#interface fa0/1
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#channel-group 1 mode desirable
wg1(config)#interface fa0/2
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#channel-group 1 mode desirable
wg1(config)#interface fa0/3
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#channel-group 1 mode desirable
wg1(config)#interface fa0/4
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#channel-group 1 mode desirable
wg1(config-if)#exit
wg1(config)#port-channel load-balance dst-mac
```

Switch wg2:

```
wg2#configure terminal
wg2(config)#interface fa0/1
wg2(config-if)#switchport trunk encapsulation dot1q
wg2(config-if)#switchport mode trunk
wg2(config-if)#channel-group 1 mode auto
wg2(config)#interface fa0/2
wg2(config-if)#switchport trunk encapsulation dot1q
wg2(config-if)#switchport mode trunk
wg2(config-if)#channel-group 1 mode auto
wg2(config)#interface fa0/3
wg2(config-if)#switchport trunk encapsulation dot1q
wg2(config-if)#switchport mode trunk
wg2(config-if)#channel-group 1 mode auto
wg2(config)#interface fa0/4
wg2(config-if)#switchport trunk encapsulation dot1q
wg2(config-if)#switchport mode trunk
wg2(config-if)#channel-group 1 mode auto
wg2(config-if)#exit
wg2(config)#port-channel load-balance dst-mac
```

Let's verify our PAgP configuration using some show commands:

```
wg2#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        u - unsuitable for bundling
        U - in use      f - failed to allocate aggregator
        d - default port

Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(SU)        PAgP      Fa0/1(P)   Fa0/2(P)   Fa0/3(P)
                                Fa0/4(P)
```

```
wg2#show etherchannel protocol
                Channel-group listing:
                ----------------------

Group: 1
----------
Protocol:  PAgP
```

```
wg2#show etherchannel load-balance
Destination MAC address
```

LACP Configuration

LACP is an IEEE 802.1ad standard protocol. It works virtually the same way as PAgP. There are two ways to setup LACP communications between interfaces:

Active: Actively negotiates with other interfaces by sending LACP messages.

Passive: Places the interface into listening mode that waits to hear LACP messages from Active ports.

Note: If both sides of the channel are configured for passive, the channel will never establish.

For our LACP configuration, we'll configure a Layer 3 LACP EtherChannel between 2 routers using 4 FE connections. *Figure 9* will be the physical setup for the example.

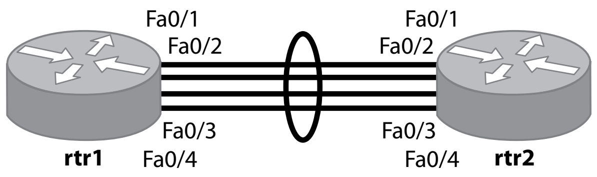


Figure 9

Example:

Router **rtr1**:

```
rtr1#configuration terminal
rtr1(config)#int port-channel 1
rtr1(config-if)#ip add 192.168.1.1 255.255.255.0
rtr1(config-if)#int fa0/1
rtr1(config-if)#no ip address
rtr1(config-if)#channel-group 1 mode active
rtr1(config-if)#int fa0/2
rtr1(config-if)#no ip address
rtr1(config-if)#channel-group 1 mode active
rtr1(config-if)#int fa0/3
rtr1(config-if)#no ip address
rtr1(config-if)#channel-group 1 mode active
rtr1(config-if)#int fa0/4
rtr1(config-if)#no ip address
rtr1(config-if)#channel-group 1 mode active
```

Router **rtr2**:

```
rtr2#configuration terminal
rtr2(config)#int port-channel 1
rtr2(config-if)#no switchport
rtr2(config-if)#ip add 192.168.1.2 255.255.255.0
rtr2(config-if)#int fa0/1
rtr2(config-if)#no ip address
rtr2(config-if)#channel-group 1 mode passive
rtr2(config-if)#int fa0/2
rtr2(config-if)#no ip address
rtr2(config-if)#channel-group 1 mode passive
rtr2(config-if)#int fa0/3
rtr2(config-if)#no ip address
rtr2(config-if)#channel-group 1 mode passive
rtr2(config-if)#int fa0/4
rtr2(config-if)#no ip address
rtr2(config-if)#channel-group 1 mode passive
```

Let's verify our LACP configuration using some show commands:

```
rtr2#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        u - unsuitable for bundling
        U - in use      f - failed to allocate aggregator
        d - default port

Number of channel-groups in use: 1
Number of aggregators:           1

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------
1      Po1(RU)        LACP      Fa0/1(P)    Fa0/2(P)    Fa0/3(P)
                                Fa0/4(P)
```

```
rtr2#show etherchannel protocol
                Channel-group listing:
                ----------------------
Group: 1
----------
Protocol:  LACP
```

```
rtr2#show etherchannel load-balance
Source MAC address
```

Inter-VLAN Routing

Explanation of Inter-VLAN Routing

As we have learned, VLANs are logical broadcast segmentations of a network. In order to send traffic from a device on VLAN A to a device on VLAN B, the traffic must go through a layer 3 routed interface. Inter-VLAN routing provides the mechanism to enable routing of traffic between VLANs.

Configuring Inter-VLAN Routing

There are basically two ways to configure Inter-VLAN routing. The first method is to configure a router on a stick. This method uses a standard layer 2 switch that is configured for multiple VLANs. Each VLAN has either a separate or trunk port to a router that has the VLAN default gateway IP address(es) configured on it. When a device on a VLAN needs to communicate to a device on another VLAN, that traffic is sent up to the router on one VLAN and returns to the switch on the other VLAN. *Figure 10* is an example Inter-VLAN routing using the router on a stick method.

Router Syntax:

```
Router#configure terminal
Router(config)#interface interface-id
Router(config-if)#no shutdown
Router(config-if)#interface interface-id.sub-int-id
rtr1(config-subif)#encapsulation {dot1q| isl } vlan-id
Router(config-subif)#ip address ip-address subnet-mask
Router(config-subif)#end
```

Switch Syntax:

```
Switch#configure terminal
Switch(config)#interface vlan vlan-id
Switch(config-if)#no shutdown
Switch(config)# interface interface-id
Switch(config-if)# switchport trunk encapsulation {isl | dot1q | negotiate}
Switch(config-if)# switchport mode {dynamic {auto | desirable} | trunk}
```

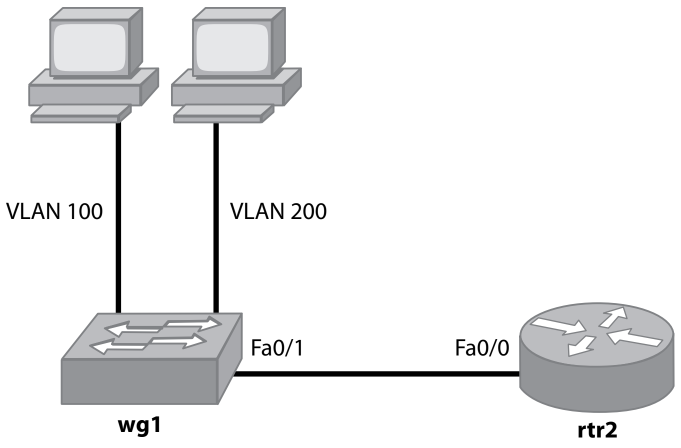


Figure 10

In this example, we will configure an 802.1q trun between wg2 and rtr1. Use sub-interface fa0/0.1 for VLAN 100 and fa0/0.2 for vlan 200.

Example:

Router Configuration:

```
rtr1#configure terminal
rtr1(config)#interface FastEthernet0/0
rtr1(config-if)#no shutdown
rtr1(config-if)#interface FastEthernet0/0.1
rtr1(config-subif)#encapsulation dot1Q 100
rtr1(config-subif)#ip address 192.168.100.1 255.255.255.0
rtr1(config-subif)#interface FastEthernet0/0.2
rtr1(config-subif)#encapsulation dot1Q 200
rtr1(config-subif)#ip address 192.168.200.1 255.255.255.0
```

Switch Configuration:

```
wg1#configure terminal
wg1(config)#interface vlan 100
wg1(config-if)#no shutdown
wg1(config-if)#interface vlan 200
wg1(config-if)#no shutdown
wg1(config-if)#int fa0/20
wg1(config-if)#switchport access vlan 100
wg1(config-if)#spanning-tree portfast
wg1(config-if)#int fa0/21
wg1(config-if)#switchport access vlan 200
wg1(config-if)#spanning-tree portfast
wg1(config-if)#int fa0/1
wg1(config-if)#switchport trunk encapsulation dot1q
wg1(config-if)#switchport mode trunk
wg1(config-if)#end
```

To verify inter-VLAN routing, do the fowllowing router and switch commands:

```
rtr1#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
   D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
   N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
   E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
   i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
   * - candidate default, U - per-user static route, o - ODR
   P - periodic downloaded static route

Gateway of last resort is not set

C   192.168.200.0/24 is directly connected, FastEthernet0/0.2
C   192.168.100.0/24 is directly connected, FastEthernet0/0.1
```

```
wg1#sh vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/2, Fa0/3, Fa0/4, Fa0/5
                                                Fa0/6, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10, Fa0/11, Fa0/12, Fa0/13
                                                Fa0/14, Fa0/15, Fa0/16, Fa0/17
                                                Fa0/18, Fa0/19, Fa0/22, Fa0/23
                                                Fa0/24, Fa0/25, Fa0/26, Fa0/27
                                                Fa0/28, Fa0/29, Fa0/30, Fa0/31
                                                Fa0/32, Fa0/33, Fa0/34, Fa0/35
                                                Fa0/36, Fa0/37, Fa0/38, Fa0/39
                                                Fa0/40, Fa0/41, Fa0/42, Fa0/43
                                                Fa0/44, Fa0/45, Fa0/46, Fa0/47
                                                Fa0/48, Gi0/1, Gi0/2
100  VLAN0100                         active    Fa0/20
200  VLAN0200                         active    Fa0/21
```

```
wg1#show interfaces trunk

Port      Mode         Encapsulation  Status        Native vlan
Fa0/1     on           802.1q         trunking      1

