CISCO NETWORKING ACADEMY PROGRAM

CCNP:

Optimizing Converged Networks v5.0

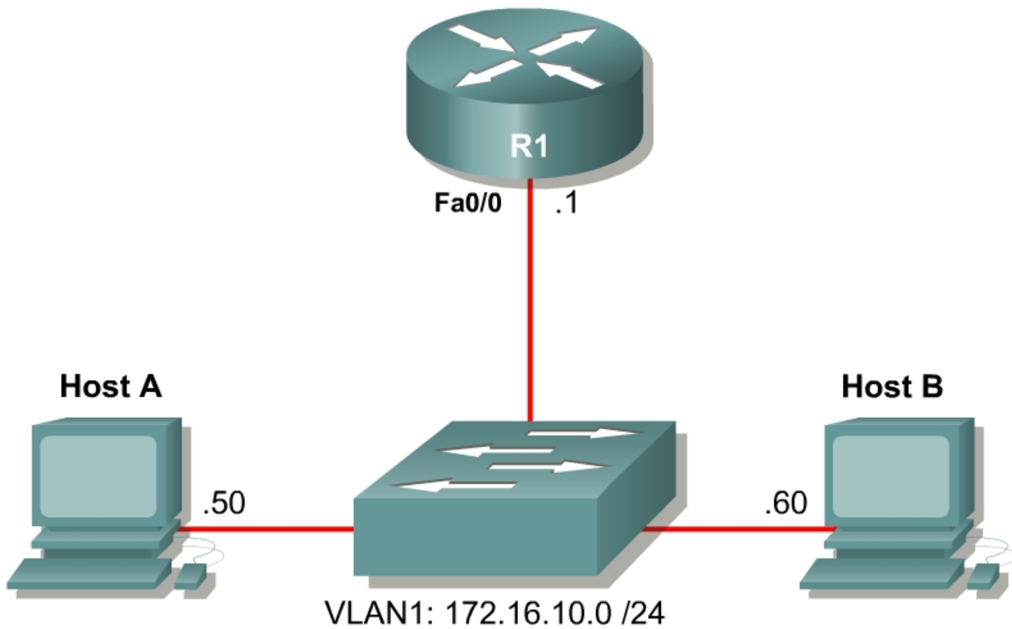
Student Lab Manual

Lab 2.1 Configure CME using the CLI and Cisco IP Communicator

Learning Objectives

- Configure Cisco Unified Call Manager Express (CME)
- Install Cisco IP Communicator (CIPC) on a host
- Verify CME and CIPC Operation

Topology Diagram



Scenario

In this lab, you will configure Cisco Unified Call Manager Express using the IOS command line. On the two hosts, you will install Cisco IP Communicator and have one host call the other. Cisco IP Communicator is a software telephony application to simulate a Cisco IP Phone on the desktop of a PC running Microsoft Windows.

This lab uses Cisco's newest version of Cisco Unified Call Manager Express at the time of this writing (CME 4.0(2)) which was tested using Cisco IOS Release 12.4(9)T1 running on a Cisco 2800 Series router. The IP Voice image is required in order to be able to manipulate codecs.

Step 1: Configure Addressing

Configure the router with the IP address shown in the diagram.


```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
```

Next, assign IP addresses to the hosts. If the hosts already have IP addresses in the same subnet as the router, you may skip this step. These steps may vary depending on your Windows version and theme.

First, open the **Control Panel** on Host A and choose **Network Connections**.

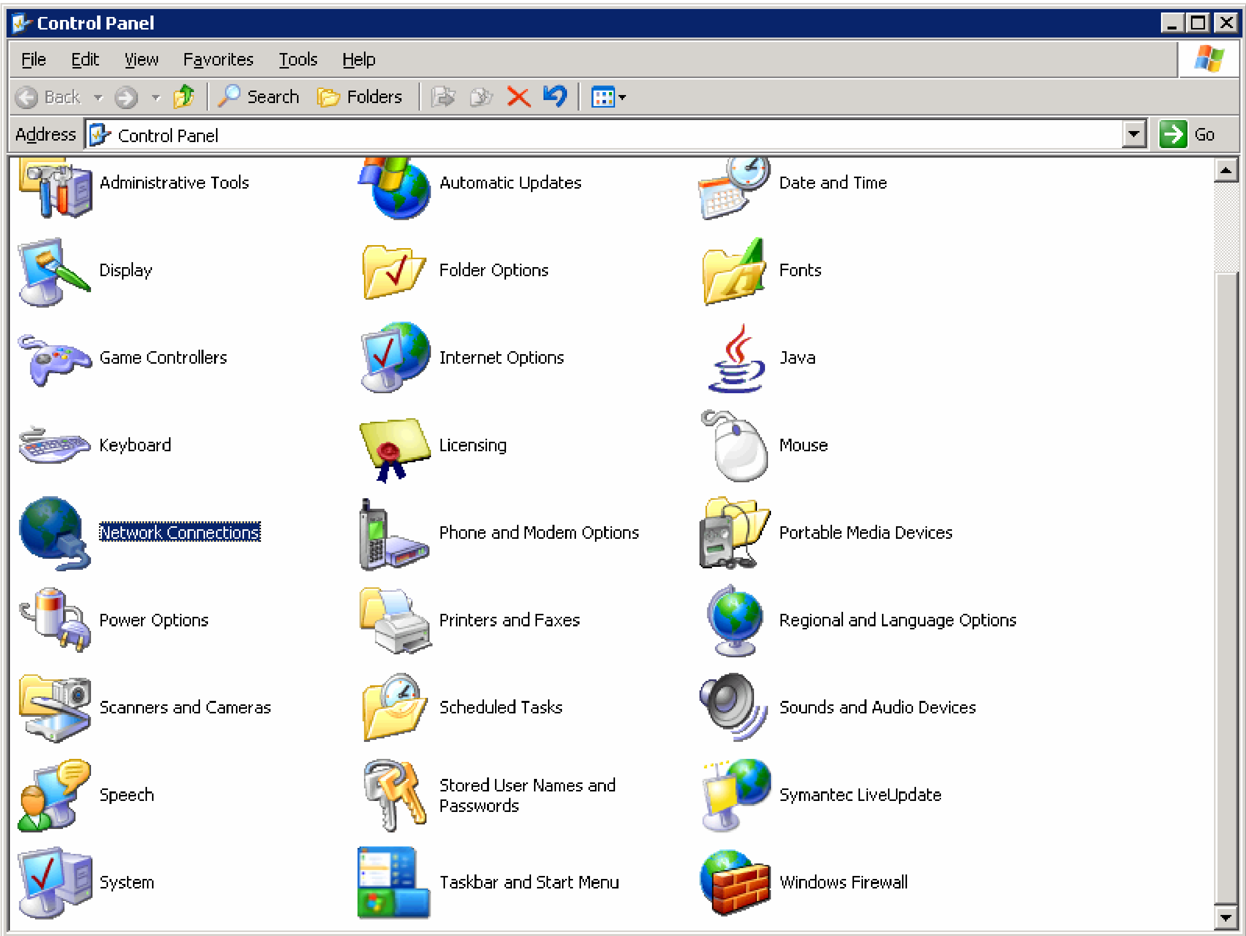


Figure 1-1: Microsoft Windows Control Panel

Next, right-click on the LAN interface that connects to the switch and click **Properties**. In the list of protocols, choose **Internet Protocol (TCP/IP)** and click **Properties**.

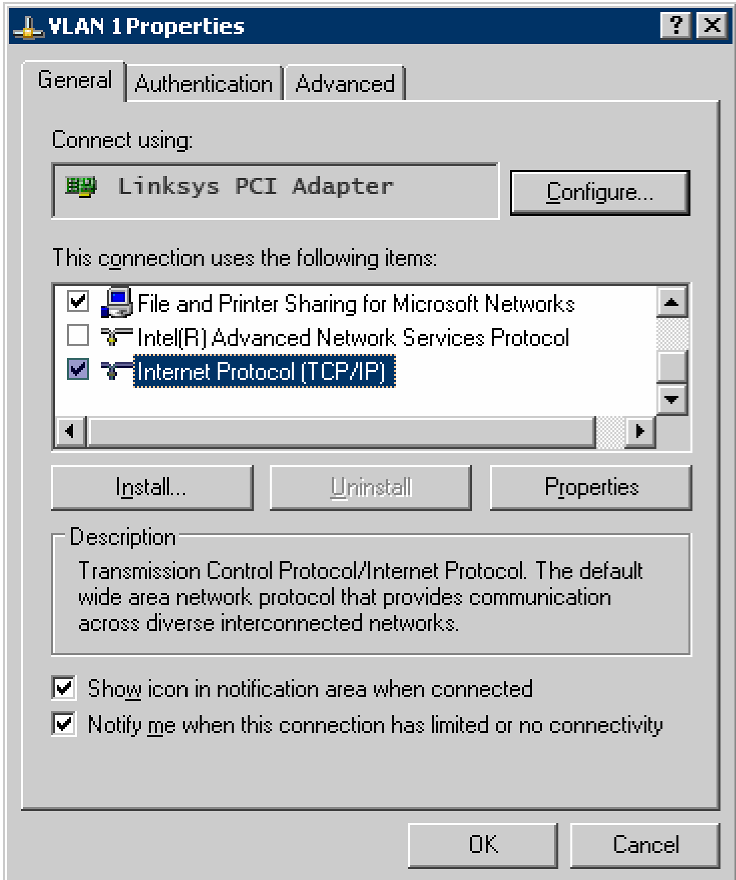


Figure 1-2: LAN Adapter Properties

Finally, configure the IP address 172.16.10.50/24 below on the interface.

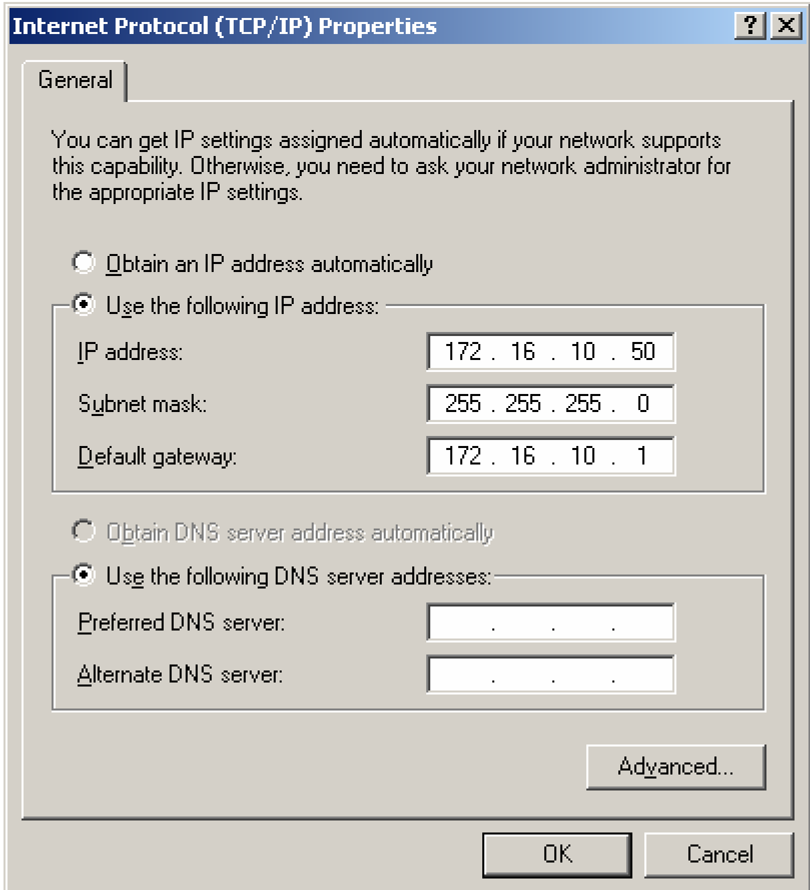


Figure 1-3: TCP/IP Settings for LAN Adapter

Click **OK** once to apply the TCP/IP settings and again to exit the LAN interface properties dialog box.

Configure Host B similarly, using 172.16.10.60/24 as the IP address.

Step 2: Configure Router Telephony Service

Cisco's Call Manager Express (CME) is a slimmed-down version of the Call Manager (CM) server application. CM runs on a dedicated server, while CME runs on a router. CME possesses much of the basic functionality of CM, which may be all that is needed in a smaller network without a large number of phones. CME may also be much more cost-effective in many environments where the full power of CM is not necessary. CM and CME both act as servers whose main function is to establish calls between phones, as well as many other voice-related functions. A Cisco IP phone deployment requires either a deployment of CME or CM to provide telephony services to the IP phones.

Cisco IP phones rely on Call Manager or Call Manager Express primarily during their boot sequence and dialing procedure to provide configuration and directory services.

To enable the CME functionality of a Cisco router running a CME-installed image, use the **telephony-service** command in global configuration mode. This will bring you into the telephony service configuration prompt. If you issue the **?** character at this prompt, you will see that there are many CME-specific commands available to customize a CME installation.

```
R1(config)# telephony-service
R1(config-telephony)# ?
Cisco Unified CallManager Express configuration commands.
For detailed documentation see:
www.cisco.com/univercd/cc/td/doc/product/access/ip_ph/ip_ks/index.htm

  after-hours              define after-hours patterns, date, etc
  application              The selected application
  auto                     Define dn range for auto assignment
  auto-reg-ephone          Enable Ephone Auto-Registration
  bulk-speed-dial          Bulk Speed dial config
  call-forward             Configure parameters for call forwarding
  call-park                Configure parameters for call park
  caller-id                Configure caller id parameters
  calling-number           Replace calling number with local for hairpin
  cnf-file                 Ephone CNF file config options
  ...
```

Since there are two hosts running Cisco IP Communicator, configure the maximum number of phones to be 2 using the **max-ephones** *number* command. Configure the maximum number of directory numbers to be 10 using **max-dn** *number*. Later in the lab exercise, you will demonstrate what the configuration of ephones and directory numbers represent.

```
R1(config-telephony)# max-ephones 2
R1(config-telephony)# max-dn 10
```

Configure the phone keepalive timeout period to be 15 seconds by issuing the **keepalive** *seconds* command. This timer specifies how long CME will wait before considering an IP phone unreachable and taking action to deregister it. The default timeout is 30 seconds.

```
R1(config-telephony)# keepalive 15
```

Configure a system message using the **system message** *line* command. This line will appear on phones associated with the CME.

```
R1(config-telephony)# system message Cisco VOIP
```

Next, tell the router to generate the configuration files for phones that associate with the CME using the **create cnf-files** command. It may take a couple minutes for the configuration process to be enabled.

```
R1(config-telephony)# create cnf-files
```

Finally, configure the source address for SCCP using the **ip source address** *address* **port** *port* command. Use the local Fast Ethernet address with a port number of 2000.


```
R1(config-telephony)# ip source-address 172.16.10.1 port 2000
```

Step 3: Create Directory Numbers

When CME configuration references an "ephone," it is referring to an Ethernet phone connected via an IP network. An ephone represents the physical phone, and can be associated with a phone MAC address and other physical properties. A phone will only have one globally-unique, hard-coded MAC address, so to uniquely identify an ephone on your network, refer to the MAC address.

At the logical layer of the VoIP model, a directory number represents a logical phone with an associated phone number and name (label). A Cisco IP phone can be associated with more than one directory number at a time, effectively making it a multi-line device with each line possessing its own directory number. The soft buttons on an IP phone each represent a single line. To configure a directory number, use the global configuration **ephone-dn** *tag* command. Use a tag of 1 for the first phone.

```
R1(config)# ephone-dn 1
```

At the ephone-dn configuration prompt, use the **number** *number* command to configure a phone number of 5001. Assign a name of "Host A" with the **name** *name* command. This will be the directory number associated with host A's phone, which we will configure shortly.

```
R1(config-ephone-dn)# number 5001
R1(config-ephone-dn)# name Host A
```

Configure ephone-dn 2 similarly.

```
R1(config-ephone-dn)# ephone-dn 2
R1(config-ephone-dn)# number 5002
R1(config-ephone-dn)# name Host B
```

Step 4: Create Phones

Before configuring the phones on the router, you will need to find out the MAC addresses of the hosts. Choose the **Start** > **Run...**, then type in **cmd**. At the command prompt, type the **ipconfig /all** command.

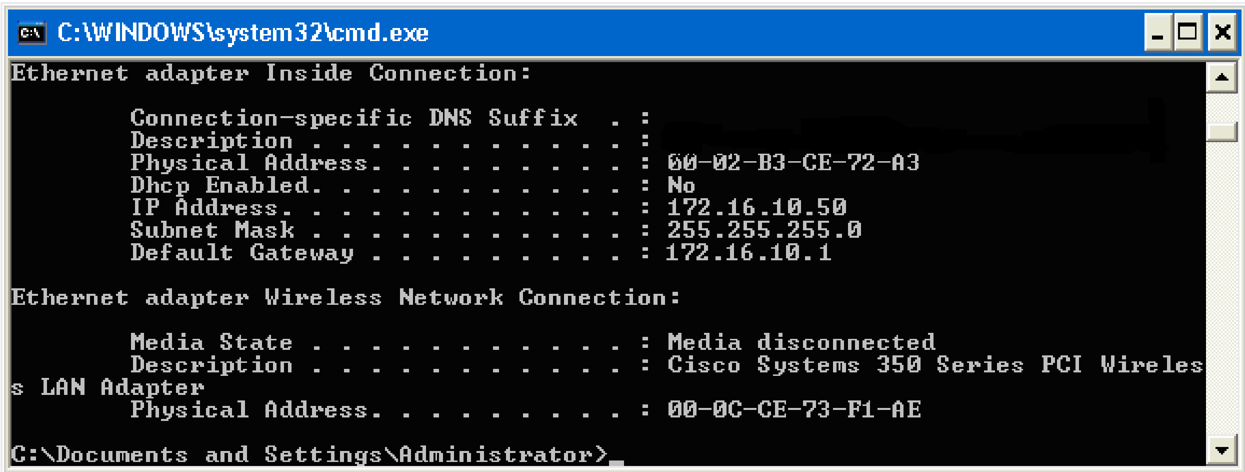


Figure 4-1: IP Configuration on Host A

The hexadecimal string listed as the physical address is the MAC address of the interface. Verify that the interface is the one configured with the correct IP address. Write down the MAC addresses for both hosts, since you will need them in this step.

Note: Your MAC addresses will be different from the addresses shown in the sample commands.

On R1, enter the ephone configuration prompt by typing the **ephone** *tag* command in global configuration mode.

```
R1(config)# ephone 1
```

Associate the MAC address with this ephone using the **mac-address** *address* command. The address must be in the format HHHH.HHHH.HHHH.

```
R1(config-ephone)# mac-address 0002.B3CE.72A3
```

Use the **type** *type* command to configure the type of phone. Since you are configuring Cisco IP Communicator to simulate Ethernet phones, use **cipc** as the phone type.

```
R1(config-ephone)# type cipc
```

Assign the first button on the phone to directory number 1 using the **button** *line* command. The button command assigns buttons to phone lines, as well as determines the type of ringer assigned to that phone line. The format for the button command we will use is “1:1”. The first 1 indicates the first button. The colon indicates a normal ringer. The second 1 represents directory number 1, previously configured with the **ephone-dn 1** command.

```
R1(config-ephone)# button 1:1
```


Apply a similar configuration for ephone 2. Change the configuration parameters where appropriate.

```
R1(config-ephone)# ephone 2
R1(config-ephone)# mac-address 0009.5B1B.67BD
R1(config-ephone)# type cipc
R1(config-ephone)# button 1:2
```

Step 5: Install Cisco IP Communicator

Download Cisco IP Communicator (CIPC) from the Cisco.com website and run the installer using the executable you downloaded. In the version used to write this lab, the name of the installer was CiscoIPCommunicatorSetup.exe, however, the filename of the installer may vary. If you have already installed CIPC, skip this step.

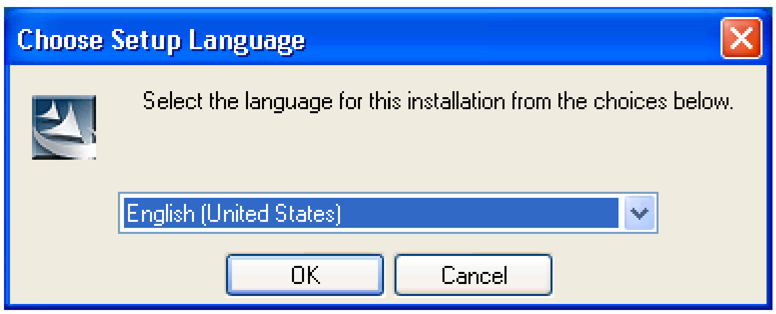


Figure 5-1: CIPC Language for Setup Program

Click **OK** after selecting the installation language of your choice.

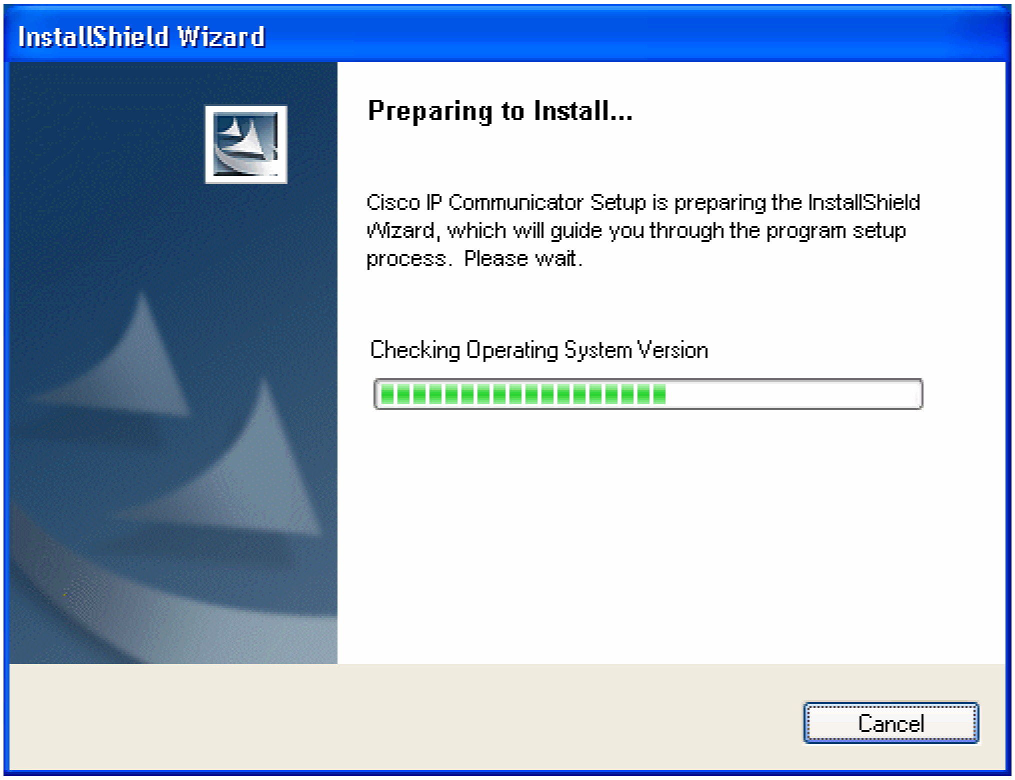


Figure 5-2: InstallShield System Check Progress Indicator

Allow the installer to prepare the InstallShield Wizard.

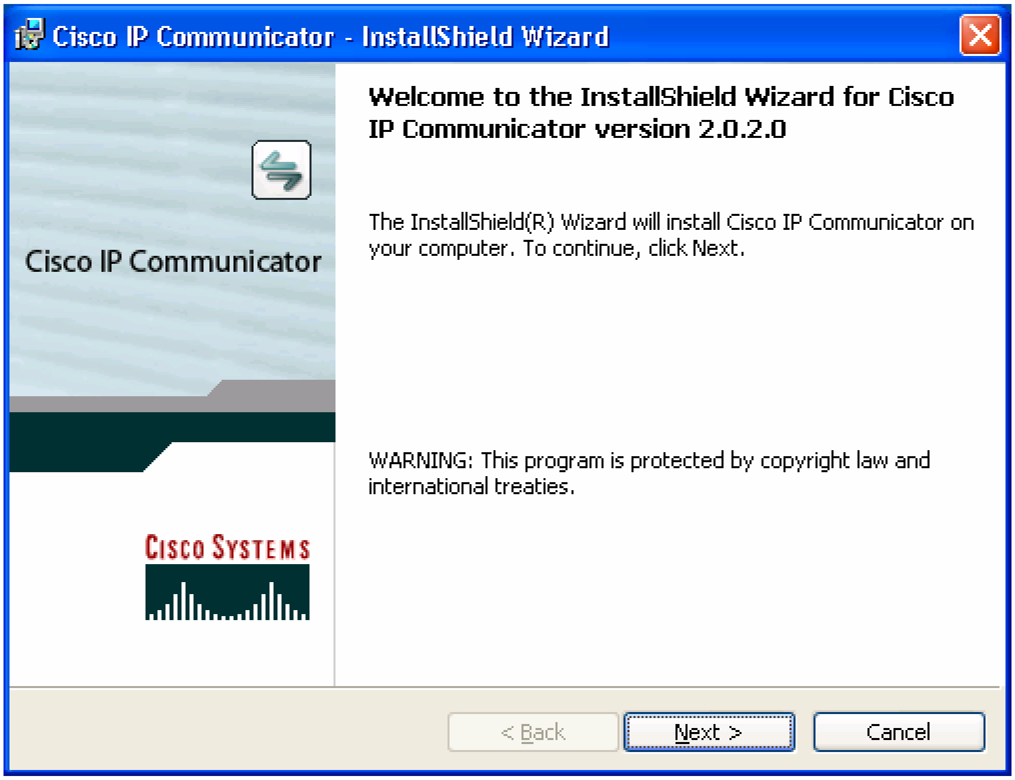


Figure 5-3: CIPC Installer

Click **Next** to continue the installation process.

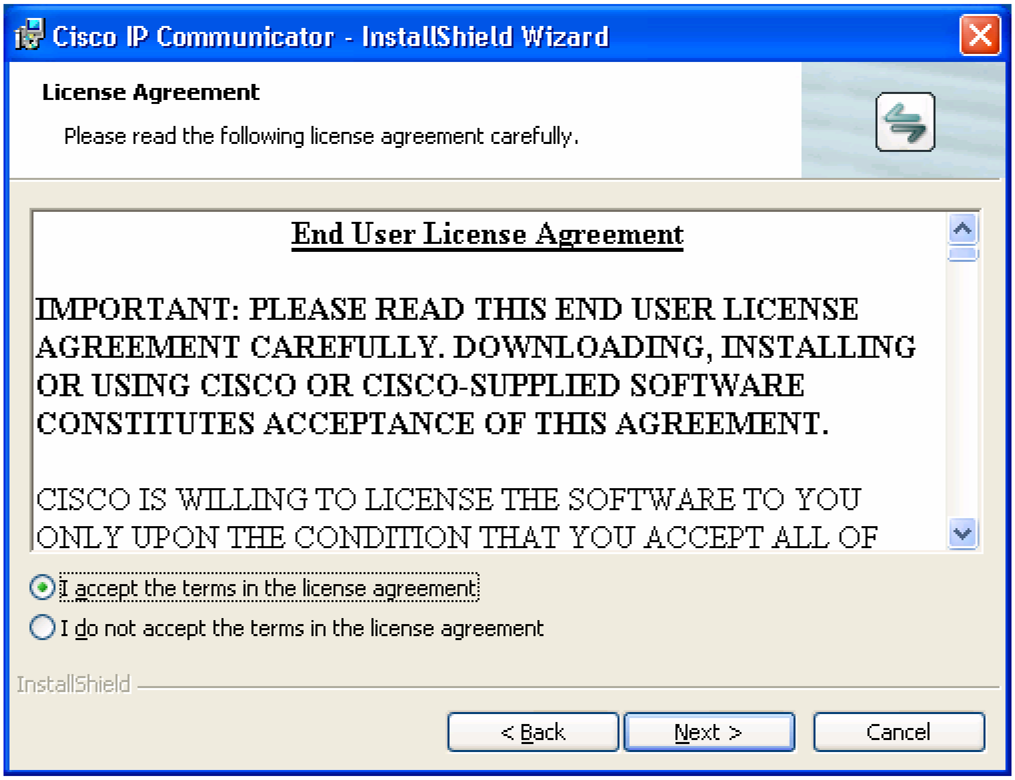


Figure 5-4: CIPC End-User License Agreement

Accept the terms in the license agreement and click **Next**.

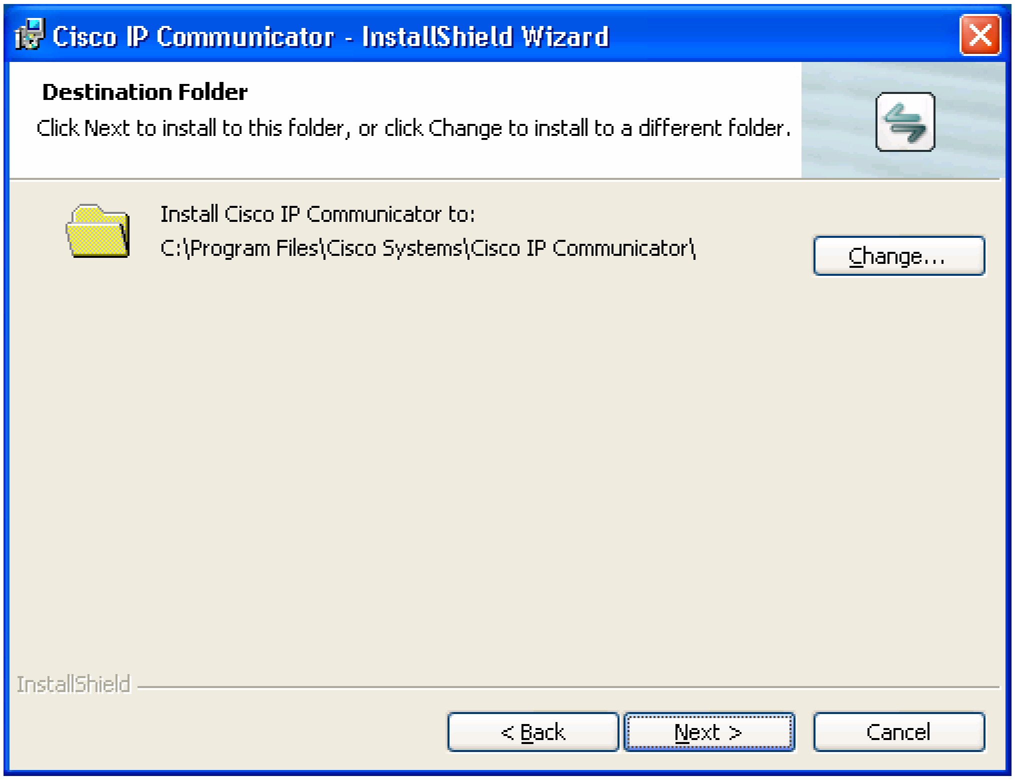


Figure 5-5: CIPC Installation Location

Use the default installation directory and click **Next**.

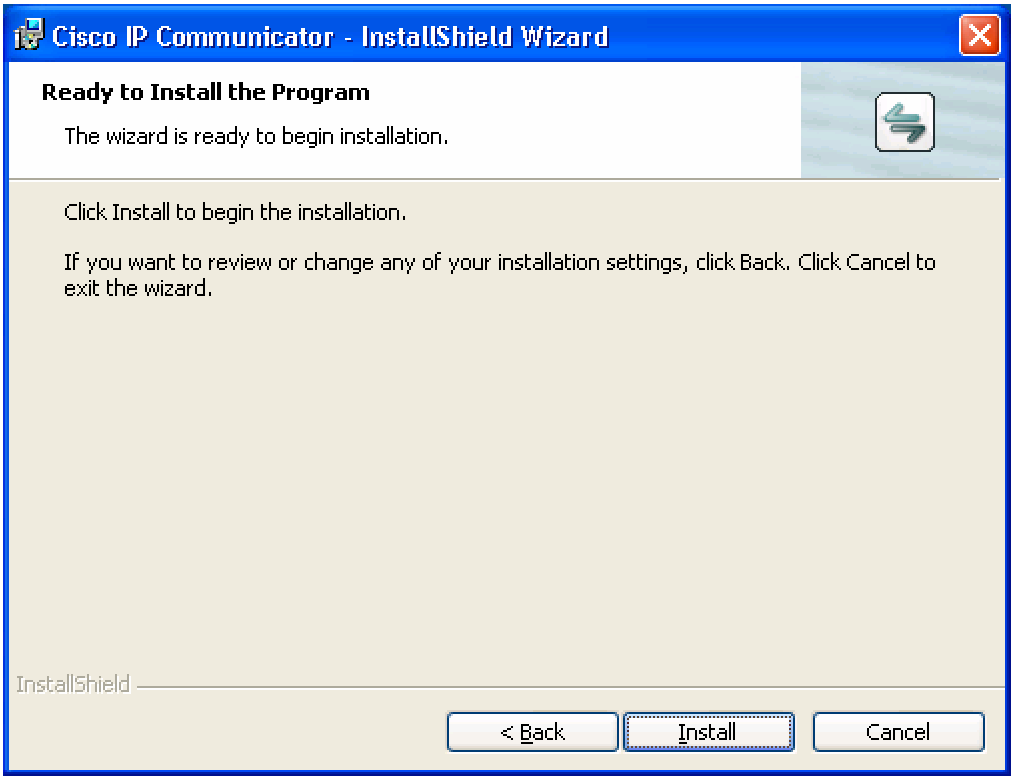


Figure 5-6: CIPC Installation Prompt

Click **Install** to begin installing CIPC.

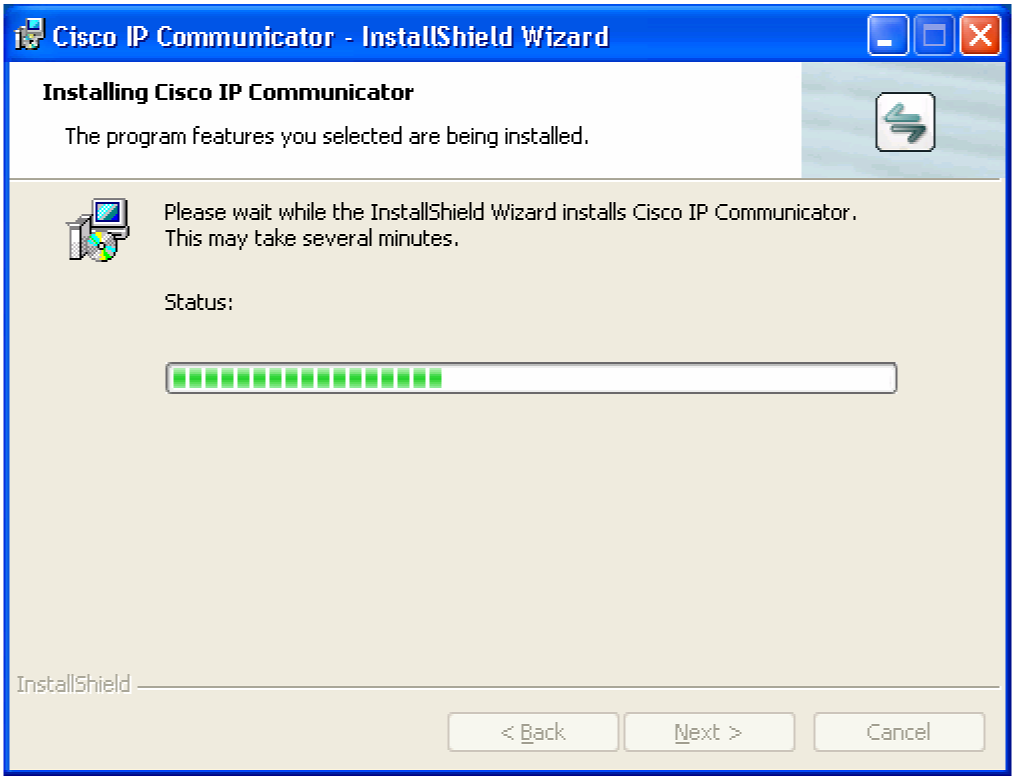


Figure 5-7: CIPC Installation Progress Indicator

Allow CIPC to install.

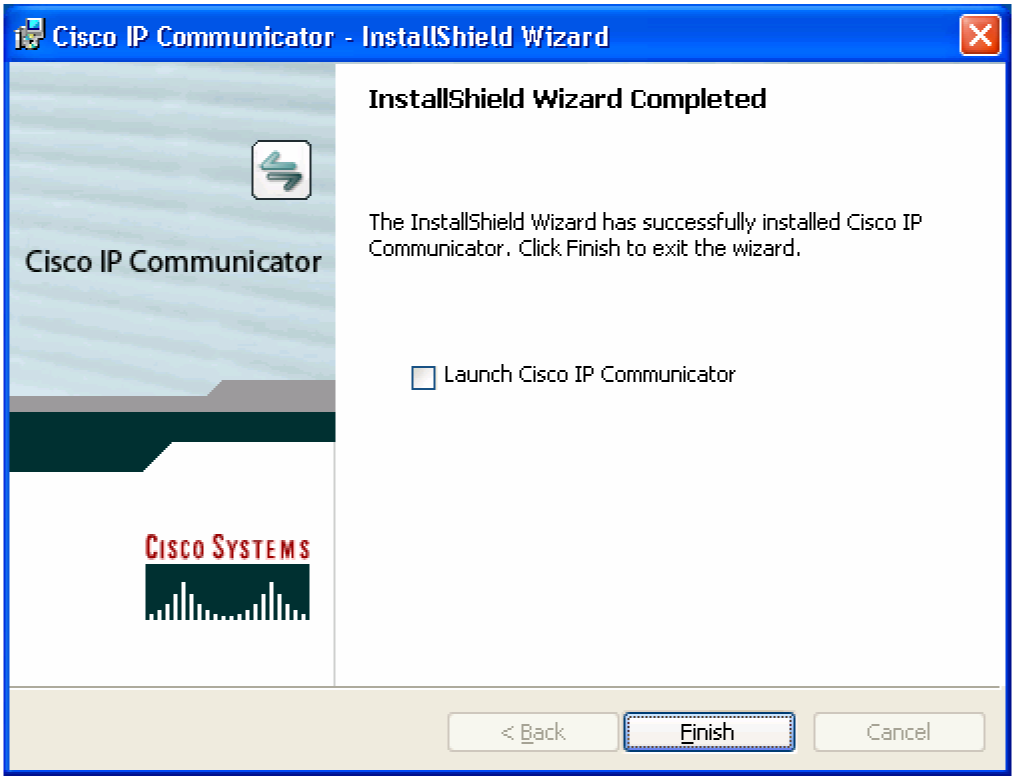


Figure 5-8: CIPC Successful Installation Notification

At the end of the installation process, do not choose to launch CIPC.

Click **Finish**.

Repeat this installation process on Host B if it does not yet have CIPC installed.

Step 6: Run Cisco IP Communicator

Cisco IP Communicator is a simulated Ethernet phone residing in software on a PC.

Before running CIPC, enable debugging for ephone registration on R1 using the **debug ephone register** command. This will let you see ephone registration output.

```
R1# debug ephone register
EPHONE registration debugging is enabled
```

Start CIPC by double clicking the **Cisco IP Communicator** icon installed on the desktop of Host A.

Follow the steps through the Audio Tuning Wizard. This lab will not guide you through the wizard because everyone's audio settings will be different, however, the wizard is self-explanatory.

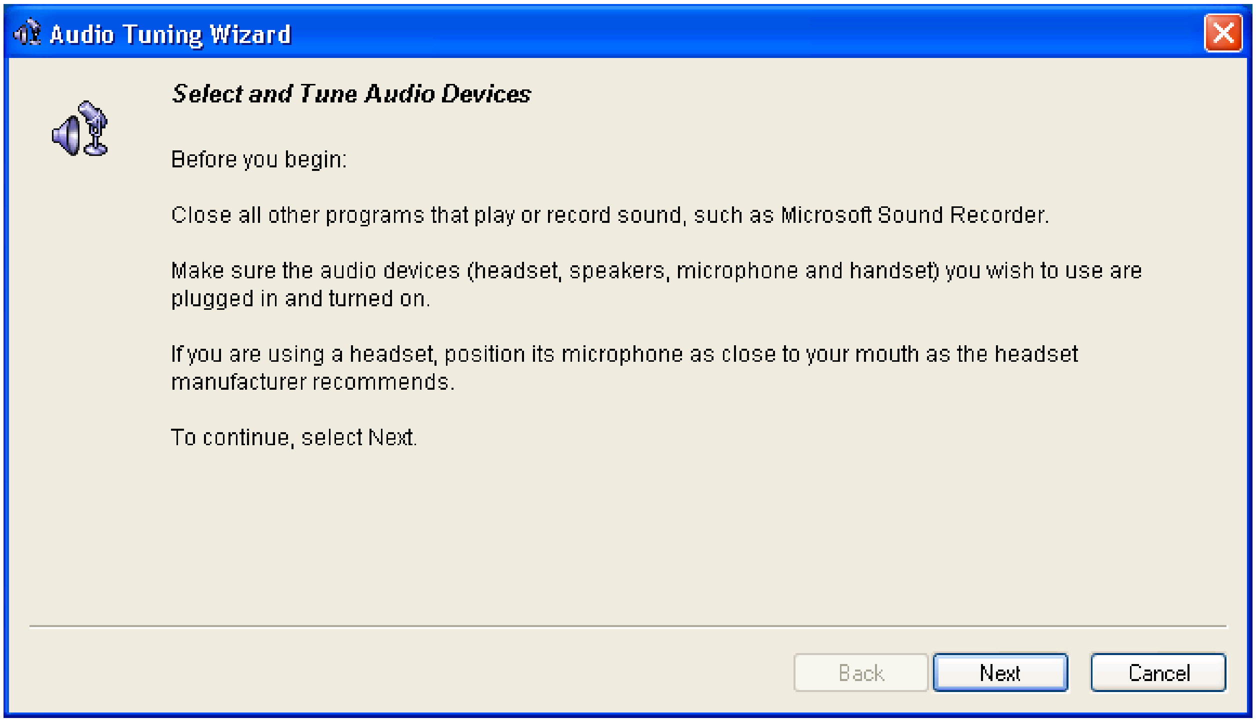


Figure 6-1: CIPC Audio Tuning Wizard

After the Audio Tuning Wizard, the splash screen for CIPC appears while CIPC loads.

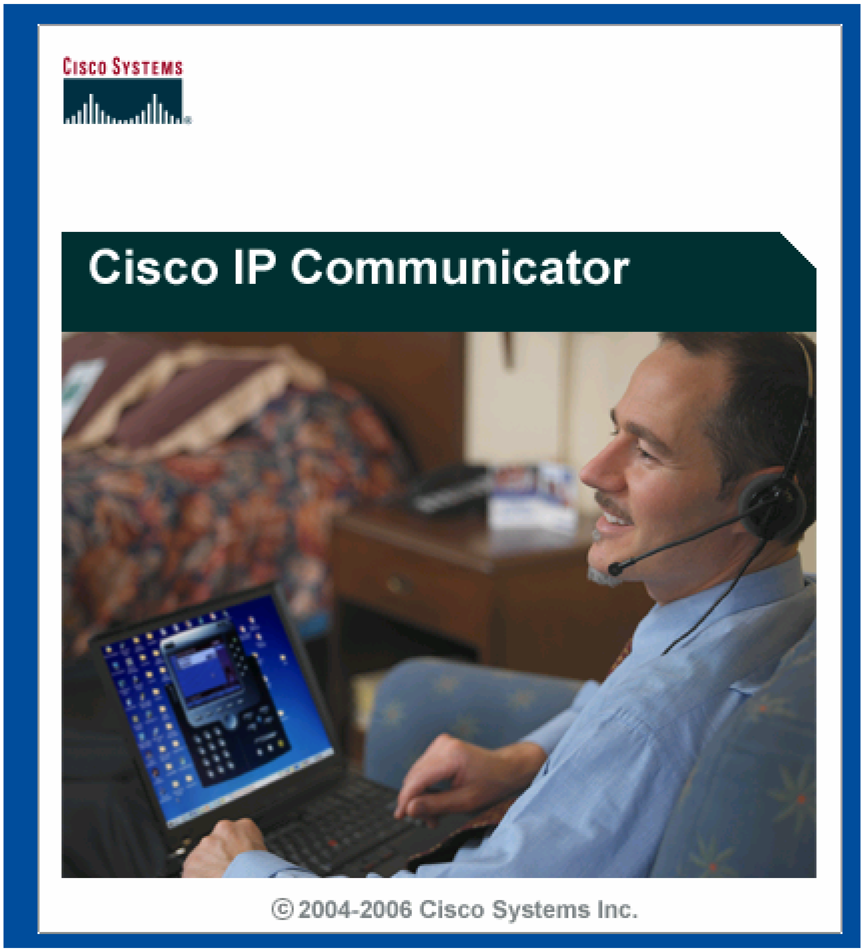
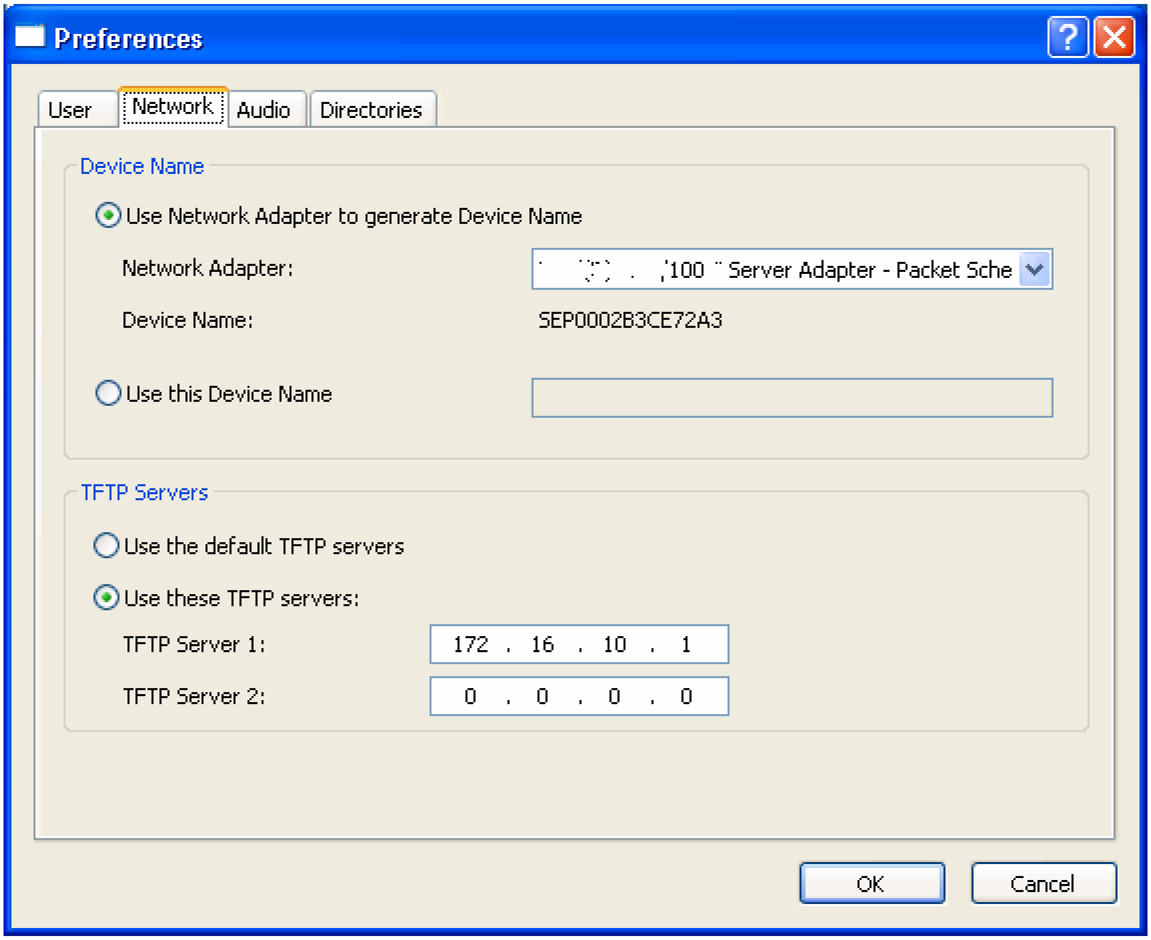


Figure 6-2: CIPC Splash Screen

If this is your first time running Cisco IP Communicator, you will be directed to the preferences page automatically. If you are not and you are presented with the main program (an IP phone image), right-click on the image and choose **Preferences...** to edit CIPC preferences.

Under the **Network** tab of the preferences screen, use the drop-down box to select the correct interface that is used in the lab. Also, under **TFTP Servers**, check **Use these TFTP servers:** and make sure the IP address belongs to R1. Click **OK** once you have changed these settings. Be sure to record any TFTP server settings that are already configured so that these can be restored after the lab.


Preferences
User
Network
Audio
Directories
Device Name
Use Network Adapter to generate Device Name
Network Adapter:
Server Adapter - Packet Sche
Device Name:
SEP0002B3CE72A3
Use this Device Name
TFTP Servers
Use the default TFTP servers
Use these TFTP servers:
TFTP Server 1:
172 . 16 . 10 . 1
TFTP Server 2:
0 . 0 . 0 . 0
OK
Cancel

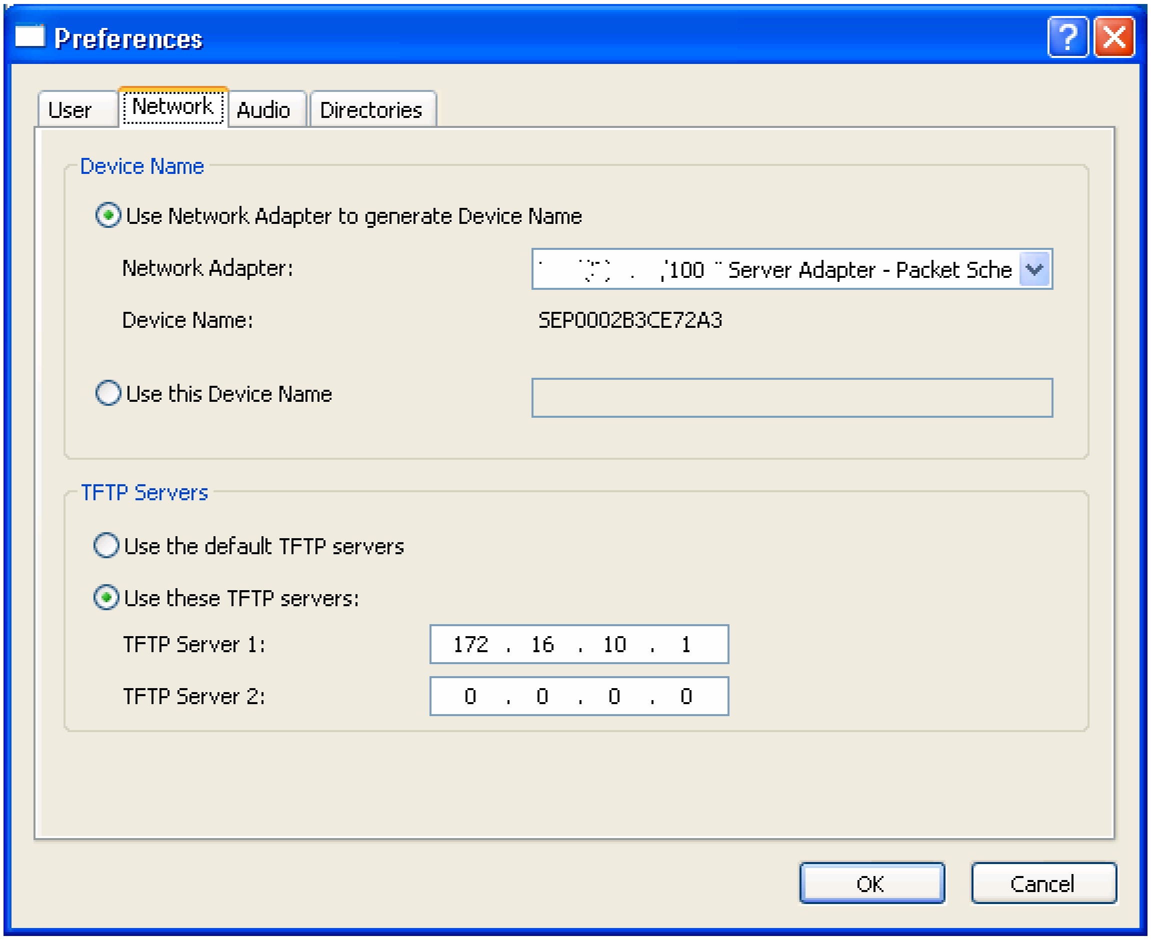


Figure 6-3: CIPC Network Preferences

Figure 6-4: CIPC Main Screen on Host A

If your screen looks similar to this, then the IP phone has successfully registered with R1. Note the correct banner at the bottom of the color display and the correct directory number in the upper-right corner. On R1, look at the debug output generated when R1 registered. The output is rather lengthy, so not all of it is included here.

```
*Jan 30 06:47:37.155: New Skinny socket accepted [2] (0 active)
*Jan 30 06:47:37.155: sin_family 2, sin_port 1034, in_addr 172.16.10.50
```



```
*Jan 30 06:47:37.155: skinny_add_socket 2 172.16.10.50 1034
*Jan 30 06:47:37.211: %IPPHONE-6-REG_ALARM: 25: Name=SEP0002B3CE72A3 Load=
2.0.2.0 Last=Initialized
*Jan 30 06:47:37.211:
Skinny StationAlarmMessage on socket [1] 172.16.10.50
*Jan 30 06:47:37.211: severityInformational p1=0 [0x0] p2=0 [0x0]
*Jan 30 06:47:37.211: 25: Name=SEP0002B3CE72A3 Load= 2.0.2.0 Last=Initialized
*Jan 30 06:47:37.411: ephone-(1)[1] StationRegisterMessage (0/0/4) from
172.16.10.50
*Jan 30 06:47:37.411: ephone-(1)[1] Register StationIdentifier DeviceName
SEP0002B3CE72A3
*Jan 30 06:47:37.411: ephone-(1)[1] StationIdentifier Instance 0    deviceType
30016
*Jan 30 06:47:37.411: ephone-1[-1]:stationIpAddr 172.16.10.50
*Jan 30 06:47:37.411: ephone-1[-1]:maxStreams 3
*Jan 30 06:47:37.411: ephone-1[-1]:protocol Ver 0x84000006
*Jan 30 06:47:37.411: ephone-1[-1]:phone-size 4700 dn-size 568
*Jan 30 06:47:37.411: ephone-(1) Allow any Skinny Server IP address
172.16.10.1
*Jan 30 06:47:37.411: ephone-1[-1]:Found entry 0 for 0002B3CE72A3
*Jan 30 06:47:37.411: ephone-1[-1]:socket change -1 to 1
*Jan 30 06:47:37.411: ephone-1[-1]:FAILED: CLOSED old socket -1
*Jan 30 06:47:37.411: ephone-1[1]:phone SEP0002B3CE72A3 re-associate OK on
socket [1]
*Jan 30 06:47:37.411: %IPPHONE-6-REGISTER: ephone-1:SEP0002B3CE72A3
IP:172.16.10.50 Socket:1 DeviceType:Phone has registered.
<OUTPUT OMITTED>
```

You may disable debugging using **undebug all**, or leave it on if you wish to see the other phone as well (just remember to undebug when you are done with the lab).

Configure Host B similarly and it should receive the correct directory number.

Figure 6-5: CIPC Main Screen on Host B

Step 7: Establish a Call from Host A to Host B

On Host A, dial extension 5002 (Host B's) by typing in the numbers on your keyboard or using the visual keypad in CIPC. Then click the **Dial** softkey.

Figure 7-1: Dialing from Host A to Host B

On host B, you should hear the phone ringing or see it receiving a call. Click the **Answer** softkey to pick up.

Figure 7-2: Host B Receiving the Call from Host A

On both phones, the call timers should increment while on the phone.

Figure 7-3: In-Call Display on Host A

Step 8: Change the Codec Being Used (OPTIONAL - Requires a version of the IOS that has Call Manager Express (CME))

There are multiple codecs that can be used for VOIP. A codec is the method used to encode and decode between analog (sound) voice data and a digital format. To find out the codec currently being used, establish a VOIP call between the two hosts as shown before and double click the **?** button on the phone.

Figure 8-1: Call Statistics

End the call. On R1, under both ephone prompts, use the **codec** *type* command to change the codec from the default, **g711ulaw**, to **g729r8**.

```
R1(config)# ephone 1
R1(config-ephone)# codec g729r8
R1(config-ephone)# ephone 2
```

```
R1(config-ephone)# codec g729r8
```

Close and reopen IP communicator on both hosts. Now, try establishing a call between the two hosts, then clicking the **?** button.

Figure 8-2: Call Statistics on Host A with Codec Change Applied

Notice the codecs listed now on the phone. G.729 only uses 8Kb of bandwidth, versus G.711, which uses 64Kb. Of course, there must be a tradeoff to decrease bandwidth usage, which in this case is sound quality. Once you are done observing the statistics, you may hang up the call.

Final Configurations

```
R1# show run
!
hostname R1
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
telephony-service
 max-ephones 4
 max-dn 10
 ip source-address 172.16.10.1 port 2000
 system message Cisco VOIP
 keepalive 15
 max-conferences 8 gain -6
 transfer-system full-consult
!
ephone-dn  1
 number 5001
 name Host A
!
ephone-dn  2
 number 5002
 name Host B
!
ephone  1
 device-security-mode none
 mac-address 0002.B3CE.72A3
 codec g729r8
 type CIPC
 button  1:1
!
ephone  2
 device-security-mode none
 mac-address 0009.5B1B.67BD
 codec g729r8
 type CIPC
 button  1:2
!
end
```


Lab 3.1 Preparing for QoS

Learning Objectives

- Create complete configurations to be used with later Quality of Service labs
- Use Pagent tools to create traffic flows for test purposes
- Load and store Pagent configurations
- View statistics on traffic flows during network tests

Topology Diagram

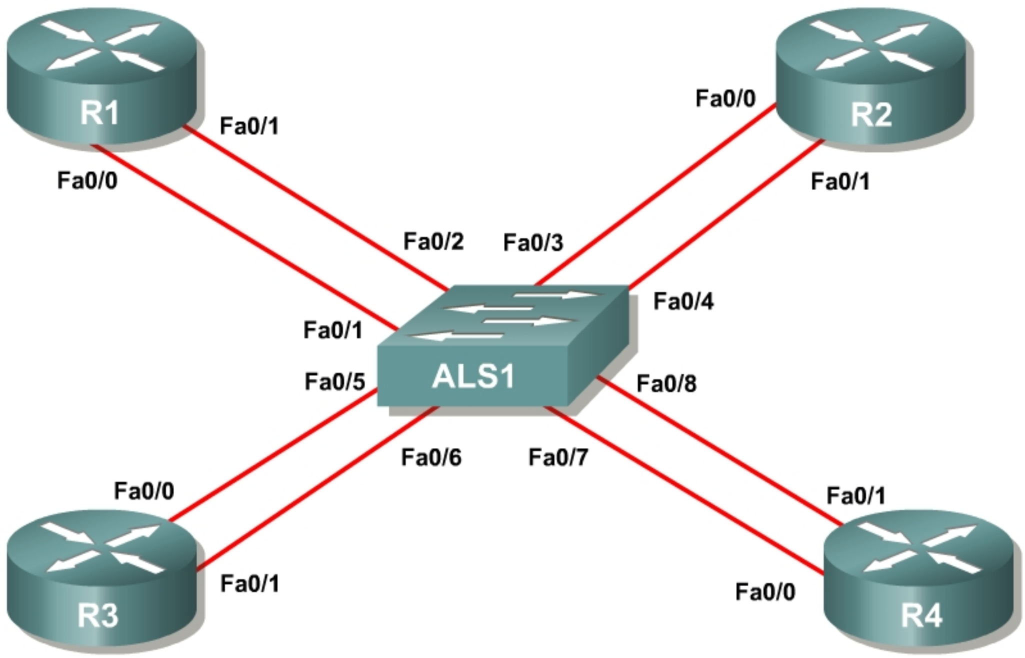


Figure 1-1: Ethernet Connectivity Diagram

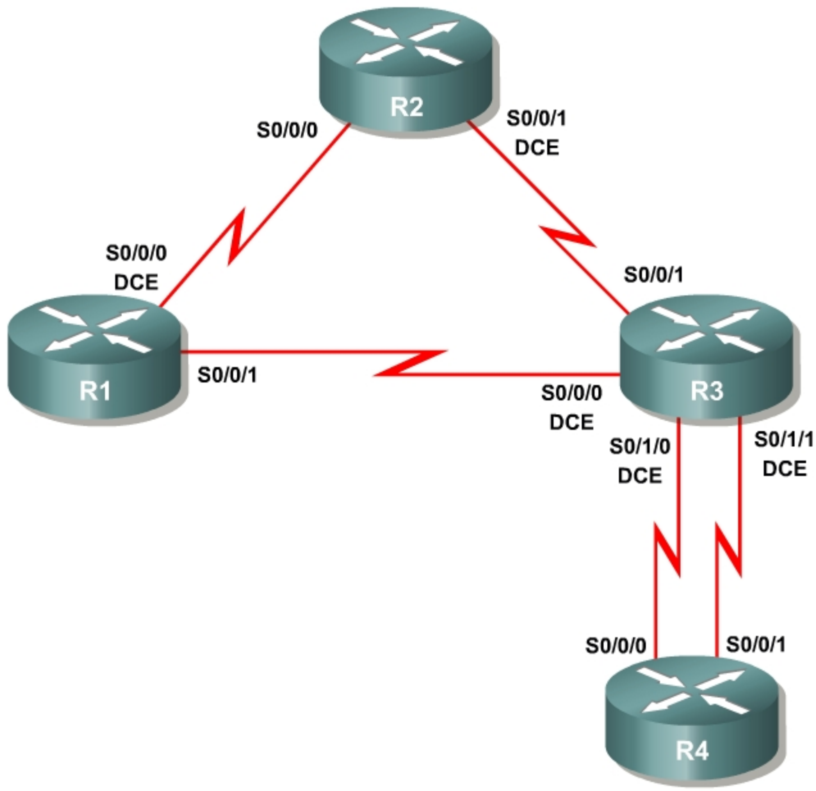


Figure 1-2: Serial Connectivity Diagram

Overview

The Quality of Service (QoS) labs for Modules 3, 4, and 5 have been designed to rely on traffic generation and measuring tools for testing purposes. Traffic generation will be used to create streams of traffic that will flow through your network unidirectionally.

The authors **highly recommend** that you use the Cisco Pagent image and toolset for the QoS labs in the QoS modules. Pagent is a set of traffic generation and testing tools that runs on top of a Cisco IOS image. Booting a router with Pagent can be done by acquiring the image through the Cisco Networking Academy program, loading it into the router's flash memory, and entering a license key when prompted during system boot.

When using the lab configuration suggested in the "CCNP: Optimizing Converged Networks Lab Configuration Guide," you should load the Pagent image on R4.

Key point: Each router booted with Pagent requires a machine-specific license key. It is important to have the license key for R4 before beginning this lab.

This lab guides you through creating configurations for the QoS labs and includes two different configurations.

You will employ the Basic Pagent Configuration in labs that demonstrate each QoS tool separately through two or three routers. You will use the Advanced Pagent Configuration in labs that integrate QoS components across four routers, with R4 acting as both the traffic generator and as a router. The interfaces involved in traffic generation will be isolated from normal routing to ensure that you can use R4 in both roles.

For purposes of this lab, it is assumed that you already have obtained, installed, and activated a Pagent IOS image with a license key on the TrafGen/R4 router.

Finally, labs in these modules may be completed without using any traffic generation. The same configuration steps in each lab will be followed. However, without packet generation tools, you will not see real-time command output.

Step 1: Preliminaries

Erase the startup configurations on any routers involved in this lab. You may need to reactivate Pagent because the activation key is stored in the running configuration of the router.

Traffic generated from TGN, the traffic generation component of Pagent, requires almost all header fields to be hardcoded. Since the packets will be generated over Ethernet, you need to set the destination MAC address of the packets so that they are not broadcast. Remember that this is only the destination for the first hop, not the final destination MAC address. Use the **show interfaces** command to discover the following values.

Example:

```
R1# show interfaces fastethernet0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is MV96340 Ethernet, address is 0019.0623.4380 (bia 0019.0623.4380)
<OUTPUT OMITTED>
```

Record the following value since you will need it at various points throughout this lab:

R1 FastEthernet 0/0, MAC Address: ____________________

Step 2: Create Basic Pagent IOS and TGN Configurations

This step guides you through creating the Basic Pagent Configuration. In this lab, traffic will flow solely through R1, which will function as the entire network "cloud." That is, generated traffic will go through R1 and directly back to TrafGen. In the actual QoS labs, the generated traffic will go to the first hop router, traverse the network topology, and then end back at the TrafGen router (or another destination) as shown in the following diagram:

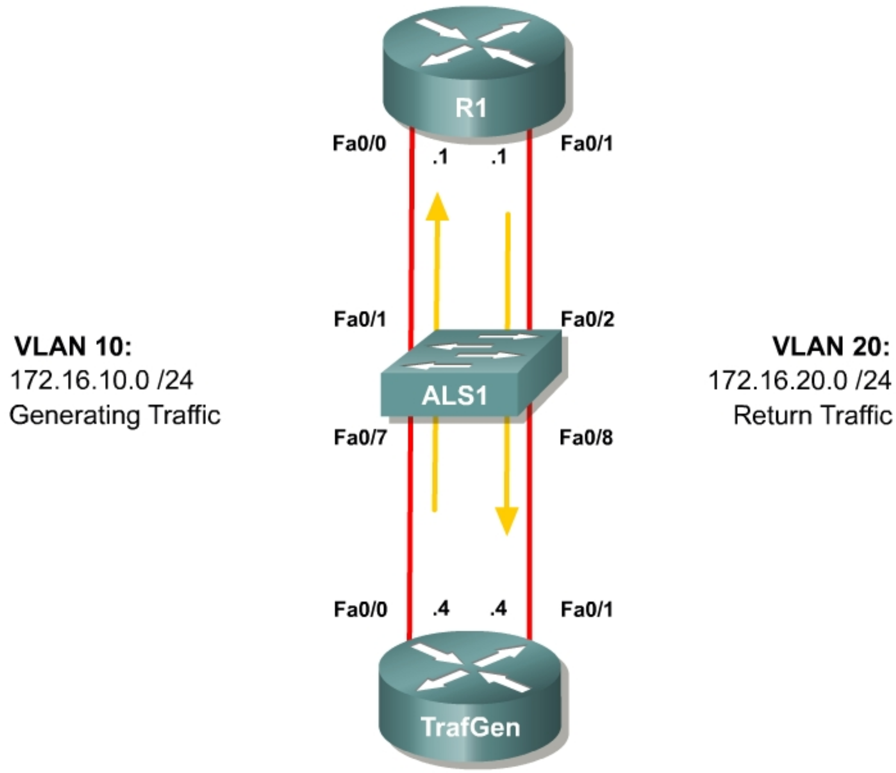


Figure 2-1: Basic Pagent Configuration

- VLAN 10 will be used to send traffic from TrafGen to R1.
- VLAN 20 will be used for traffic returning to the TrafGen router after passing through the last router in the network topology.

You need to assign switchports into the VLANs shown in the diagram.

In order to test connectivity in this scenario, configure TrafGen to send traffic to R1 and then directly back to TrafGen.

Configure the switch to provide Ethernet connectivity for VLANs 10 and 20 as shown in the diagram. Do not configure the FastEthernet 0/2 interface on the switch yet.

```
ALS1# configure terminal
ALS1(config)# interface fastethernet0/1
ALS1(config-if)# switchport access vlan 10
ALS1(config-if)# switchport mode access
ALS1(config-if)# interface fastethernet0/7
ALS1(config-if)# switchport access vlan 10
ALS1(config-if)# switchport mode access
ALS1(config-if)# interface fastethernet0/8
ALS1(config-if)# switchport access vlan 20
ALS1(config-if)# switchport mode access
```


This configuration will be used to begin labs that use the Basic Pagent Configuration. Since the network topology's exit point will change from lab to lab, only TrafGen's FastEthernet 0/1 interface will be placed in VLAN 20 for your template to load at the beginning of each lab that uses the Basic Pagent Configuration. Save this configuration on the switch to a file in flash memory named *flash:basic.cfg*.

```
ALS1# copy run flash:basic.cfg
Destination filename [basic.cfg]?

1391 bytes copied in 0.730 secs (1905 bytes/sec)
ALS1#
```

For this lab only, R1's FastEthernet 0/1 will be the exit point for the network topology while traffic is forwarded back to TrafGen. Therefore add the FastEthernet 0/2 interface on the switch to access VLAN 20.

```
ALS1(config)# interface fastethernet 0/2
ALS1(config-if)# switchport access vlan 20
ALS1(config-if)# switchport mode access
```

At this point, your switch configuration should be complete.

Put TrafGen into configuration mode.

```
Router> enable
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
```

Copy and paste the following configuration into TrafGen. Adjust the **interface** statements for your lab setup if necessary. You will use the same configuration to begin every QoS lab that uses the Basic Pagent Configuration.

```
hostname TrafGen
!
! Replace this interface with the outgoing interface for generated traffic
interface fastethernet0/0
ip address 172.16.10.4 255.255.255.0
no shutdown
!
! Replace this interface with the incoming interface for generated traffic
!  (return traffic)
interface fastethernet0/1
ip address 172.16.20.4 255.255.255.0
no shutdown
```

Copy and paste the following configuration into R1. This configuration is for this guide only. Normally non-Pagent routers should be configured according to the lab. Replace the interface names as necessary if the physical topology of your lab is different.

```
hostname R1
interface fastethernet0/0
ip address 172.16.10.1 255.255.255.0
no shutdown
```



```
interface fastethernet0/1
ip address 172.16.20.1 255.255.255.0
no shutdown
```

TGN is the bulk packet generator tool of Pagent. On the TrafGen router, enter the TGN configuration prompt by using the privileged EXEC command **tgn**.

```
TrafGen# tgn
TrafGen(TGN:OFF,Fa0/0:none)#
```

Copy and paste the following configuration to a text editor. Replace $R1-MAC$ in the highlighted line in the configuration below with R1's MAC address from Step 1. If you are using a different source interface for generated traffic, replace all instances of "fastethernet0/0" with the appropriate port. If you are using an outbound serial interface, you do not need to specify an l2-dest and should remove the highlighted line entirely. To exit the TGN prompt, use the **end** command.

```
fastethernet0/0
add tcp
rate 1000
l2-dest $R1-MAC$
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 23
length random 16 to 1500
burst on
burst duration off 1000 to 2000
burst duration on 1000 to 3000
add fastethernet0/0 1
l4-dest 80
data ascii 0 GET /index.html HTTP/1.1
add fastethernet0/0 1
l4-dest 21
add fastethernet0/0 1
l4-dest 123
add fastethernet0/0 1
l4-dest 110
add fastethernet0/0 1
l4-dest 25
add fastethernet0/0 1
l4-dest 22
add fastethernet0/0 1
l4-dest 6000
!
end
```

Now that you have configured TGN, starting and stopping traffic generation in a lab is very simple. To start traffic generation, use the privileged EXEC command **tgn start**. To stop traffic generation, use the privileged EXEC command **tgn stop**. Or, enter the TGN prompt using the privileged exec command **tgn**, and then use the **start** and **stop** commands. Either method is acceptable, since both perform the same task.

```
TrafGen# tgn start
TrafGen# tgn stop
TrafGen# tgn
```



```
TrafGen(TGN:OFF,Fa0/0:8/8)# start
TrafGen(TGN:ON,Fa0/0:8/8)# stop
TrafGen(TGN:OFF,Fa0/0:8/8)# end
TrafGen#
```

On R1, use the **show interfaces** command for both the inbound and outbound interfaces to make sure that packets are being generated correctly and routed appropriately. This test should be done while traffic generation is on. For the inbound interface (receiving newly generated packets), make sure the inbound packet counters are incrementing. For the outbound interface (routing the generated packets back to TrafGen), make sure the outbound packet counters are incrementing.

```
TrafGen# tgn start

R1# show interfaces fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is MV96340 Ethernet, address is 0019.0623.4380 (bia 0019.0623.4380)
  Internet address is 172.16.10.1/24
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 2/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 100Mb/s, 100BaseTX/FX
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:16, output 00:00:01, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 874000 bits/sec, 139 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     46701 packets input, 36522488 bytes
<OUTPUT OMITTED>

R1# show interfaces fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is MV96340 Ethernet, address is 0019.0623.4380 (bia 0019.0623.4380)
  Internet address is 172.16.10.1/24
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 2/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 100Mb/s, 100BaseTX/FX
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:26, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 952000 bits/sec, 152 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     55017 packets input, 43066713 bytes
<OUTPUT OMITTED>

R1# show interfaces fastethernet 0/1
FastEthernet0/1 is up, line protocol is up
  Hardware is MV96340 Ethernet, address is 0019.0623.4381 (bia 0019.0623.4381)
  Internet address is 172.16.20.1/24
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
```



```
     reliability 255/255, txload 4/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 100Mb/s, 100BaseTX/FX
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:19, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 1666000 bits/sec, 270 packets/sec
     48 packets input, 17808 bytes
     Received 47 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog
     0 input packets with dribble condition detected
     97245 packets output, 75956525 bytes, 0 underruns
<OUTPUT OMITTED>

R1# show interfaces fastethernet 0/1
FastEthernet0/1 is up, line protocol is up
  Hardware is MV96340 Ethernet, address is 0019.0623.4381 (bia 0019.0623.4381)
  Internet address is 172.16.20.1/24
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 4/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 100Mb/s, 100BaseTX/FX
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:29, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 1794000 bits/sec, 292 packets/sec
     48 packets input, 17808 bytes
     Received 47 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog
     0 input packets with dribble condition detected
     106314 packets output, 82995904 bytes, 0 underruns
<OUTPUT OMITTED>
```

Step 3: Store Basic Pagent Configurations

First, store the Basic Pagent Configuration in flash memory with a filename of *basic-ios.cfg* using the **copy running-config flash:basic-ios.cfg** command. When you require the Basic Pagent Configuration, your first step should be to replace the configuration in NVRAM with this file. Then you would reload your router and load the Pagent configurations.

Caution: Make sure you do not erase the flash file system when you replace the configuration. If you do, you will have to stop the lab and install a Pagent IOS image on the router before continuing.

```
TrafGen# copy running-config flash:basic-ios.cfg
Destination filename [basic-ios.cfg]?
```



```
Erase flash: before copying? [confirm] n
Verifying checksum...  OK (0x3FD3)
2875 bytes copied in 0.600 secs (4792 bytes/sec)
```

As you may have guessed, TGN configurations are stored separately from the running configuration, so they are not saved to the router when you type **copy run start** or **write memory** to save the running configuration to the NVRAM of the router. To save a TGN configuration, use the TGN command **save-config** *location*. To load a TGN configuration from a file, use the TGN command **load-config** *location*. The following example shows the TGN configuration being saved to a file on the flash named *basic-tgn.cfg*, and shows it being loaded back in. Use this filename if you want to be able to load the configuration from the menu in the previous step.

```
TrafGen# tgn
TrafGen(TGN:OFF,Fa0/0:8/8)# save-config flash:basic-tgn.cfg
Save complete.
TrafGen(TGN:OFF,Fa0/0:8/8)# load-config flash:basic-tgn.cfg
   Please wait until 'Load Complete' message.

TrafGen(TGN:OFF,Fa0/0:none)#
   Load Complete.
TrafGen(TGN:OFF,Fa0/0:8/8)#
```

Clear the current TGN configuration before you proceed to the next step. Use the TGN command **clear config**, as shown in the following output.

```
TrafGen(TGN:OFF,Fa0/0:8/8)# clear config
TrafGen(TGN:OFF,Fa0/0:none)#
```

Along with the ALS1's *basic.cfg* file, the configurations saved in this step will be loaded initially at the beginning of each of the labs which use the Basic Pagent Configuration.

Step 4: Create Advanced Pagent IOS, TGN, and NQR Configurations

Keep in mind that the Basic Pagent Configuration will be used in the labs that demonstrate individual QoS tools; the Advanced Pagent Configuration will be used in labs that integrate QoS topics across a larger topology. You will use R4 as both a transit router on which you will configure some QoS tools and as the Pagent host on VLANs 10 and 20 with which you will generate and capture traffic. The interfaces you configure to generate and capture Pagent traffic will be isolated from the default routing table. They will be contained in another routing table, essentially virtualizing the router into two devices. One virtual device will be acting as a host generating traffic on one interface and receiving it back on another after the traffic passes through the network topology. The other virtual router will act as R4 in the topology, associating with the other routers through routing protocols. If you are confused about this concept, discuss it with classmates and study the topology diagram in Figure 5-1 and the conceptual diagram in Figure 5-2. Do not proceed until you understand the concept.