Port      Vlans allowed on trunk
Fa0/1     1-4094

Port      Vlans allowed and active in management domain
Fa0/1     1,100,200,300

Port      Vlans in spanning tree forwarding state and not pruned
Fa0/1     1,100,200
```

The other method to configure Inter-VLAN routing is to use a multi-layer switch (MLS) to configure switch virtual interfaces (SVI). SVIs are virtual layer 3 interfaces that have the ability to route between VLANs. Think of an MLS with SVIs as a combination of a switch and a router. Now, you only need one switch to perform inter-VLAN routing as shown in *Figure 11*.

Syntax:

```
Switch#configure terminal
Switch(config)#ip routing
Switch(config)#interface vlan vlan-id
Switch(config-if)#ip address ip-address subnet-mask
Switch(config-if)#no shutdown
Switch(config)# interface interface-id
Switch (config-if)#interface interface-id
Switch(config-if)#switchport access vlan vlan-id
Switch(config-if)#end
```

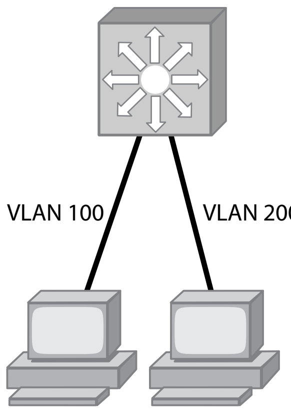


Figure 11

In this example, we will configure VLAN 100 and VLAN 200 on MLS-1 to route between each other. Assume PC100 is on Fa0/20 and PC200 is on Fa0/21.

Note: The VLAN interfaces will not go into the up/up state until a device on that VLAN is linked.

Example:

```
MLS-1(config)#ip routing
MLS-1(config)#int vlan 100
MLS-1(config-if)#ip add 192.168.100.1 255.255.255.0
MLS-1(config-if)#no shut
MLS-1(config-if)#int vlan 200
MLS-1(config-if)#ip add 192.168.200.1 255.255.255.0
MLS-1(config-if)#no shut
MLS-1(config-if)#interface fa0/20
MLS-1(config-if)#switchport access vlan 100
MLS-1(config-if)#spanning-tree portfast
MLS-1(config-if)#interface fa0/21
MLS-1(config-if)#switchport access vlan 200
MLS-1(config-if)#spanning-tree portfast
MLS-1(config-if)#end

To verify inter-VLAN routing, use the following commands:
MLS-1#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

C    192.168.200.0/24 is directly connected, Vlan200
C    192.168.100.0/24 is directly connected, Vlan100
```

```
MLS-1#show ip interface brief
Interface          IP-Address     OK?  Method   Status                 Protocol
Vlan1              unassigned     YES  unset    administratively down  down
Vlan100            192.168.100.1  YES  manual   up                     up
Vlan200            192.168.200.1  YES  manual   up                     up
FastEthernet0/1    unassigned     YES  unset    up                     down
FastEthernet0/2    unassigned     YES  unset    up                     down
FastEthernet0/3    unassigned     YES  unset    up                     down
FastEthernet0/4    unassigned     YES  unset    up                     down
FastEthernet0/5    unassigned     YES  unset    down                   down
FastEthernet0/6    unassigned     YES  unset    down                   down
FastEthernet0/7    unassigned     YES  unset    down                   down
FastEthernet0/8    unassigned     YES  unset    down                   down
FastEthernet0/9    unassigned     YES  unset    down                   down
FastEthernet0/10   unassigned     YES  unset    down                   down
FastEthernet0/11   unassigned     YES  unset    down                   down
FastEthernet0/12   unassigned     YES  unset    down                   down
FastEthernet0/13   unassigned     YES  unset    down                   down
FastEthernet0/14   unassigned     YES  unset    down                   down
FastEthernet0/15   unassigned     YES  unset    down                   down
FastEthernet0/16   unassigned     YES  unset    down                   down
FastEthernet0/17   unassigned     YES  unset    down                   down
FastEthernet0/18   unassigned     YES  unset    down                   down
FastEthernet0/19   unassigned     YES  unset    down                   down
FastEthernet0/20   unassigned     YES  unset    up                     up
FastEthernet0/21   unassigned     YES  unset    up                     up
FastEthernet0/22   unassigned     YES  unset    down                   down
FastEthernet0/23   unassigned     YES  unset    down                   down
FastEthernet0/24   unassigned     YES  unset    down                   down
```

Explain and Enable CEF

Cisco Express Forwarding (CEF) is a layer 3 switching technology that improves switching performance on CEF capable MLS switches. As we know, routing is slower than switching due to the fact that each packet in a router has to do a routing lookup prior to putting it on the wire. CEF attempts to achieve wire speeds by building a forwarding information base (FIB) that pre-populates destination routing information. Now, packets look to the FIB that already knows what route a particular packet is going to take. This speeds up the layer 3 routing process and lessens the load on the MLS CPU.

CEF is enabled by default on all Cisco layer 3 switches. The only reason that it should ever be turned off is for troubleshooting using the **debug ip packet detail** command. In order to see all forwarded packets in the debug, you must turn off CEF so every packet will be processed by the CPU. To turn off CEF on an interface, use the following syntax:

Syntax:

```
Switch#configure terminal
Switch(config)# interface interface-id
Switch(config-if)#no ip route-cache cef
```

You can verify CEF status and adjacencies by using the following commands:

```
show ip cef
show adjacency
```

Gateway Redundancy Technologies

Gateway redundancy is also known as first hop redundancy due to the fact that it provides end-device redundancy from a default gateway standpoint. The three main redundancy protocols that Cisco layer 3 devices utilize are Hot Standby Routing Protocol (HSRP), Virtual Router Redundancy Protocol (VRRP) and the Gateway Load Balancing Protocol (GLBP).

HSRP

HSRP is a Cisco proprietary protocol that performs automatic router backup when you configure it on Cisco capable devices. There are three interface types that can be configured for HSRP:

1. Standard Routed Ports
2. MLS switched virtual interfaces (SVI)
3. Layer 3 EtherChannels

HSRP takes 2 redundant layer 3 gateways and groups them together. Once they are inside an HSRP group, they are configured to share a Virtual MAC and IP address. That virtual IP address is what the end stations use as the gateway address. One of the HSRP interfaces is the active and the other is the standby interface. If for some reason, the active fails, the standby interface detects the failure and takes over the gateway duties. HSRP uses a priority configuration to determine which layer 3 interface is primary. When HSRP is initially configured, the device has a default priority of 100. The highest priority becomes the active interface.

HSRP communicates to its peers by sending multicast messages. These messages include:

- Hello: Verifies that the routers are still functioning properly.
- Coup: When a standby router becomes the active router.
- Resign: When a router that is the active router sends this message when it is about to give up being the active due to another router with a higher priority coming online.

When new HSRP routers come online, they can immediately try to become the active router if the **standby preempt** command is used. You want to use this command on the router with the highest priority.

Syntax:

```
Switch#configure terminal
Switch(config)#interface interface-id
Switch(config-if)# standby group-number ip virtual-ip-address
Switch(config-if)#standby group-number priority priority [preempt [delay delay]]
```

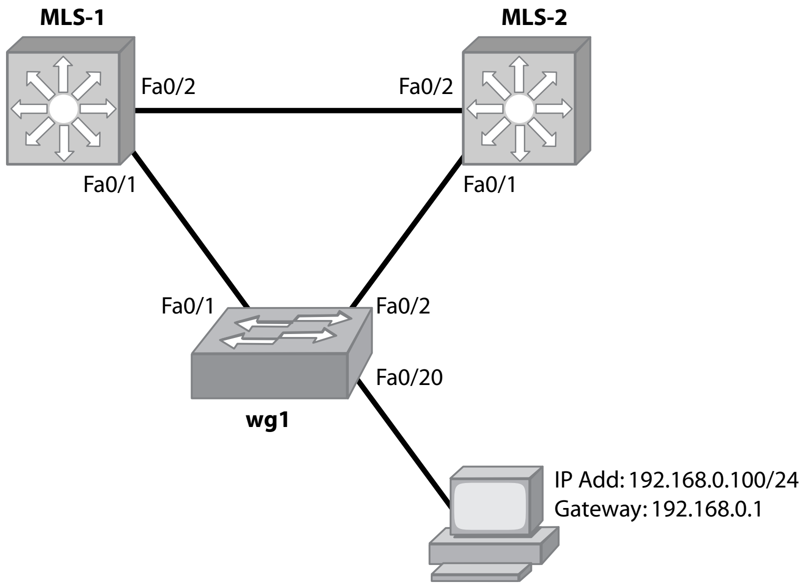


Figure 12

In our HSRP Example, we'll configure 2 MLS switches with an SVI on VLAN 100 to run HSRP between the two. 192.168.0.1 will be the HSRP virtual interface. We'd like MLS-1 to be the primary HSRP router. Use *Figure 12* as the physical layout for this example.

Example:

Switch **MLS-1**:

```
MLS-1#configure terminal
MLS-1(config)#interface FastEthernet0/1
MLS-1(config-if)#switchport access vlan 100
MLS-1(config-if)#interface FastEthernet0/2
MLS-1(config-if)#switchport access vlan 100
MLS-1(config-if)#interface Vlan100
MLS-1(config-if)#ip address 192.168.0.3 255.255.255.0
MLS-1(config-if)#standby 1 ip 192.168.0.1
MLS-1(config-if)#standby 1 priority 110 preempt
```

Switch **MLS-2**:

```
MLS-2#configure terminal
MLS-2(config)#interface FastEthernet0/1
MLS-2(config-if)#switchport access vlan 100
MLS-2(config-if)#interface FastEthernet0/2
MLS-2(config-if)#switchport access vlan 100
MLS-2(config-if)#interface Vlan100
MLS-2(config-if)#ip address 192.168.0.4 255.255.255.0
MLS-2(config-if)#standby 1 ip 192.168.0.1
```

Switch wg1:

```
wg1#configure terminal
wg1config-if)#interface fa0/1
wg1 (config-if)#switchport access vlan 100
wg1 (config-if)#interface fa0/2
wg1 (config-if)#switchport access vlan 100
```

Let’s verify our HSRP configuration using the **show standby** command on MLS-1 and MLS-2.