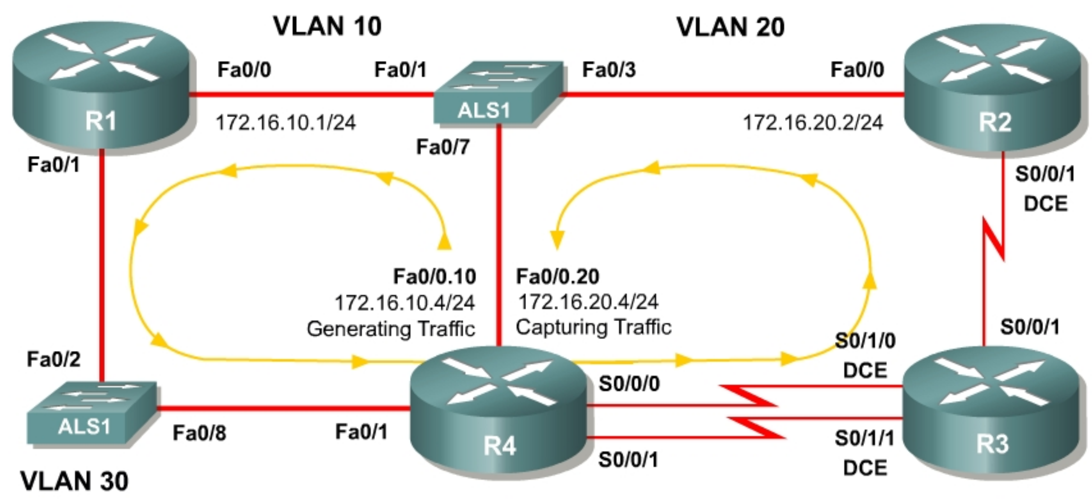


Figure 5-1: Advanced Pagent Configuration

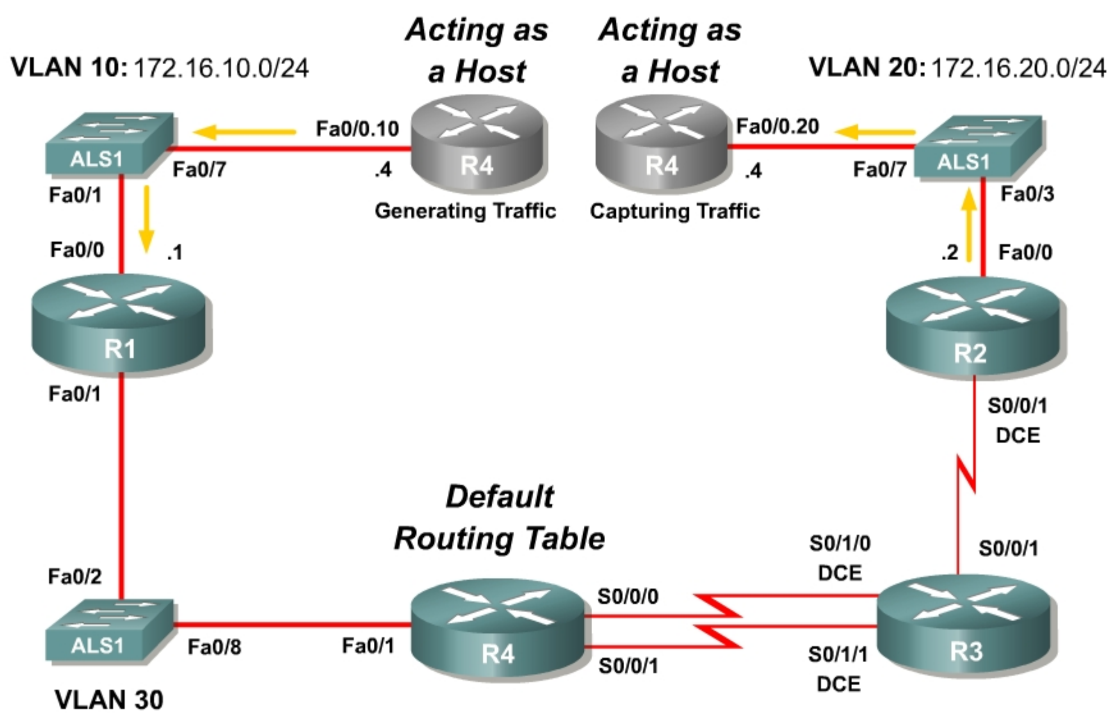


Figure 5-2: Advanced Pagent Configuration, Conceptual Diagram

The recommended configuration uses trunking. If you are using a pod in which you may not manipulate switchports to trunking mode, you may consider using more than one subnet on a single VLAN as shown in Appendix C. Appendix C configurations are NetLab-compatible.

Use the **erase startup-config** command followed by the **reload** command to reset the R4 with a blank configuration. You will need to re-enter the Pagent license key that you first entered in Step 1.

Next, copy and paste the following Advanced Pagent Configuration onto R4 (TrafGen) at the configure prompt. This configuration only includes the commands relevant to Pagent's setup but not those that relate to specific connectivity between R4 and the routers with which it will communicate. This configuration isolates the traffic generation to a separate routing table from the main routing table using virtual routing and forwarding tables, or VRFs. VRFs are outside the scope of this course. To learn more about VRFs, consult cisco.com.

```
hostname R4
!
ip vrf PAGENT
!
interface fastethernet0/0.10
description Interface generating traffic
encapsulation dot1q 10
ip vrf forwarding PAGENT
ip address 172.16.10.4 255.255.255.0
!
interface fastethernet0/0.20
description Interface capturing traffic
encapsulation dot1q 20
ip vrf forwarding PAGENT
ip address 172.16.20.4 255.255.255.0
!
interface fastethernet0/0
no shutdown
```

Configure the switch connected to R4's Fast Ethernet 0/0 port to trunk VLANs 10 and 20 to R4. Also, configure switchports connected to R1 and R2 as access ports and in the VLANs diagrammed above. Finally, place Fast Ethernet interfaces 0/2 and 0/8 on the switch in VLAN 30. Fast Ethernet interfaces 0/2 and 0/8 will be in VLAN 30 for all of the QoS labs that require the Advanced Pagent Configuration.

Copy and paste the following configuration onto the switch in global configuration mode to accomplish these tasks.

```
hostname ALS1
!
vtp mode transparent
vtp domain CISCO
!
vlan 10,20,30
!
interface fastethernet0/1
 switchport mode access
 switchport access vlan 10
!
interface fastethernet 0/2
 switchport mode access
 switchport access vlan 30
!
interface fastethernet0/3
 switchport mode access
 switchport access vlan 20
!
```



```
interface fastethernet0/7
! switchport trunk encapsulation dot1q
! Remove the exclamation point in the previous line
!   if the switch supports multiple trunk encapsulations
 switchport mode trunk
!
interface fastethernet 0/8
 switchport mode access
 switchport access vlan 30
!
end
```

On R4, you will now configure TGN. The configuration you will use is almost identical to the basic one, except modified because we are using subinterfaces. You will not need to put in R1's MAC address because the packets are being encapsulated differently. Use the privileged EXEC command **tgn** to get into the TGN prompt.

```
fastethernet0/0
add tcp
rate 1000
datalink ios-dependent fastethernet0/0.10
l2-arp-for 172.16.10.1
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 23
length random 16 to 1500
burst on
burst duration off 1000 to 2000
burst duration on 1000 to 3000
add fastethernet0/0 1
l4-dest 80
data ascii 0 GET /index.html HTTP/1.1
add fastethernet0/0 1
l4-dest 21
add fastethernet0/0 1
l4-dest 123
add fastethernet0/0 1
l4-dest 110
add fastethernet0/0 1
l4-dest 25
add fastethernet0/0 1
l4-dest 22
add fastethernet0/0 1
l4-dest 6000
!
end
```

Refer to Step 2 to find out how to use TGN.

Step 5: Store Advanced Pagent Configurations

Store the advanced TGN configuration to the file in flash memory named *advanced-tgn.cfg*, and save the advanced IOS configuration to the file in flash named *advanced-ios.cfg*.

```
R4# tgn save-config flash:advanced-tgn.cfg
Save complete.
R4# copy running-config flash:advanced-ios.cfg
```



```
Destination filename [advanced-ios.cfg]?
Erase flash: before copying? [confirm]n
Verifying checksum...  OK (0xDCE7)
1103 bytes copied in 1.228 secs (898 bytes/sec)
```

This configuration will be used to begin labs that use the Advanced Pagent Configuration. Save this configuration on the switch to a file in flash memory named *flash:advanced.cfg*.

```
ALS1# copy run flash:advanced.cfg
Destination filename [advanced.cfg]?

1458 bytes copied in 0.730 secs (1997 bytes/sec)
```

Step 6: Display Traffic Statistics

In many labs using the advanced configuration, you can use NQR to gather traffic statistics. NQR is a Pagent tool that allows you to send and then capture packets. It combines TGN (the traffic generation tool you have already been using) and PKTS (a packet capturing tool you have not set up). Configuration of NQR is similar to that of TGN except that you select one interface for generating the packets and another for capturing them. Unlike the TGN configuration for this course, NQR labs may vary from lab to lab so this configuration is just an example, not a template to be used in all labs.

Before you configure NQR, apply the configurations in Appendix D to each of your routers to set up an end-to-end routing topology using Open Shortest Path First (OSPF).

NQR can be run at the same time as TGN. They work together for QoS testing, in that TGN can generate the bulk background traffic but statistics can be run for the more limited NQR traffic. For this part of the lab, shut off TGN so that its traffic will not interfere with the NQR traffic. If you decide to try this part of the lab, you will also have to configure all of the routers the same way as shown in the Figure 5-1, with IP addresses and a routing protocol (including R4). These configurations are shown in Appendix D. Otherwise, just look at the commands below to get an idea of how NQR works.

To get into the NQR configuration prompt, type **nqr** at the privileged EXEC prompt. After you get into the NQR configuration prompt, copy and paste in the following configuration. Please see appendix C for NETLAB compatible version

```
fastethernet0/0
add tcp
datalink ios-dependent fastethernet0/0.10
l2-arp-for 172.16.10.1
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 23
fastethernet0/0.20 ios-dependent capture
```


This configuration instructs NQR to generate traffic destined towards TCP port 23 (Telnet), similar to the stream configured for TGN. You may notice that an interface was selected for capturing packets.

To start the traffic stream, type **start** (just like TGN). To stop the traffic stream, type **stop**.

Note that in NQR, once traffic generation is stopped, it will keep collecting data and the status will change to “WAIT” before it is done. Once complete, you can view traffic statistics on loss, delay, reordering, jitter, and so forth.

```
R4#nqr
R4(NQR:OFF,Fa0/0:none)# fastethernet0/0
R4(NQR:OFF,Fa0/0:none)# add tcp
R4(NQR:OFF,Fa0/0:1/1)# datalink ios-dependent fastethernet0/0.10
R4(NQR:OFF,Fa0/0:1/1)# l2-arp-for 172.16.10.1
R4(NQR:OFF,Fa0/0:1/1)# l3-src 172.16.10.4
R4(NQR:OFF,Fa0/0:1/1)# l3-dest 172.16.20.4
R4(NQR:OFF,Fa0/0:1/1)# l4-dest 23
R4(NQR:OFF,Fa0/0:1/1)# fastethernet0/0.20 ios-dependent capture
R4(NQR:OFF,Fa0/0:1/1)#
R4(NQR:OFF,Fa0/0:1/1)# start
R4(NQR:ON,Fa0/0:1/1)# stop
R4(NQR:WAIT,Fa0/0:1/1)#
R4(NQR:OFF,Fa0/0:1/1)#
```

Verify packet drop and reordering statistics using the command **show pkt-seq-drop-stats**. You should have zero dropped packets (as seen in the following output) since there is no other traffic running through the network. If all packets are dropped, you have a problem: either they are not getting routed correctly through the network or something else is taking up the bandwidth (which should not happen since TGN was turned off).

```
R4(NQR:OFF,Fa0/0:1/1)# show pkt-seq-drop-stats

Summary of packet sequence/drop stats of traffic streams
  ts#   template interface      sent     recvd   dropped   out-of-seq  max-seq
  1     TCP      Fa0/0.10*        31        31         0            0       31
```

You can also look at delay and jitter statistics with the commands **show delay-stats** and **show jitter-stats** respectively.

```
R4(NQR:OFF,Fa0/0:1/1)# show delay-stats

Summary of delay-stats of traffic streams
 ts#    template interface    min-delay    max-delay    avg-delay  stdev-delay
  1     TCP      Fa0/0.10*     0.009561     0.009771     0.009653     0.000060

R4(NQR:OFF,Fa0/0:1/1)# show jitter-stats

Summary of jitter-stats of traffic streams
 ts#    template interface   min-jitter   max-jitter   avg-jitter stdev-jitter
  1     TCP      Fa0/0.10*     0.000001     0.000144     0.000054     0.000036
```


Appendix A: Basic Pagent Configurations

IOS Configuration on R4 — Stored in **flash:basic-ios.cfg**

```
!
hostname TrafGen
!
username cisco password cisco
username pagent privilege 15 password pagent
username pagent autocommand menu pagentmenu
!
! Replace this interface with the outgoing interface for generated traffic
interface fastethernet0/0
 ip address 172.16.10.4 255.255.255.0
 no shutdown
!
! Replace this interface with the incoming interface for generated traffic
!  (return traffic)
interface fastethernet0/1
 ip address 172.16.20.4 255.255.255.0
 no shutdown
!
line con 0
 login local
!
end
```

TGN Configuration on R4 — Stored in **flash:basic-tgn.cfg**

```
fastethernet0/0
add tcp
rate 1000
l2-dest $R1-MAC$
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 23
length random 16 to 1500
burst on
burst duration off 1000 to 2000
burst duration on 1000 to 3000
add fastethernet0/0 1
l4-dest 80
data ascii 0 GET /index.html HTTP/1.1
add fastethernet0/0 1
l4-dest 21
add fastethernet0/0 1
l4-dest 123
add fastethernet0/0 1
l4-dest 110
add fastethernet0/0 1
l4-dest 25
add fastethernet0/0 1
l4-dest 22
add fastethernet0/0 1
l4-dest 6000
```

IOS Configuration on ALS1 — Stored in **flash:basic.cfg.** You may have to add additional switchports to VLAN 20 in future labs based on specific lab topologies.

```
!
hostname ALS1
!
```



```
interface fastethernet 0/1
 description R1 FastEthernet0/0
 switchport access vlan 10
 switchport mode access
!
interface fastethernet 0/7
 description TrafGen FastEthernet 0/0
 switchport access vlan 10
 switchport mode access
!
interface fastethernet 0/8
 description TrafGen FastEthernet0/1
 switchport access vlan 20
 switchport mode access
!
end
```

Appendix B: Advanced Pagent Configurations

IOS Configuration on R4 (TrafGen) — Stored in **flash:advanced-ios.cfg**. This does not include the sample configuration for NQR (shown in Appendix D)

```
hostname R4
!
username cisco password cisco
username pagent privilege 15 password pagent
username pagent autocommand menu pagentmenu
!
ip vrf PAGENT
!
interface fastethernet0/0
 no shutdown
!
interface fastethernet0/0.10
 encapsulation dot1q 10
 ip vrf forwarding PAGENT
 ip address 172.16.10.4 255.255.255.0
!
interface fastethernet0/0.20
 encapsulation dot1q 20
 ip vrf forwarding PAGENT
 ip address 172.16.20.4 255.255.255.0
!
line con 0
 login local
!
end
```

TGN Configuration on R4 (TrafGen) — Stored in **flash:advanced-tgn.cfg**

```
fastethernet0/0
add tcp
rate 1000
datalink ios-dependent fastethernet0/0.10
l2-arp-for 172.16.10.1
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 23
length random 16 to 1500
burst on
burst duration off 1000 to 2000
```



```
burst duration on 1000 to 3000
add fastethernet0/0 1
l4-dest 80
data ascii 0 GET /index.html HTTP/1.1
add fastethernet0/0 1
l4-dest 21
add fastethernet0/0 1
l4-dest 123
add fastethernet0/0 1
l4-dest 110
add fastethernet0/0 1
l4-dest 25
add fastethernet0/0 1
l4-dest 22
add fastethernet0/0 1
l4-dest 6000
```

IOS Configuration on ALS1 — Stored in **flash:advanced.cfg**

```
hostname ALS1
!
vtp mode transparent
vtp domain CISCO
!
vlan 10,20,30
!
interface fastethernet0/1
 switchport mode access
 switchport access vlan 10
!
interface fastethernet 0/2
 switchport mode access
 switchport access vlan 30
!
interface fastethernet0/3
 switchport mode access
 switchport access vlan 20
!
interface fastethernet0/7
! switchport trunk encapsulation dot1q
! Remove the exclamation point in the previous line
!   if the switch supports multiple trunk encapsulations
 switchport mode trunk
!
interface fastethernet 0/8
 switchport mode access
 switchport access vlan 30
!
end
```

Appendix C: NetLab-compatible Advanced Pagent Configurations

IOS Configuration on R4 — Stored in **flash:advanced-ios.cfg**

```
hostname R4
!
username cisco password cisco
username pagent privilege 15 password pagent
username pagent autocommand menu pagentmenu
!
```



```
ip vrf PAGENT
!
!
interface fastethernet0/0
 ip vrf forwarding PAGENT
 ip address 172.16.20.4 255.255.255.0
 ip address 172.16.10.4 255.255.255.0 secondary
 no shutdown
!
!
line con 0
 login local
!
end
```

TGN Configuration on R4 — Stored in **flash:advanced-tgn.cfg**

```
fastethernet0/0
add tcp
rate 1000
l2-dest $ R1 Fa0/0's MAC$
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 23
length random 16 to 1500
burst on
burst duration off 1000 to 2000
burst duration on 1000 to 3000
add fastethernet0/0 1
l4-dest 80
data ascii 0 GET /index.html HTTP/1.1
add fastethernet0/0 1
l4-dest 21
add fastethernet0/0 1
l4-dest 123
add fastethernet0/0 1
l4-dest 110
add fastethernet0/0 1
l4-dest 25
add fastethernet0/0 1
l4-dest 22
add fastethernet0/0 1
l4-dest 6000
```

NOTE: NETLAB+ would automatically load the following configuration to the switch for this exercise. Notice that VLAN 10 connects two IP subnets 172.16.10.0 and 172.16.20.0.

IOS Configuration on ALS1 – Stored in **flash:advanced.cfg**

```
!
hostname ALS1
!
vtp mode transparent
vtp domain CISCO
!
vlan 10,30
!
interface FastEthernet0/1
```



```
 description Connection to R1 (FastEthernet0/0)
 switchport access vlan 10
 switchport mode access
!
interface FastEthernet0/2
 description Connection to R1 (FastEthernet0/1)
 switchport access vlan 30
 switchport mode access
!
interface FastEthernet0/3
 description Connection to R2 (FastEthernet0/0)
 switchport access vlan 10
 switchport mode access
!
interface FastEthernet0/7
 description Connection to R4 (FastEthernet0/0) - for Pagent Generation
 switchport access vlan 10
 switchport mode access
!
interface FastEthernet0/8
 description Connection to R4 (FastEthernet0/1)
 switchport access vlan 30
 switchport mode access
!
end
```

NQR Configuration

```
Fastethernet0/0
add tcp
l2-dest $R1 Fa0/0's MAC$
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 23
fasethernet0/0 capture
```

Appendix D: Sample Advanced Pagent Configuration

Copy and paste these configurations into their respective routers in the configure prompt. This configuration is only for trying out the advanced topology. These configurations may vary from lab to lab. For the configuration that can be used as a template, consult Appendix B. The switch configurations are not shown.

R1:

```
!
hostname R1
!
interface fastethernet0/0
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface fastethernet0/1
 ip address 172.16.14.1 255.255.255.0
 no shutdown
!
router ospf 1
```



```
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

R2:

```
!
hostname R2
!
interface FastEthernet0/0
 ip address 172.16.20.2 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 ip address 172.16.23.2 255.255.255.0
 clockrate 64000
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

R3:

```
!
hostname R3
!
interface Serial0/0/1
 ip address 172.16.23.3 255.255.255.0
 no shutdown
!
interface Serial0/1/0
 ip address 172.16.34.3 255.255.255.0
 clock rate 64000
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

R4:

```
!
hostname R4
!
interface Serial0/0/0
 ip address 172.16.34.4 255.255.255.0
 no shutdown
!
interface FastEthernet0/1
 ip address 172.16.14.4 255.255.255.0
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
ip route vrf PAGENT 172.16.14.0 255.255.255.0 172.16.10.1
ip route vrf PAGENT 172.16.23.0 255.255.255.0 172.16.10.1
ip route vrf PAGENT 172.16.34.0 255.255.255.0 172.16.10.1
!
end
```


R4 NQR:

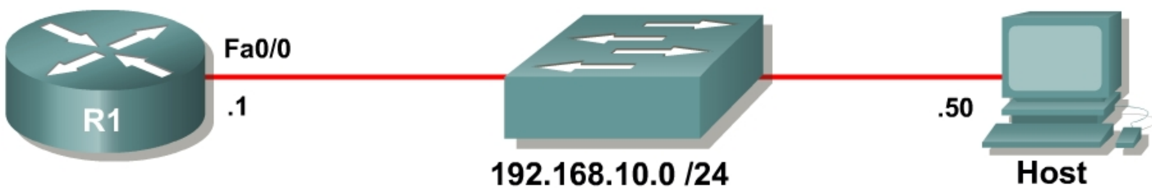
```
fastethernet0/0
add tcp
datalink ios-dependent fastethernet0/0.10
l2-arp-for 172.16.10.1
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 23
fastethernet0/0.20 ios-dependent capture
```


Lab 3.2 Installing SDM

Learning Objectives

- Prepare a router for access with Cisco Security Device Manager
- Install SDM onto a PC
- Install SDM onto a router through a Windows host

Topology Diagram



Scenario

In this lab, you will prepare a router for access via the Cisco Security Device Manager (SDM), using some basic commands, to allow connectivity from the SDM to the router. You will then install the SDM application locally on a host computer. Finally, you will install SDM onto the flash memory of a router.

Step 1: Lab Preparation

Start this lab by erasing any previous configurations and reloading your devices. Once your devices are reloaded, set the appropriate hostnames. Ensure that the switch is set up so that both the router and host are in the same VLAN. By default, all ports on the switch are assigned to VLAN 1.

Step 2: Prepare the Router for SDM

The Cisco SDM application uses the virtual terminal lines and HTTP server to manipulate the configuration of the device. Since a user must log in to access or change the configuration, some basic commands must be issued to allow remote access.

These are basic IOS commands and are not SDM-specific. However, without these commands, SDM will not be able to access the router, and will not work properly.

First, create a username and password on the router for SDM to use. This login will need to have a privilege level of 15 so that SDM can change configuration settings on the router. Make the password argument of this command the last argument on the line, since everything after the password argument will become part of the password. The username and password combination will be used later when accessing the router.

```
R1(config)# username ciscosdm privilege 15 password 0 ciscosdm
```

HTTP access to the router must be configured for SDM to work. If your image supports it (you will need to have an IOS image that supports crypto functionality), you should also enable secure HTTPS access using the **ip http secure-server** command. Enabling HTTPS generates some output about RSA encryption keys. This is normal. Also, make sure the HTTP server uses the local database for authentication purposes.

```
R1(config)# ip http server
R1(config)# ip http secure-server
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
*Jan 14 20:19:45.310: %SSH-5-ENABLED: SSH 1.99 has been enabled
*Jan 14 20:19:46.406: %PKI-4-NOAUTOSAVE: Configuration was modified.  Issue
"write memory" to save new certificate
R1(config)# ip http authentication local
```

Finally, configure the virtual terminal lines of the router to authenticate using the local authentication database. Allow virtual terminal input through both telnet and SSH.

```
R1(config)# line vty 0 4
R1(config-line)# login local
R1(config-line)# transport input telnet ssh
```

Based on your knowledge of SDM, why do you think that the router needs to have these non-SDM specific commands entered in?

Step 3: Configure Addressing

Now that the router has all of the commands necessary for remote access, connectivity will need to be established between the PC and the router. The first thing we will need to do is configure the Fast Ethernet interface on the router with the IP address shown in the diagram. If you have already configured the correct IP address, skip this step.

```
R1(config)# interface fastethernet0/0
R1(config-if)# ip address 192.168.10.1 255.255.255.0
R1(config-if)# no shutdown
```


Next, assign an IP address to the PC. If the PC already has an IP address in the same subnet as the router, you may skip this step. These steps may vary depending on your Windows version and theme.

First, access the PC Control Panel window and open the Network Connections management interface.

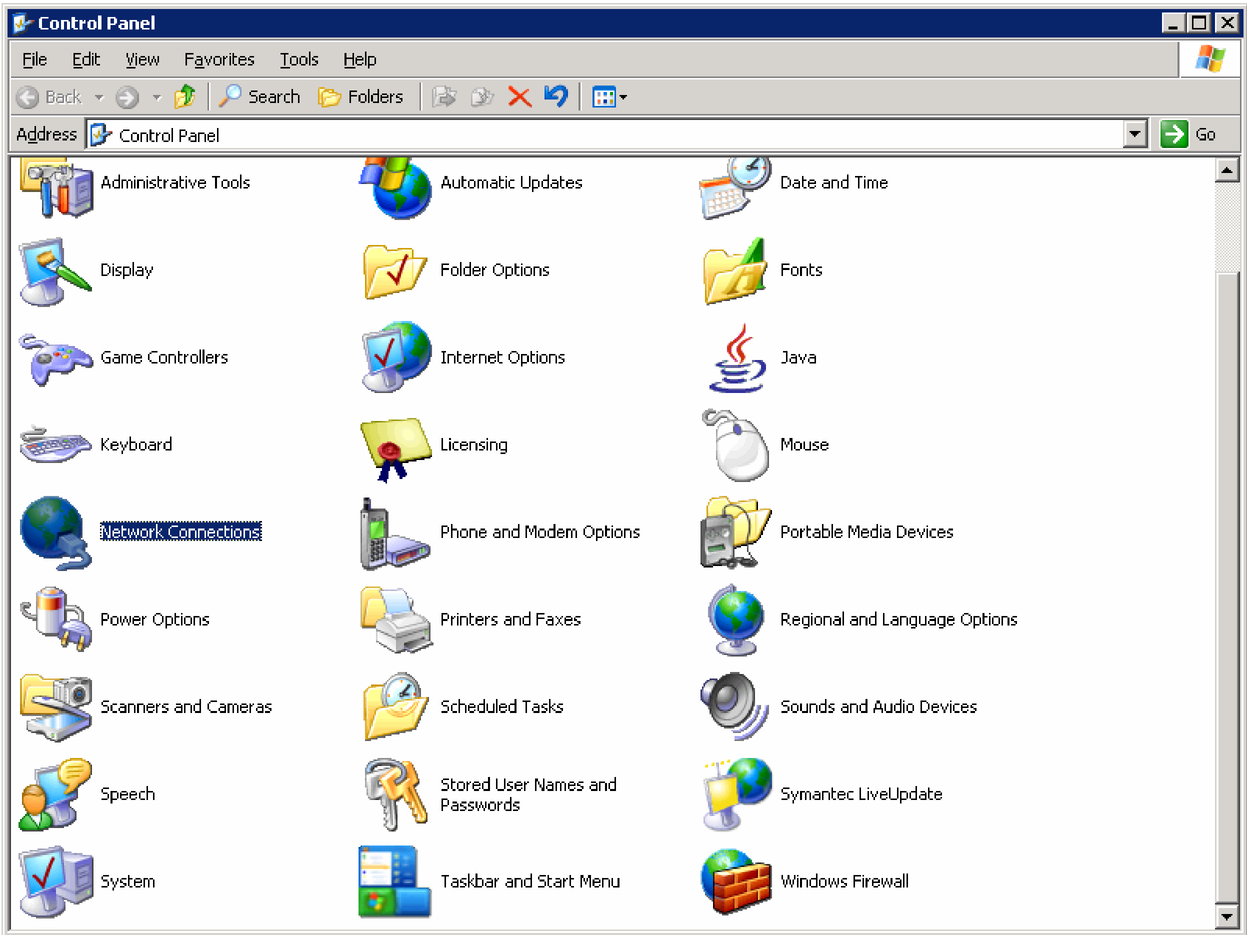


Figure 3-1: Microsoft Windows Control Panel

Right-click the LAN interface that connects to the Catalyst switch and click **Properties**. Choose **Internet Protocol (TCP/IP),** and then click the **Properties** button.

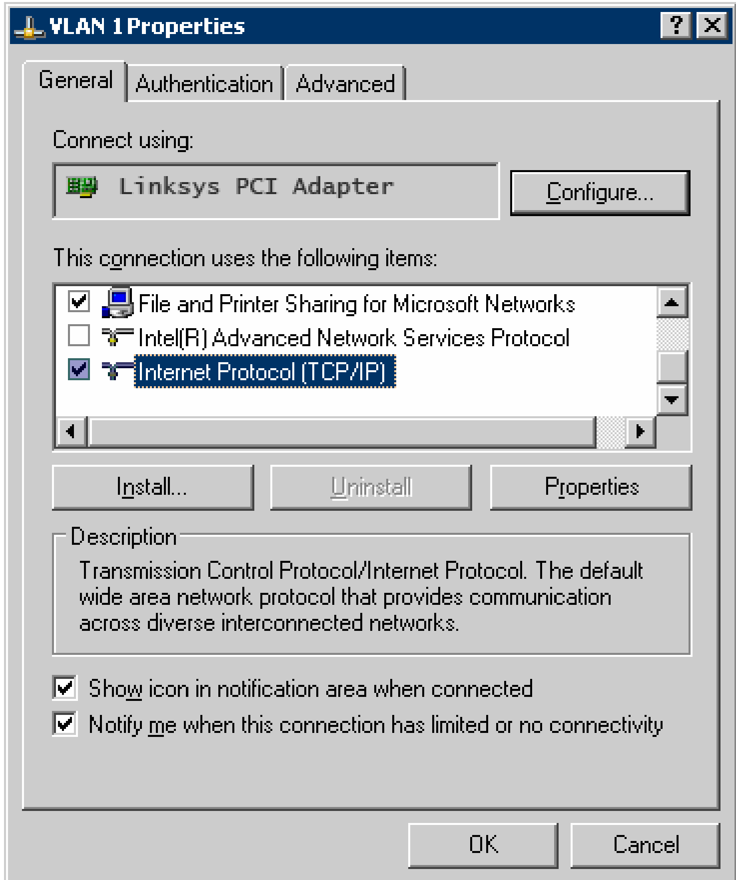


Figure 3-2: Network Connection Properties

Finally, configure the IP address shown in the diagram on the interface.

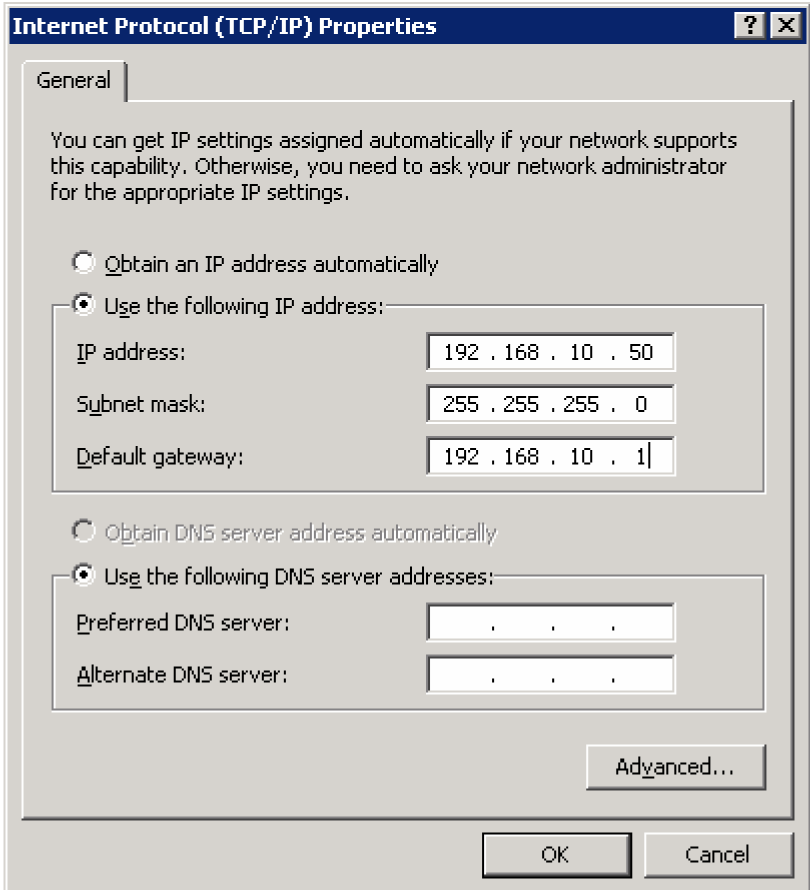


Figure 3-3: IP Properties

Click **OK** once to apply the TCP/IP settings and again to exit the configuration dialog box for the LAN interface. Open the Start Menu, and then click **Run**.... Issue the **cmd** command and press the [Return] key. At the Windows command-line prompt, ping the R1 Ethernet interface. You should receive responses. If you do not receive a response, troubleshoot by verifying the VLAN of the switchports and the IP address and subnet mask on each of the devices attached to the switch.

```
C:\Documents and Settings\Administrator> ping 192.168.10.1

Pinging 192.168.10.1 with 32 bytes of data:

Reply from 192.168.10.1: bytes=32 time=1ms TTL=255
Reply from 192.168.10.1: bytes=32 time<1ms TTL=255
Reply from 192.168.10.1: bytes=32 time<1ms TTL=255
Reply from 192.168.10.1: bytes=32 time<1ms TTL=255

Ping statistics for 192.168.10.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```


Step 4: Extract SDM on the Host

Now that the router is ready to be accessed from SDM and there is connectivity between the router and the PC, you can use SDM to configure the router.

You should start by extracting the SDM zip file to a directory on your hard drive. In this example, the directory used is “C:\sdm\,” although you can use any path you want. If your version of Windows has a built-in zip utility, you can use that to extract it, or if you don’t have it built in, you can use a third-party tool such as WinZip. To get to the built in Windows Extraction Wizard, right-click the SDM zip file and click **Extract All...**. If you decide to use a third-party tool, extract the file to the directory of your choice and skip to the next step.

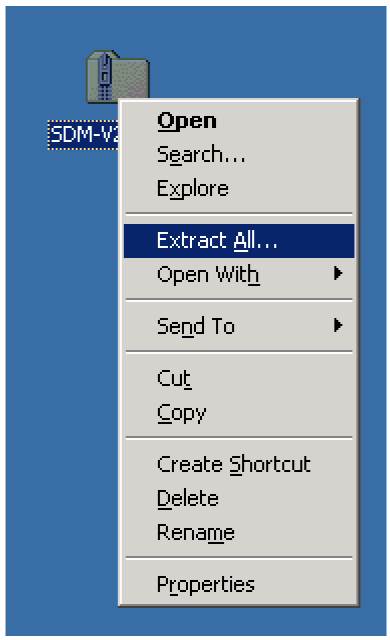


Figure 4-1: Zip File Menu

Once the extraction wizard has opened, click **Next** to get to the destination selection screen.

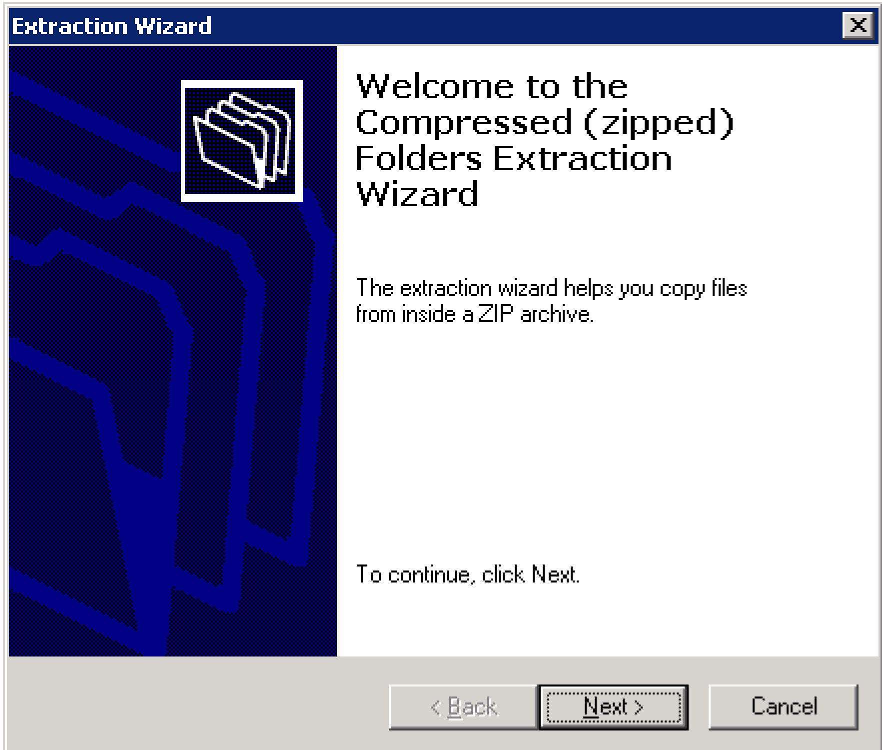


Figure 4-2: Windows Extraction Wizard

Select the folder you want to use as the destination directory, and then click **Next**.

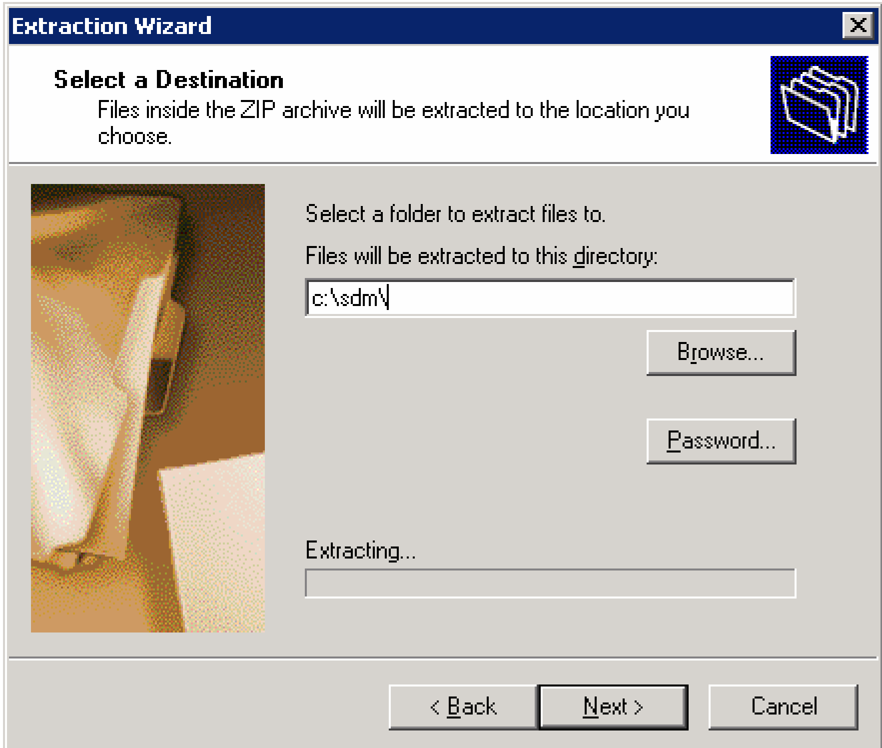


Figure 4-3: Destination Selection Dialog

The files are extracted. It may take a few seconds for the extraction to finish.

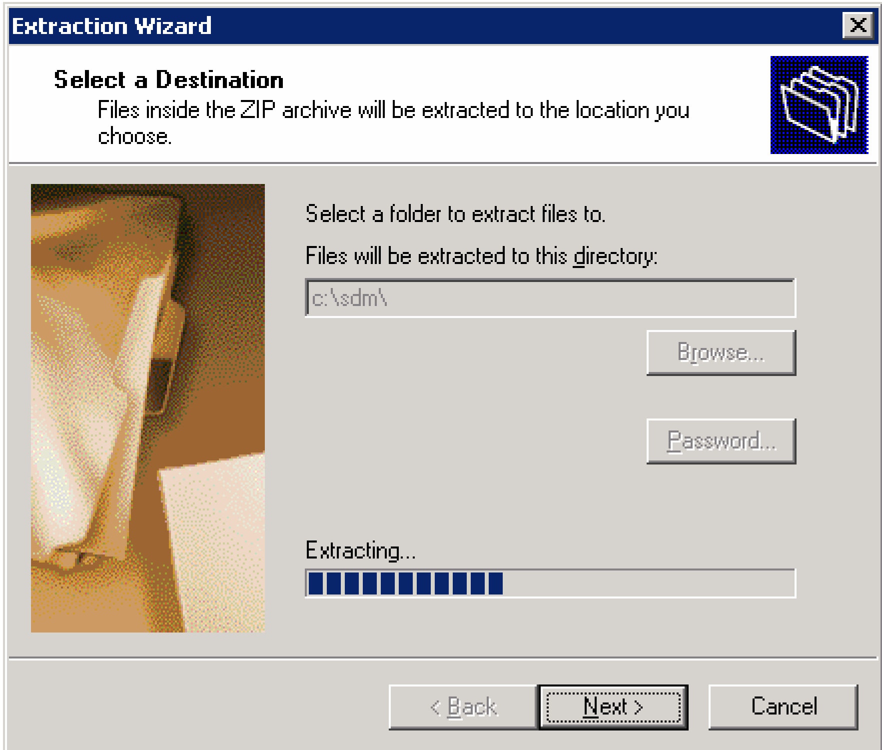


Figure 4-4: Windows Extraction Wizard

Afterwards, you are prompted to decide if you want to show the extracted files. Check this option if it is not already checked, and then click **Finish**.

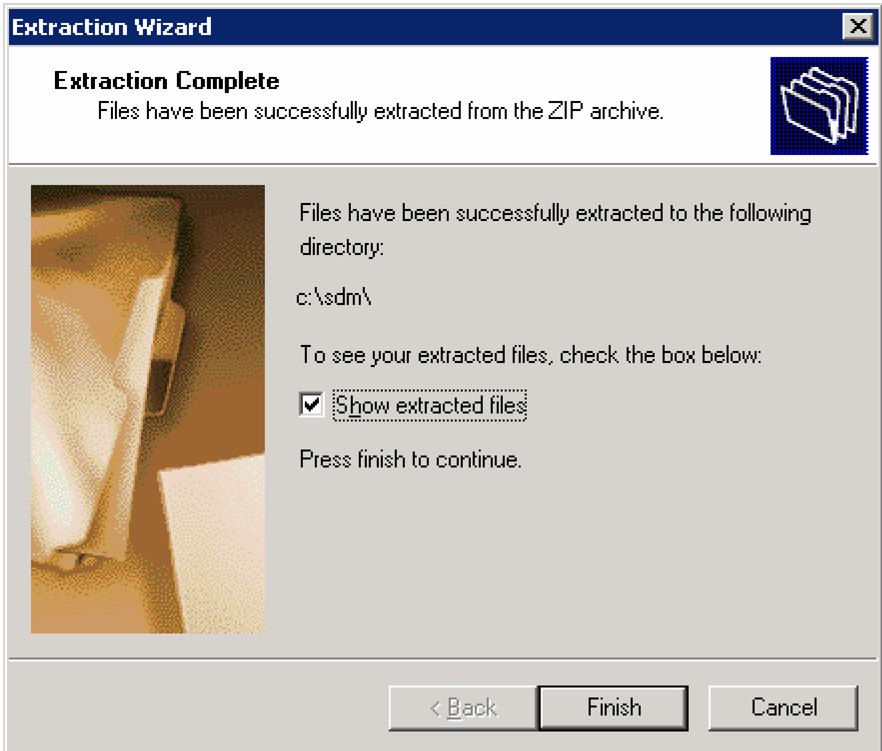


Figure 4-5: Final Extraction Wizard Dialog

After you have extracted the file, open the directory to which the file was extracted. The files in this directory may look different depending on the version of SDM you have.

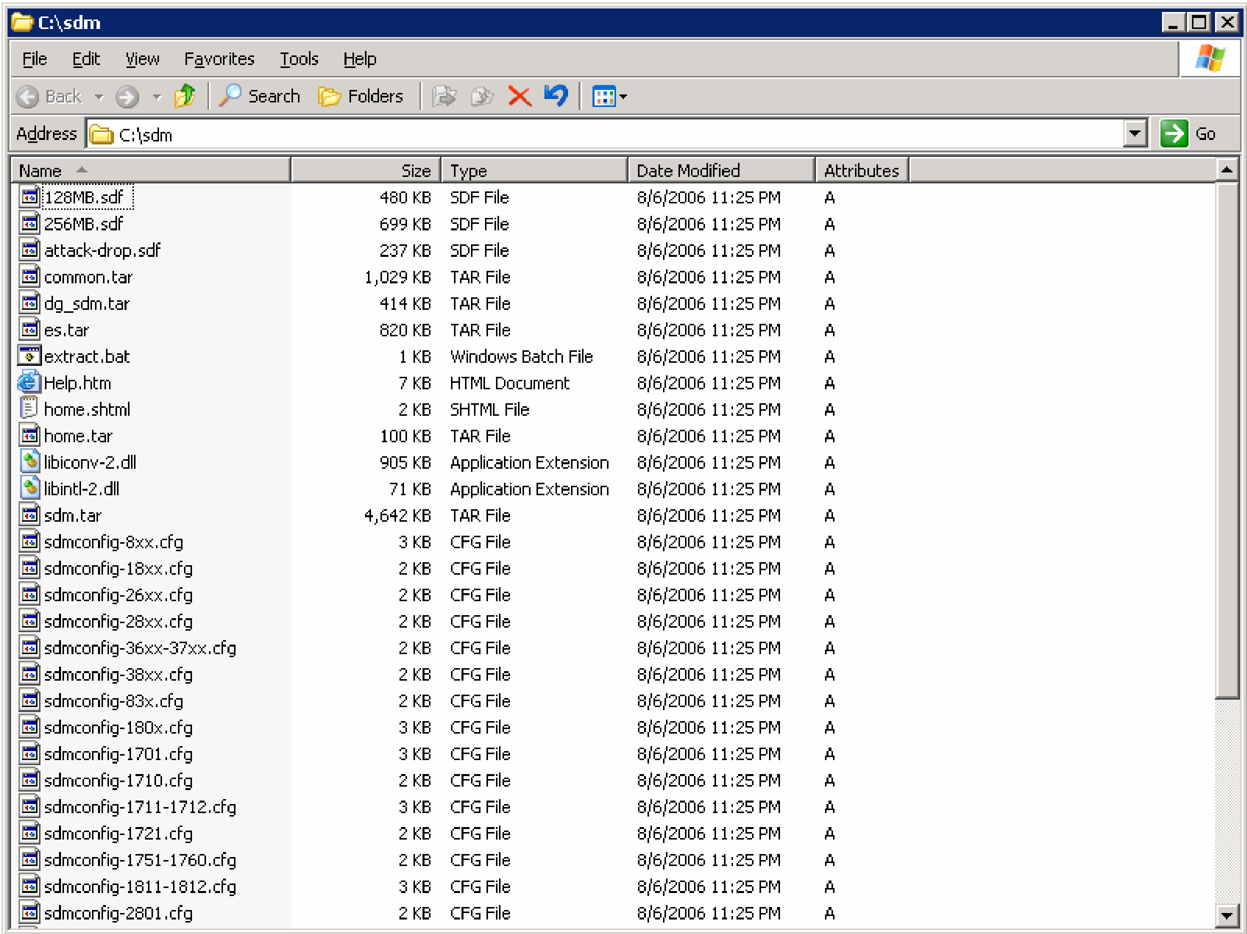


Figure 4-6: Directory of SDM Extraction

You are almost ready to use SDM to configure the router. The last step is installing the SDM application on the PC.

Step 5: Install SDM on the PC

Double-click the **setup.exe** executable program to open the installation wizard. Once the installation wizard screen opens, click **Next**.

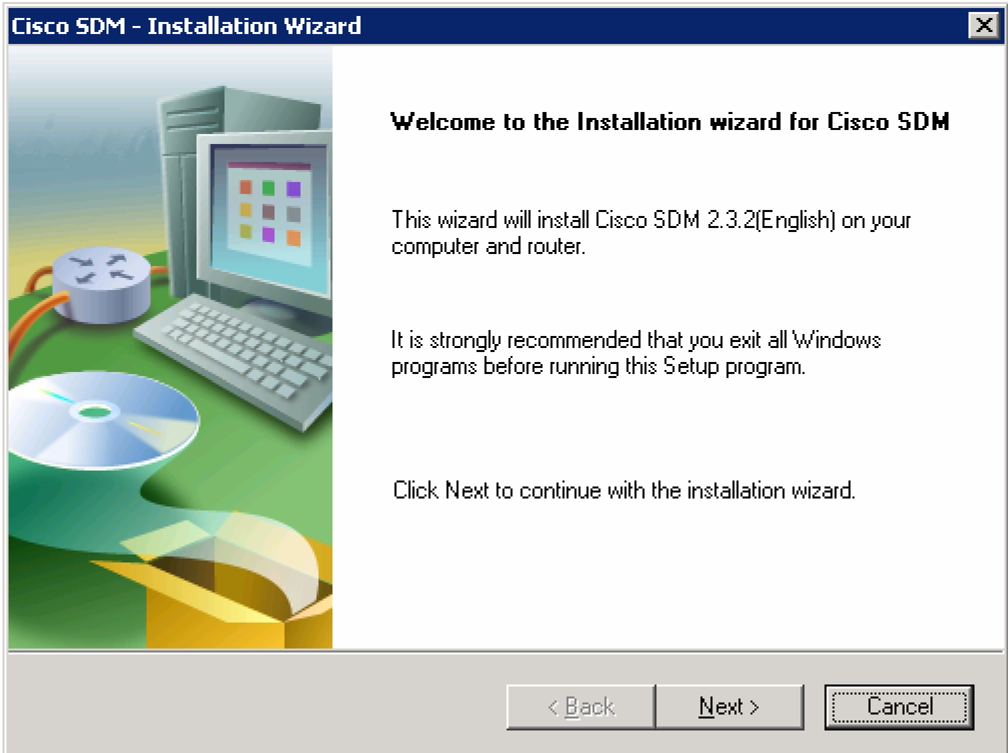


Figure 5-1: Welcome Screen for SDM Installation Wizard

Accept the terms of the license agreement, and then click **Next**.

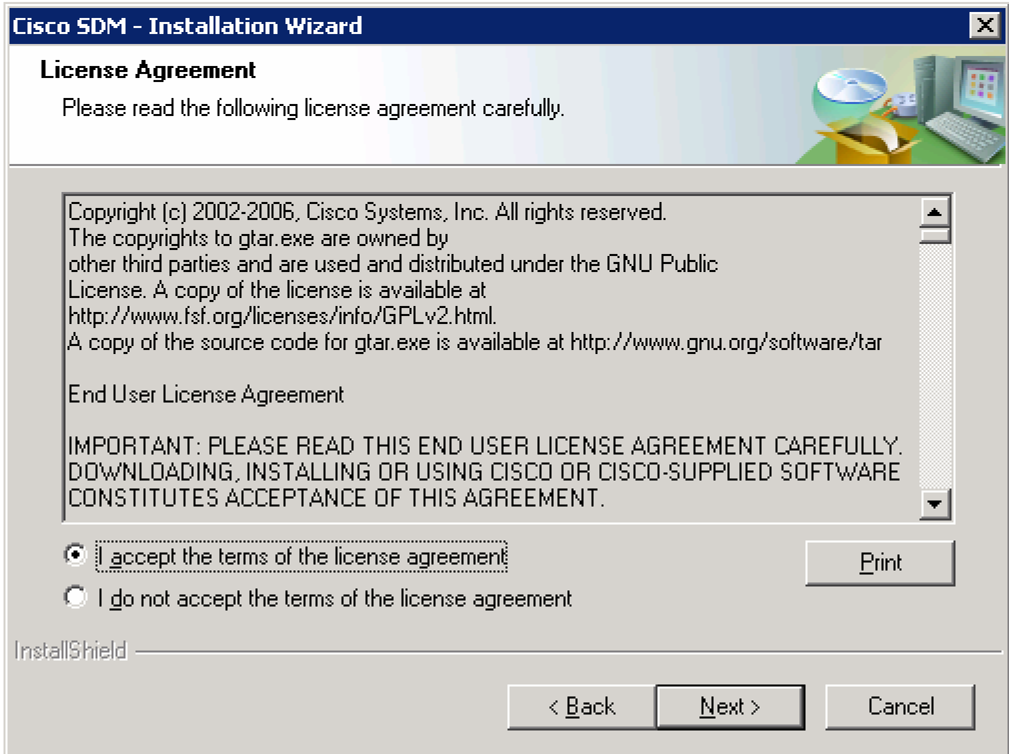


Figure 5-2: SDM License Agreement

The next screen prompts you to choose from three options where you want to install SDM.

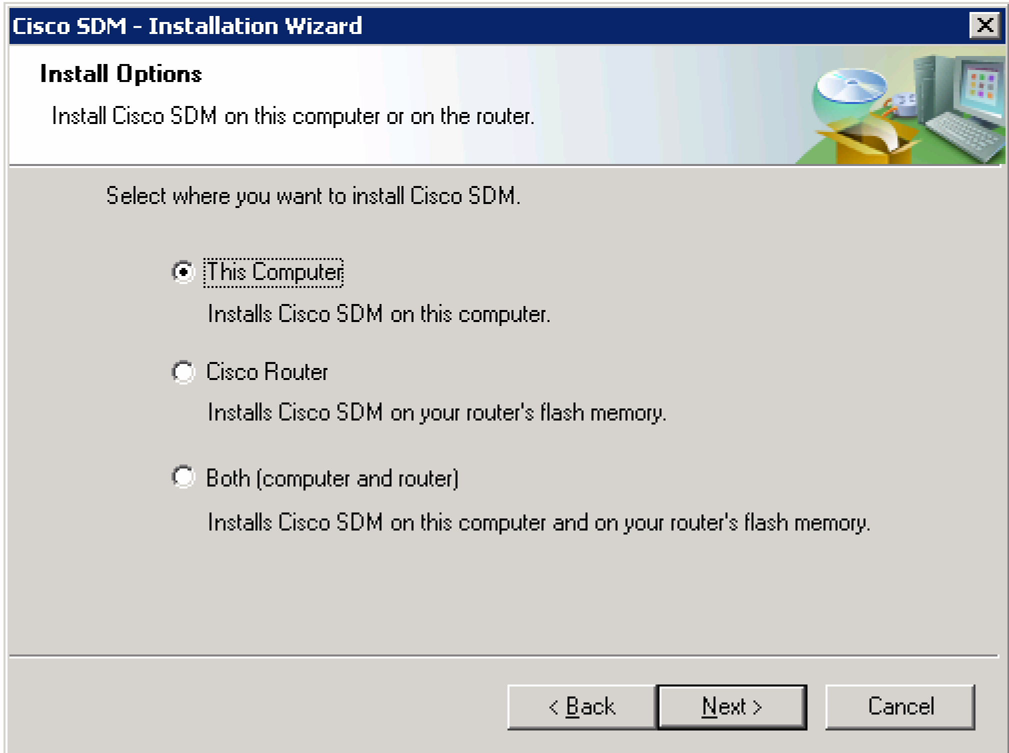


Figure 5-3: Installation Location Options

When installing SDM, you can install the application on the computer and not place it on the flash memory of the router, or you can install it on the router without affecting the computer, or you can install it to both. Both installation types are very similar. This lab explains how to install SDM on your computer and on the Cisco router. It is not necessary to explain how to install it on both because that is self-evident once you have learned how to install to one or the other. If you do not want to install SDM to your computer, skip to step 7.

What are the advantages and disadvantages of installing SDM on the computer only?

What are the advantages and disadvantages of installing SDM on the router only?

What are the advantages and disadvantages of installing SDM on both the router and PC?

For now, click **This computer**, and then click **Next**. Use the default destination folder and click **Next** again.

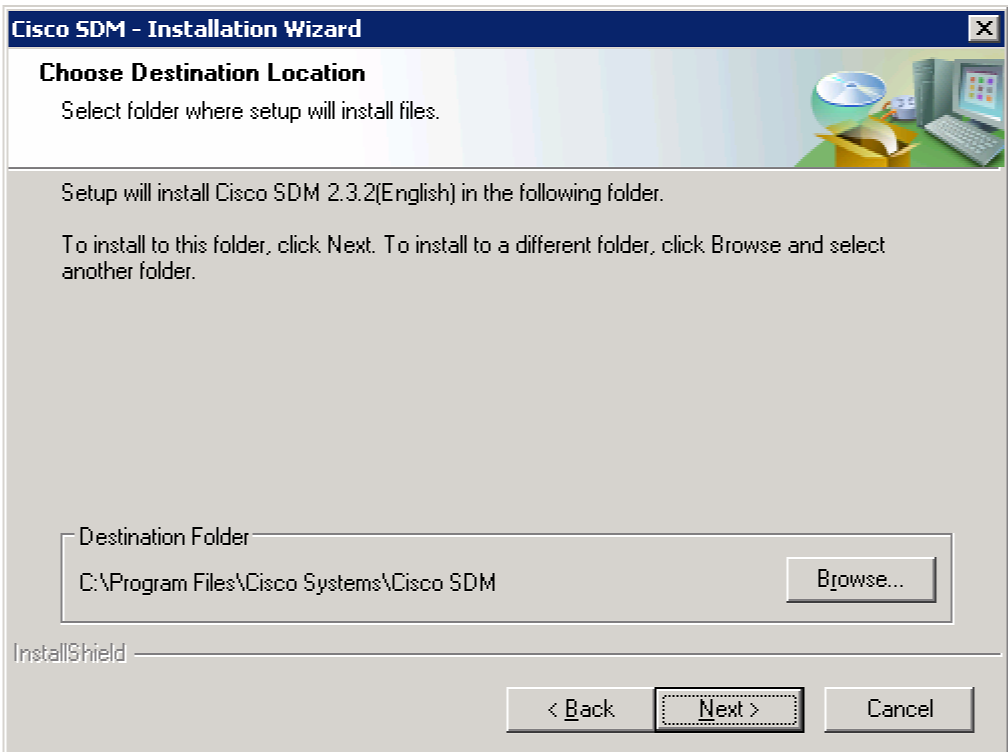


Figure 5-4: Local Installation Location Dialog

Click **Install** to begin the installation.

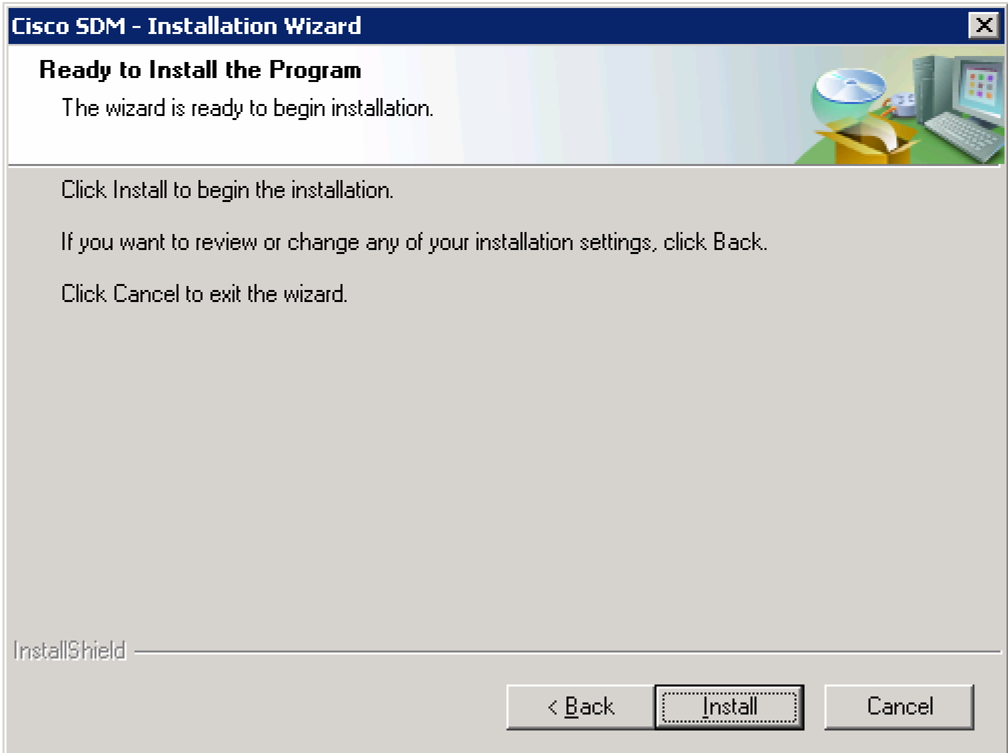


Figure 5-5: Installation Prompt

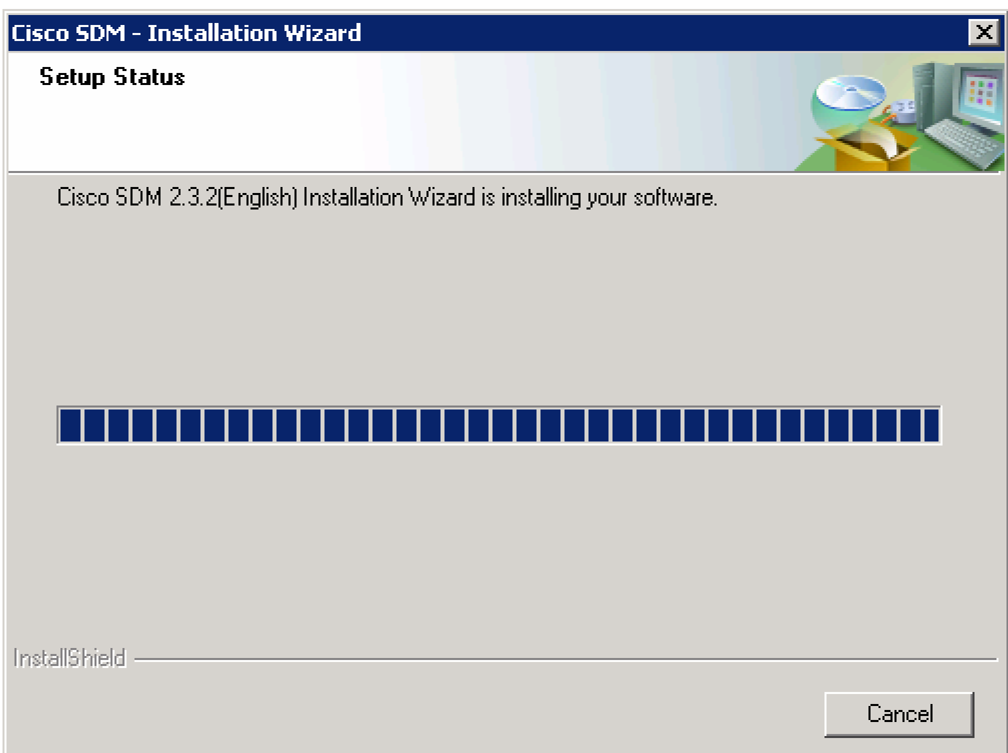


Figure 5-6: Installation Progress Information

The software installs, and then you are prompted with a final dialog box to launch SDM. Check the **Launch Cisco SDM** box, and then click **Finish**.

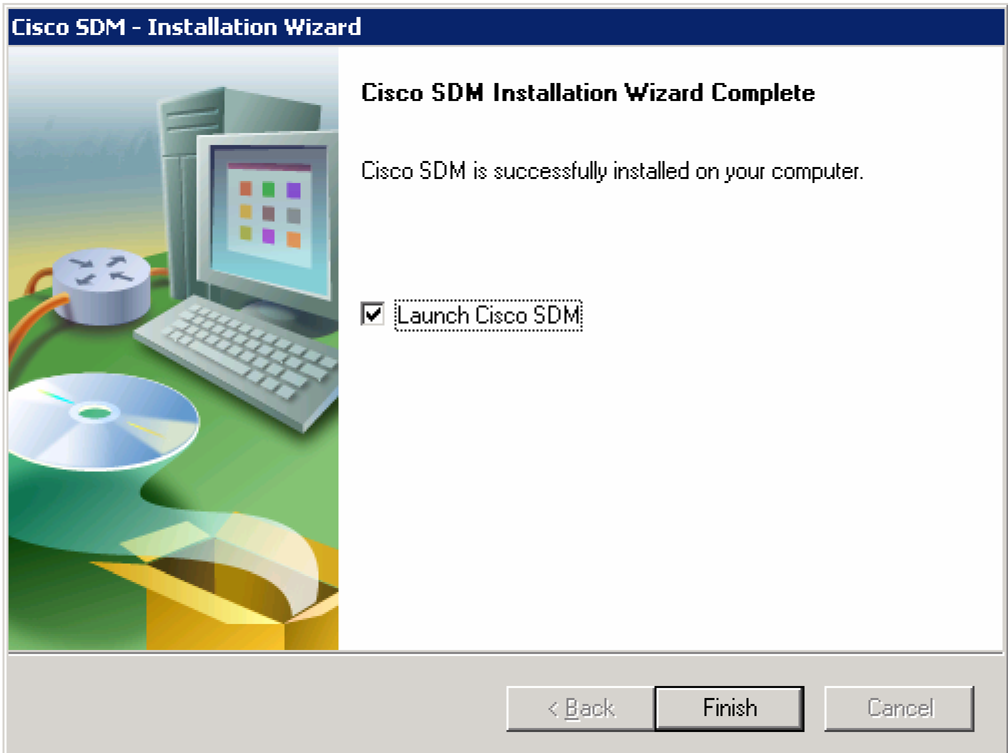


Figure 5-7: Final Installation Wizard Report

Step 6: Run SDM from the PC

SDM should start up from the installer when you have completed step 5 if you checked the Launch Cisco SDM option. If you did not, or you are running SDM without just installing it, click the icon on the desktop labeled **Cisco SDM**. The SDM Launcher dialog box will open. Type the IP address of the router shown in the diagram as a Device IP Address. Check **This device has HTTPS enabled and I want to use it** if you enabled the HTTP secure server in step 2. Then click the **Launch** button.

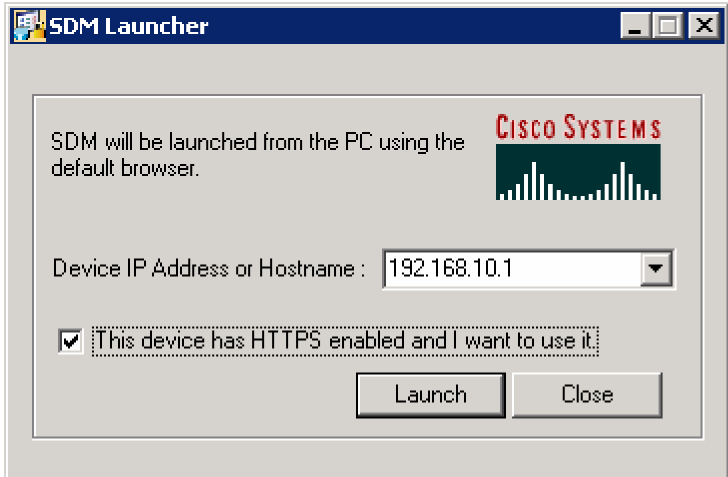


Figure 6-1: SDM Launcher Window

Click **Yes** when the security warning appears. Note that Internet Explorer may block SDM at first, and you will need to allow it or adjust your Internet Explorer security settings accordingly to use it. Depending on the version of Internet Explorer you are running, one of these settings is especially important for running SDM locally, and it is on the Tools menu, under Internet Options.... Click the **Advanced** tab, and under the Security heading, check **Allow active content to be run in files on My Computer** if it is not already checked.

Enter in the username and password you created in step 2.

Figure 6-2: HTTP Authentication Screen

You may be prompted to accept a certificate from this router. Accept the certificate to proceed. After this, give the username and password for the router and click **Yes**.

Figure 6-3: Internet Explorer Security Alert Prompt

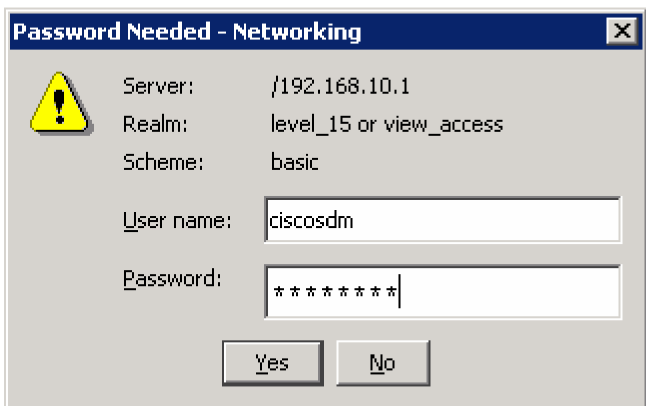


Figure 6-4: SDM Authentication Dialog

SDM reads the configuration off the router.

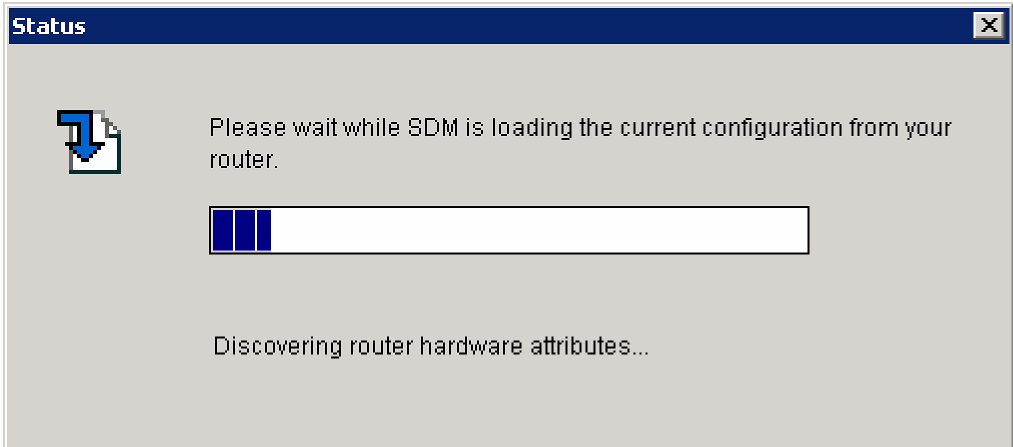


Figure 6-5: SDM Load Progress Indicator

If everything was configured correctly in step 2, you will be able to access the SDM dashboard. If your configuration here looks correct, it means you have successfully configured and connected to SDM. Your information may vary depending upon which version of SDM you are running.

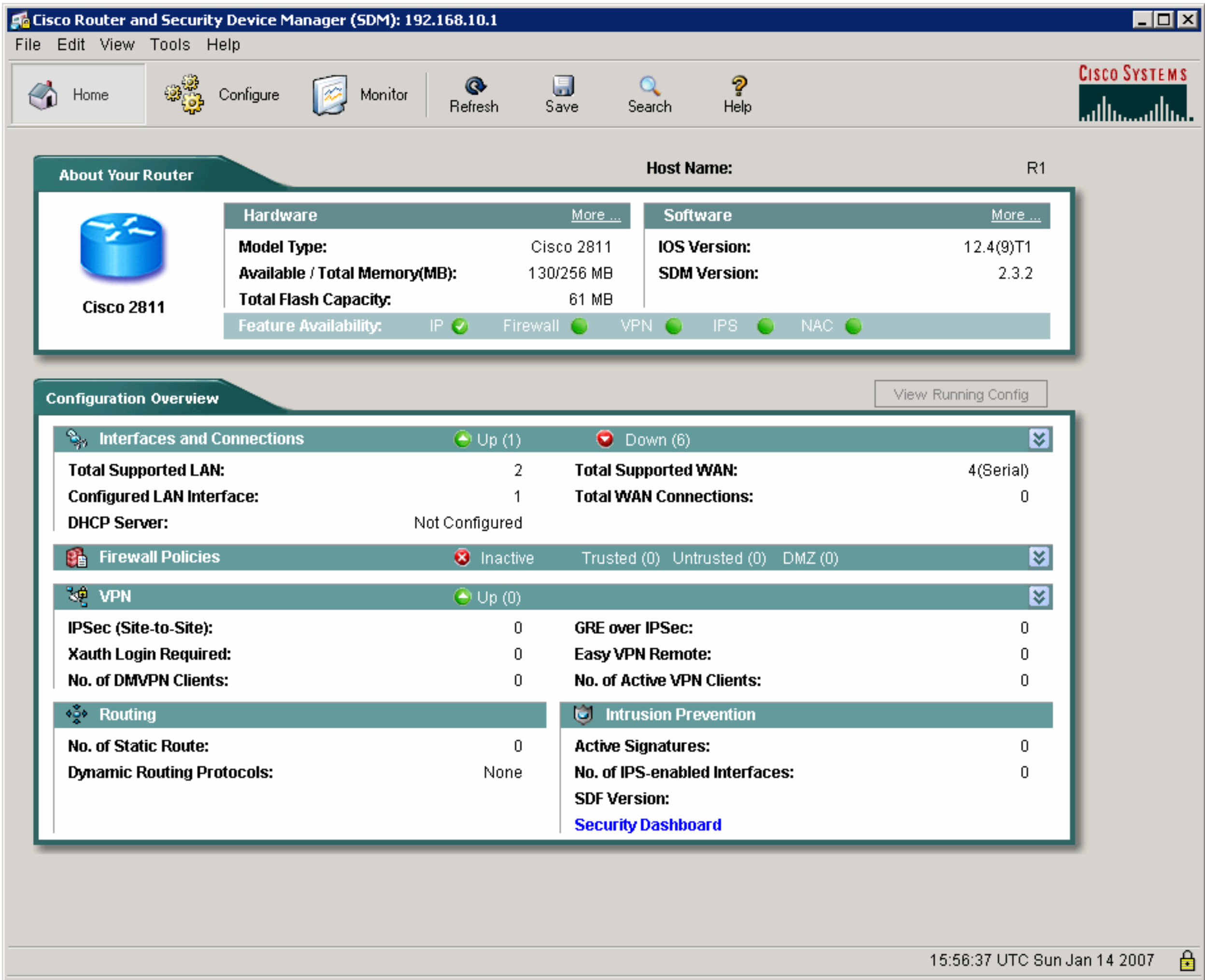


Figure 6-6: SDM Dashboard

Step 7: Install SDM to the Router

Follow step 6 until the prompt shown in the following figure appears.. When this window appears, click **Cisco Router** to install SDM to your router's flash memory. If you don't want to install SDM to your router's flash memory, or do not have the available space on the flash drive, then do not attempt to install SDM to the router.

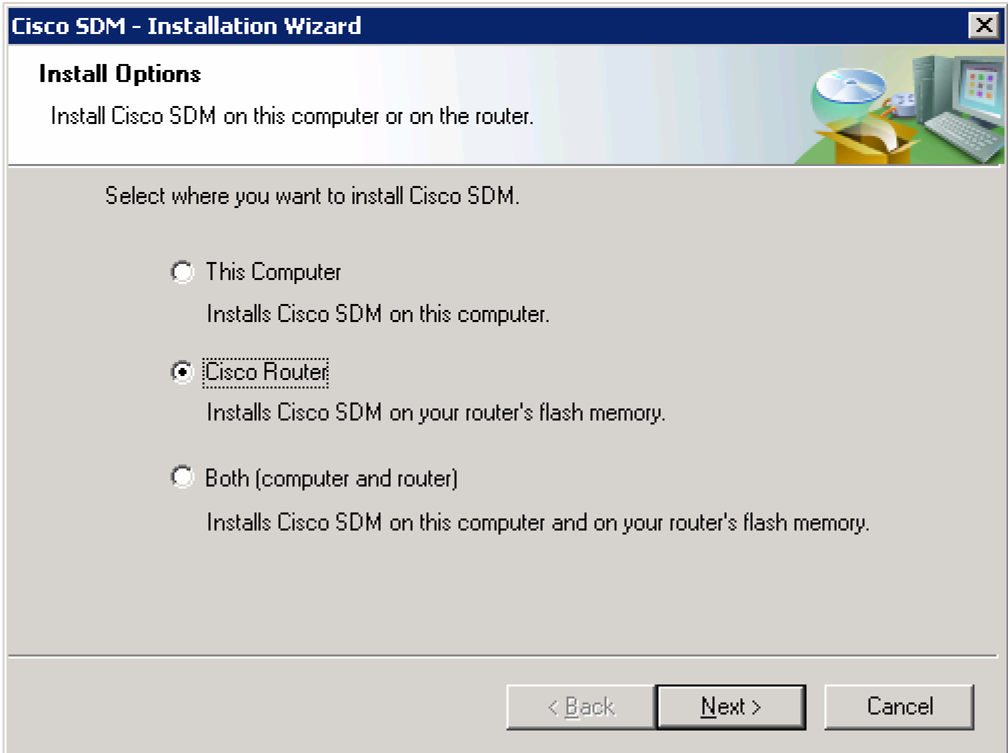


Figure 7-1: Installation Location Options

Enter your router's information so that the installer can remotely access and install SDM to the router.

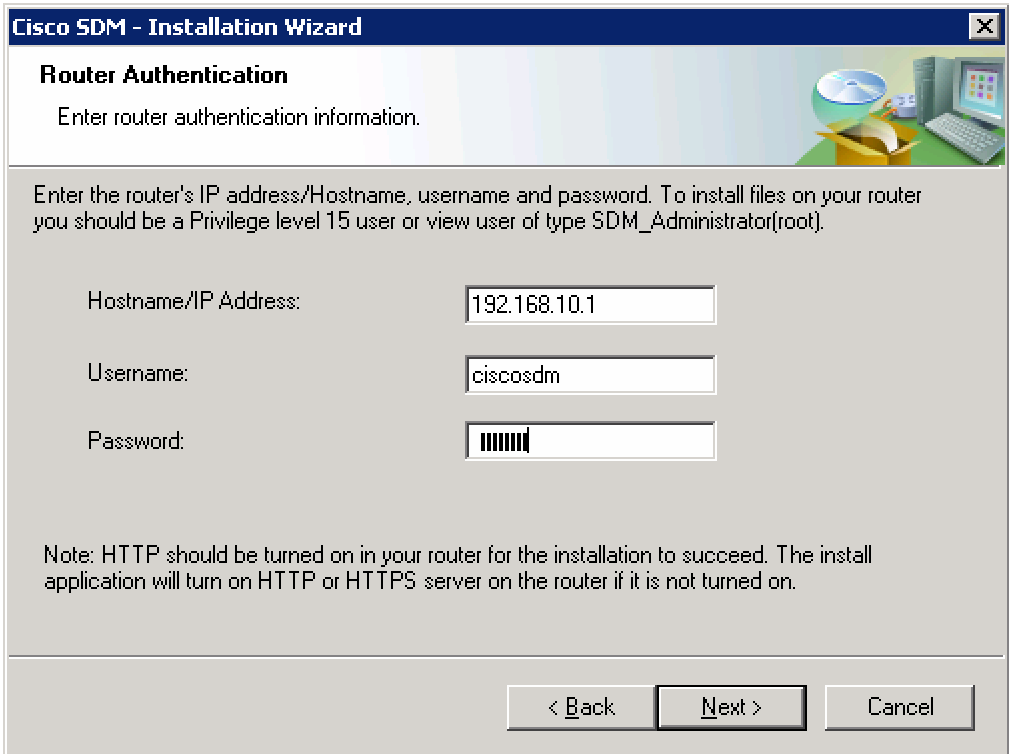


Figure 7-2: Router Authentication Dialog

Cisco SDM connects to the router. You may notice some messages being logged to the console. This is normal.