```
MLS-1#show standby
Vlan100 - Group 1
  Local state is Active, priority 110, may preempt
  Hellotime 3 sec, holdtime 10 sec
  Next hello sent in 1.110
  Virtual IP address is 192.168.0.1 configured
  Active router is local
  Standby router is 192.168.0.4 expires in 8.552
  Virtual mac address is 0000.0c07.ac01
  1 state changes, last state change 00:10:02
  IP redundancy name is "hsrp-Vl100-1" (default)
```

```
MLS-2#show standby
Vlan100 - Group 1
  Local state is Standby, priority 100
  Hellotime 3 sec, holdtime 10 sec
  Next hello sent in 1.852
  Virtual IP address is 192.168.0.1 configured
  Active router is 192.168.0.3, priority 110 expires in 8.824
  Standby router is local
  4 state changes, last state change 00:10:41
  IP redundancy name is "hsrp-Vl100-1" (default)
```

VRRP

VRRP is an IETF standard gateway redundancy protocol. Because it’s a standard, it can be used in multi-vendor environments. VRRP configuration enables a group of layer 3 devices to form a single virtual router. The end devices then use the virtual router as their default gateway address. VRRP works very much like HSRP but instead of creating a Virtual IP address, VRRP uses the master interface physical IP address. If for some reason, the master VRRP router fails, the backup VRRP router takes over the IP address and applies it to its physical interface.

VRRP uses a priority feature to determine which device is the master and which ones are the backups. The highest priority is the master. The priority can be configured between 1 and 254. If two routers have the same priority, the router with the highest IP address becomes the master. The default priority is 100.

Below is the syntax to configure basic VRRP:

Syntax:

```
Router#configure terminal
Router(config)#interface interface-id
Router(config-if)# ip address ip-address mask
Router(config-if)#vrrp group ip ip-address
Router(config-if)# vrrp group priority level
Router(config-if)# vrrp group preempt
```

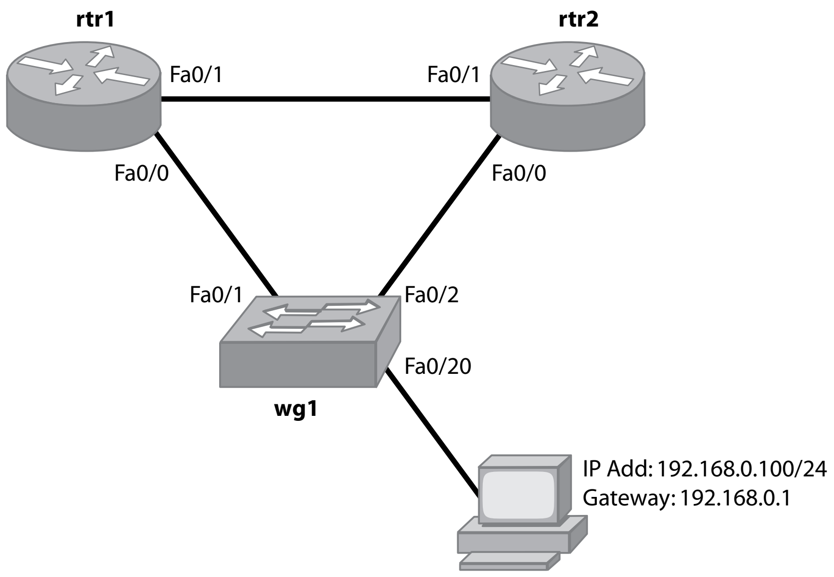


Figure 13

In our VRRP Example, we'll configure 2 routers with ip addresses of 192.168.0.1 and 192.168.0.2. 192.168.0.1 will be the VRRP master router configured on router rtr1. The network connecting the 2 routers is 10.0.0.1/30. Use *Figure 13* as the physical layout for this example.

Example:

Router **rtr1**:

```
rtr1(config)#int fa0/1
rtr1(config-if)#ip address 10.0.0.1 255.255.255.252
rtr1(config-if)#no shutdown
rtr1(config)#interface fa0/0
rtr1(config-if)#ip address 192.168.0.1 255.255.255.0
rtr1(config-if)#no shutdown
rtr1(config-if)#vrrp 1 ip 192.168.0.1
rtr1(config-if)#vrrp 1 priority 110
rtr1(config-if)#vrrp 1 preempt
```

Router rtr2:

```
rtr2(config)#int fa0/1
rtr2(config-if)#ip address 10.0.0.2 255.255.255.252
rtr2(config-if)#no shutdown
rtr2(config)#interface fa0/0
rtr2(config-if)#ip address 192.168.0.2 255.255.255.0
rtr2(config-if)#no shutdown
rtr2(config-if)#vrrp 1 ip 192.168.0.1
```

Switch wg1:

```
wg1#configure terminal
wg1config-if)#interface fa0/1
wg1 (config-if)#switchport access vlan 100
wg1 (config-if)#interface fa0/2
wg1 (config-if)#switchport access vlan 100
```

Let's verify our VRRP configuration using the **show vrrp** command on rtr1 and rtr2.

```
rtr1#show vrrp
Ethernet0/0 - Group 1
 State is Master
 Virtual IP address is 192.168.0.1
 Virtual MAC address is 0000.5e00.0101
 Advertisement interval is 1.000 sec
 Preemption enabled
 Priority is 255 (cfgd 110)
 Master Router is 192.168.0.1 (local), priority is 255
 Master Advertisement interval is 1.000 sec
 Master Down interval is 3.003 sec
```

```
rtr2#show vrrp
Ethernet0/0 - Group 1
 State is Backup
 Virtual IP address is 192.168.0.1
 Virtual MAC address is 0000.5e00.0101
 Advertisement interval is 1.000 sec
 Preemption enabled
 Priority is 100
 Master Router is 192.168.0.1, priority is 255
 Master Advertisement interval is 1.000 sec
 Master Down interval is 3.609 sec (expires in 2.583 sec)
```

GLBP

The main difference between GLBP and HSRP/VRRP is in its ability to load balance by default. With HSRP and VRRP, there are ways to design your network to load balance at the VLAN level, but this can lead to an imbalance if one VLAN utilizes the vast majority of traffic. GLBP is a per-MAC address way to load balance while still providing gateway redundancy. GLBP routers configured in the same group communicate between each other using multicast hello packets.

With GLBP, a router is elected to be the active virtual gateway (AVG). The AVGs responsibility is to hand out virtual MAC addresses to the other routers in the GLBP group. All the other routers in the group are called active virtual forwarders (AVF). They use the same gateway IP address as the rest but with a different MAC address. End devices are all configured with the same gateway address. The difference is that when the end device does an ARP lookup to get the MAC address of its gateway IP, the AVG hands out the different MAC addresses of itself and the other AVFs that are configured. This is how load balancing is achieved. A single GLBP group can have up to four AVFs. If one of the GLBP routers becomes unavailable, the other router assumes responsibility to handle traffic for the downed MAC address and its own MAC address. The end devices never notice because they're still sending their traffic to the same layer 2 MAC and layer 3 IP address.

Syntax:

```
Router#configure terminal
Router(config)#interface interface-id
Router(config-if)# ip address ip-address mask
Router(config-if)#glbp group ip ip-address
Router(config-if)#glbp group priority level preempt
```

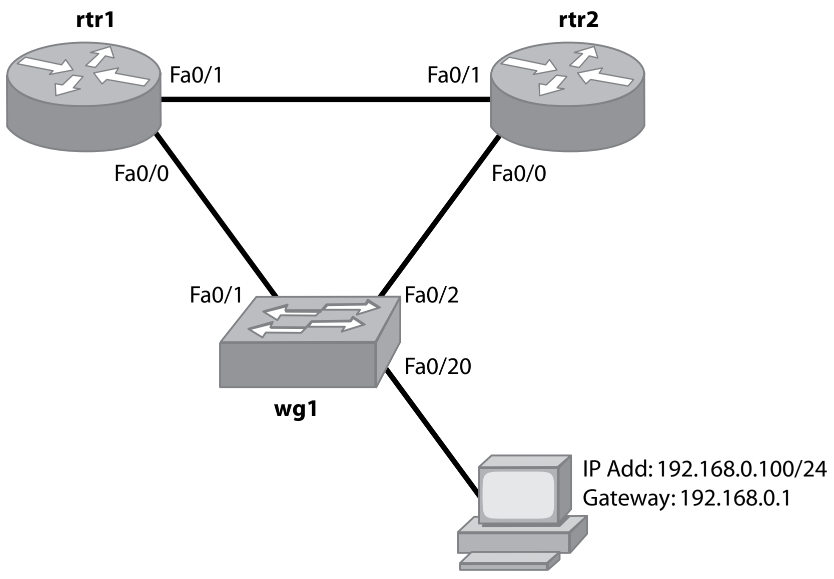


Figure 14

In our GLBP Example, we'll configure 2 routers with ip addresses of 192.168.0.3 and 192.168.0.4. 192.168.0.1 will be the GLBP group IP address that the routers will share. Router rtr1 will be configured as the AVG. The network connecting the 2 routers is 10.0.0.1/30. Use *Figure 14* as the physical layout for this example.

Example:

Router rtr1:

```
rtr1(config)#int fa0/1
rtr1(config-if)#ip address 10.0.0.1 255.255.255.252
rtr1(config-if)#no shutdown
rtr1(config)#interface fa0/0
rtr1(config-if)#ip address 192.168.0.3 255.255.255.0
rtr1(config-if)#no shutdown
rtr1(config-if)#glbp 1 ip 192.168.0.1
rtr1(config-if)#glbp 1 priority 110
rtr1(config-if)#glbp 1 preempt
```

Router rtr2:

```
rtr2(config)#int fa0/1
rtr2(config-if)#ip address 10.0.0.2 255.255.255.252
rtr2(config-if)#no shutdown
rtr2(config)#interface fa0/0
rtr2(config-if)#ip address 192.168.0.4 255.255.255.0
rtr2(config-if)#no shutdown
rtr2(config-if)#glbp 1 ip 192.168.0.1
```

Let's verify our GLBP configuration using the **show glbp** command on rtr1 and rtr2.