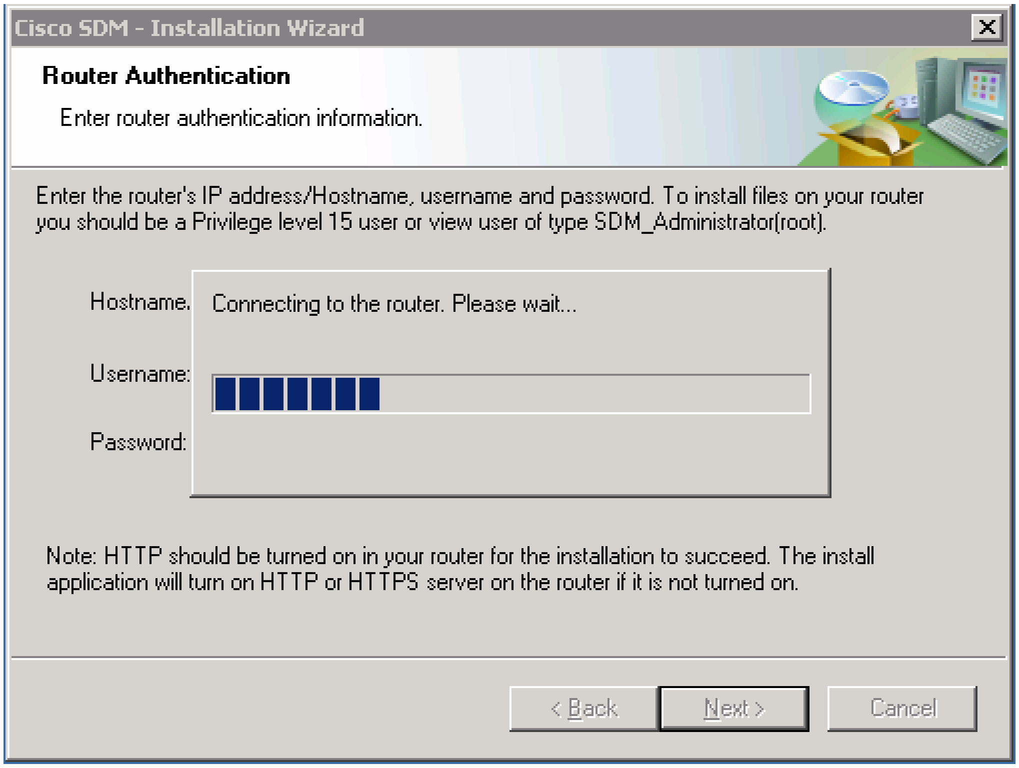


Figure 7-3: Router Connection Indicator

```
Jan 14 16:15:26.367: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:15:30.943: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:15:36.227: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:15:39.211: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:15:44.583: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
```

As shown in the following figure, choose **Typical** as your installation type, and then click **Next**.

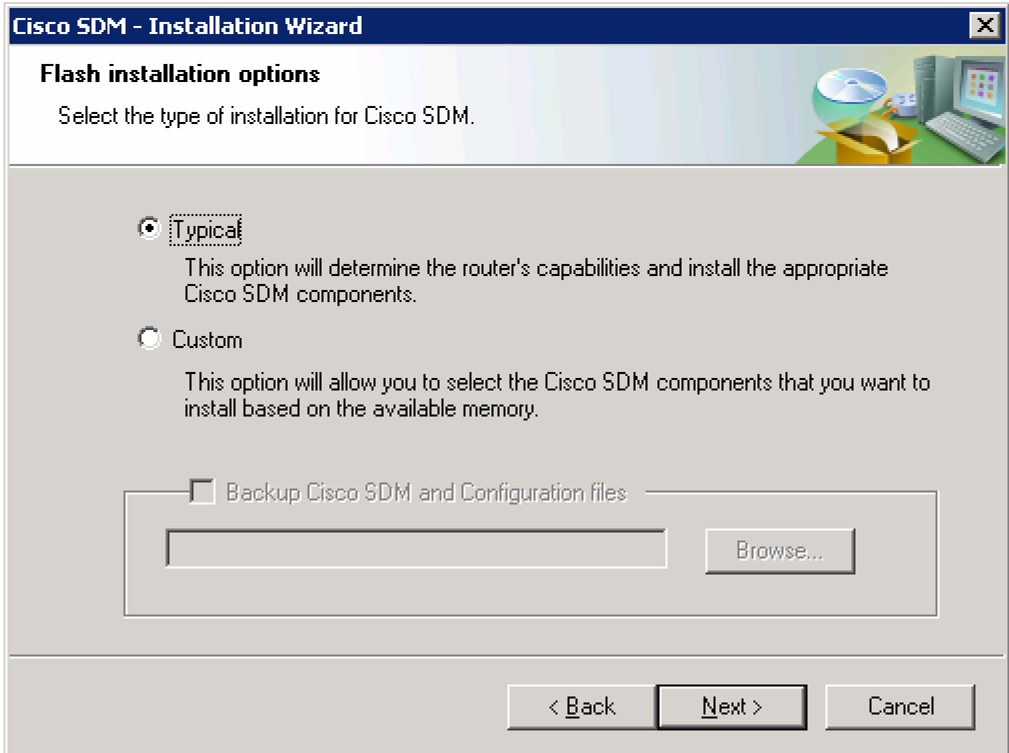


Figure 7-4: SDM Installation Options, Step 1

Leave the default installation options checked and click **Next**.

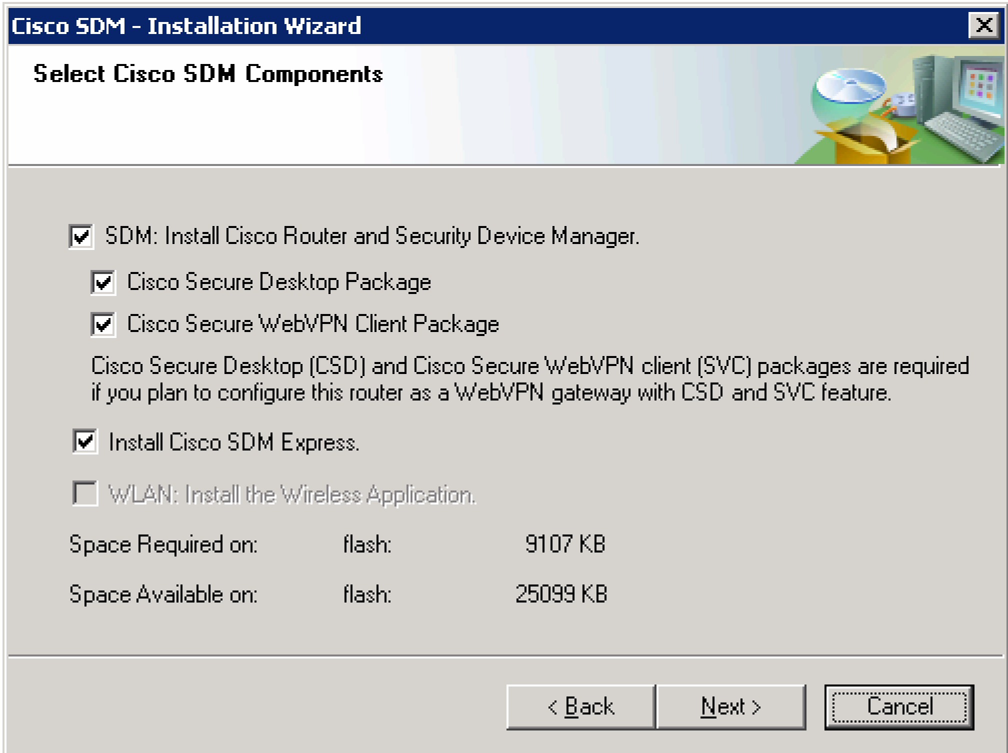


Figure 7-5: SDM Installation Options, Step 2

Finally, click **Install** for the installation process to begin. During the installation, more messages may be logged to the console. This installation process takes a little while (look at the timestamps in the console output below to estimate the duration on a Cisco 2811). The time will vary by router model.

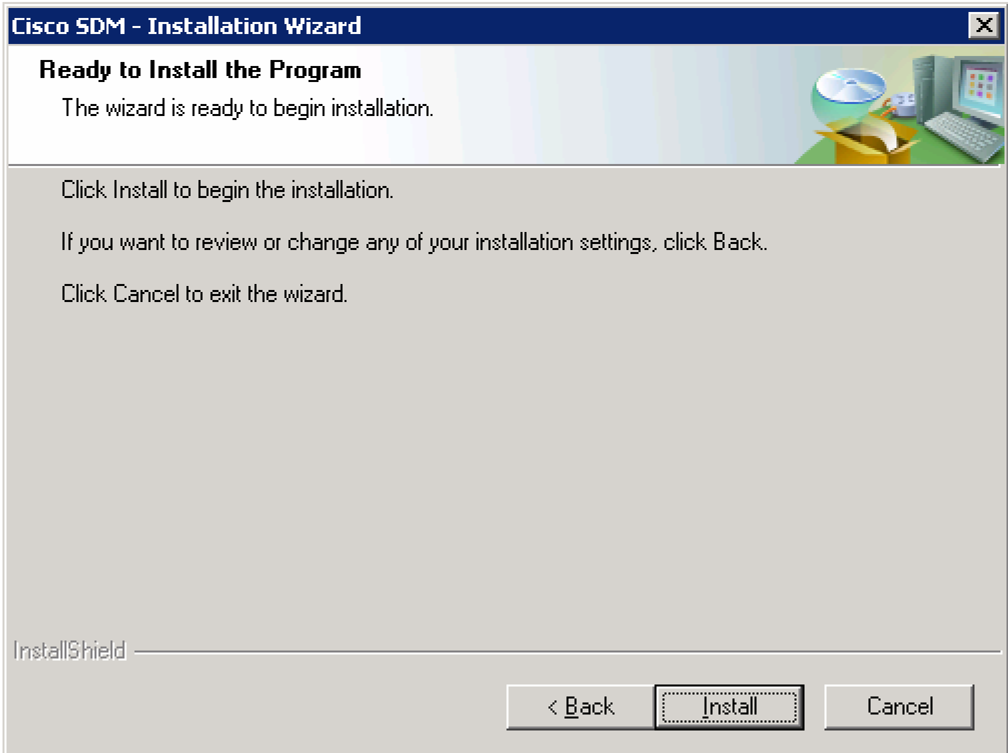


Figure 7-6: Confirmation Prompt

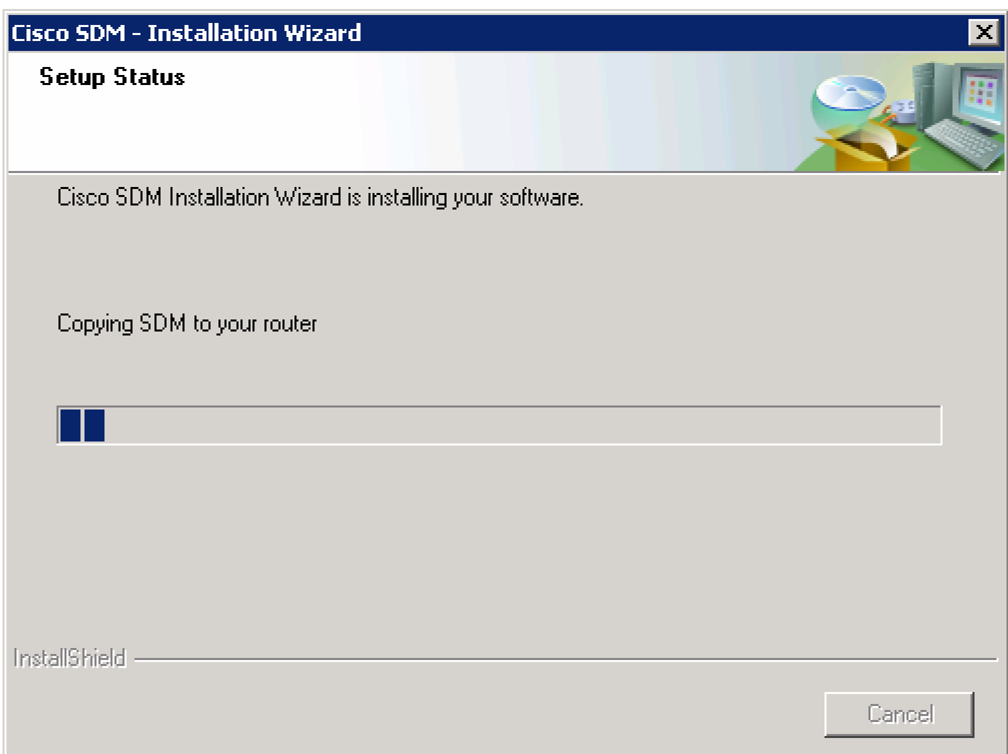


Figure 7-7: Installation Progress Indicator


```
Jan 14 16:19:40.795: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:19:43.855: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:19:49.483: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:25:57.823: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:26:02.331: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:27:42.279: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:27:46.767: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:28:11.403: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:28:15.795: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
Jan 14 16:29:04.391: %SYS-5-CONFIG_I: Configured from console by ciscosdm on
vty0 (192.168.10.50)
```

At the end of the installation, you are prompted to launch SDM on the router. Before you do this, go onto the console and issue the **show flash:** command. Notice all the files that SDM installed to flash. Before the installation, the only file listed was the first file, the IOS image.

```
R1# show flash:

CompactFlash directory:
File  Length   Name/status
  1   38523272 c2800nm-advipservicesk9-mz.124-9.T1.bin
  2   1038     home.shtml
  3   1823     sdmconfig-2811.cfg
  4   102400   home.tar
  5   491213   128MB.sdf
  6   1053184  common.tar
  7   4753408  sdm.tar
  8   1684577  securedesktop-ios-3.1.1.27-k9.pkg
  9   398305   sslclient-win-1.1.0.154.pkg
 10   839680   es.tar
[47849552 bytes used, 16375724 available, 64225276 total]
62720K bytes of ATA CompactFlash (Read/Write)
```

As shown in the following figure, make sure that the **Launch Cisco SDM** option is checked, and then click the **Finish** button to launch SDM.

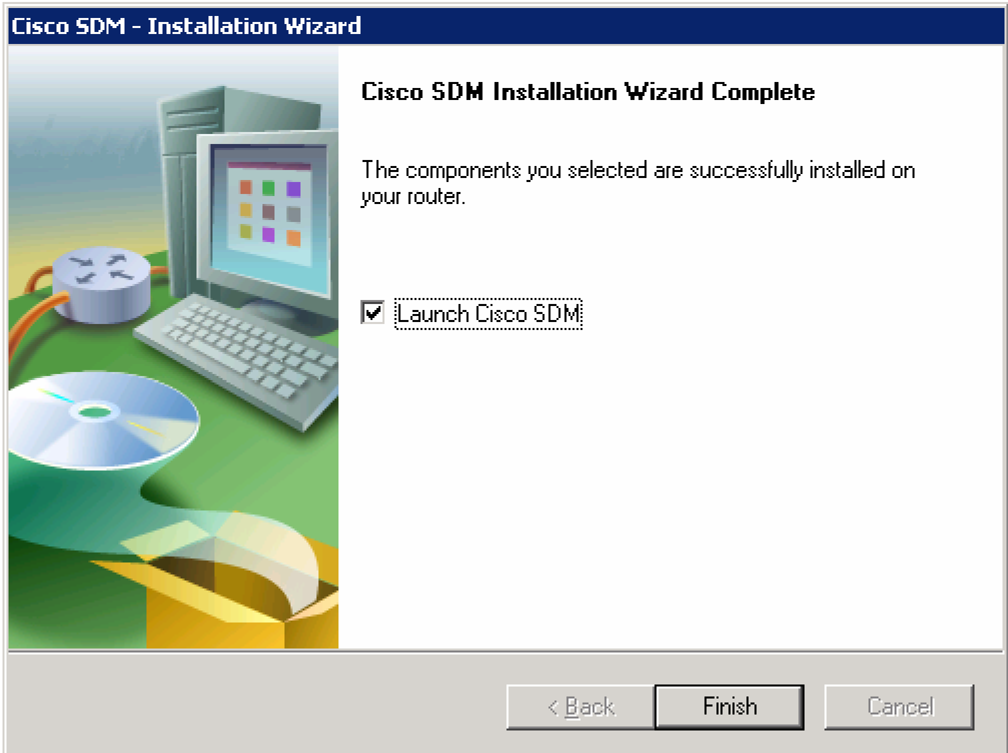


Figure 7-8: Final SDM Installation Dialog

Step 8: Run SDM from the Router

SDM should start up from the installer when you have completed the previous step if you checked the **Launch Cisco SDM** option. If you did not, or you are running SDM without installing it, open up Internet Explorer and navigate to the URL “https://<IP address>/” or “http://<IP address>/” depending on whether you enabled the HTTP secure server in step 2. When you are prompted to accept the certificate, click **Yes**.

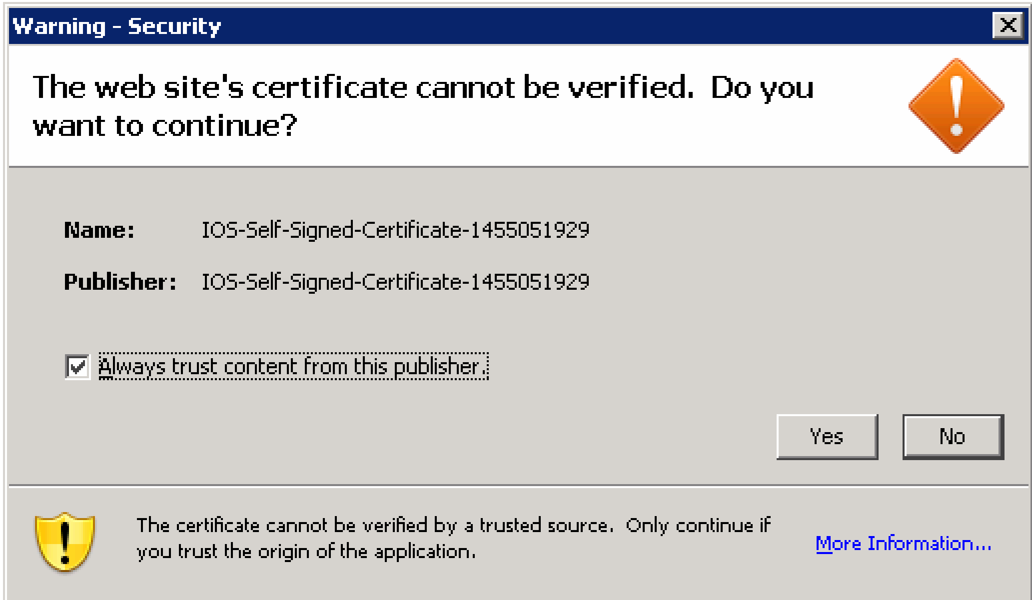


Figure 8-1: Internet Explorer Certificate Confirmation

Ignore the security warnings and click **Run**.

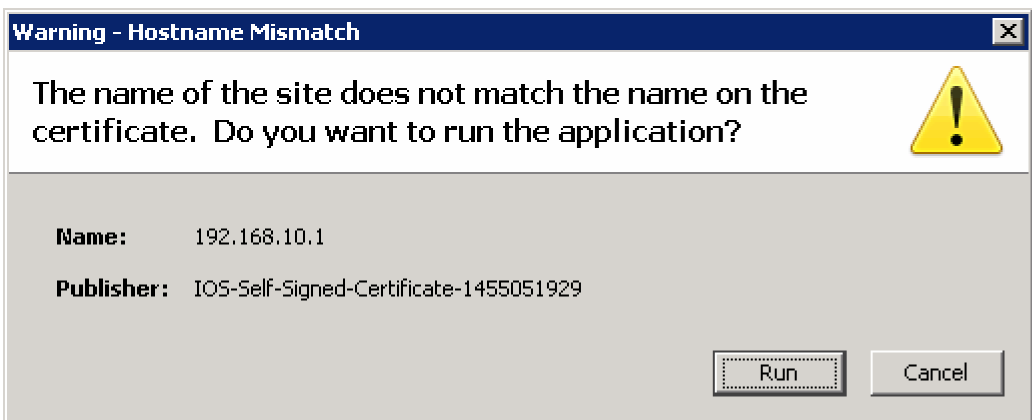


Figure 8-2: Internet Explorer Security Confirmation

Enter in the username and password you configured in step 2.

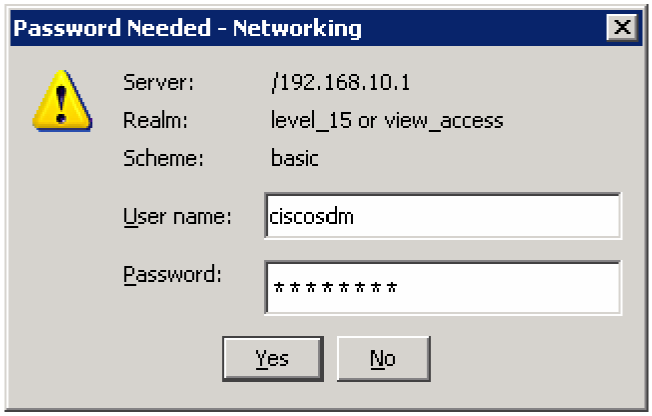


Figure 8-3: SDM Authentication Dialog

SDM will read the configuration off the router.

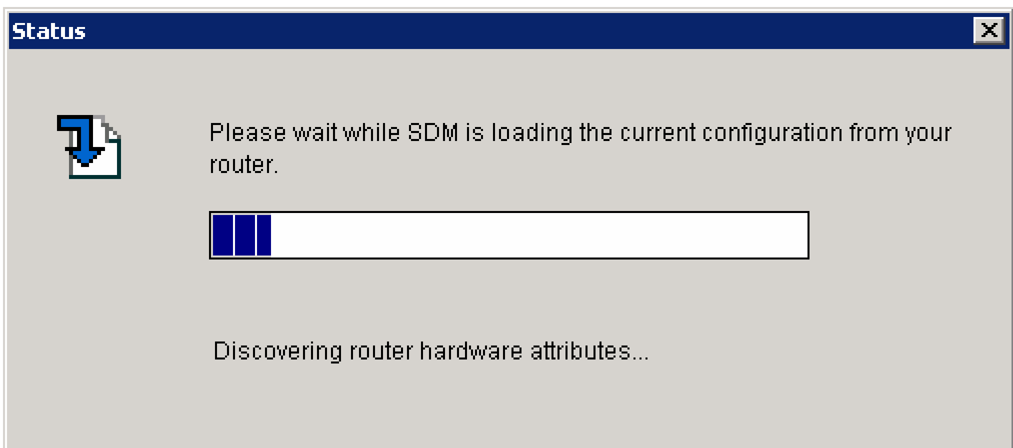


Figure 8-4: SDM Load Progress Indicator

Once SDM is finished loading the current configuration of your router, the SDM homepage appears. If your configuration here looks correct, it means you have successfully configured and connected to SDM. What you see may differ from what appears in the following figure depending upon router model number, IOS version, and so forth.

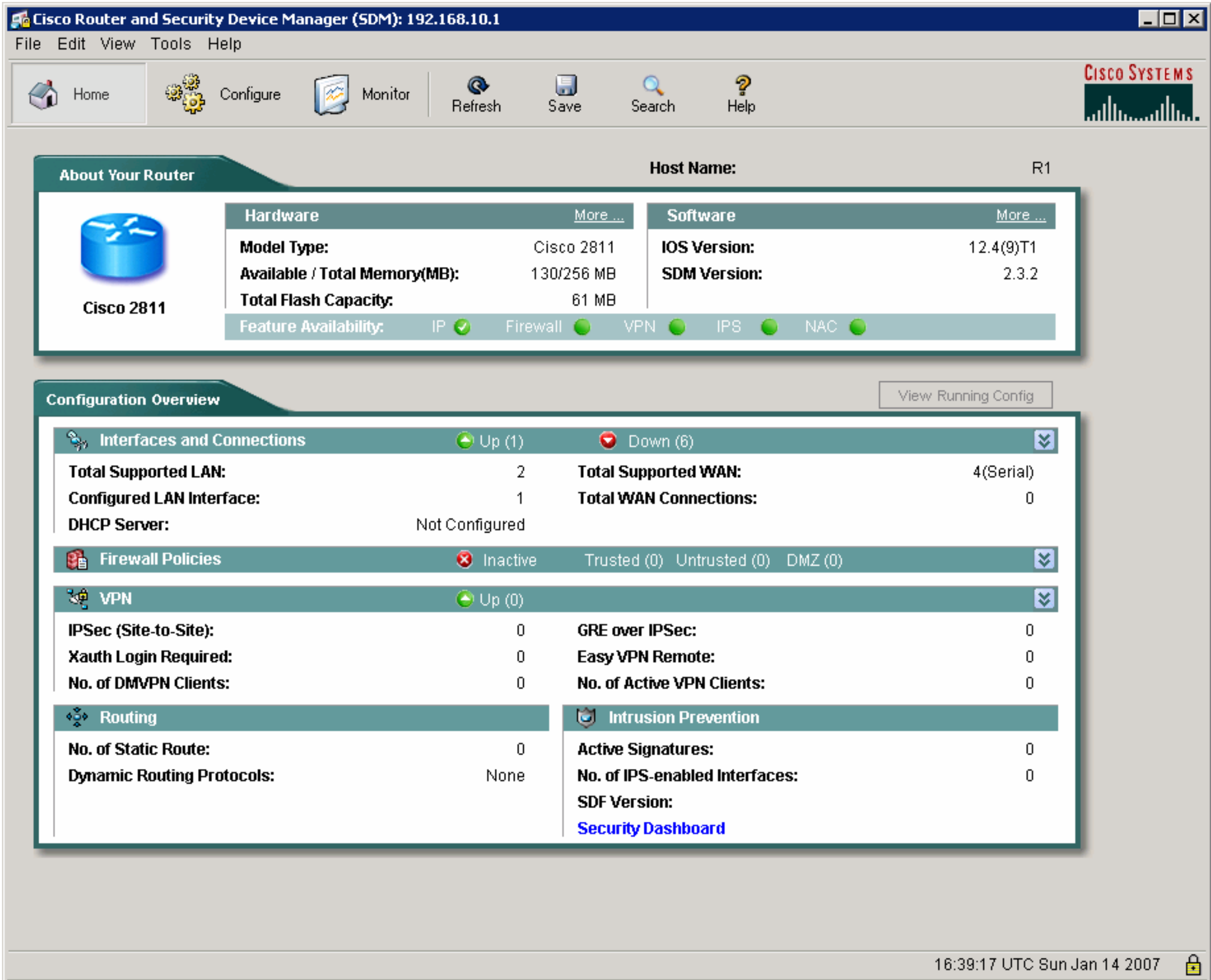


Figure 8-5: SDM Dashboard

Step 9: Monitor an Interface in SDM

In SDM, you can look at an interface to verify that SDM is working and communicating with the router properly. To do this, click the **Monitor** tab at the top, and then click **Interface Status** on the left sidebar. You should see the graphs start to populate when FastEthernet0/0 is selected.

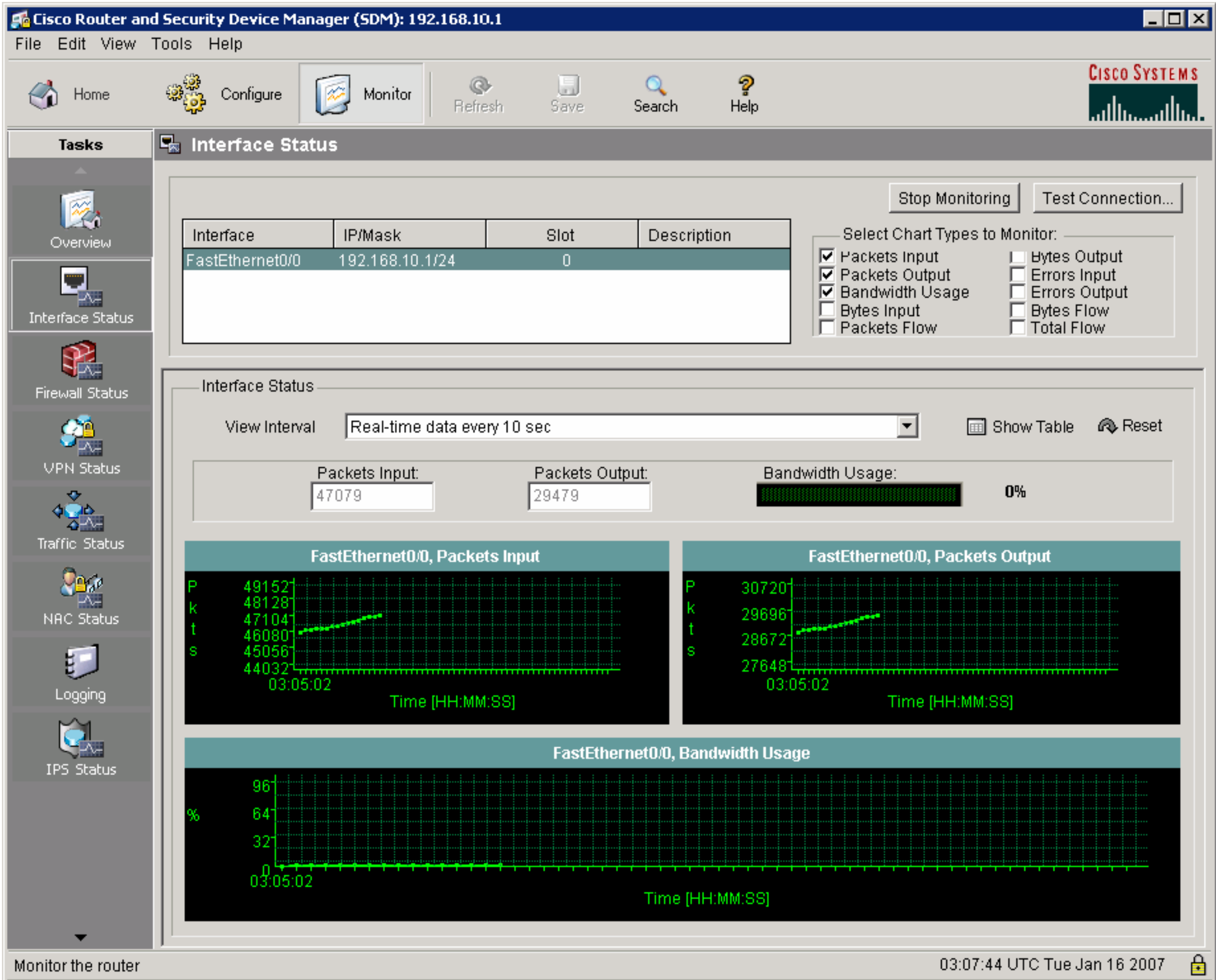


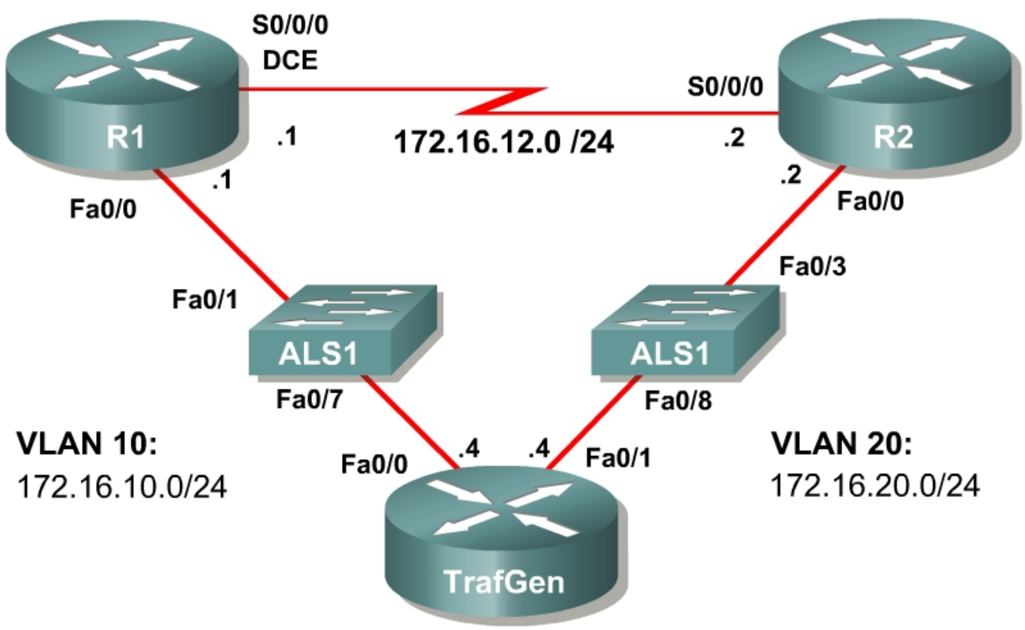
Figure 9-1: SDM Dashboard

Lab 3.3 Configuring QoS with SDM

Learning Objectives

- Configure Quality of Service tools with the SDM QoS wizard
- Monitor traffic patterns using the SDM QoS interface

Topology Diagram



Scenario

Cisco Security Device Manager employs a basic Quality of Service (QoS) configuration wizard that can be used to apply some basic QoS tools to a router's interfaces.

Normally, you would configure and deploy QoS tools on the command-line interface (CLI) without the benefit of a graphical user interface (GUI). However, SDM's QoS wizard provides a useful introduction to QoS tools. Thus, we begin our exploration of QoS tools using the SDM GUI.

Preparation

This lab uses the Basic Pagent Configuration for TrafGen and the Switch to generate and facilitate lab traffic in a stream from TrafGen to R1 to R2. Prior to beginning this lab, configure TrafGen (R4) and the switch according to the Basic Pagent Configuration in Lab 3.1: Preparing for QoS. You may simply accomplish this on R4 by loading the *basic-ios.cfg* file from Flash memory into the NVRAM, and reloading.

```
TrafGen# copy flash:basic-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

Next, instruct TGN to load the *basic-tgn.cfg* file and to start generating traffic.

```
TrafGen> enable
TrafGen# tgn load-config flash:basic-tgn.cfg
TrafGen# tgn start
```

On the switch, load the *basic.cfg* file into NVRAM and reload the device.

```
ALS1# copy flash:basic.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
ALS1# reload
Proceed with reload? [confirm]
```

In addition, add the Fast Ethernet 0/3 interface on the switch to VLAN 20 since R2 will be the exit point from the network topology in this lab.

```
ALS1# configure terminal
ALS1(config)# interface fastethernet 0/3
ALS1(config-if)# switchport access vlan 20
ALS1(config-if)# switchport mode access
```

Step 1: Configure Physical Interfaces

Configure all of the physical interfaces shown in the diagram. Set the clock rate on the serial link to 800Kbps, and use the **no shutdown** command on all interfaces.

```
R1(config)# interface fastethernet0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface serial0/0/0
R1(config-if)# ip address 172.16.12.1 255.255.255.0
R1(config-if)# clock rate 800000
R1(config-if)# no shutdown

R2(config)# interface fastethernet0/1
R2(config-if)# ip address 172.16.20.2 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# interface serial0/0/0
R2(config-if)# ip address 172.16.12.2 255.255.255.0
R2(config-if)# no shutdown
```


Step 2: Configure Routing with EIGRP

Configure R1 and R2 to participate in EIGRP AS 1. Disable automatic summarization and add the entire major 172.16.0.0 network.