```
rtr1#show glbp
Ethernet0/0 - Group 1
 State is Active
  2 state changes, last state change 00:07:16
 Virtual IP address is 192.168.0.1
 Hello time 3 sec, hold time 10 sec
  Next hello sent in 1.489 secs
 Redirect time 600 sec, forwarder time-out 14400 sec
 Preemption enabled, min delay 0 sec
 Active is local
 Standby is 192.168.0.4, priority 100 (expires in 8.601 sec)
 Priority 110 (configured)
 Weighting 100 (default 100), thresholds: lower 1, upper 100
 Load balancing: round-robin
 There are 2 forwarders (1 active)
 Forwarder 1
  State is Active
   1 state change, last state change 00:07:06
  MAC address is 0007.b400.0101 (default)
  Owner ID is 0002.fd79.fba0
  Redirection enabled
  Preemption enabled, min delay 30 sec
  Active is local, weighting 100
```

Continued on next page.

Continued.

```
  Forwarder 2
    State is Listen
    MAC address is 0007.b400.0102 (learnt)
    Owner ID is 0002.b90d.5960
    Redirection enabled, 597.399 sec remaining (maximum 600 sec)
    Time to live: 14397.399 sec (maximum 14400 sec)
    Preemption enabled, min delay 30 sec
    Active is 192.168.0.4 (primary), weighting 100 (expires in 7.399 sec)
```