```
R1(config)# router eigrp 1
R1(config-router)# no auto-summary
R1(config-router)# network 172.16.0.0

R2(config)# router eigrp 1
R2(config-router)# no auto-summary
R2(config-router)# network 172.16.0.0
```

Step 3: Connect to R1 using SDM

Set up a host using R1 as its default gateway. Set up R1 for SDM access and connect to it using the host. If you do not know how to set the IP address on a host or connect to a router using SDM, consult Lab 3.2: Installing SDM.

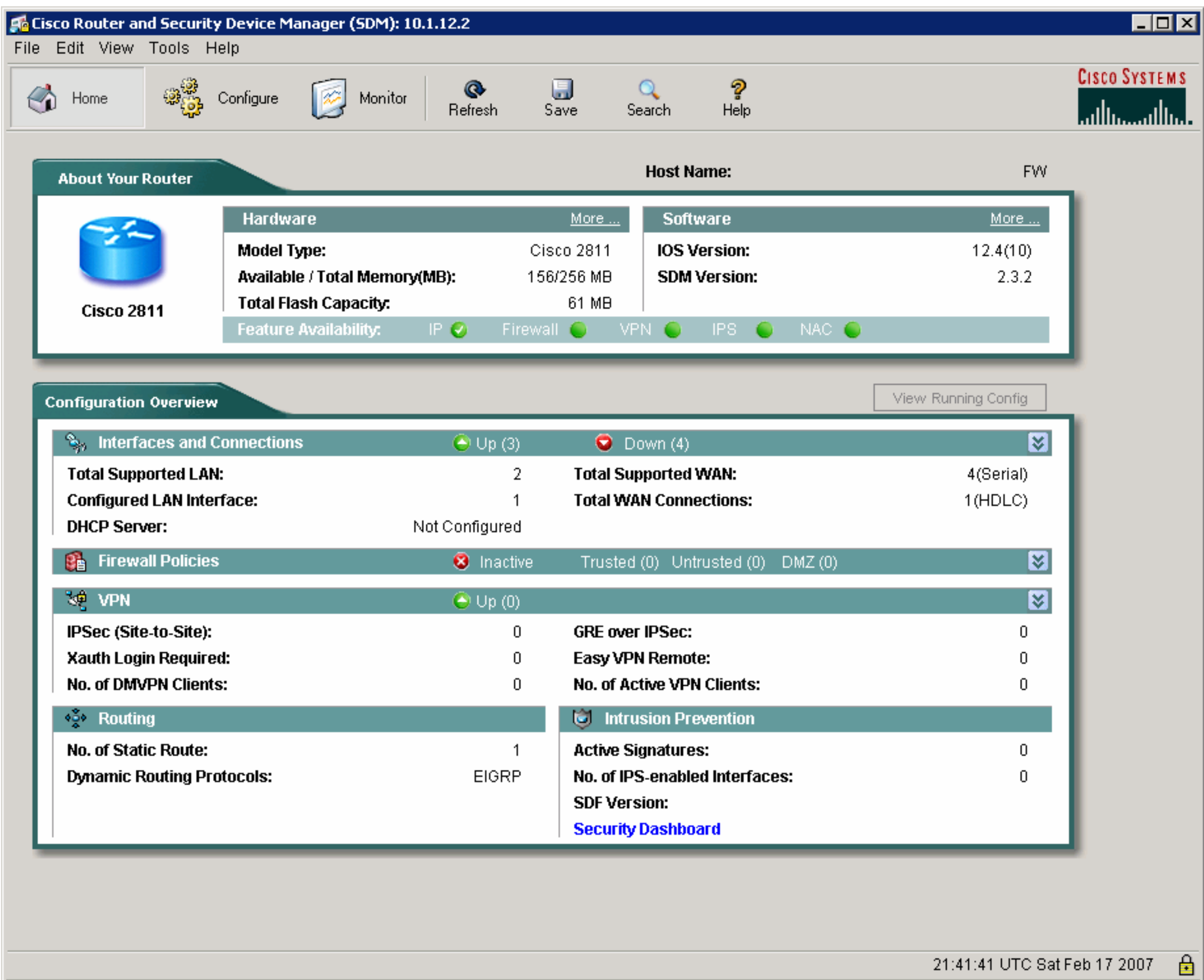


Figure 3-1: SDM Home Page

Choose **Edit > Preferences**. Make sure that **Preview commands before delivering to router** is checked, and then click **OK**. Now, you are able to preview exactly what configuration lines the SDM delivers to the router.

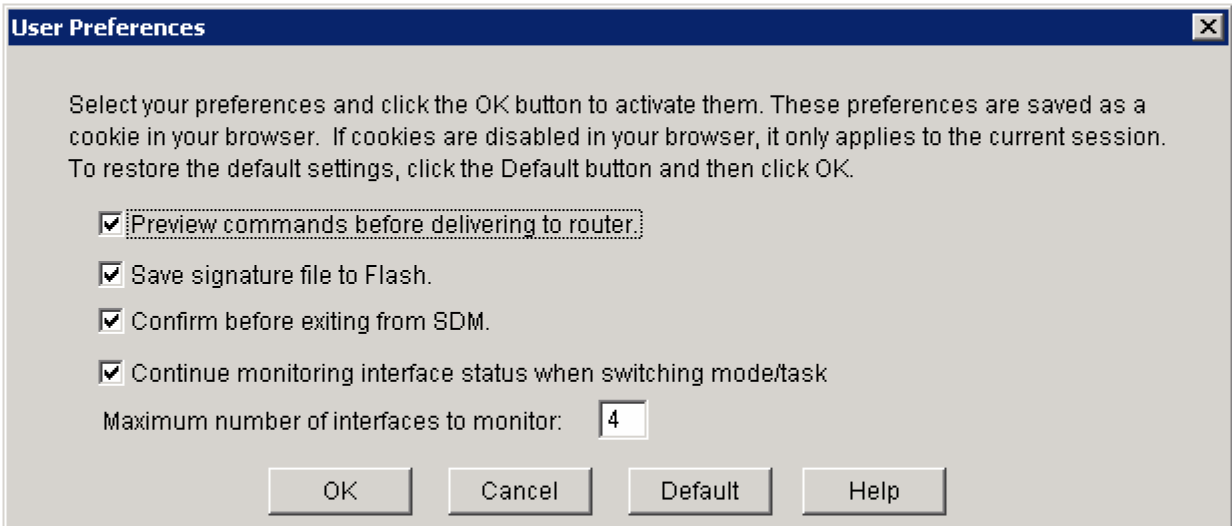


Figure 5-2: SDM User Preferences

Step 4: Use the SDM QoS Wizard

SDM facilitates the implementation of a class-based QoS policy on router interfaces. The QoS wizard uses Network-based Application Recognition (NBAR) to classify packets based on application protocol and implements bandwidth guarantees for each type of traffic.

To begin, click the **Configure** icon at the top of the SDM home page, and then choose **Quality of Service** in the Tasks sidebar. On the **Create QoS Policy** tab, click the **Launch QoS Wizard** button.

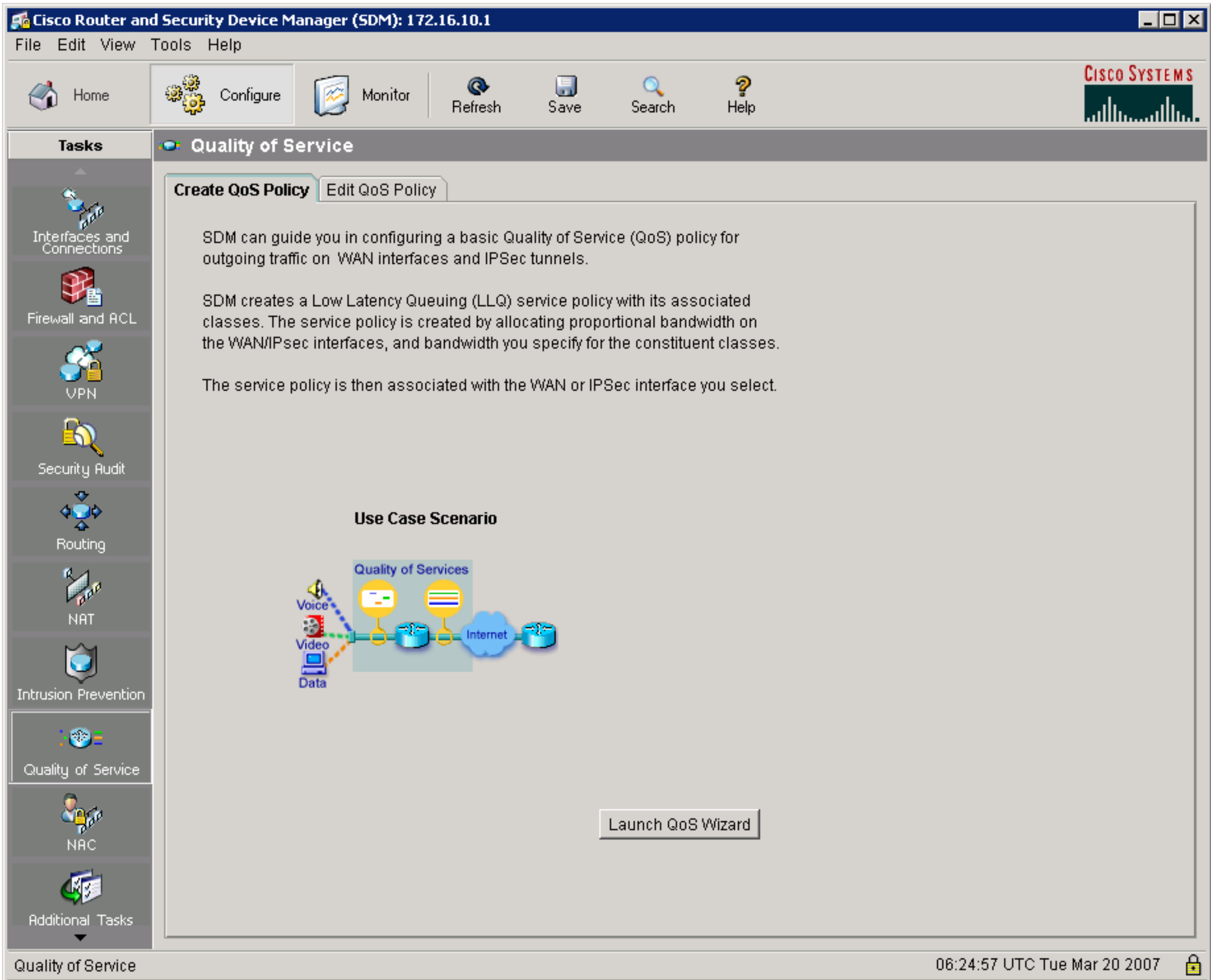


Figure 4-1: Create QoS Policy Tab

After reading the introduction to the SDM QOS Wizard, click the **Next** button.

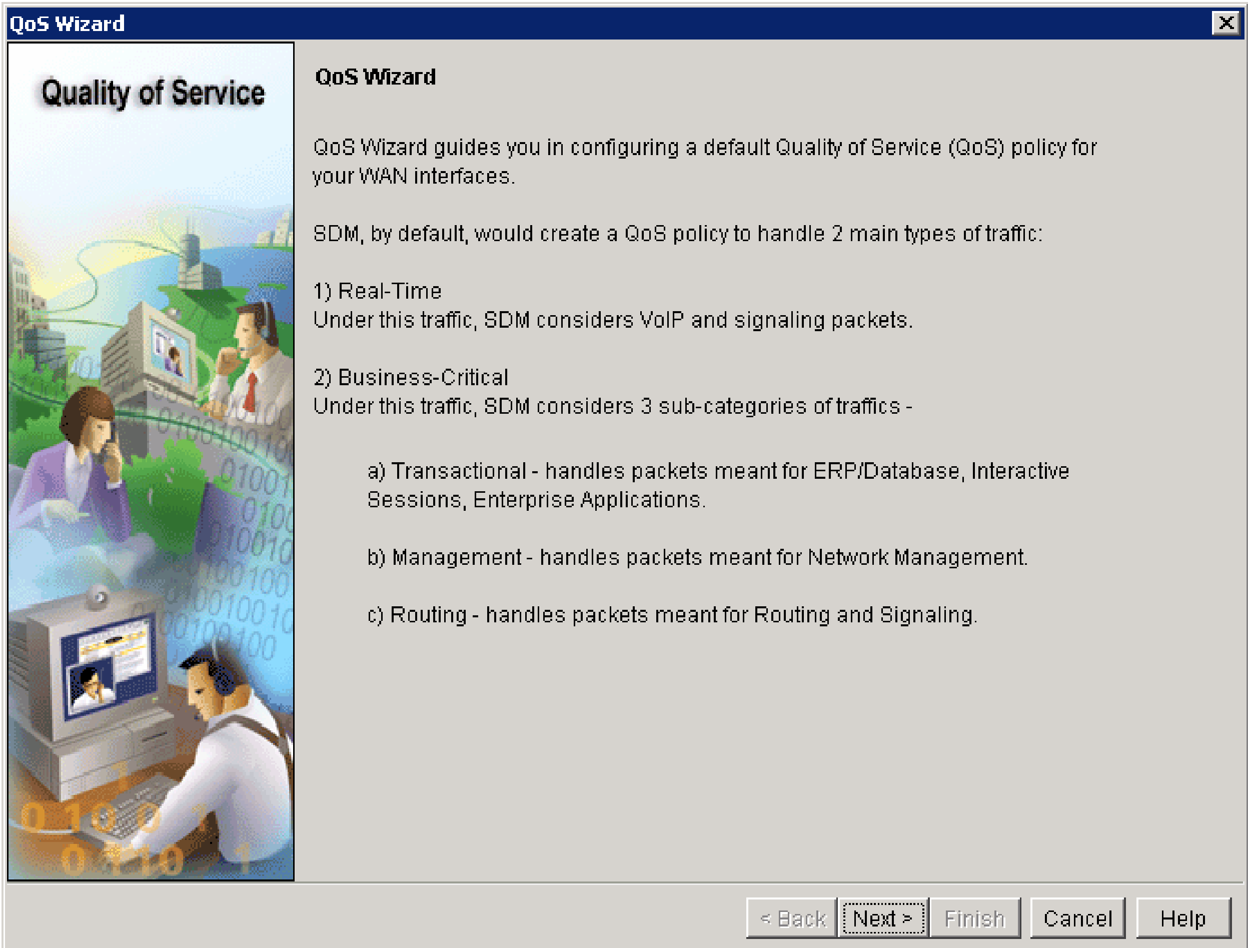


Figure 4-2: SDM QoS Wizard

Select the Serial 0/0/0 interface as the egress interface for QoS policy. This interface will be the egress interface at which packets generated by Pagent will create congestion.

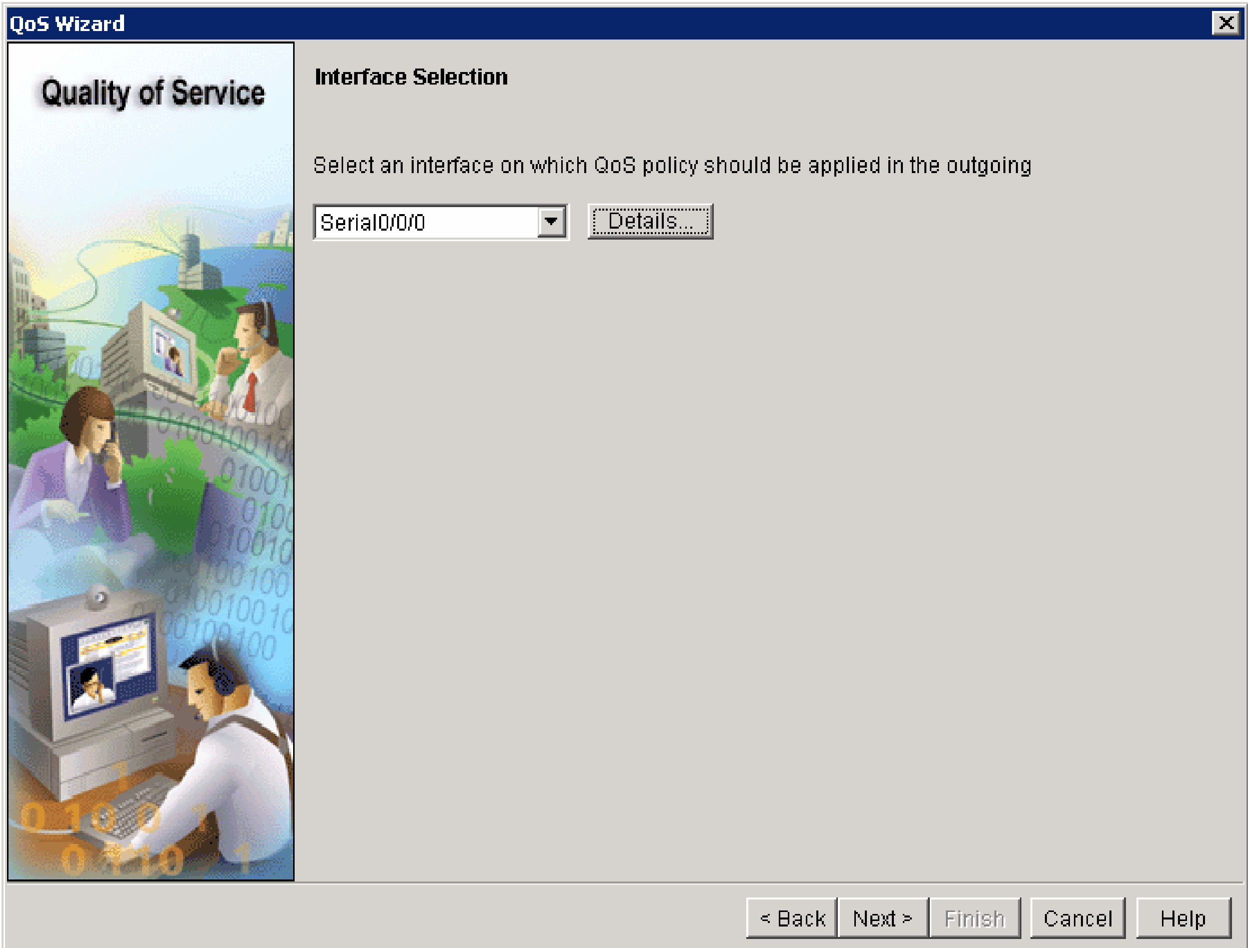


Figure 4-3: Egress Interface Selection for QoS Policy

Cisco routers automatically enable weighted fair queuing (WFQ) on low-speed serial interfaces. SDM displays a dialog box to prompt you to decide if you want to disable WFQ to replace it with another QoS policy on this interface. Click the **Yes** button in response to the dialog box.

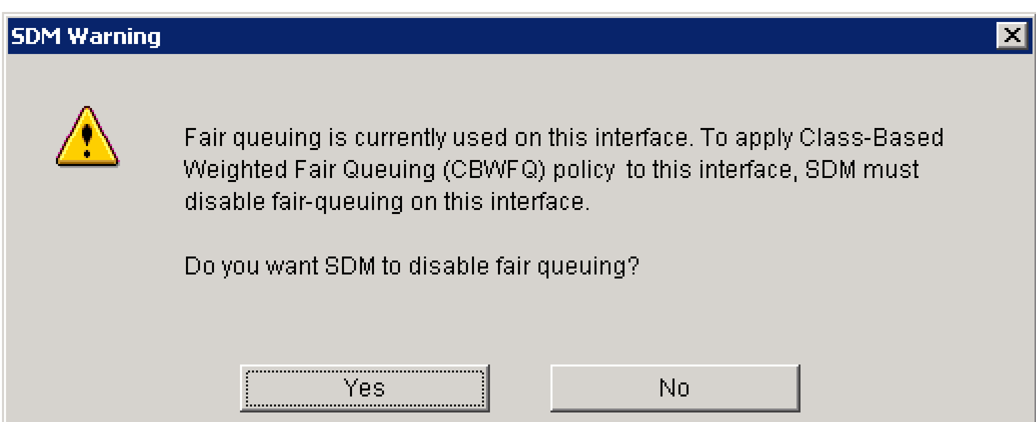


Figure 4-4: Disable Fair Queuing Dialog

Accept the default bandwidth percentage allocations and click **View Details…**.

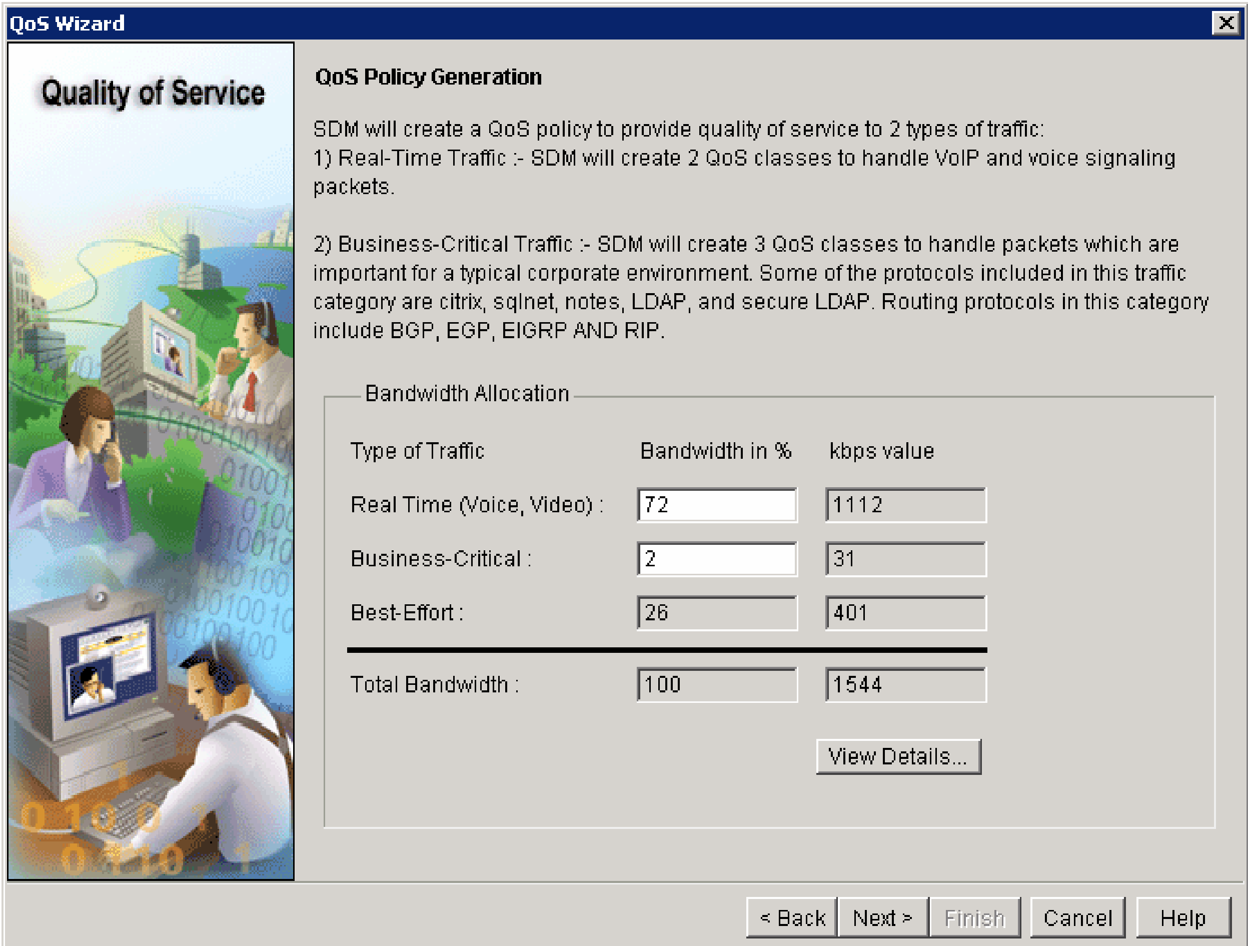


Figure 4-5: QoS Policy Configuration

SDM displays another dialog box to prompt you that it needs to enable NBAR on the interface to discover protocols. Click **Yes** in response to this dialog box. SDM may pause for a few moments.

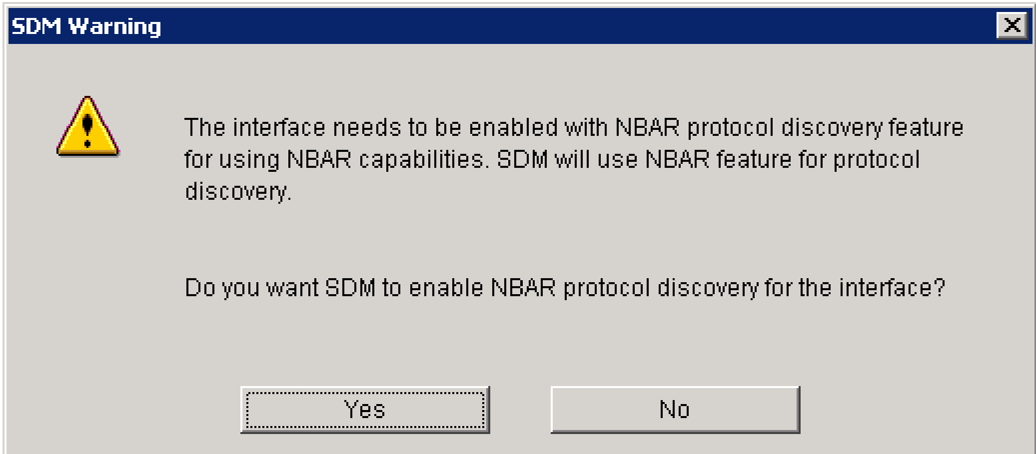


Figure 4-6: NBAR Confirmation

Verify the SDM classes for both tabs, and then click **Close**.

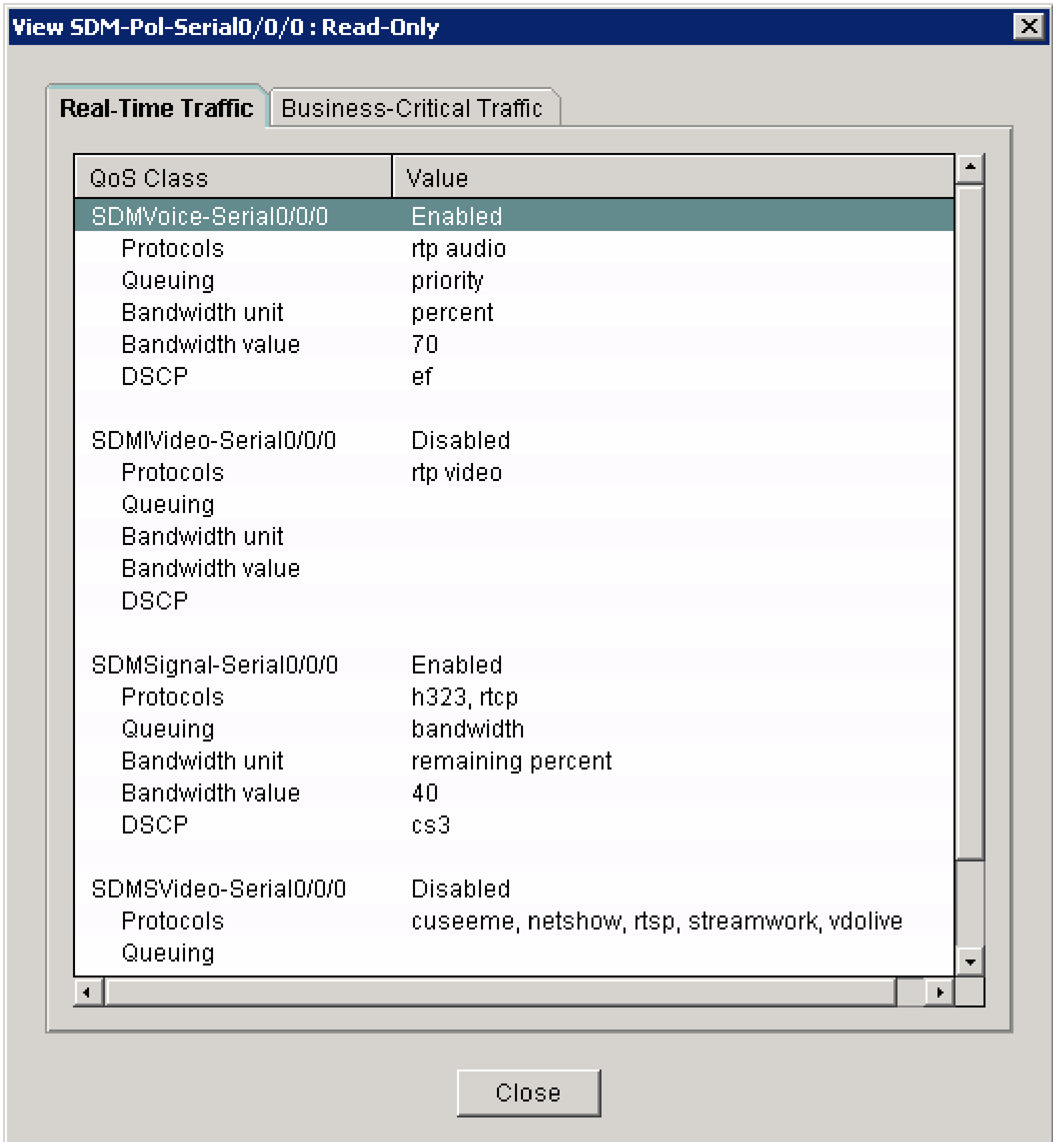


Figure 4-7: QoS Policy, Summarized by Interface

Click **Finish** once you have gone over the changes SDM will make.

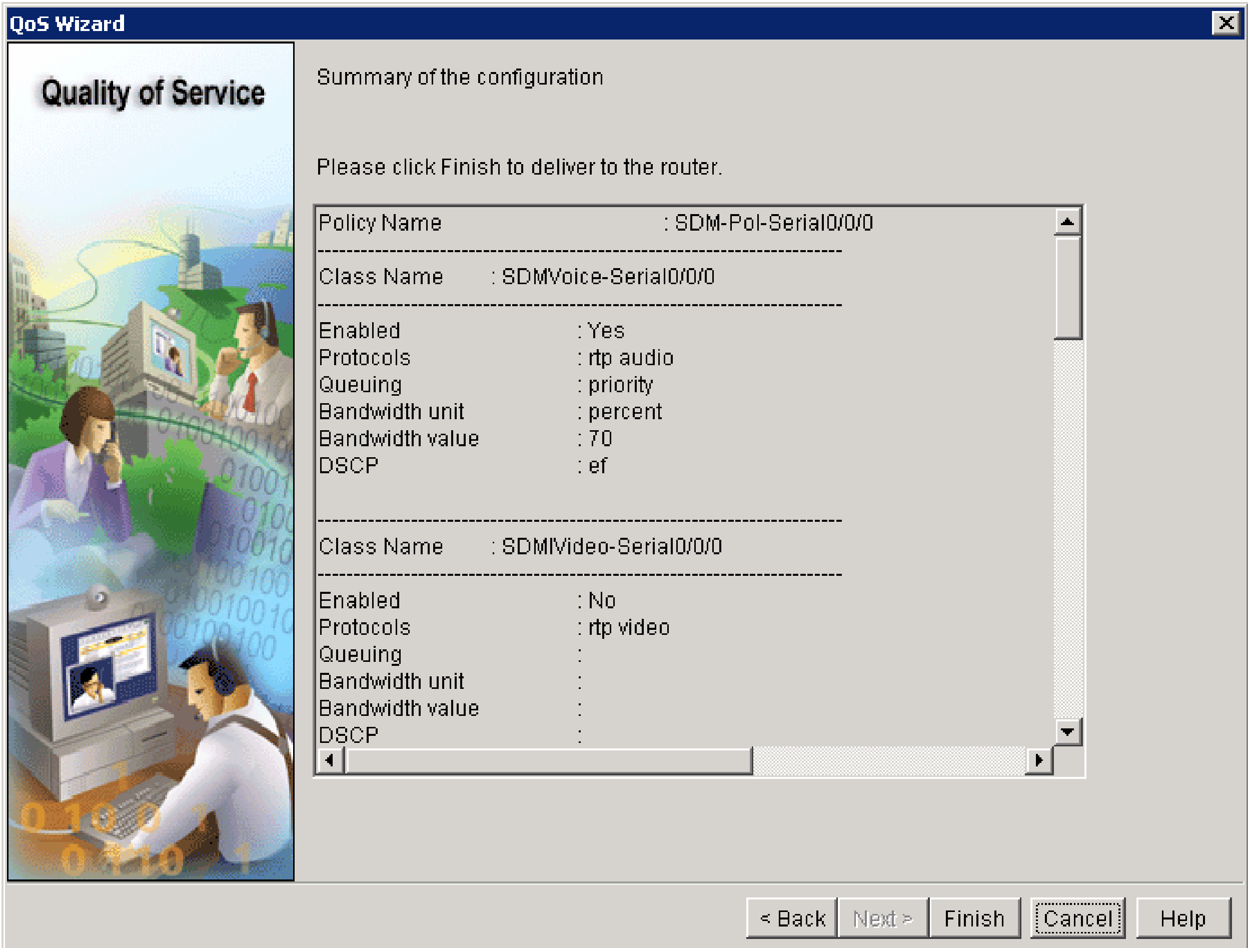


Figure 4-8: Configuration Summary

View the actual commands SDM will add to the configuration, and then click **Deliver**.

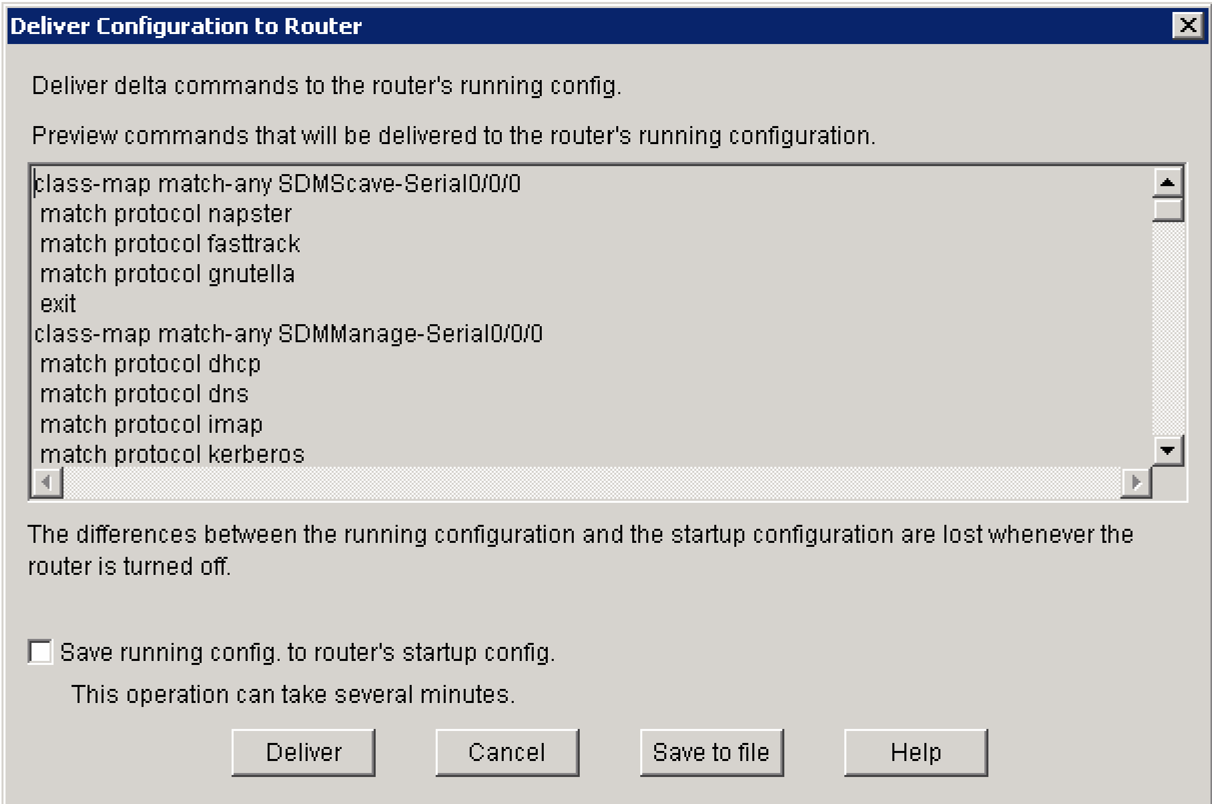


Figure 4-9: Command Delivery Notification

When the commands have been delivered, click **OK** to leave the wizard.

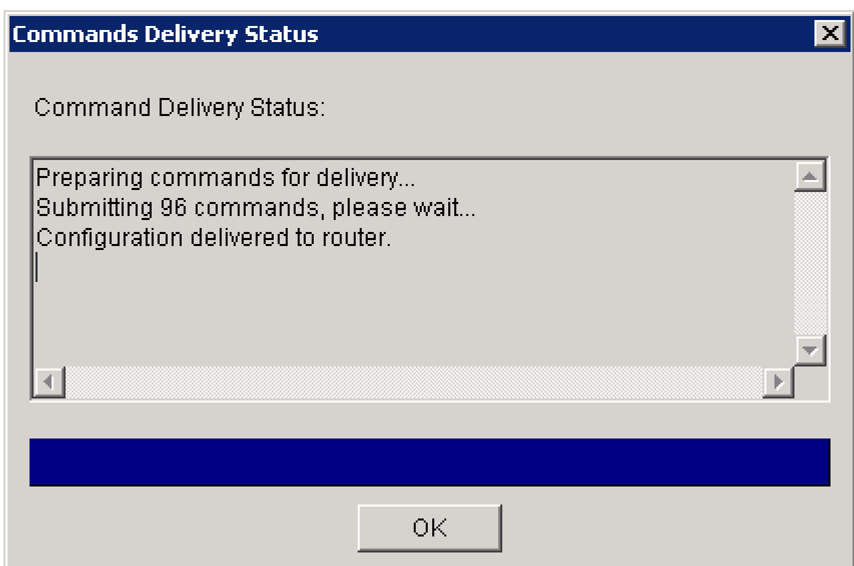


Figure 4-10: Command Delivery Progress Indicator

SDM brings you to the **Edit QoS Policy** tab.

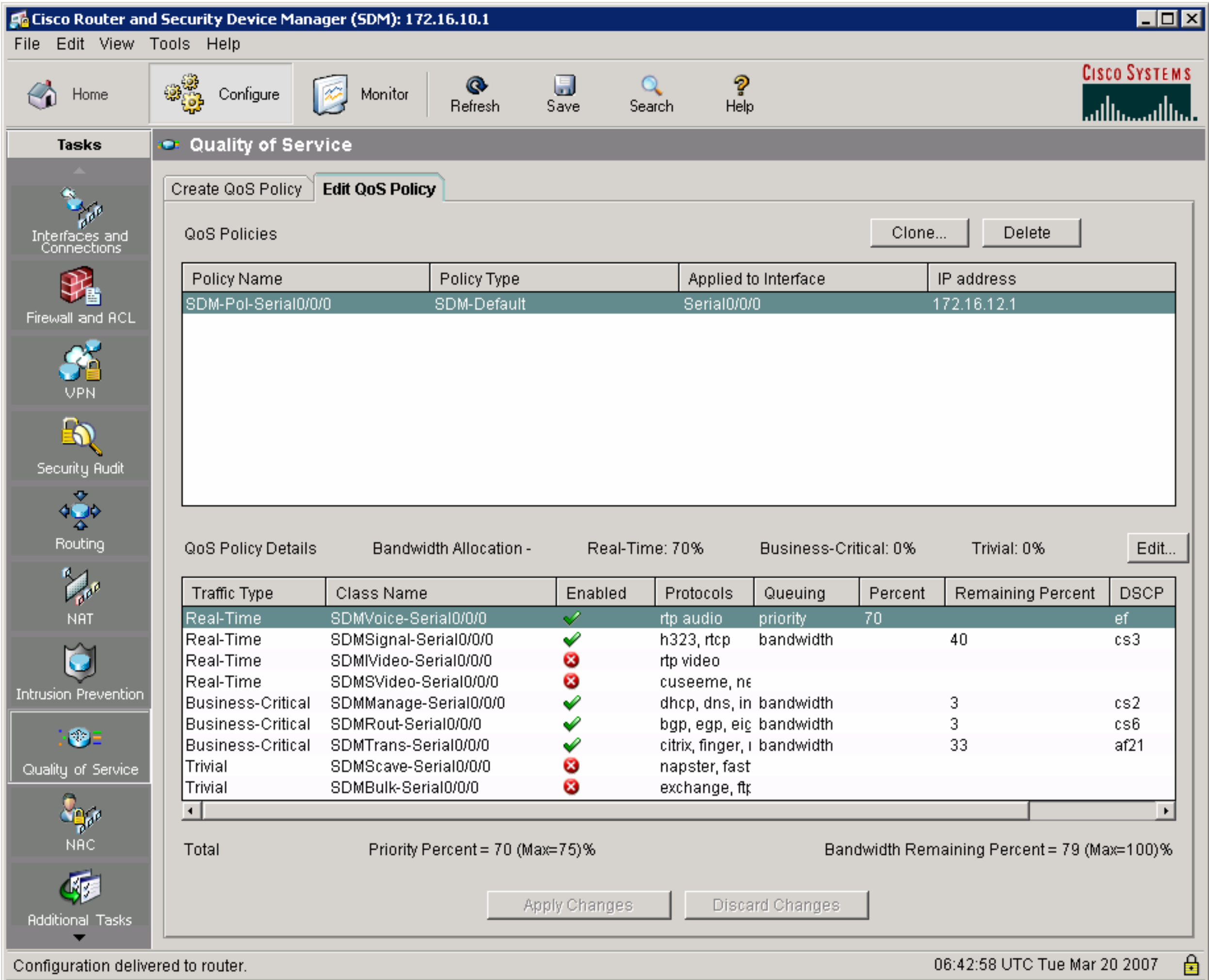


Figure 4-11: Edit QoS Policy Tab

Step 5: Verify QoS Operation with SDM

In SDM, click the **Monitor** icon at the toolbar at the top of the window. Choose **Traffic Status** on the Tasks sidebar, and then in the next pane, choose **QoS**. Clicking QoS will display some graphs and statistics that show how much bandwidth different traffic classes are using.

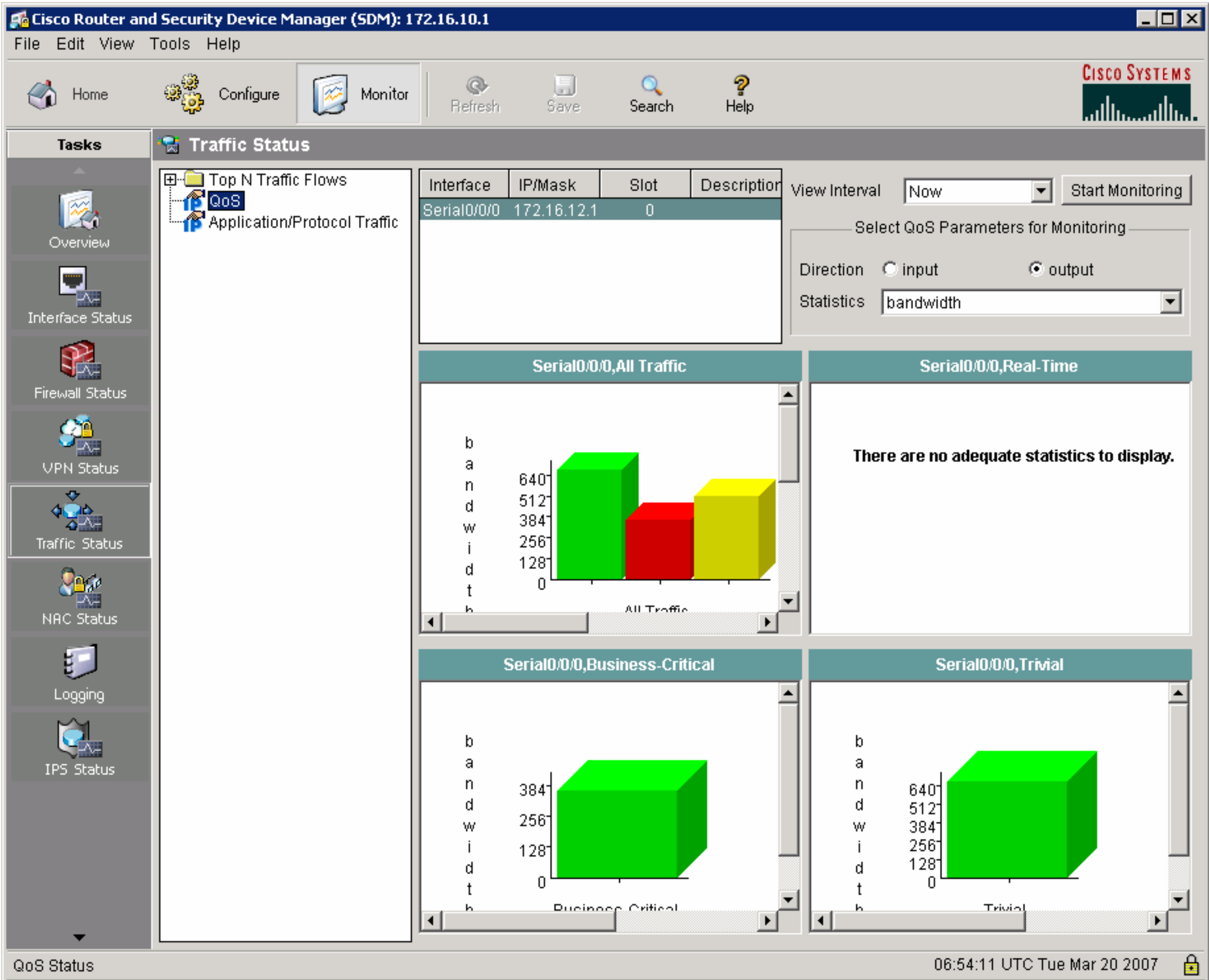


Figure 5-1: Interface Traffic Statistics

Choose **Application/Protocol Traffic** to see a graphical breakdown of different traffic types. Due to the TGN configuration, all traffic classes should be roughly equal in bandwidth usage.

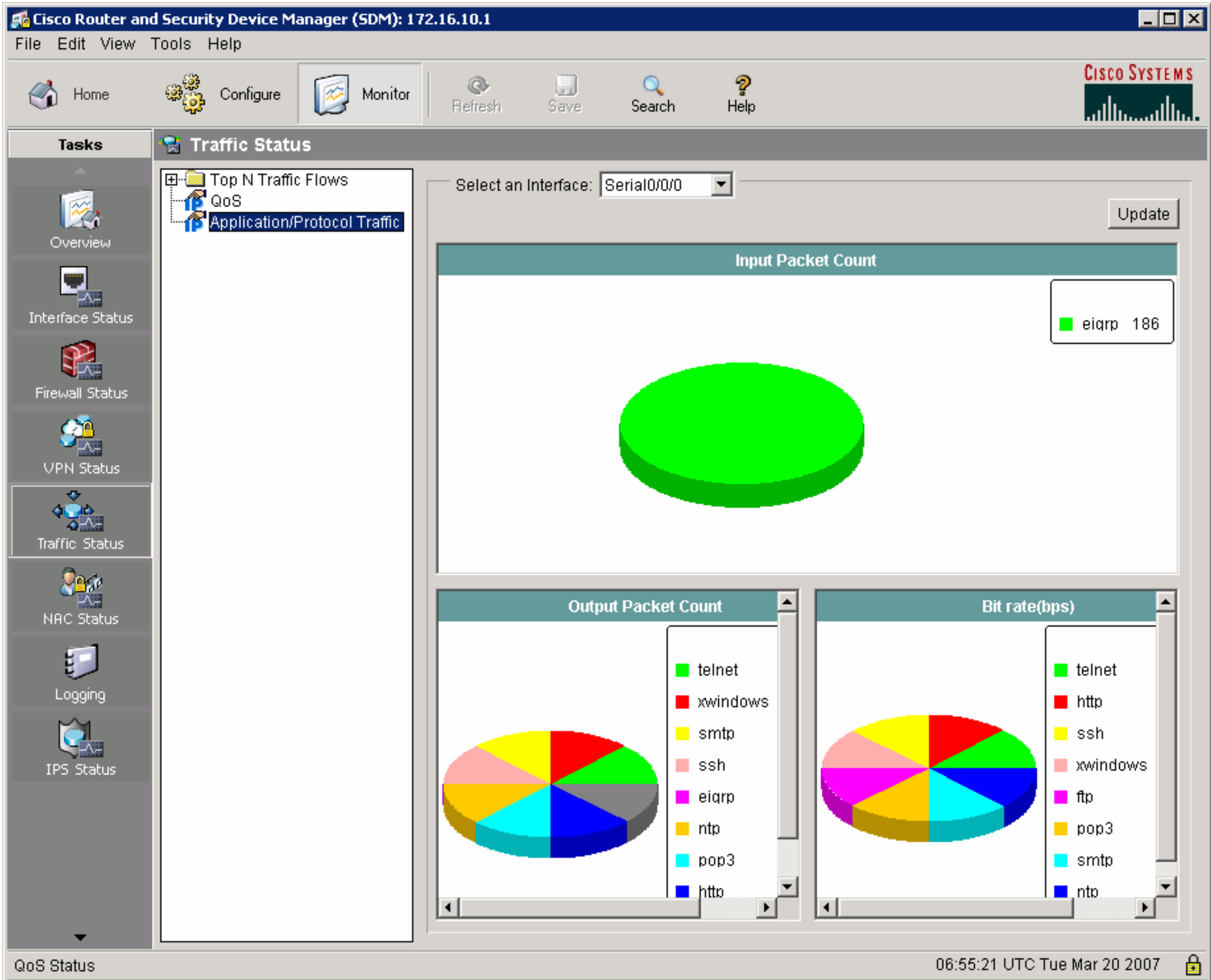


Figure 5-2: Bandwidth Distibution by Application

Final Configurations