```
rtr2#show glbp
Ethernet0/0 - Group 1
  State is Standby
    1 state change, last state change 00:00:02
  Virtual IP address is 192.168.0.1
  Hello time 3 sec, hold time 10 sec
    Next hello sent in 0.908 secs
  Redirect time 600 sec, forwarder time-out 14400 sec
  Preemption disabled
  Active is 192.168.0.3, priority 110 (expires in 7.760 sec)
  Standby is local
  Priority 100 (default)
  Weighting 100 (default 100), thresholds: lower 1, upper 100
  Load balancing: round-robin
  There are 2 forwarders (1 active)
  Forwarder 1
    State is Listen
    MAC address is 0007.b400.0101 (learnt)
    Owner ID is 0002.fd79.fba0
    Time to live: 14397.756 sec (maximum 14400 sec)
    Preemption enabled, min delay 30 sec
    Active is 192.168.0.3 (primary), weighting 100 (expires in 7.752 sec)
  Forwarder 2
    State is Active
      1 state change, last state change 00:00:13
    MAC address is 0007.b400.0102 (default)
    Owner ID is 0002.b90d.5960
    Preemption enabled, min delay 30 sec
    Active is local, weighting 100
```

Wireless Standards

The IEEE defines all 802.11 wireless data standards. All current wireless standards are based off of the original 802.11 standard also known as 802.11 Legacy standard. All of the 802.11 standards run within the 2.4 GHz or 5 GHz unlicensed ISM frequencies. Because the frequencies are unlicensed, they must be designed to tolerate interference.

Data, Throughput and Coverage of 802.11 Standards

- 802.11b: This is the first major revision of the 802.11 standard in 1999. It runs within the 2.4 GHz frequency and has a maximum data rate of 11 Mbps. It has a maximum range of 300 feet.
- 802.11g: This standard was released in 2003. It runs in the 2.4 GHz frequency. It has a maximum data rate of 54 Mbps and a range of 300 feet.
- 802.11a: This standard operates within the 5 GHz range. It has a maximum data rate of 54 Mbps and a range of 300 feet although the signal strength weakens quickly therefore it is said to have a range slightly less than the 802.11b and 802.11g standards. This standard also has 8 channels that it uses within the 5 GHz range as opposed to 14 (14 in Japan) channels that the standards in the 2.4 GHz range use.
- 802.11i: This is a wireless amendment to the 802.11 Legacy standard that includes security features for wireless. Due to the security problems found with the original security specification: Wired Equivalent Privacy (WEP), 802.11i was put in place to offer a better method to secure wireless data. 802.11i is also referred to as Wired Protected Access 2 (WPA2).
- 802.11n: This standard is still in the draft stages and therefore is not yet an official amendment. However, the draft is currently in revision 2 and it is likely that very little will change within this draft before it becomes official. There are already Cisco products available that use the 802.11n draft 2 specifications. This amendment draft runs either in the 2.4 GHz or 5 GHz frequencies. It has a maximum data rate of 300 Mbps and a range of up to 600 feet. The biggest difference between 802.11n and the other 802.11 amendments is the fact that 802.11n utilizes Multiple-input-Multiple-output (MIMO) technology. This technology uses multiple transmit and receive antennas to improve overall wireless performance.

Components and Operations of WLAN Topologies

- <u>Cisco Aironet Wireless Access Point</u>: This is a wireless LAN device that transmits and receives wireless signals from wireless clients and places them on the Local Ethernet LAN for transmission. It acts as the connection point between the wired and wireless networks. *Figure 15* shows a typical WAP setup on a LAN:

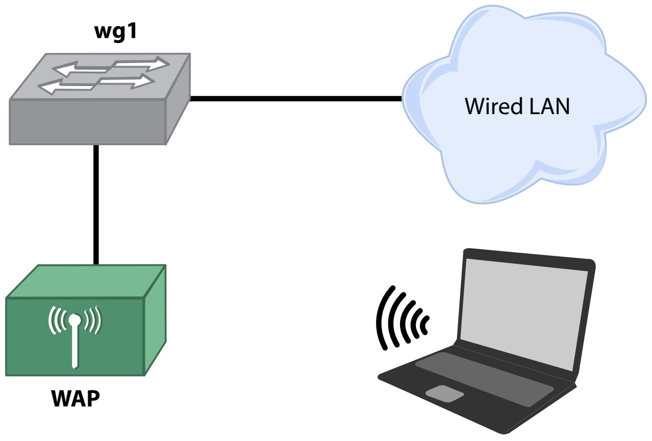


Figure 15

- <u>Cisco Aironet Wireless Workgroup Bridge:</u> The workgroup bridge provides a stand-alone wired Ethernet segment that is connected back to the main Ethernet LAN over a wireless connection. This option is best used when you have several wired Ethernet devices in a location that is difficult to pull physical cabling. *Figure 16* details a workgroup bridge design:

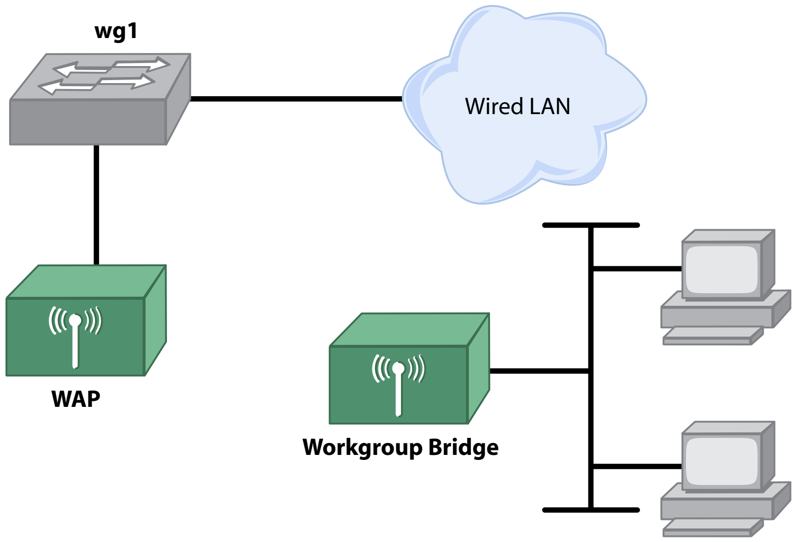


Figure 16

- Wireless LAN Controller (WLC): This device helps to simplify the management of a large wireless network infrastructure. Using a Cisco Wireless LAN controller, you can use special access points called Lightweight Access Points (LWAP) that are all controlled through the WLC. So instead of having hundreds or even thousands of autonomous Access Points (aAP) to individually maintain, all the intelligence and configuration of the LWAP is done at the WLC. The WLC allows for ease of wireless mobility between access points.This appliance is part of the Cisco Unified Wireless Network.

- Wireless Control System (WCS): The WCS is where network engineers monitor, support and maintain their LWAP network. A configuration change can be made one time on the WCS and pushed out to some or all of the wireless access points on the network. This appliance is part of the Cisco Unified Wireless Network.
- Wireless Location Appliance (WLA):This appliance works in conjunction with the WLC and LWAP devices to physically track the location of wireless devices in a large wireless network. It has the ability to locate the wireless devices within a few meters. This appliance keeps track of location information over time so it can be used for trending.This appliance is part of the Cisco Unified Wireless Network.
- Wireless LAN Client Adapters:The wireless LAN adapters are for end devices that do not have built-in wireless capabilities required to join a wireless network. These wireless cards come in various PCI and CardBus forms.

Cisco Recommended Designs

Cisco wireless AP's can be configured to belong in basically one of two major designs. The first design is the autonomous access point (aAP) design. The other design is called the Cisco Unified Wireless Network.

- aAP (autonomous) Access Point:The autonomous AP design is when a full-blown wireless IOS image is placed on every access point. Each device is managed separately from a configuration and wireless intelligence point of view. This type of network is perfect for wireless hot-spots or small wireless mesh designs. It does not scale well from a support standpoint however.
- Cisco Unified Wireless Network:The Cisco unified network is for large wireless deployments in an enterprise infrastructure. The network uses a lightweight version of the wireless IOS on each access point. All the LWAPs are maintained by the wireless LAN controller (WLC) and wireless control system (WCS). The wireless location appliance (WLA) is optional but can add the very useful wireless tracking features to the network. *Figure 17* depicts a typical Cisco unified wireless network design.

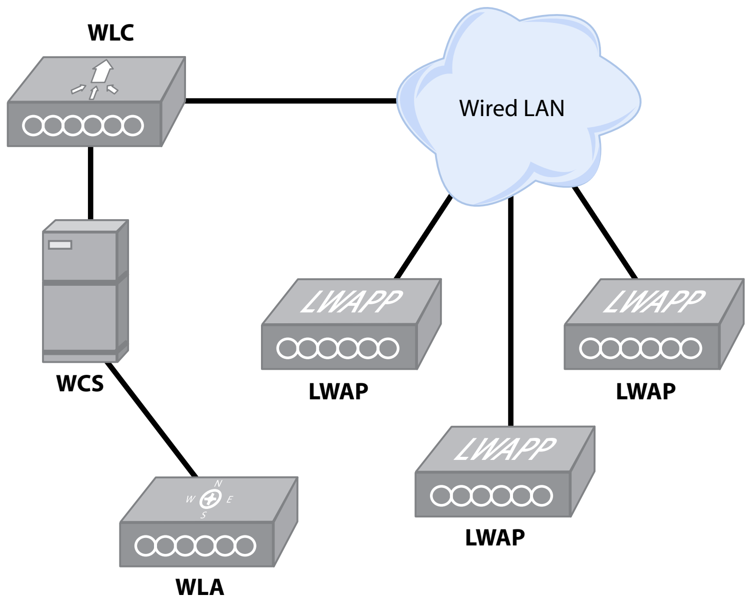


Figure 17

Wireless Mobility

Network unification from a wireless standpoint includes security policies, quality of service (QOS) and wireless mobility. The Cisco unified wireless network also incorporates what's known as wireless mobility. This wireless mobility allows for wireless uses to be completely mobile so they can seamlessly move from location to location within a wireless infrastructure.