```
R1# show run
hostname R1
!
crypto pki trustpoint TP-self-signed-1455051929
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-1455051929
 revocation-check none
 rsakeypair TP-self-signed-1455051929
!
crypto pki certificate chain TP-self-signed-1455051929
 certificate self-signed 01
  3082023A 308201A3 A0030201 02020101 300D0609 2A864886 F70D0101 04050030
<OUTPUT OMITTED>
  quit
username ciscosdm privilege 15 password 0 ciscosdm
!
class-map match-any SDMScave-Serial0/0/0
 match protocol napster
 match protocol fasttrack
```



```
 match protocol gnutella
class-map match-any SDMVoice-Serial0/0/0
 match protocol rtp audio
class-map match-any SDMTrans-Serial0/0/0
 match protocol citrix
 match protocol finger
 match protocol notes
 match protocol novadigm
 match protocol pcanywhere
 match protocol secure-telnet
 match protocol sqlnet
 match protocol sqlserver
 match protocol ssh
 match protocol telnet
 match protocol xwindows
class-map match-any SDMManage-Serial0/0/0
 match protocol dhcp
 match protocol dns
 match protocol imap
 match protocol kerberos
 match protocol ldap
 match protocol secure-imap
 match protocol secure-ldap
 match protocol snmp
 match protocol socks
 match protocol syslog
class-map match-any SDMBulk-Serial0/0/0
 match protocol exchange
 match protocol ftp
 match protocol irc
 match protocol nntp
 match protocol pop3
 match protocol printer
 match protocol secure-ftp
 match protocol secure-irc
 match protocol secure-nntp
 match protocol secure-pop3
 match protocol smtp
 match protocol tftp
class-map match-any SDMSignal-Serial0/0/0
 match protocol h323
 match protocol rtcp
class-map match-any SDMRout-Serial0/0/0
 match protocol bgp
 match protocol egp
 match protocol eigrp
 match protocol ospf
 match protocol rip
 match protocol rsvp
class-map match-any SDMSVideo-Serial0/0/0
 match protocol cuseeme
 match protocol netshow
 match protocol rtsp
 match protocol streamwork
 match protocol vdolive
class-map match-any SDMIVideo-Serial0/0/0
 match protocol rtp video
!
policy-map SDM-Pol-Serial0/0/0
 class SDMTrans-Serial0/0/0
  bandwidth remaining percent 33
  set dscp af21
 class SDMSignal-Serial0/0/0
```



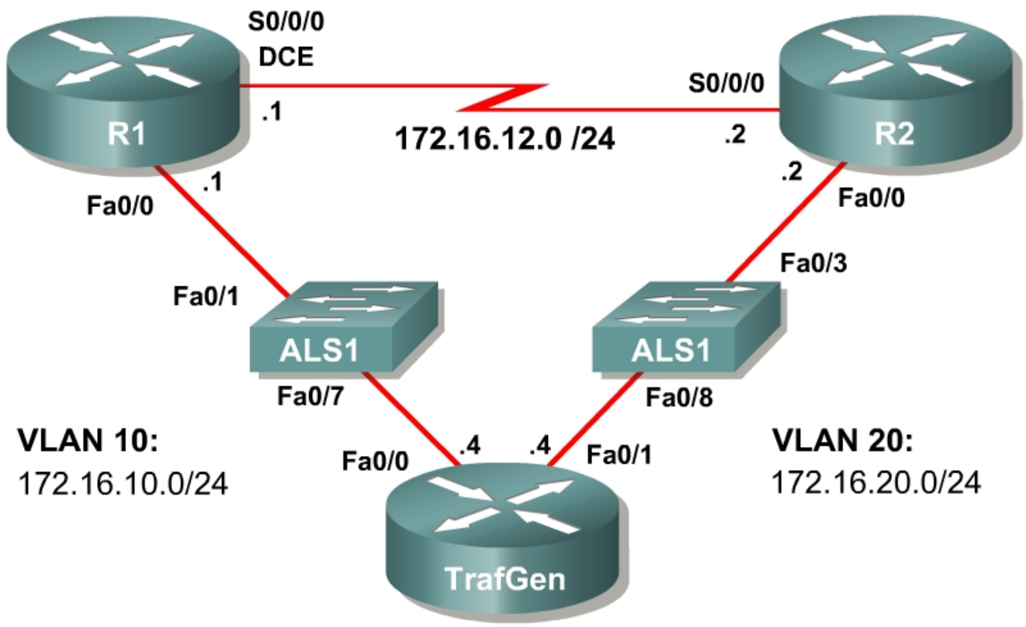
```
  bandwidth remaining percent 40
  set dscp cs3
 class SDMVoice-Serial0/0/0
  priority percent 70
  set dscp ef
 class SDMRout-Serial0/0/0
  bandwidth remaining percent 3
  set dscp cs6
 class SDMManage-Serial0/0/0
  bandwidth remaining percent 3
  set dscp cs2
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 ip address 172.16.12.1 255.255.255.0
 ip nbar protocol-discovery
 clock rate 800000
 service-policy output SDM-Pol-Serial0/0/0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
ip http server
ip http secure-server
end

R2# show run
hostname R2
!
interface FastEthernet0/1
 ip address 172.16.20.2 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 ip address 172.16.12.2 255.255.255.0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
end
```


Lab 4.1 Default Queuing Tools

Learning Objectives

- Verify interface queuing configuration
- Observe statistics over multiple software queues
- Consider differences between FIFO and WFQ
- Change interface queuing types

Topology Diagram



Scenario

When configuring quality of service (QoS) on router interfaces, you will find two queuing mechanisms that are used by default on particular types of interfaces in Cisco IOS software.

This operating system defaults to first-in first-out (FIFO) operation for most interfaces and selects weighted fair queuing (WFQ) for serial interfaces at E1 speeds (2.048 Mbps) and below. In this lab, you will explore the operation of these mechanisms with live traffic generation.

Preparation

This lab uses the Basic Pagent Configuration for TrafGen and the switch ALS1 to generate and facilitate lab traffic in a stream from TrafGen to R1 to R2. Prior to beginning this lab, configure TrafGen (R4) and ALS1 according to the Basic

Pagent Configuration in Lab 3.1: Preparing for QoS. You can accomplish this on R4 by loading the *basic-ios.cfg* file from flash memory into the NVRAM and reloading.

```
TrafGen# copy flash:basic-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

Next, instruct TGN to load the *basic-tgn.cfg* file and to start generating traffic.

```
TrafGen> enable
TrafGen# tgn load-config basic-tgn.cfg
TrafGen# tgn start
```

On the switch, load the *basic.cfg* file into NVRAM and reload the device.

```
ALS1# copy flash:basic.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
ALS1# reload
Proceed with reload? [confirm]
```

In addition, add the Fast Ethernet 0/3 interface on the switch to VLAN 20 since R2 will be the exit point from the network topology in this lab.

```
ALS1# configure terminal
ALS1(config)# interface fastethernet 0/3
ALS1(config-if)# switchport access vlan 20
ALS1(config-if)# switchport mode access
```

Step 1: Configure Addressing

Configure all of the physical interfaces shown in the diagram. Set the clocking bit rate on the serial link to 800,000 bps and use the **no shutdown** command to enable all of the interfaces in the topology diagram.

```
R1(config)# interface fastethernet0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface serial0/0/0
R1(config-if)# ip address 172.16.12.1 255.255.255.0
R1(config-if)# clock rate 800000
R1(config-if)# no shutdown

R2(config)# interface fastethernet0/0
R2(config-if)# ip address 172.16.20.2 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# interface serial0/0/0
R2(config-if)# ip address 172.16.12.2 255.255.255.0
R2(config-if)# no shutdown
```

Best QoS practices dictate that the **bandwidth** command be applied to a serial interface. Serial interfaces do not default their bandwidth parameter to the applied clock rate, but rather allow the administrator to set the reference amount of usable bandwidth for QoS provisioning tools with the **bandwidth** command.

The **bandwidth** command assigns an informational value that will not be used at the physical layer, but will be communicated to and used by upper-layer protocols.

Display the bandwidth value for R1's Serial 0/0/0 interface with the **show interfaces serial 0/0/0** command. Notice that by default R1's serial interface maintains a reference bandwidth of 1.544 Mbps—T1 speed—regardless of the access rate configured with the **clock rate** command.

```
R1# show interfaces serial 0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  Internet address is 172.16.12.1/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 130/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  CRC checking enabled
  Last input 00:00:02, output 00:00:01, output hang never
  Last clearing of "show interface" counters 01:42:53
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 16792618
  Queueing strategy: weighted fair
  Output queue: 71/1000/64/16792618 (size/max total/threshold/drops)
     Conversations  6/9/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 790000 bits/sec, 234 packets/sec
     1724 packets input, 112900 bytes, 0 no buffer
     Received 892 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     741417 packets output, 378715957 bytes, 0 underruns
     0 output errors, 0 collisions, 4 interface resets
     0 output buffer failures, 0 output buffers swapped out
     2 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Approximately 800 Kbps of traffic—the maximum amount of traffic that can be sent at the current clocking access rate—is flowing across a link with a bandwidth parameter of 1544 Kbps. Therefore, the transmit load ratio defined as (output rate) ÷ (bandwidth parameter) is approximately one-half, represented as a fraction of the value 255 so that it can be stored as an 8-bit value by the operating system.

In the output shown above, what percentage of the bandwidth is available for forwarding packets in the output queue?

If you were to enable weighted fair queuing (WFQ) on a Fast Ethernet interface, you would find that the following bandwidth information would be shown.

```
R1# show interfaces FastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is MV96340 Ethernet, address is 0019.0623.4380 (bia 0019.0623.4380)
  Internet address is 172.16.10.1/24
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 72/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 100Mb/s, 100BaseTX/FX
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:01:02, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: weighted fair
  Output queue: 0/1000/64/0 (size/max total/threshold/drops)
     Conversations  0/1/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 75000 kilobits/sec
```

What percentage of the bandwidth is available for forwarding packets in the output queue for this Fast Ethernet interface?

Various upper-layer protocols and mechanisms, such as Enhanced Interior Gateway Routing Protocol (EIGRP) and WFQ, use the bandwidth parameter to accomplish tasks such as metric calculation and bandwidth provisioning.

What traffic must flow over a link that is not Layer 3 traffic forwarded from another network? Give at least two examples.

Obviously, since the number of bits sent in an interval can not exceed the clocking access rate for that interval, all local control traffic, including Layer 2 traffic, must also be sent within the limit of the total access rate configured with the **clock rate** command. Therefore, it is advisable not to reserve more than 75 percent of the total bandwidth for queued traffic so that such control traffic can be sent.

The Cisco product documentation for Cisco IOS version 12.4 Command Reference summarizes this as follows:

> "The sum of all bandwidth allocation on an interface should not exceed 75 percent of the available bandwidth on an interface. The remaining 25 percent of bandwidth is used for overhead, including Layer 2 overhead, control traffic, and best-effort traffic.
>
> If you need to allocate more than 75 percent for RSVP, CBWFQ, LLQ, IP RTP Priority, Frame Relay IP RTP Priority, and Frame Relay PIPQ, you can use the **max-reserved-bandwidth** command. The *percent* argument specifies the maximum percentage of the total interface bandwidth that can be used.
>
> If you do use the **max-reserved-bandwidth** command, make sure that not too much bandwidth is taken away from best-effort and control traffic."[1]

Configure the bandwidth parameter on the serial interface to match the access rate of the interface.

```
R1(config)# interface serial 0/0/0
R1(config-if)# bandwidth 800

R2(config)# interface serial 0/0/0
R2(config-if)# bandwidth 800
```

Step 2: Configure EIGRP AS 1

Provide routing connectivity at Layer 3 between all networks using EIGRP as the routing protocol.

Assign EIGRP AS 1 to connected networks on R1 and R2. Disable automatic summarization and add the entire major 172.16.0.0 network with a classful **network** statement.

```
R1(config)# router eigrp 1
R1(config-router)# no auto-summary
R1(config-router)# network 172.16.0.0

R2(config)# router eigrp 1
R2(config-router)# no auto-summary
R2(config-router)# network 172.16.0.0
```

Note: If you do not use the basic-ios.cfg and basic-tgn.cfg files, enter these commands on R4 to configure it for traffic generation.

```
TrafGen(config)#interface fastethernet 0/0
TrafGen(config-if)# ip address 172.16.10.4 255.255.255.0
TrafGen(config-if)# no shutdown
TrafGen(config-if)# interface fastethernet 0/1
```

---

[1] Cisco Product Documentation, Cisco IOS version 12.4 Command Reference.
http://www.cisco.com/univercd/cc/td/doc/product/software/ios124/124cr/hqos_r/qos_m1h.htm#wp1113113


```
TrafGen(config-if)# ip address 172.16.20.4
TrafGen(config-if)# no shutdown
```

From global configuration mode on TrafGen, enter TGN configuration mode:

```
TrafGen# tgn
TrafGen(TGN:OFF<Fa0/0:none)#
```

Enter (or copy and paste) the following commands at the prompt. Note that you will need to enter the MAC address of R1's FastEthernet 0/0 interface in the highlighted field.

```
fastethernet 0/0
add tcp
rate 1000
L2-dest [enter MAC address of R1 Fa0/0]
L3-src 172.16.10.4
L3-dest 172.16.20.4
L4-dest 23
length random 16 to 1500
burst on
burst duration off 1000 to 2000
burst duration on 1000 to 3000
add fastethernet0/0 1
l4-dest 80
data ascii 0 GET /index.html HTTP/1.1
add fastethernet0/0 1
l4-dest 21
add fastethernet0/0 1
l4-dest 123
add fastethernet0/0 1
l4-dest 110
add fastethernet0/0 1
l4-dest 25
add fastethernet0/0 1
l4-dest 22
add fastethernet0/0 1
l4-dest 6000
!
end
```

Start generating traffic by entering the "start" command at the TGN prompt:

```
TrafGen(TGN:ON,Fa0/0:8/8)# start
```

Step 3: Contrast Interface Queuing Strategies

On R1, contrast the output of **show interfaces** *interface* for the serial connection to R2 and the Fast Ethernet connection to TrafGen.

```
R1# show interfaces serial 0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  Internet address is 172.16.12.1/24
  MTU 1500 bytes, BW 800 Kbit, DLY 20000 usec,
     reliability 255/255, txload 253/255, rxload 1/255
```



```
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  CRC checking enabled
  Last input 00:00:00, output 00:00:04, output hang never
  Last clearing of "show interface" counters 03:51:37
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 50313796
  Queueing strategy: weighted fair
  Output queue: 66/1000/64/50313798 (size/max total/threshold/drops)
     Conversations  4/7/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 600 kilobits/sec
  <OUTPUT OMITTED>

R2# show interfaces fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is MV96340 Ethernet, address is 0018.b992.28d8 (bia 0018.b992.28d8)
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex, 100Mb/s, 100BaseTX/FX
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:32, output 00:00:06, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  <OUTPUT OMITTED>
```

Why do the interfaces have different queuing strategies?

Briefly explain the FIFO logic.

Without detailing the algorithm, list the benefits of weighted fair queuing (WFQ) on an interface.

Discuss possible reasons why Cisco implemented WFQ as the default on links where DiffServ has not been implemented.

Why is there an excessive number of packet drops on R1's serial interface?

Imagine that you ping from router R1 to the R2 serial interface.

Do you predict that the packets would be forwarded to R2 or not?

The Cisco IOS provides the **hold-queue** *packets* {**in** | **out**} command to configure the number of packets that can be stored in the FIFO software queue.

Will increasing the number of packets stored in the FIFO queue have a positive or negative impact upon the overall quality of service? Keep in mind that the link is completely saturated.

When links are completely saturated, as in this scenario, congestion management features cannot solve the true problem: lack of bandwidth. Congestion management features can help smaller packets sneak ahead of larger ones, as in WFQ, but if the queues are always packed, the result is that packets that are forwarded are forwarded with greater delays and the packets that are not forwarded are dropped. Do not implement queuing strategies on an interface that is already saturated expecting miraculous QoS improvements. Any benefits you gain will be offset by losses. Congestion management strategies will not resolve most problems created by a lack of bandwidth.

If you wish to discover how the QoS tools explored in any of the Module 4 labs perform under less saturated conditions, police the Pagent-generated traffic at the ingress router interface to a rate less than that of the egress interface. You may find the command **rate-limit input 700000 2000 2000 conform-action transmit exceed-action transmit** helpful for your testing.

Step 4: Verify and Change Queuing Modes

Test your answers from the previous step by pinging across the serial link.

Ping from R1 to the IP address on R2's serial interface. The ICMP packets should solicit successful replies with low latency, regardless of whether the link is saturated with traffic from TrafGen or not. You can see that the link is saturated because the number of egress drops counted in the output of the **show interfaces** command for that interface increases as more traffic comes from TrafGen.

```
R1# ping 172.16.12.2 repeat 100

Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 172.16.12.2, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 4/19/84 ms

R1# show interfaces serial 0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  Internet address is 172.16.12.1/24
  MTU 1500 bytes, BW 800 Kbit, DLY 20000 usec,
     reliability 255/255, txload 252/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  CRC checking enabled
  Last input 00:00:00, output 00:00:02, output hang never
  Last clearing of "show interface" counters 00:07:53
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 2059241
  Queueing strategy: weighted fair
  Output queue: 70/1000/64/2059241 (size/max total/threshold/drops)
     Conversations  5/9/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 600 kilobits/sec
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 791000 bits/sec, 221 packets/sec
     158 packets input, 10312 bytes, 0 no buffer
     Received 55 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     107548 packets output, 46910536 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

The reason you achieve successful results though bulk traffic is also traversing the link is that WFQ provisions traffic on a per-flow basis. WFQ has multiple output queues—up to 4096 queues—that it provisions on a per-flow basis to produce a weighted queuing strategy.

WFQ dynamically creates conversation queues when it receives a packet with a flow for which it does not currently have a conversation queue open on this interface. WFQ dynamically destroys a conversation queue when it sends the last packet in that queue.

The amount of bandwidth that IOS provisions for each queue depends on the size of the packets and its IP precedence marking.

The Cisco IOS classifies the ICMP traffic from R1's serial interface to R2's serial interface into a separate queue and sends it according to a predefined scheduling operation.

On the interfaces running WFQ, make use of some WFQ-specific **show** commands to view the details of the queuing strategy. One of these is the **show queueing** command, which gives an overview of different interfaces queuing strategies. Note the spelling of this command for future reference.

```
R1# show queueing
Current fair queue configuration:

  Interface           Discard    Dynamic  Reserved  Link    Priority
                      threshold  queues   queues    queues  queues
  Serial0/0/0         64         256      0         8       1
  Serial0/0/1         64         256      0         8       1
  Serial0/1/0         64         256      0         8       1
  Serial0/1/1         64         256      0         8       1

Current DLCI priority queue configuration:
Current priority queue configuration:
Current custom queue configuration:
Current random-detect configuration:
Current per-SID queue configuration:
```

The **show queue** *interface* command displays detailed information about individual queues for an interface. Notice how each conversation (flow) gets its own queue.

```
R1# show queue serial 0/0/0
 Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 11593695
 Queueing strategy: weighted fair
 Output queue: 269/1000/256/11593695 (size/max total/threshold/drops)
    Conversations  8/10/32 (active/max active/max total)
    Reserved Conversations 0/0 (allocated/max allocated)
    Available Bandwidth 1158 kilobits/sec

 (depth/weight/total drops/no-buffer drops/interleaves) 36/32384/1449009/376/0
 Conversation 27, linktype: ip, length: 321
 source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
 TOS: 0 prot: 6, source port 0, destination port 23
```



```
(depth/weight/total drops/no-buffer drops/interleaves) 40/32384/1460389/495/0
Conversation 20, linktype: ip, length: 764
source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
TOS: 0 prot: 6, source port 0, destination port 80

(depth/weight/total drops/no-buffer drops/interleaves) 45/32384/1450829/350/0
Conversation 18, linktype: ip, length: 581
source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
TOS: 0 prot: 6, source port 0, destination port 110

(depth/weight/total drops/no-buffer drops/interleaves) 30/32384/1463060/474/0
Conversation 11, linktype: ip, length: 1340
source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
TOS: 0 prot: 6, source port 0, destination port 6000

(depth/weight/total drops/no-buffer drops/interleaves) 25/32384/1444400/510/0
Conversation 29, linktype: ip, length: 855
source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
TOS: 0 prot: 6, source port 0, destination port 25

(depth/weight/total drops/no-buffer drops/interleaves) 36/32384/1442437/369/0
Conversation 26, linktype: ip, length: 932
source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
TOS: 0 prot: 6, source port 0, destination port 22

(depth/weight/total drops/no-buffer drops/interleaves) 23/32384/1445168/375/0
Conversation 25, linktype: ip, length: 825
source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
TOS: 0 prot: 6, source port 0, destination port 21

(depth/weight/total drops/no-buffer drops/interleaves) 34/32384/1442882/376/0
Conversation 31, linktype: ip, length: 1289
source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
TOS: 0 prot: 6, source port 0, destination port 123
```

All of the conversation queues shown in the above output have the same source and destination addresses. On what basis does WFQ distinguish these conversations from each other?

From which conversation(s) is R1 dropping traffic?

Why is there no conversation queue for ICMP traffic?

On the basis of your answer to the previous question, explain why no ICMP packets were dropped.

Based on the output of the **show queue** command, does WFQ create conversation queues for Layer 2 control traffic?

Now, change the queuing strategy of the serial interface to FIFO by disabling fair queuing on the interface. When you have removed the **fair-queue** command from an interface's configuration, the FIFO mechanism will begin queuing packets. Disable fair queuing on R1's serial interface with the **no fair-queue** command.

```
R1(config)# interface serial 0/0/0
R1(config-if)# no fair-queue
```

First, clear the interface counters. Then, verify the change with the **show interfaces** command. Notice that the queue is full with 40 packets. In our output, we waited over 5 minutes to ensure that the statistics would be correct.

```
R1# clear counters
Clear "show interface" counters on all interfaces [confirm]

R1# show interfaces serial 0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  Internet address is 172.16.12.1/24
  MTU 1500 bytes, BW 800 Kbit, DLY 20000 usec,
     reliability 255/255, txload 250/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  CRC checking enabled
  Last input 00:00:01, output 00:00:02, output hang never
  Last clearing of "show interface" counters 00:17:04
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 4567134
  Queueing strategy: fifo
  Output queue: 40/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 791000 bits/sec, 111 packets/sec
     340 packets input, 22100 bytes, 0 no buffer
     Received 119 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     113797 packets output, 101428413 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
```



```
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

If you try to **ping** across the link, it should not work. You may get a ping to work once in a while by chance (due to the varying sizes of generated traffic).

```
R1# ping 172.16.12.2 repeat 20

Type escape sequence to abort.
Sending 20, 100-byte ICMP Echos to 172.16.12.2, timeout is 2 seconds:
....................
Success rate is 0 percent (0/20)
```

Why are these ICMP packets dropped by the interface queue?

Notice that FIFO displays nearly the same transmit load ratio as WFQ, but the number of packets per second is less than half of that under WFQ.

Why has the throughput in terms of packets per second dropped while the load has not?

At any given point, there are most likely 39 to 40 packets in the input queue. Verify this with the **show interfaces** *interface-name* **summary** command.

```
R1# show interfaces serial 0/0/0 summary

 *: interface is up
 IHQ: pkts in input hold queue     IQD: pkts dropped from input queue
 OHQ: pkts in output hold queue    OQD: pkts dropped from output queue
 RXBS: rx rate (bits/sec)          RXPS: rx rate (pkts/sec)
 TXBS: tx rate (bits/sec)          TXPS: tx rate (pkts/sec)
 TRTL: throttle count

  Interface              IHQ   IQD  OHQ   OQD  RXBS RXPS  TXBS TXPS TRTL
------------------------------------------------------------------------
* Serial0/0/0              0     0   40 986667     0    0 788000  110    0
```

You may modify the size of the output hold queue using the **hold-queue** *depth* **out** command to provision a number of packets.

```
R1(config)# interface serial 0/0/0
R1(config-if)# hold-queue 1000 out
```


Notice the change in the queue depth by viewing the output of the **show interfaces serial 0/0/0 summary** command.

```
R1# show interfaces serial 0/0/0 summary

 *: interface is up
 IHQ: pkts in input hold queue     IQD: pkts dropped from input queue
 OHQ: pkts in output hold queue    OQD: pkts dropped from output queue
 RXBS: rx rate (bits/sec)          RXPS: rx rate (pkts/sec)
 TXBS: tx rate (bits/sec)          TXPS: tx rate (pkts/sec)
 TRTL: throttle count

  Interface              IHQ   IQD  OHQ   OQD  RXBS RXPS  TXBS TXPS TRTL
------------------------------------------------------------------------
* Serial0/0/0              0     0 1000 4522356     0    0 793000  118    0
```

Step 5: Modify Default Queuing Settings

Fair-queuing can be customized based on the congestive discard threshold, number of dynamic queues, and the number of reservable queues. The congestive discard threshold is the maximum size of each queue, and the default number is 64 packets per queue. The number of dynamic queues is the maximum number of queues that can be dynamically allocated for traffic, and the default number for this is set based on interface speed.

From previous output of the **show interfaces** command, you can determine that the maximum total conversations for the serial interface on R1 is 256. Conversation queues may be reserved in the Integrated Services (IntServ) model via the Resource Reservation Protocol (RSVP), but that exceeds the scope of this lab. The default number of reservable queues is zero.

On the serial interface, make the queue size 256 packets each (queue sizes must be an exponent of 2), and have 32 queues available for dynamic allocation. Do not create any reservable queues. To adjust the fair queuing parameters on an interface, use the **fair-queue** [*congestive-discard-threshold* [*dynamic-queues* [*reservable-queues*]]] command in interface configuration mode. All of the numerical arguments are optional; however, to set one argument, all the other arguments before it must also be entered.

```
R1(config)# interface serial 0/0/0
R1(config-if)# fair-queue 256 32
```

Verify using the **show** commands we used earlier.

```
R1# show interfaces serial 0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  Internet address is 172.16.12.1/24
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
     reliability 255/255, txload 130/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  CRC checking enabled
  Last input 00:00:00, output 00:00:00, output hang never
```



```
  Last clearing of "show interface" counters 05:02:53
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 80279227
  Queueing strategy: weighted fair
  Output queue: 266/1000/256/80279228 (size/max total/threshold/drops)
     Conversations  8/9/32 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec
<OUTPUT OMITTED>

R1# show queueing
Current fair queue configuration:

  Interface           Discard    Dynamic  Reserved  Link    Priority
                      threshold  queues   queues    queues  queues
  Serial0/0/0         256        32       0         8       1
<OUTPUT OMITTED>

R1# show queue serial 0/0/0
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 2740715
  Queueing strategy: weighted fair
  Output queue: 257/1000/256/2740715 (size/max total/threshold/drops)
     Conversations  6/9/32 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 70/32384/18753/0/0
  Conversation 31, linktype: ip, length: 416
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 123

  (depth/weight/total drops/no-buffer drops/interleaves) 27/32384/21003/0/0
  Conversation 26, linktype: ip, length: 716
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 22
<OUTPUT OMITTED>
```

If you now try the same **ping** from earlier, you may receive mixed results, with some success, and some failures. Try this multiple times with different repeat counts because you may get different results each time depending on how the traffic is queued.

```
R1# ping 172.16.12.2 repeat 20

Type escape sequence to abort.
Sending 20, 100-byte ICMP Echos to 172.16.12.2, timeout is 2 seconds:
!...!..!!!!!!!!!!..!
Success rate is 65 percent (13/20), round-trip min/avg/max = 12/393/1396 ms
```

Since you have limited the number of dynamic conversation queues that can be created to 32, ICMP traffic will not get allocated a dynamic queue when it needs it. Thus, some packets will be dropped.

Final Configurations

```
R1# show run
hostname R1
!
interface FastEthernet0/0
```



```
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 bandwidth 800
 ip address 172.16.12.1 255.255.255.0
 fair-queue 256 32 0
 clock rate 800000
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
end
```

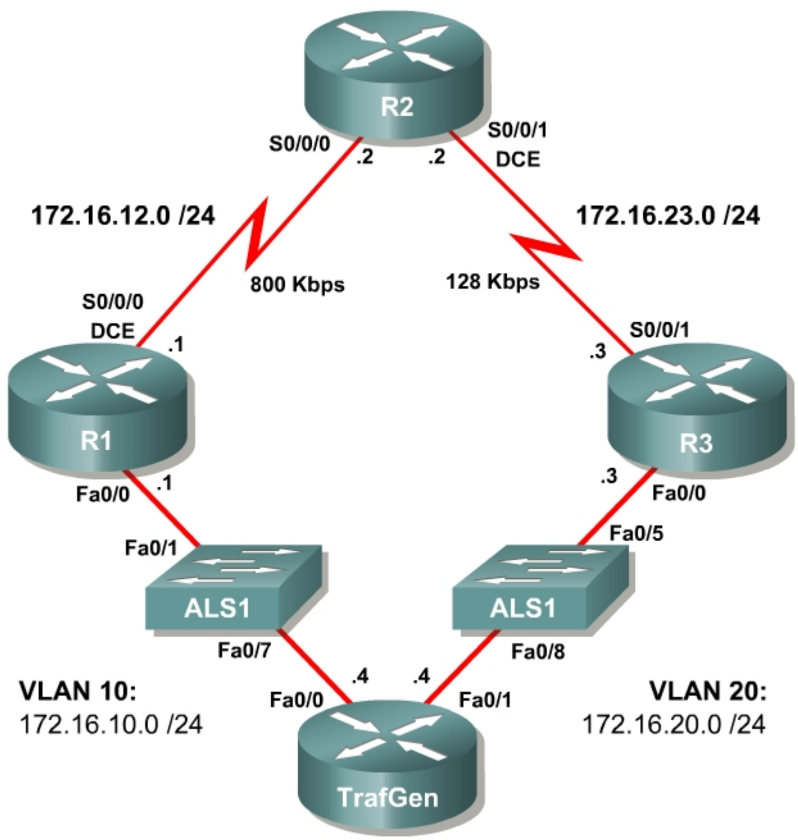
```
R2# show run
hostname R2
!
interface FastEthernet0/1
 ip address 172.16.20.2 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 bandwidth 800
 ip address 172.16.12.2 255.255.255.0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
end
```


Lab 4.2 Intermediate Queuing Tools

Learning Objectives

- Configure and verify custom queuing
- Configure and verify priority queuing

Topology Diagram



Scenario

In this lab, you will configure two IOS quality of service (QoS) queuing tools. First-in, first-out (FIFO) and weighted fair queuing (WFQ) require very little configuration to implement. Priority queuing and custom queuing require decisions about classification and priority or weighting in order to properly apply the tools. These two tools are configured similarly but function very differently.

Preparation

This lab uses the Basic Pagent Configuration for TrafGen and the switch ALS1 to generate and facilitate lab traffic in a stream from TrafGen to R1 to R2 to R3.

Prior to beginning this lab, configure TrafGen (R4) and the switch according to the Basic Pagent Configuration in Lab 3.1: Preparing for QoS. You may accomplish this on R4 by loading the *basic-ios.cfg* file from flash memory into the NVRAM and reloading.