- Cisco Fast Secure Roaming (FSR): FSR provides the ability for a wireless client to roam from 1 AP to another without the need to re-authenticate back to a centralized authentication server. Now applications that require uninterrupted network service can be offered over a wireless network. These applications include wireless voice and streaming applications.

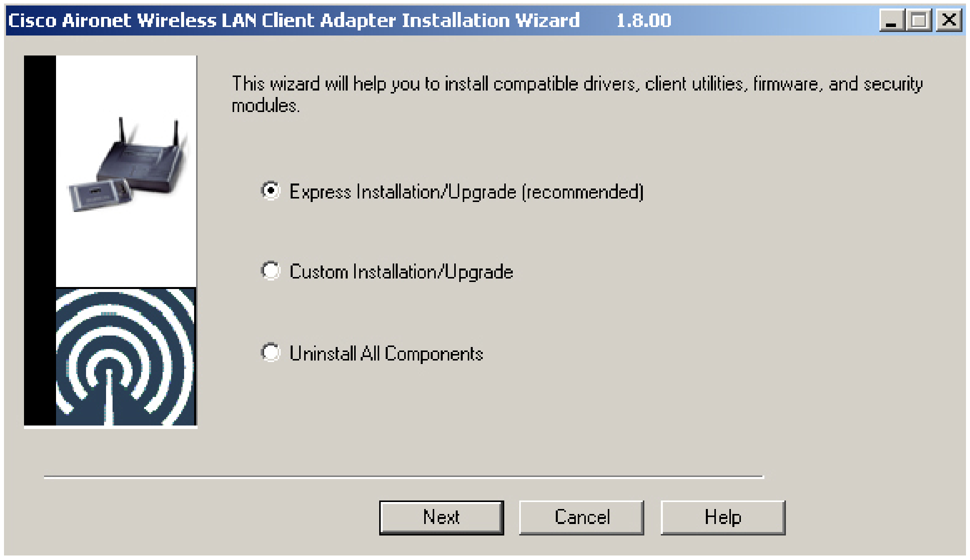
- Inter-Subnet Roaming: Cisco wireless mobility allows a wireless client to keep it's wireless session up while moving from one IP subnet to another. The WLC creates an IP subnet "anchor" for each wireless client that comes on the unified wireless network. When a wireless client roams to a foreign subnet, the traffic is forwarded to the anchor subnet and moved across the LAN. This process is completely hidden from the end user and is highly efficient from a latency perspective.

Configuring a Wireless Client

The setup and configuration of the Cisco Aironet wireless LAN client adapter is shown below. This will guide you through the actual configuration for a wireless client that will connect to an AP with the following settings:

Encryption: 128 bit WEP key
Authentication: Open
SSID: BCMSN

- Step 1: Use the Installation wizard to perform The Express Installation/Upgrade

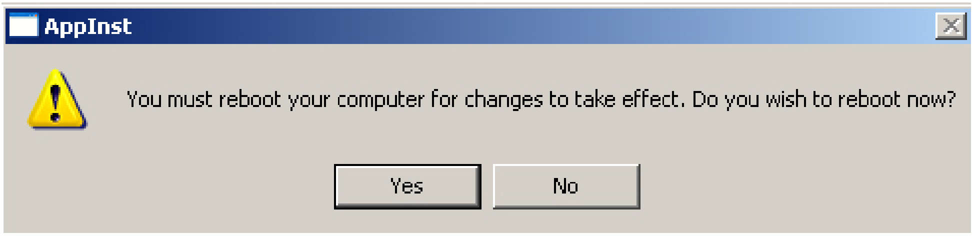


Cisco Aironet Wireless LAN Client Adapter Installation Wizard 1.8.00

This wizard will help you to install compatible drivers, client utilities, firmware, and security modules.

(●) Express Installation/Upgrade (recommended)
( ) Custom Installation/Upgrade
( ) Uninstall All Components

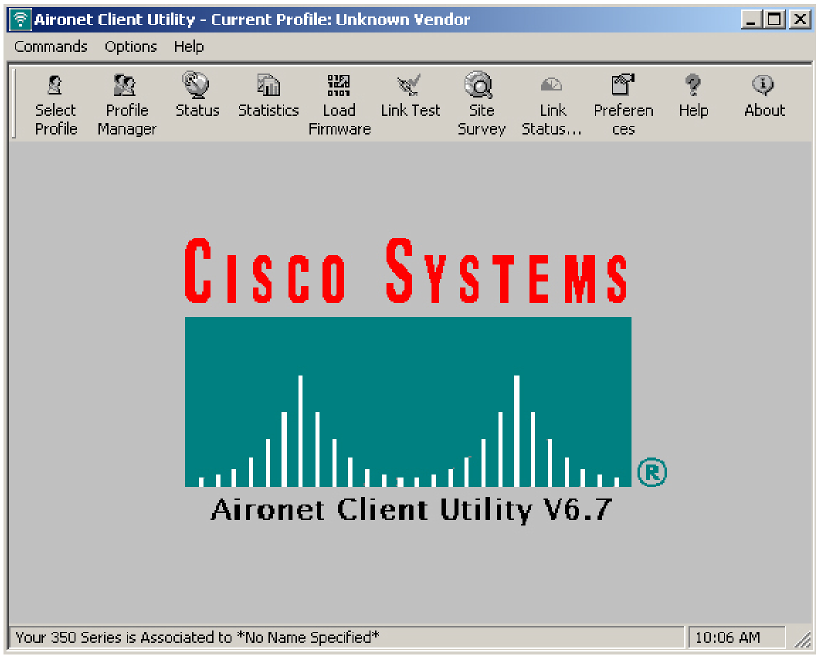
- Step 2: Once the drivers have been successfully installed, reboot the PC.



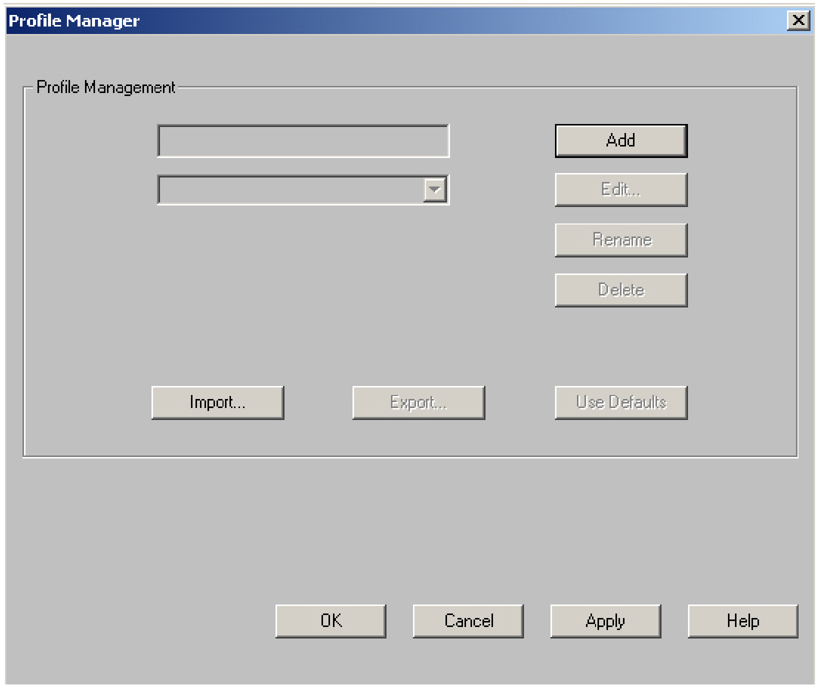
AppInst

You must reboot your computer for changes to take effect. Do you wish to reboot now?

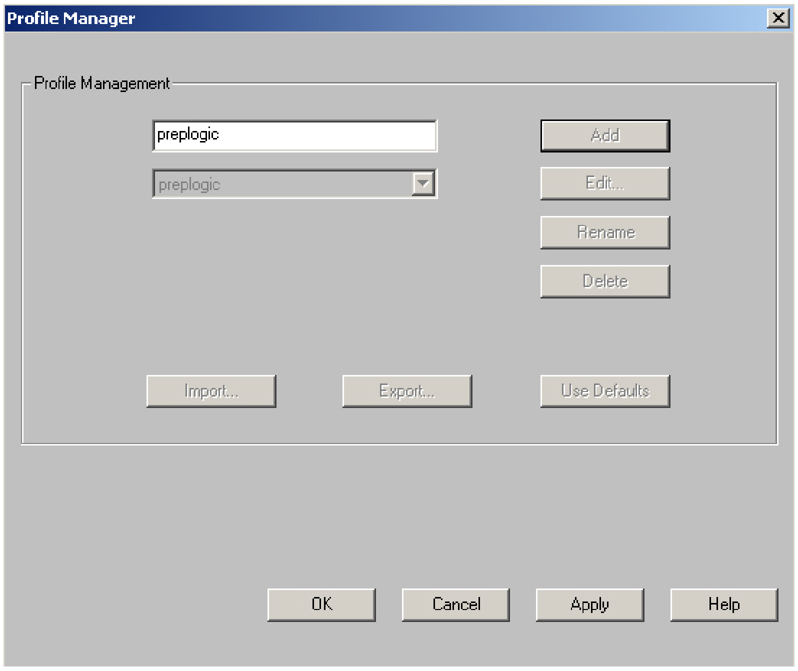
- Step 3: After the PC has rebooted, launch the Aironet Client Utility (ACU). Click on the Profile manager button.



- Step 4: Click the Add button to make a new Profile.



- Step 5: Enter a profile name. In this case we used "preplogic" as our profile name. Click the OK button.



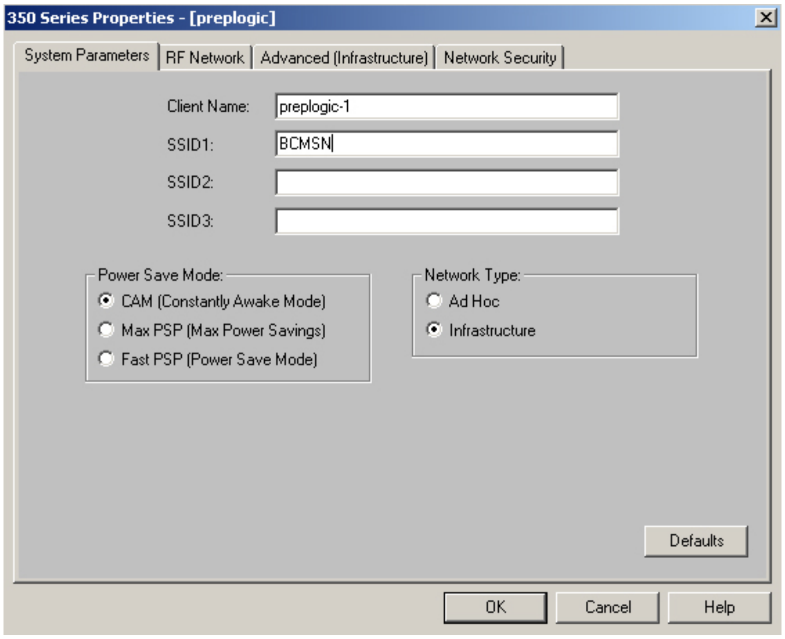
Profile Manager

Profile Management

preplogic

preplogic

- Step 6: Enter the Client Name of the PC you are doing the configuration on. We used preplogic-1 as our name. Enter the SSID. The AP is configured with an SSID of BCMSN in this example.



350 Series Properties - [preplogic]

Client Name: preplogic-1

SSID1: BCMSN

SSID2:

SSID3:

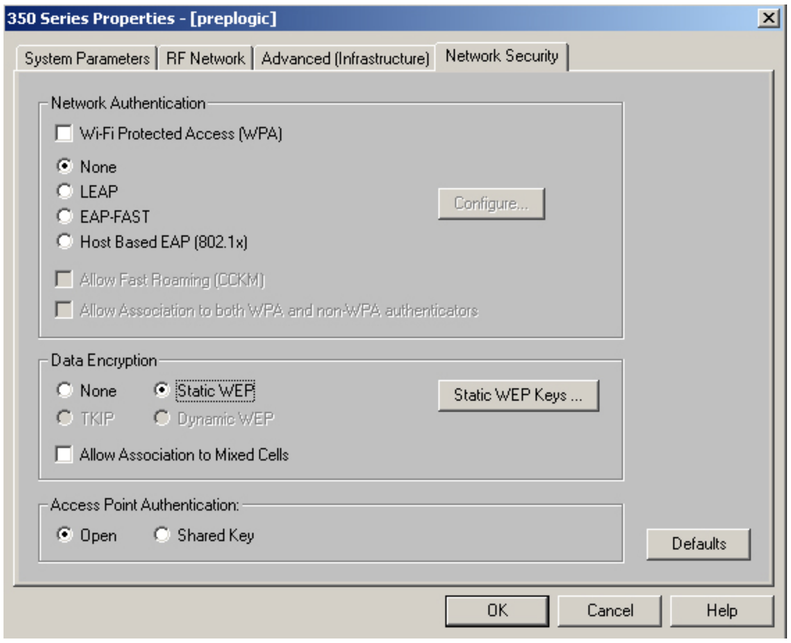
Power Save Mode:

- [✓ CAM (Constantly Awake Mode)]
- Max PSP (Max Power Savings)
- Fast PSP (Power Save Mode)

Network Type:

- Ad Hoc
- [✓ Infrastructure]

- Step 7: Click on the Network Security tab at the top. Select the Static WEP radio button and click the button that says Static WEP Keys.



350 Series Properties - [preplogic]

Network Authentication
- [ ] Wi-Fi Protected Access (WPA)
- [✓ None] LEAP EAP-FAST Host Based EAP (802.1x)
- [ ] Allow Fast Roaming (CCKM)
- [ ] Allow Association to both WPA and non-WPA authenticators

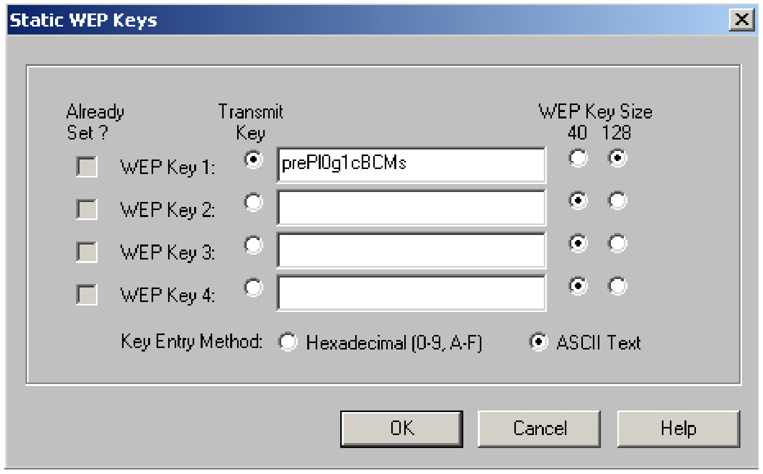
Data Encryption
- None [✓ Static WEP] TKIP Dynamic WEP
- [ ] Allow Association to Mixed Cells

Access Point Authentication:
- [✓ Open] Shared Key

- Step 8: Enter the static WEP key. You can enter the keys with either 40 or 128 bit key sizes. You can also enter the key using ASCII or Hexadecimal. Depending on the method used, the actual number of characters used to enter the WEP key varies:

  - 128-bit ASCII – 13 Characters
  - 40-bit ASCII – 5 Characters
  - 128-bit Hex – 26 Characters
  - 40-bit Hex – 10 Characters

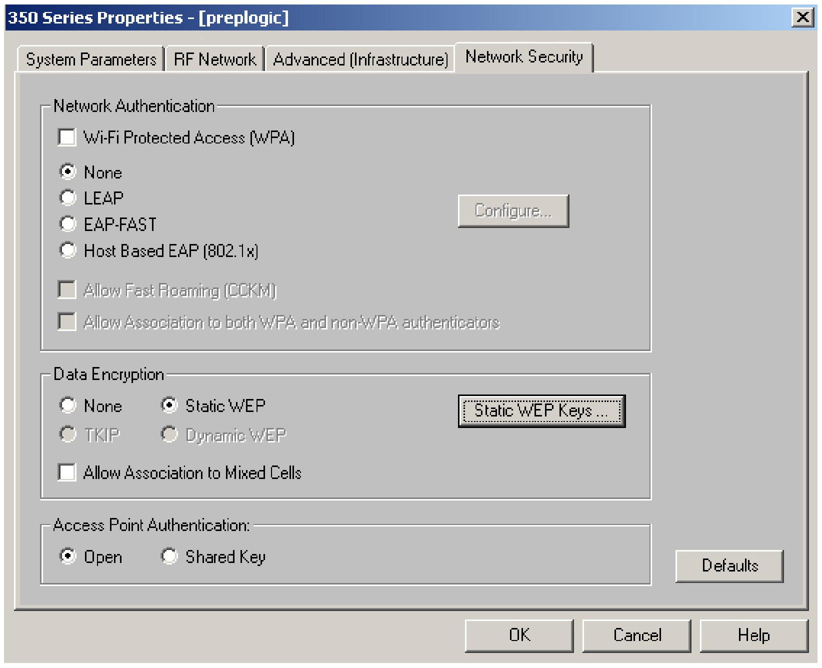
The AP is set with a 128-bit key. We'll enter the key using ASCII which is "prePl0g1cBCMs". Note: this key is exactly 13 characters in length. Click OK.


Static WEP Keys

Already Set ?		Transmit Key		WEP Key Size 40	128
☐	WEP Key 1:	●	prePl0g1cBCMs	○	●
☐	WEP Key 2:	○		●	○
☐	WEP Key 3:	○		●	○
☐	WEP Key 4:	○		●	○

Key Entry Method: Hexadecimal (0-9, A-F) [✓ ASCII Text]

- Step 9: Click OK once we're done setting up the new profile.

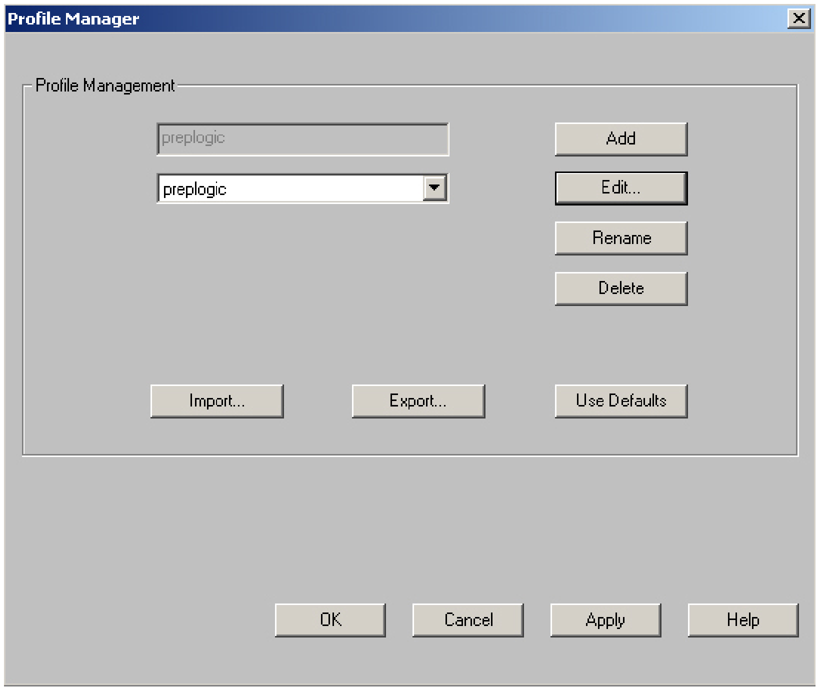

350 Series Properties - [preplogic]

Network Authentication
- ☐ Wi-Fi Protected Access (WPA)
- [✓ None] LEAP EAP-FAST Host Based EAP (802.1x)
- ☐ Allow Fast Roaming (CCKM)
- ☐ Allow Association to both WPA and non-WPA authenticators

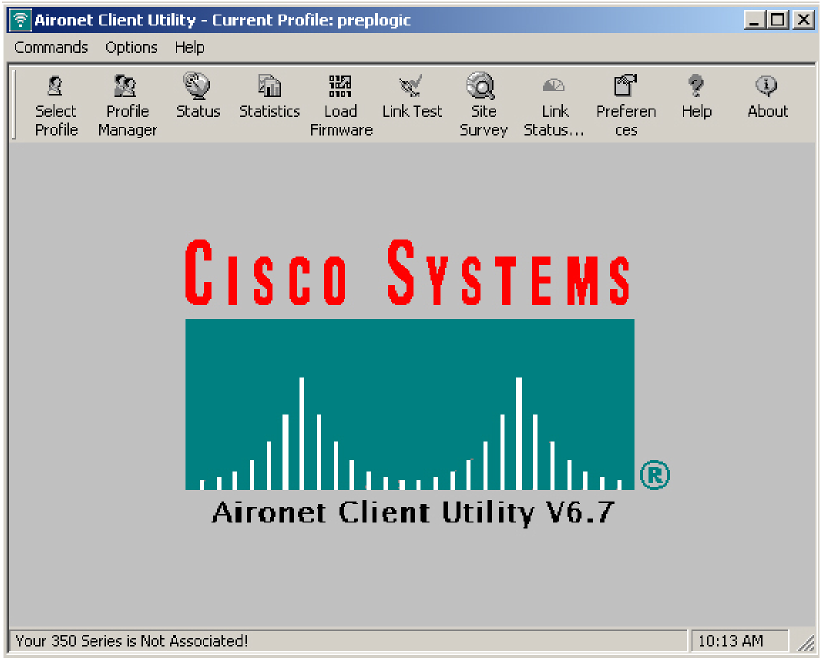
Data Encryption
- None [✓ Static WEP] TKIP Dynamic WEP
- ☐ Allow Association to Mixed Cells

Access Point Authentication:
- [✓ Open] Shared Key

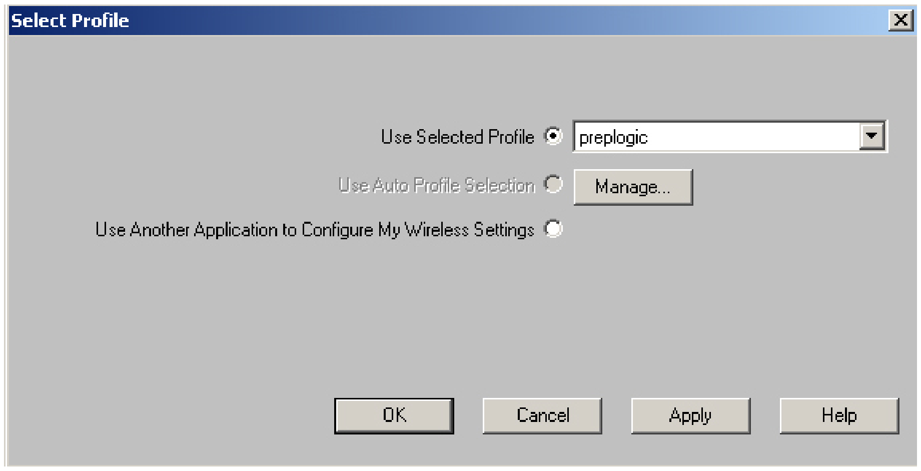
- Step 10: Click OK to save the new profile named "preplogic".



- Step 11: Click the Button named Select Profile.



Select the profile named "preplogic" from the dropdown list. Click the OK button.


Select Profile
Use Selected Profile
preplogic
Use Auto Profile Selection
Manage...
Use Another Application to Configure My Wireless Settings
OK
Cancel
Apply
Help

Your new profile is now complete. You should be able to connect to the wireless AP.

Security Features in a Switched Network

Switch security is an important part of the BCMSN exam. This section will detail some of the typical security problems and how to setup your equipment to prevent attacks from occurring.

MAC Address Flooding

The content addressable memory (CAM) table holds the port to MAC address mappings on a switch. Like all memory, it is finite in nature and can only hold a certain number of mappings before it fills up completely. If a CAM table becomes full, the switch basically ignores the CAM table and forwards traffic out all ports on the same VLAN. It basically turns the intelligent switch into a bridge or hub. An attacker can then put a sniffer on the switch and see all traffic on that VLAN. There are two recommended methods that can protect a network from the MAC address flooding attack:

Port Security

This interface command feature allows the engineer to limit the number of MAC addresses it allows on a particular port. It is a good idea to limit the number of MAC addresses on your access ports. Not only does it protect against hackers attempting to flood the CAM table but it also guards against users from adding an unauthorized switch or wireless access point.

Port security can be configured to allow any number of MAC addresses on a port. It can also be setup so specific MAC addresses are manually entered into the switch (**switchport port-security mac-address** command) or they can be dynamically learned. For example, A network engineer can set the port security MAC address limit to 1 (using the **switchport port-security maximum** command) and hard code the specific MAC addresses of the device that will connect to that particular switch port. Alternatively, he/she can set the port security to dynamically accept the first MAC address that is seen on the port. If a port security violation occurs (either too many MAC addresses seen or the wrong address) the switch port can be set to either shutdown permanently (err-disable), shutdown for a specific period of time, or drops all inbound packets from the insecure device. This is done using the **switchport port-security violation** command. The default violation is to shutdown the port.

Syntax:

```
Switch#configure terminal
Switch(config)# interface interface_id
Switch(config-if)#switchport mode access
Switch(config-if)#switchport port-security
Switch(config-if)#switchport port-security maximum value
Switch(config-if)#switchport port-security violation {protect | restrict | shutdown}
Switch(config-if)#switchport port-security mac-address mac_address
```

This example will configure port security on port fa0/10 and hard code a MAC address to the port. This is the only address that can use swithport 0/10. All other devices or additional devices will shut the port down by default.

Example:

```
wg1#configure terminal
wg1(config)# interface fa0/10
wg1(config-if)# switchport mode access
wg1(config-if)# switchport port-security
wg1(config-if)# switchport port-security maximum 1
wg1(config-if)# switchport port-security mac-address 1234.5678.9012
wg1(config-if)# end
```

We can now verify our configuration:

```
wg1#sh port-security address
```

```
    Secure Mac Address Table
-------------------------------------------------------------
Vlan   Mac Address      Type                 Ports
----   -----------      ----                 -----
1      1234.5678.9012   SecureConfigured     Fa0/10
```

802.1x Authentication

The 802.1x standard uses Extensible Authentication Protocol (EAP) to authenticate end users prior to giving them access to the network. Authentication happens through a RADIUS server such as the Cisco ACS server. Once authenticated, the user has access to the network. If authentication fails, the end device cannot connect to the switch.

Address Resolution Protocol (ARP) Spoofing

ARP spoofing is when a device sends an ARP broadcast message to the entire VLAN announcing its IP address. Hackers can attempt to send out an ARP broadcasting that it is the default gateway IP address. This can cause other devices in the VLAN to send all of their gateway bound traffic to the attacking device instead of the real default gateway. This is referred to as a man in the middle attack. To prevent this, an engineer can use port security and 802.1x features to prevent the attacker from ever getting on the network. The engineer can also use private VLANs and ARP entry rules to protect against the spoof attack. We'll focus on using static ARP entries, ARP inspection and PVLANs.

Static ARP Entries

Static ARP entries are just like the name says. Instead of dynamically learning the MAC address to IP address entry, you can statically assign the MAC to Static address. This static ARP entry takes precedence over any dynamically learned entry. Note that this security method greatly increases the switch management overhead required. The global command used to create static ARP entries on an MLS switch is:

Syntax:

```
Switch#configure terminal
Switch(config)#arp {ip-address | vrf vrf-name} hardware-address encap-type [interface-type]
```

Let's configure an MLS switch to have a static mapping on IP 192.168.1.1 and a MAC address of 1234.5678.9012:

Example:

```
MLS-1(config)# arp 192.168.1.1 1234.5678.9012 arpa
```

Now we can verify our configuration by doing a **show ip arp**:

```
MLS-1#show ip arp
Protocol  Address      Age (min)   Hardware Addr    Type    Interface
Internet  192.168.1.1      -       1234.5678.9012   ARPA
```

Dynamic ARP Inspection (DAI)

This feature uses VLAN access control lists (ACLs) to filter ARP traffic on a particular VLAN. ARP inspection can be configured to allow ARP traffic to a specific MAC address only and deny all other traffic. A hacker would not be able to become the man in the middle using this security feature. To configure DAI, you need to first make an ARP ACL mapping the IP address to a MAC address. You can then enable ARP inspection on a single VLAN or multiple VLANs as shown in the syntax section.

Syntax:

```
Switch#configure terminal
Switch(config)#ip arp inspection filter arp_acl_name vlan {vlan_ID | vlan_range} [static]
Switch(config)#ip arp inspection vlan {vlan_ID | vlan_range}
```

Below is an example where we apply an ARP ACL called "preplogic" to VLAN 10:

Example:

```
MLS-1# configure terminal
MLS-1(config)# ip arp inspection filter preplogic vlan 10
MLS-1(config)# ip arp inspection vlan 10
```

Private VLANs (PVLAN)

PVLANs have the ability to isolate layer 2 traffic in the same VLAN. Each port within a PVLAN can be configured as promiscuous, isolated or community.

- Promiscuous Port: The end device can talk to any isolated, community or any device outside of the VLAN.
- Isolated Port: The device can only talk to hosts outside the VLAN and to any ports configured as promiscuous in the same PVLAN.
- Community Port: The device can talk to other devices in the same community VLAN as well as hosts outside the VLAN and any promiscuous ports.

To go along with the three types of PVLAN ports, there are three types of PVLANs:

- Isolated VLAN: Transports traffic from isolated to promiscuous ports only.
- Community VLAN: Transports traffic from community/promiscuous ports.
- Primary VLAN: Transports traffic to all PVLAN ports.

Now that we know what PVLANs are, here is how to configure them.

Note: The VTP mode of the switch must be Transparent in order to configure PVANs.

Syntax:

```
Switch#configure terminal
Switch(config)# interface vlan vlan_ID
Switch(config-vlan)# private-vlan {community | isolated | primary}
Switch(config)# interface interface-type
Switch(config-if)# switchport mode private-vlan {host | promiscuous}
```

VLAN Hopping

VLAN hopping is a method attackers can use on 802.1q trunked networks to access VLANs other than the one they are directly connected to. By default, an 802.1q trunk has a native VLAN and VTP on VLAN 1. All access ports are also configured by default to be in VLAN 1. Hackers can use an end device to send 802.1q frames to the switch that are encapsulated with the VLAN they want to reach. They then "double" encapsulate the frame by adding a VLAN 1 tag in front of the VLAN 10 tag. So when the attacker plugs into a port that is on VLAN 1 and successfully sets up a trunk link using DTP, they send this double encapsulated frame to the switch. The switch strips off the VLAN 1 tag and what's left is a frame that is destined for VLAN 10. This frame is sent to the L3 interface and routed accordingly. The hacker has successfully made it look like his end port is on VLAN 10 when it is actually on VLAN 1.

Two simple steps can fix this VLAN hopping problem. The first is to insure that the Native VLAN is not sent across the trunk port:

Syntax:

```
Switch#configure terminal
Switch(config)# interface interface-id
Switch(config-if)#switchport trunk allowed vlan vlan-id
```

So let's say that we have a trunk port (gi0/1) with a native VLAN of 1 and we have our end users on VLANs 100 and 200. To prevent against VLAN hopping using the native VLAN, we would only allow VLAN 100 and 200 across the trunk. Use *Figure 18* as the physical layout:

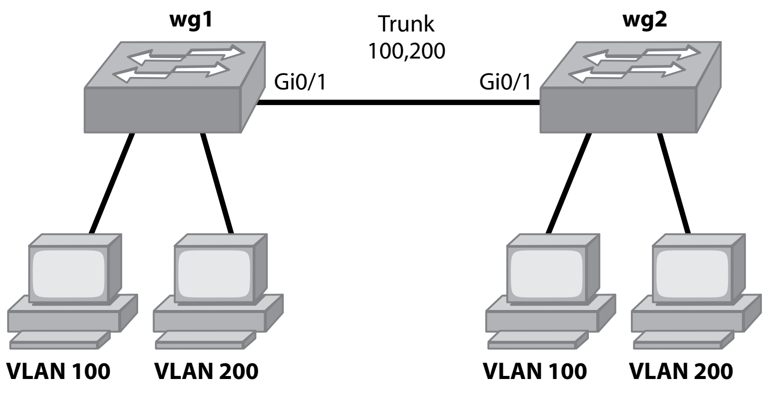


Figure 18

Example:

Switch wg1:

```
wg1#configure terminal
wg1(config)#interface gi0/1
wg1(config-if)#switchport trunk allowed vlan 100, 200
```

Switch wg2:

```
wg2#configure terminal
wg2(config)#interface gi0/1
wg2(config-if)#switchport trunk allowed vlan 100, 200
```

We can now verify that our trunk is only sending VLAN 100 and 200 across the trunk using the **show interfaces trunk command**:

```
wg1#show interfaces trunk

Port     Mode     Encapsulation  Status     Native vlan
Fa0/1     on          802.1q     trunking    1

Port    Vlans allowed on trunk
Fa0/1     100, 200

Port     Vlans allowed and active in management domain
Fa0/1     100,200

Port     Vlans in spanning tree forwarding state and not pruned
Fa0/1     100,200
```

The other step that can be done to help prevent against VLAN hopping is to insure that your user device ports can never become trunk ports dynamically. To accomplish this, you need to configure each port as an access port:

Syntax:

```
Switch#configure terminal
Switch(config)# interface interface-id
Switch(config-if)#switchport mode access
```

Configure Support for Voice

Characteristics of Voice in the Campus Network

Internet Packet Telephony (IPT) is the convergence of voice and data onto a single network. Traditionally, data transmissions across the network could be sent on fairly unreliable networks in terms of bandwidth and latency. The TCP stack helps recover any dropped or malformed packets simply by retransmitting them. Also, the packet sizes were increased as well to pack in more information to make the data stream more efficient by having less overhead.

When we discuss voice traffic on the network, there are several differences when compared to data. First of all, voice runs over the UDP protocol, which does not retransmit lost packets. This is because voice traffic must get to the end device in exactly the same order that it was sent.

Second, latency plays a much larger factor in the quality of the voice data. The higher the latency, the worse the quality of the voice call will be. When we discuss voice latency, the term jitter is often used. Jitter is the variation in the delay of packets at the destination. When an IPT phone call is established. The user begins speaking into the phone. The information is placed onto the wire in evenly spaced UDP packets. These UDP packets are then sent across the network and to the destination phone. Somewhere along the network from the sending phone to the receiving phone can be network congestion or incorrect queuing mechanisms that can throw the evenly spaced packets off to where they become bunched up or out of sequence. This causes the voice quality to suffer at the end device phone. This is basically what jitter is. For voice calls, the maximum amount of latency between two IPT phones is 200 ms. When voice traffic is put onto the same wire as data, the smaller voice packets are intermixed with the often much larger TCP data packets. This can also cause jitter on an IPT call.

Functions of Voice VLANs and Trust Boundaries

Fortunately, there are mechanisms available to prioritize voice traffic so it is considered a higher priority over other forms of data on the network. Quality of Service (QoS) tools can protect against any jitter or latency problems that network convergence causes. The most popular method of performing QoS on a LAN is to setup voice VLANs to carry all voice traffic. Along with the voice VLAN broadcast separation, a QoS mechanism called class of service (CoS) is used to prioritize voice traffic end to end on the LAN. CoS is an IEEE 802.1P (a subset of the 802.1Q, the telephone uses priority tagging and uses an 802.1Q VLAN ID of 0) specification. It uses classification of traffic to properly schedule and place network traffic on the wire in a very predictable way. If a packet has a CoS tag, it will be placed in a priority queue based on the priority assigned to it. CoS priorities can be assigned priorities of 0 through 7. By default, all non-tagged packets are given a CoS of 0. Voice traffic is typically assigned a CoS of 5. The higher the CoS value, the higher priority the traffic is assumed to be.

Cisco phones such as the 7961 and 7971's have a built-in 3 port switch. One port of this switch connects to the access layer switch. The second port is dedicated to the IPT phone itself. The third port is for user devices such as a desktop PC. You can get a visual image of the Cisco phone design by viewing *Figure 19*.

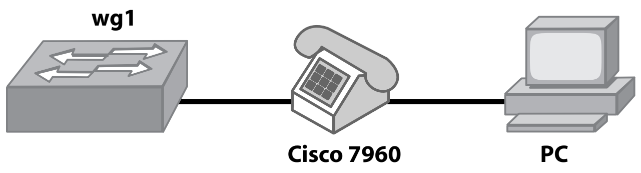


Figure 19

As you can see, the Cisco phone and the PC share the same physical connection back to the switch. When a voice VLAN is setup, the CoS can either be set to be trusted at the phone itself or at the access switch. This is referred to as setting the trust boundary. The trust boundary is set at the access switch. You can configure the switch to either trust the CoS that is sent from the phone or to override the CoS the phone sends and assign your own priority of the traffic.

Voice VLAN Configuration and Verification

You need to turn QoS on globally to be able to configure it at the interface level. Also keep in mind that as soon as the voice VLAN is enabled on a port, PortFast is automatically enabled as well. The two different methods to carry voice traffic are either on standard 802.1q frames or 802.1p frames. To use 802.1q frames use the **switchport voice vlan** *vlan-id* command. To use 802.1p priority-tagged frames use **switchport voice vlan dot1p.**

Syntax:

```
Switch#configure terminal
Switch(config)#mls qos
Switch(config)#interface interface-id
Switch(config-if)#switchport voice vlan {vlan-id | dot1p}
Switch(config-if)#mls qos trust cos
Switch(config-if)#switchport priority extend {trust | cos priority}
```

Let's configure an 802.1p priority-tagged switchport for a Cisco 7961 phone that is plugged into port fa0/1 of switch wg1. The port will carry both voice and data traffic on the phone as the phone has a PC connected to it as shown in *Figure 19*. We'll set the phone port to trust the CoS coming into it. We will force all the data coming from the PC to have a priority of 0.

Example:

```
wg1#configure terminal
wg1(config)#mls qos
wg1(config)#interface fa0/1
wg1(config-if)#switchport voice vlan dot1p
wg1(config-if)#mls qos trust cos
wg1(config-if)#switchport priority extend cos 0
```

Now we can verify our configuration using the **show mls qos** and **show mls qos interface** commands:

```
wg1#show mls qos
QoS is enabled
```

```
wg1#show mls qos interface fastEthernet 0/1
FastEthernet0/1
trust state: trust cos
trust mode: trust cos
COS override: dis
default COS: 0
DSCP Mutation Map: Default DSCP Mutation Map
trust device: none
```

The next example shows how to configure a switch using standard 802.1q tagging on the phone. The voice VLAN is 101. We'll hard-code the COS on the switchport to 5. This means that all traffic that gets sent from both the phone and PC are tagged with a COS of 5.

Example:

```
wg1#configure terminal
wg1(config)#mls qos
wg1(config)#interface fa0/1
wg1(config-if)#switchport voice vlan 101
wg1(config-if)#mls qos cos 5
wg1(config-if)#mls qos cos override
```

Now let's look at the output of the show commands:

```
wg1#show mls qos
QoS is enabled
```

```
wg1#show mls qos interface fastEthernet 0/1
FastEthernet0/1
trust state: not trusted
trust mode: not trusted
COS override: ena
default COS: 5
DSCP Mutation Map: Default DSCP Mutation Map
trust device: none
```