```
TrafGen# copy flash:basic-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

Next, instruct TGN to load the *basic-tgn.cfg* file and to start generating traffic.

```
TrafGen> enable
TrafGen# tgn load-config
TrafGen# tgn start
```

On the switch, load the *basic.cfg* file into NVRAM and reload the device.

```
ALS1# copy flash:basic.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
ALS1# reload
Proceed with reload? [confirm]
```

In addition, add the Fast Ethernet 0/5 interface on the switch to VLAN 20 since R3 will be the exit point from the network topology in this lab.

```
ALS1# configure terminal
ALS1(config)# interface fastethernet 0/5
ALS1(config-if)# switchport access vlan 20
ALS1(config-if)# switchport mode access
```

Step 1: Configure the Physical Interfaces

Configure all of the physical interfaces shown in the diagram. Set the clock rate on the serial link between R1 and R2 to 800000, and the clock rate of the serial link between R2 and R3 to be 128000, and use the **no shutdown** command on all interfaces. Set the informational bandwidth parameter on the serial interfaces.

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface serial 0/0/0
R1(config-if)# bandwidth 800
R1(config-if)# ip address 172.16.12.1 255.255.255.0
R1(config-if)# clock rate 800000
R1(config-if)# no shutdown

R2(config)# interface serial 0/0/0
R2(config-if)# bandwidth 800
R2(config-if)# ip address 172.16.12.2 255.255.255.0
```



```
R2(config-if)# no shutdown
R2(config-if)# interface serial 0/0/1
R2(config-if)# bandwidth 128
R2(config-if)# ip address 172.16.23.2 255.255.255.0
R2(config-if)# clock rate 128000
R2(config-if)# no shutdown

R3(config)# interface fastethernet 0/0
R3(config-if)# ip address 172.16.20.3 255.255.255.0
R3(config-if)# no shutdown
R3(config-if)# interface serial 0/0/1
R3(config-if)# bandwidth 128
R3(config-if)# ip address 172.16.23.3 255.255.255.0
R3(config-if)# no shutdown
```

Step 2: Configure EIGRP AS 1

Configure routing between R1, R2, and R3 using Enhanced Interior Gateway Routing Protocol (EIGRP). Include the entire 172.16.0.0/16 major network in AS 1 and disable automatic summarization.

```
R1(config)# router eigrp 1
R1(config-router)# no auto-summary
R1(config-router)# network 172.16.0.0

R2(config)# router eigrp 1
R2(config-router)# no auto-summary
R2(config-router)# network 172.16.0.0

R3(config)# router eigrp 1
R3(config-router)# no auto-summary
R3(config-router)# network 172.16.0.0
```

Verify that the number of packets counted is increasing on the outbound interface of R3 using the **show interfaces fastethernet 0/0** command. Issue the command twice to make sure the number of packets output has changed. If the number is not increasing, troubleshoot Layer 1, 2, and 3 connectivity and the EIGRP topology.

Step 3: Configure Custom Queuing

Custom queuing (CQ) is an egress queuing tool that allows you to classify traffic into various queues based on the types of information that can be selected by an access list. These properties include transport or application protocol, port numbers, differentiated services code point (DSCP) or IP Precedence markings, and input interface. Many of these parameters can be referenced with an access list, so you may prefer to specify such attributes in a single access list rather than entering multiple classification lines for each protocol. The goal of custom queuing is to allocate bandwidth proportionally amongst various classes of traffic.

CQ may use up to 16 queues for IP forwarding, and the queues are serviced in a round-robin fashion. Each queue has a configurable maximum size in bytes specified and a configurable byte count for sending traffic during each round.

This effectively allows you to proportionally designate how much bandwidth you want to allocate to each queue.

Custom queuing is configured in three steps:

1. Globally define classification methods to select traffic for particular queues.

2. Globally define the byte count and packet limit for each queue. This step is optional and only needs to be configured where desired.

3. Apply the CQ that you created globally to a particular interface, where it will replace the current outbound queuing strategy.

In this lab, you will configure R1 to use custom queuing as the queuing method on the serial link facing R2.

You may configure up to 16 queues in each queue list. A queue list represents a set of queues that together may be applied as a CQ strategy on an interface. The configuration in this lab will use queue list 7.

Traffic is sent from each queue in sequence until the byte count is met or exceeded, and then the next queue is processed. Refer to Figure 3-1 for a conceptual diagram.

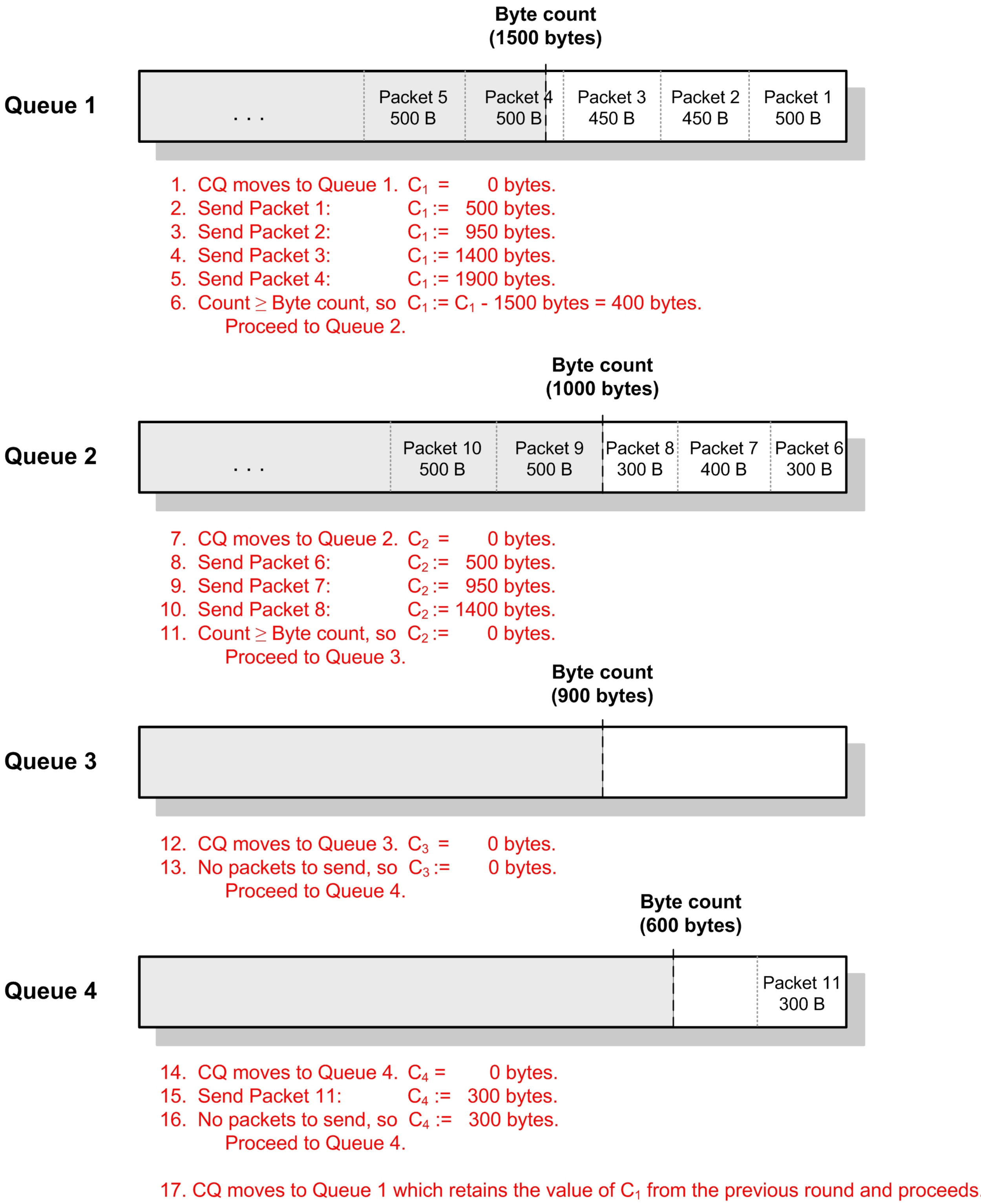


Figure 3-1: Custom Queuing

When Telnet traffic is sent from one router to another, the IP packets are labeled with an IP Precedence of 6, Internet Control. Later in this step, you will test your queuing configurations with Telnet.

Create an extended access control list (ACL) to select traffic with an IP Precedence of 6.

```
R1(config)# access-list 101 permit ip any any precedence internet
```

Apply this ACL to CQ classification by issuing the **queue-list** *queue-list-number* **protocol ip** *queue-number* **list** *access-list-number* command.

```
R1(config)# queue-list 7 protocol ip 1 list 101
```

The rest of the queues you configure in this queue list will match on TCP port number. Classification based on port number is fairly simple using the **queue-list** *queue-list-number* **protocol** *protocol queue-number* **tcp** *port-number* command. You could also replace the **tcp** keyword with **udp** to match on UDP port numbers, although this method will not be used in this lab because all of the traffic generated by TrafGen uses TCP as the transport protocol.

Classify SSH (TCP port 22) and telnet into queue 2, NTP traffic (TCP port 123) into queue 3, and XWindows (TCP port 6000) and HTTP into queue 4. Do not place any other traffic into queues yet.

```
R1(config)# queue-list 7 protocol ip 2 tcp 22
R1(config)# queue-list 7 protocol ip 2 tcp telnet
R1(config)# queue-list 7 protocol ip 3 tcp 123
R1(config)# queue-list 7 protocol ip 3 tcp 6000
R1(config)# queue-list 7 protocol ip 4 tcp www
```

The TrafGen router also spoofs POP3 and SMTP traffic to 172.16.20.4. This traffic is not caught by any of the classification tools on the queues you have created, so assign unclassified traffic to queue 4. Issue the **queue-list** *queue-list-number* **default** *queue-number* command, selecting queue 4 as the default queue.

```
R1(config)# queue-list 7 default 4
```

Now that you have classified packets into queues, you can adjust the parameters of queues. Reduce the queue size of queue 1 to 10 packets from the default 20 packets with the **queue-list** *queue-list-number* **queue** *queue-number* **limit** *limit* command.

```
R1(config)# queue-list 7 queue 1 limit 10
```

Most important to your CQ configuration is what byte count to send from each individual queue during each round-robin pass. Beginning in IOS Release 12.1, the byte count was changed from a minimum to an average by extending its support for a deficit between round-robin passes.[1] If the size of the final packet exceeds the byte count, CQ stores the excess as the starting byte count for the next round. If CQ depletes the queue before the byte count is reached, CQ stores the deficit as a negative balance to use at the beginning of the next round-robin pass.

Since your default queue, Queue 4, will probably have more traffic than other queues, raise its byte count to 3000, which is double the default of 1500.

```
R1(config)# queue-list 7 queue 4 byte-count 3000
```

What effect will this command produce?

The last step of configuring CQ is to apply it to an interface. Issue the **custom-queue-list** *queue-list-number* command in interface configuration mode for R1's Serial 0/0/0 interface. Apply queue list 7 to R1's Serial 0/0/0 interface.

```
R1(config)# interface serial 0/0/0
R1(config-if)# custom-queue-list 7
```

You can verify the queuing configuration on a router using the **show queueing** command.

```
R1# show queueing
Current fair queue configuration:

  Interface           Discard    Dynamic  Reserved  Link    Priority
                      threshold  queues   queues    queues  queues
  Serial0/0/0         64         256      0         8       1

Current DLCI priority queue configuration:
Current priority queue configuration:
Current custom queue configuration:

List   Queue  Args
7      4      default
7      1      protocol ip          list 101
7      2      protocol ip          tcp port telnet
7      2      protocol ip          tcp port 22
7      3      protocol ip          tcp port 123
7      3      protocol ip          tcp port 6000
7      4      protocol ip          tcp port www
7      1      limit 10
7      4      byte-count 3000
```

[1] Cisco Product Documentation, Quality of Service Configuration Guide: Custom Queuing. http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/fqos_c/fqcprt2/qcfconmg.htm#wp1001355


```
Current random-detect configuration:
Current per-SID queue configuration:
```

Notice that the maximum size of Queue 1 is different than the rest of the queues, since we changed it earlier.

Notice that some of the TCP port numbers have been replaced with protocol names. When configuring CQ, you can enter the names of certain well-known protocols instead of their protocol numbers; however, the IOS contains a very small list of named protocols.

The output of **show interfaces** changes, as well, to reflect the new queuing strategy for an interface.

```
R1# show interfaces serial0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is GT96K Serial
  Internet address is 172.16.12.1/24
  MTU 1500 bytes, BW 800 Kbit, DLY 20000 usec,
     reliability 255/255, txload 252/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  CRC checking enabled
  Last input 00:00:02, output 00:00:01, output hang never
  Last clearing of "show interface" counters 00:08:45
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 2333955
  Queueing strategy: custom-list 7
  Output queues: (queue #: size/max/drops)
     0: 0/20/0 1: 0/10/0 2: 20/20/581726 3: 19/20/579996 4: 20/20/1172236
     5: 0/20/0 6: 0/20/0 7: 0/20/0 8: 0/20/0 9: 0/20/0
     10: 0/20/0 11: 0/20/0 12: 0/20/0 13: 0/20/0 14: 0/20/0
     15: 0/20/0 16: 0/20/0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 792000 bits/sec, 122 packets/sec
     175 packets input, 11460 bytes, 0 no buffer
     Received 61 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     63932 packets output, 52134752 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Why is queue 1 empty?

Which queues are actively enqueuing and sending traffic?

In addition to the queues that you organized for classification above, queue 0 is used to send link control traffic across the link outside of the 16 normal queues used by custom queuing. EIGRP hellos and Layer 2 keepalives are sent through Queue 0 so that they receive preferential treatment.[2]

According to the output of the **show interface** command shown above, what is the maximum number of packets that queue 1 can hold?

Issue the **show queue** *interface queue-number* command to view the contents of individual queues within the CQ output queues. The output below shows queue 4 (the default queue) of Serial 0/0/0 on R1.

```
R1# show queue serial 0/0/0 4
Output queue for Serial0/0/0 is 20/20

Packet 1, linktype: ip, length: 1406, flags: 0x88
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 80
    data: 0x0000 0x0050 0x0000 0x0000 0x0000 0x0000 0x5000
          0x0000 0x9BE6 0x0000 0x4745 0x5420 0x2F69 0x6E64

Packet 2, linktype: ip, length: 658, flags: 0x88
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 80
    data: 0x0000 0x0050 0x0000 0x0000 0x0000 0x0000 0x5000
          0x0000 0x3EE7 0x0000 0x4745 0x5420 0x2F69 0x6E64

Packet 3, linktype: ip, length: 1210, flags: 0x88
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 25
    data: 0x0000 0x0019 0x0000 0x0000 0x0000 0x0000 0x5000
          0x0000 0xB651 0x0000 0x0001 0x0203 0x0405 0x0607

Packet 4, linktype: ip, length: 1100, flags: 0x88
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 110
    data: 0x0000 0x006E 0x0000 0x0000 0x0000 0x0000 0x5000
          0x0000 0x432E 0x0000 0x0001 0x0203 0x0405 0x0607

<OUTPUT OMITTED>
```

Which protocols do the destination port numbers indicate?

---

[2] Cisco.com, QoS Congestion Management Design TechNote. *Custom Queuing and Routing Updates*. Document ID: 13784. http://www.cisco.com/en/US/tech/tk543/tk544/technologies_tech_note09186a0080093f90.shtml

Next, you'll demonstrate the output of the custom queuing debugging commands. Shut down the Fast Ethernet interface on R1 to reduce the amount of traffic flowing into the serial interface. After configuring the virtual terminal lines, begin a Telnet session from R2 to R1.

```
R1(config)# interface fastethernet 0/0
R1(config-if)# shutdown
R1(config-if)# exit
R1(config)# line vty 0 4
R1(config-line)# password cisco
R1(config-line)# login
```

```
R2# telnet 172.16.12.1
Trying 172.16.12.1 ... Open


User Access Verification

Password: cisco
R1>
```

Issue the **debug custom-queue** command on R1 to display packets passing though the CQ mechanism. Issue the **undebug all** command when you are done.

```
R1# debug custom-queue
R1#
*May  9 00:42:21.279: CQ: Serial0/0/0 output (Pk size/Q: 48/1) Q # was 4 now 1
*May  9 00:42:21.283: CQ: Serial0/0/0 output (Pk size/Q: 56/2) Q # was 1 now 2
*May  9 00:42:21.287: CQ: Serial0/0/0 output (Pk size/Q: 86/2) Q # was 2 now 2
*May  9 00:42:21.291: CQ: Serial0/0/0 output (Pk size/Q: 47/2) Q # was 2 now 2
*May  9 00:42:21.291: CQ: Serial0/0/0 output (Pk size/Q: 47/2) Q # was 2 now 2
*May  9 00:42:21.291: CQ: Serial0/0/0 output (Pk size/Q: 50/2) Q # was 2 now 2
*May  9 00:42:21.291: CQ: Serial0/0/0 output (Pk size/Q: 47/2) Q # was 2 now 2
R1# undebug all
```

Reactivate the Fast Ethernet interface on R1 before continuing to the next step.

```
R1(config)# interface fastethernet 0/0
R1(config-if)# no shutdown
```

Step 4: Configure Priority Queuing

Priority queuing (PQ) is an IOS queuing method that allows you to classify traffic into various queues the same way that CQ does. However, PQ implements a strict priority queuing policy.

Rather than many queues that are serviced in a round-robin fashion, there are 4 queues with different priorities—high, medium, normal, and low. A queue will not be serviced unless the queues with higher priority than it are empty. The default size of each queue gets smaller and smaller as the priority increases, although you can adjust the default queue sizes. Priority queuing can easily create bandwidth starvation for lower-priority queues.

If a packet is in the highest-priority queue, then PQ will always send that packet before others. If a packet is in the medium-priority queue and no packets are in the high-priority queue, then the medium priority packet will take strict precedence over all packets in any lower-priority queues regardless of how many there are or how long they have been queued.

Priority queuing is configured using these steps:

1. Globally define classification methods to select traffic for particular queues.

2. Establish the packet limit for each queue. This step is optional.

3. Apply the priority queuing list that you created globally to a particular interface, where it will replace the current outbound queuing strategy.

In a production environment, you would want time-sensitive packets, such as VoIP packets, to have a high priority as well as routing control packets like EIGRP. In this scenario, give priority to packets with IP Precedence of 6 since you don't want significant delay in your telnet sessions. Configure R2 to use priority queuing as the queuing method on the serial link facing R3.

Using the same extended access list you used in Step 3, select traffic with IP Precedence of 6 for the high-priority queue. Issue the **priority-list** *priority-list-number* **protocol** *protocol queue-name* **list** *access-list-number* command to configure a queue in a priority list to hold packets matched by the access list. As in custom queuing, you can create up to 16 priority lists on a router. For this lab, configure priority list 5.

```
R2(config)# access-list 101 permit ip any any precedence internet
R2(config)# priority-list 5 protocol ip high list 101
```

The rest of the queues you will configure in this queue list will match on TCP port number. Classification based on port number is fairly simple using the **priority-list** *priority-list-number* **protocol** *protocol* {**high** | **medium** | **normal** | **low**} **tcp** *port-number* command. You could also replace the **tcp** keyword with **udp** to match on UDP port numbers, although this will not be used in this lab because all of the traffic generated by TrafGen uses TCP as the transport protocol.

Classify SSH (TCP port 22) and TrafGen-generated telnet into medium-priority queue, NTP traffic (TCP port 123) into the normal-priority queue. Do not place any other traffic into queues yet.

```
R2(config)# priority-list 5 protocol ip medium tcp 22
R2(config)# priority-list 5 protocol ip medium tcp 23
R2(config)# priority-list 5 protocol ip normal tcp 123
```


Instead of explicitly assigning XWindows and HTTP traffic to the low-priority queue, simply assign the remainder of all traffic to that queue by selecting it as the default queue. Issue the **priority-list** *priority-list-number* **default** *queue-name* command in global configuration mode.

```
R2(config)# priority-list 5 default low
```

The queue sizes for a priority list can also be configured. The default queue sizes are 20, 40, 60, and 80 for high, medium, normal, and low priorities respectively. For this lab, increase the low queue size to 100. Issue the **priority-list** *priority-list-number* **queue-limit** *high-limit medium-limit normal-limit low-limit* command to change the priority list queue sizes. You must enter in all four values together and in sequence.

```
R2(config)# priority-list 5 queue-limit 20 40 60 100
```

Now that the priority list is configured, apply it to an interface by issuing the **priority-group** *priority-list-number* command in interface configuration mode. Apply priority list 5 on R2 to its serial interface facing R3.

```
R2(config)# interface serial0/0/1
R2(config-if)# priority-group 5
```

Verify the queuing configuration on the router with the **show queueing** command.

```
R2# show queueing
Current fair queue configuration:

  Interface           Discard    Dynamic  Reserved  Link    Priority
                      threshold  queues   queues    queues  queues
  Serial0/0/1         64         256      0         8       1

Current DLCI priority queue configuration:
Current priority queue configuration:

List   Queue  Args
5      low    default
5      high   protocol ip          list 101
5      medium protocol ip          tcp port 22
5      medium protocol ip          tcp port telnet
5      normal protocol ip          tcp port 123
5      low    limit 100
Current custom queue configuration:
Current random-detect configuration:
Current per-SID queue configuration:
```

Like custom queuing, priority queuing changes the output of **show interfaces**.

```
R2# show interfaces serial 0/0/1
Serial0/0/1 is up, line protocol is up
  Hardware is GT96K Serial
  Internet address is 172.16.23.2/24
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 249/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
```



```
  Last input 00:00:00, output 00:00:01, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 9660079
  Queueing strategy: priority-list 5
  Output queue (queue priority: size/max/drops):
     high: 0/20/0, medium: 39/40/171, normal: 60/60/9658709, low: 100/100/1199
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 125000 bits/sec, 21 packets/sec
     28688 packets input, 1867995 bytes, 0 no buffer
     Received 10090 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     2274841 packets output, 1332338661 bytes, 0 underruns
     0 output errors, 0 collisions, 13 interface resets
     0 output buffer failures, 0 output buffers swapped out
     6 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Also like other queuing types, you can view the contents of each queue with **show queue** *interface queue-number*. The queue numbers correspond to the four named queues, starting at 0, with 0 being the highest priority. The output below shows the contents of the low-priority queue of Serial 0/0/1 on R2.

```
R2# show queue serial0/0/1 3
Output queue for Serial0/0/1 is 100/100

Packet 1, linktype: ip, length: 1322, flags: 0x88
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 58,
  TOS: 0 prot: 6, source port 0, destination port 110
    data: 0x0000 0x006E 0x0000 0x0000 0x0000 0x0000 0x5000
          0x0000 0x7211 0x0000 0x0001 0x0203 0x0405 0x0607

Packet 2, linktype: ip, length: 1438, flags: 0x88
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 58,
  TOS: 0 prot: 6, source port 0, destination port 110
    data: 0x0000 0x006E 0x0000 0x0000 0x0000 0x0000 0x5000
          0x0000 0xEDDF 0x0000 0x0001 0x0203 0x0405 0x0607

Packet 3, linktype: ip, length: 176, flags: 0x88
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 58,
  TOS: 0 prot: 6, source port 0, destination port 6000
    data: 0x0000 0x1770 0x0000 0x0000 0x0000 0x0000 0x5000
          0x0000 0x4EB3 0x0000 0x0001 0x0203 0x0405 0x0607

<OUTPUT OMITTED>
```

Execute the previous command again.

Has there been any change in the packets in the low-priority queue?

What does this indicate?

How could you resolve this problem?

Challenge

Shut down the Serial 0/0/0 interface on R2.

Debug priority queuing with the **debug priority-queue** command.

Configure R3 for telnet access. Then, telnet from R2 to R3 to observe the enqueuing of packets into the high-priority queue.

Final Configurations

```
R1# show run
!
hostname R1
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 duplex auto
 speed auto
!
interface Serial0/0/0
 bandwidth 800
 ip address 172.16.12.1 255.255.255.0
 custom-queue-list 7
 clock rate 800000
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
access-list 101 permit ip any any precedence internet
queue-list 7 protocol ip 1 list 101
queue-list 7 protocol ip 2 tcp telnet
queue-list 7 protocol ip 2 tcp 22
queue-list 7 protocol ip 3 tcp 123
queue-list 7 protocol ip 3 tcp 6000
queue-list 7 protocol ip 4 tcp www
queue-list 7 default 4
queue-list 7 queue 1 limit 10
queue-list 7 queue 4 byte-count 3000
!
line vty 0 4
 password cisco
 login
!
end

R2# show run
!
hostname R2
!
interface Serial0/0/0
 bandwidth 800
```



```
 ip address 172.16.12.2 255.255.255.0
!
interface Serial0/0/1
 bandwidth 128
 ip address 172.16.23.2 255.255.255.0
 priority-group 5
 clock rate 128000
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
access-list 101 permit ip any any precedence internet
priority-list 5 protocol ip high list 101
priority-list 5 protocol ip medium tcp 22
priority-list 5 protocol ip medium tcp telnet
priority-list 5 protocol ip normal tcp 123
priority-list 5 default low
priority-list 5 queue-limit 20 40 60 100
!
line vty 0 4
 password cisco
 login
!
end
```

```
R3# show run
!
hostname R3
!
interface FastEthernet0/1
 ip address 172.16.20.3 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 bandwidth 128
 ip address 172.16.23.3 255.255.255.0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
end
```

```
Switch# show run
!
hostname Switch
!
vtp domain CISCO
vtp mode transparent
!
interface FastEthernet0/1
 switchport access vlan 10
 switchport mode access
 spanning-tree portfast
!
interface FastEthernet0/5
 switchport access vlan 20
 switchport mode access
 spanning-tree portfast
!
interface FastEthernet0/7
```



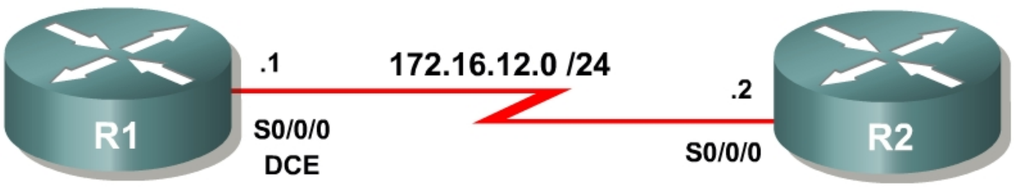
```
 switchport access vlan 10
 switchport mode access
 spanning-tree portfast
!
interface FastEthernet0/8
 switchport access vlan 20
 switchport mode access
 spanning-tree portfast
!
end
```


Lab 4.3 TCP Header Compression

Learning Objectives

- Configure TCP header compression
- Verify TCP header compression

Topology Diagram



Scenario

In this lab, you will configure TCP header compression across a link and verify it by establishing a Telnet session over the link.

This lab does *not* use the Pagent TGN application for traffic generation.

Step 1: Configure Addressing

Configure all of the physical interfaces shown in the diagram. Set the clock rate on the serial link to 64000 and use the **no shutdown** command to enable all of the interface addresses in the topology diagram.

```
R1(config)# interface serial0/0/0
R1(config-if)# ip address 172.16.12.1 255.255.255.0
R1(config-if)# clock rate 64000
R1(config-if)# no shutdown

R2(config)# interface serial0/0/0
R2(config-if)# ip address 172.16.12.2 255.255.255.0
R2(config-if)# no shutdown
```

Step 2: Enable Telnet Access on R2

Enable telnet access on R2 by setting a VTY line password to "cisco".

```
R2(config-if)# line vty 0 4
R2(config-line)# password cisco
```

Step 3: Enable TCP Header Compression

TCP header compression is used to compress TCP headers in a network to save bandwidth on a link. However, TCP header compression comes at a cost in terms of processor time.

TCP header compression must be configured on both ends of the network to compress and decompress packets. RTP header compression is configured similarly, although it is not shown in this lab.

Issue the **ip tcp header-compression** command in interface configuration mode to enable TCP header compression. A class-based form of the command is used in the modular QoS CLI (MQC), but that information will be covered in later labs. Configure this command on the Serial 0/0/0 interfaces on both R1 and R2.

```
R1(config)# interface serial0/0/0
R1(config-if)# ip tcp header-compression

R2(config)# interface serial0/0/0
R2(config-if)# ip tcp header-compression
```

Describe a traffic profile in which TCP header compression can be very useful.

Step 4: Verify TCP Header Compression

Issue the **show ip tcp header-compression** command to view statistics for compressed TCP headers.

```
R1# show ip tcp header-compression
TCP/IP header compression statistics:
  Interface Serial0/0/0 (compression on, VJ)
    Rcvd:    0 total, 0 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    0 total, 0 compressed, 0 status msgs, 0 not predicted
             0 bytes saved, 0 bytes sent
    Connect: 16 rx slots, 16 tx slots,
             0 misses, 0 collisions, 0 negative cache hits, 16 free contexts
```

Generate some TCP traffic by connecting from R1 to R2 via Telnet.

```
R1# telnet 172.16.12.2
Trying 172.16.12.2 ... Open


User Access Verification

Password: cisco
R2> exit

[Connection to 172.16.12.2 closed by foreign host]
R1#
```

Verify that the TCP traffic was compressed.


```
R1# show ip tcp header-compression
TCP/IP header compression statistics:
  Interface Serial0/0/0 (compression on, VJ)
    Rcvd:    17 total, 16 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    19 total, 18 compressed, 0 status msgs, 0 not predicted
             622 bytes saved, 181 bytes sent
             4.43 efficiency improvement factor
    Connect: 16 rx slots, 16 tx slots,
             1 misses, 0 collisions, 0 negative cache hits, 16 free contexts
             94% hit ratio, five minute miss rate 0 misses/sec, 0 max

R2# show ip tcp header-compression
TCP/IP header compression statistics:
  Interface Serial0/0/0 (compression on, VJ)
    Rcvd:    19 total, 18 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    17 total, 16 compressed, 0 status msgs, 0 not predicted
             537 bytes saved, 229 bytes sent
             3.34 efficiency improvement factor
    Connect: 16 rx slots, 16 tx slots,
             1 misses, 0 collisions, 0 negative cache hits, 16 free contexts
             94% hit ratio, five minute miss rate 0 misses/sec, 0 max
```

Given the numbers in the output of the commands shown above, identify how the efficiency improvement factor is computed:

Final Configurations

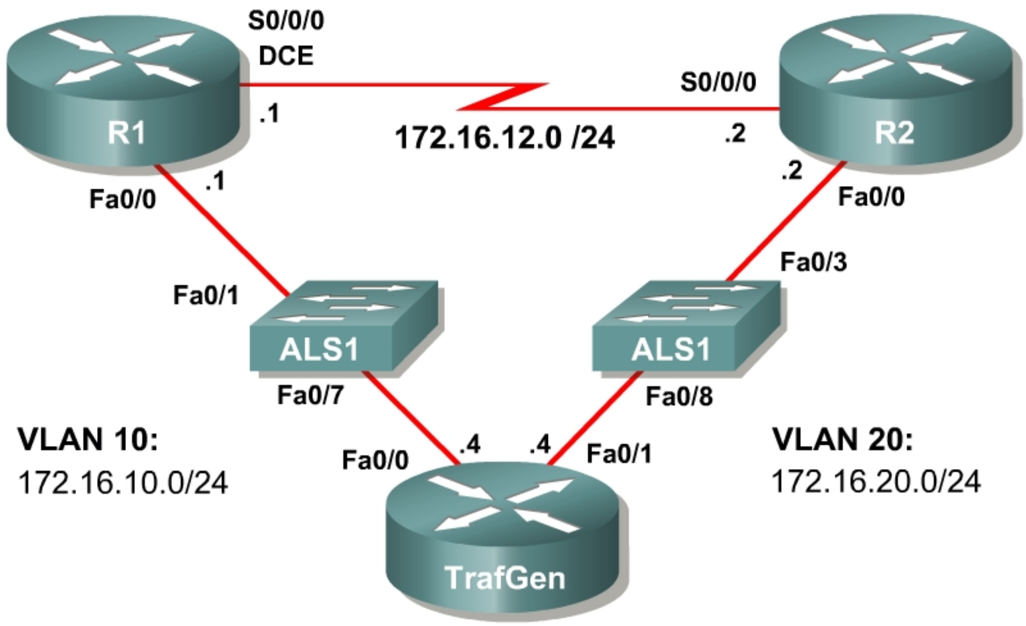
```
R1# show run
!
hostname R1
!
interface Serial0/0/0
 ip address 172.16.12.1 255.255.255.0
 ip tcp header-compression
 clock rate 64000
 no shutdown
!
end

R2# show run
!
hostname R2
!
interface Serial0/0/0
 ip address 172.16.12.2 255.255.255.0
 ip tcp header-compression
 no shutdown
!
line vty 0 4
 password cisco
 login
!
end
```


Lab 4.4 Comparing Queuing Strategies

Learning Objectives

- Implement FIFO, WFQ, CQ, and PQ queuing strategies
- Compare queuing strategies using the NQR tool

Topology Diagram



Scenario

This lab is designed as an integration lab to help you assess and recall skills learned in Labs 4.1 and 4.2. You will use some of the packet analysis tools available in the Pagent toolset to compare different queuing strategies and their impact on end-to-end quality of service (QoS). The four different queuing strategies that will be configured in this lab are first-in, first-out (FIFO), weighted fair queuing (WFQ), custom queuing (CQ), and priority queuing (PQ).

This is an investigative lab, so be sure to tweak the queuing strategies to ameliorate the results of your configurations. Compare results with classmates and contrast the configurations that provide those results.

Typically, commands and command output will only be shown if they have not been implemented in preceding labs, so it is highly recommended that you complete the previous labs to ensure knowledge of the queuing strategies and their configurations.

Preparation

This lab relies on the Basic Pagent Configuration, which you should have created in Lab 3.1: Preparing for QoS.

Prior to beginning this lab, configure R4 and the switch according to the Basic Pagent Configuration. You may easily accomplish this on R4 by loading the *basic-ios.cfg* file from flash memory into the NVRAM, and reloading.

```
TrafGen# copy flash:basic-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

On the switch, load the *basic.cfg* file into NVRAM and reload the device.

```
ALS1# copy flash:basic.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
ALS1# reload
Proceed with reload? [confirm]
```

Unlike Labs 4.1 and 4.2, this lab will use the NQR tool in the Pagent toolset rather than the TGN traffic generator. Do not load the TGN traffic generator configuration.

In addition, add the Fast Ethernet 0/3 interface on the switch to VLAN 20 since R2 will be the exit point from the network topology in this lab.

```
ALS1# configure terminal
ALS1(config)# interface fastethernet 0/3
ALS1(config-if)# switchport access vlan 20
ALS1(config-if)# switchport mode access
```

Step 1: Configure Addressing and Routing

Configure all IP addresses shown in the topology diagram and use a clock rate of 800 kbps on the serial link between R1 and R2. Set the informational bandwidth parameter appropriately on the serial interfaces.

Configure EIGRP AS 1 to include all networks shown in the diagram.

Step 2: Create NQR Configuration for Testing Purposes

Traffic generated from NQR, the traffic generation component of Pagent, requires almost all header fields to be hardcoded. Since the packets will be generated over Ethernet, you need to set the destination MAC address of the packets so that they are not broadcast. Remember that this is only the destination for the first hop, not the final destination MAC address. Use the **show interfaces** command to discover the value of the 48-bit MAC address.

Example:

```
R1# show interfaces fastethernet0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is MV96340 Ethernet, address is 0019.0623.4380 (bia 0019.0623.4380)
<OUTPUT OMITTED>
```

Use the MAC address on R1 as the Layer 2 destination of the NQR stream you will configure next.

On R4, issue the **nqr** command in privileged EXEC mode to enter NQR configuration mode. Then, copy and paste the NQR configuration shown below into a text editor, such as Notepad, and replace the $R1_MAC$ field with the MAC address you displayed in the output of the **show interfaces fastethernet 0/0** command. Then, copy and paste that configuration into the TrafGen router.

```
fastethernet0/0
add tcp
send 1000
rate 60
length random 200 to 1000
l2-dest $R1_MAC$
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 23
fastethernet0/1 capture
add clone-of 1
l4-dest 21
add clone-of 1
l4-dest 119
add clone-of 1
l4-dest 22
add clone-of 1
l4-dest 6000
```

To begin NQR testing, issue either the **start send** command in NQR configuration mode or the **nqr start send** command from privileged EXEC mode. Time will pass, and then the router will inform you when all packets have been sent. There is no need to stop the streams since they will stop on their own.

Finally, issue the **show pkt-seq-drop-stats**, **show delay**, and **show jitter** NQR commands to display drop/resequencing, delay, and jitter statistics, respectively. Example output is shown below, although this type of output will not be shown again later in the lab. Record all statistics by copying and pasting them into a text editor such as Notepad. Record a baseline reading for your current topology.

```
R4(NQR:OFF,Fa0/0:5/5)# start send
R4(NQR:SEND,Fa0/0:5/5)#

   Send process complete.
```



```
R4(NQR:WAIT,Fa0/0:5/5)#
R4(NQR:OFF,Fa0/0:5/5)# show pkt-seq-drop-stats

Summary of packet sequence/drop stats of traffic streams
  ts#   template interface      sent     recvd   dropped   out-of-seq  max-seq
  1     TCP      Fa0/0.10*      1000       625       375          271       28
  2     TCP      Fa0/0.10*      1000       637       363          271       30
  3     TCP      Fa0/0.10*      1000       638       362          254       30
  4     TCP      Fa0/0.10*      1000       598       402          265       29
  5     TCP      Fa0/0.10*      1000       604       396          267       28

R4(NQR:OFF,Fa0/0:5/5)# show delay-stats

Summary of delay-stats of traffic streams
 ts#    template interface    min-delay     max-delay     avg-delay  stdev-delay
  1     TCP      Fa0/0.10*     0.013646      0.433202      0.355633     0.047306
  2     TCP      Fa0/0.10*     0.012966      0.426203      0.352435     0.048258
  3     TCP      Fa0/0.10*     0.008824      0.436855      0.357987     0.046055
  4     TCP      Fa0/0.10*     0.028379      0.448521      0.361942     0.049450
  5     TCP      Fa0/0.10*     0.015277      0.457674      0.363785     0.046969

R4(NQR:OFF,Fa0/0:5/5)# show jitter-stats

Summary of jitter-stats of traffic streams
 ts#    template interface   min-jitter    max-jitter    avg-jitter stdev-jitter
  1     TCP      Fa0/0.10*     0.000063      0.204891      0.033416     0.034363
  2     TCP      Fa0/0.10*     0.000098      0.190365      0.034329     0.034809
  3     TCP      Fa0/0.10*     0.000015      0.172803      0.033511     0.032503
  4     TCP      Fa0/0.10*     0.000047      0.223152      0.035887     0.034892
  5     TCP      Fa0/0.10*     0.000070      0.165289      0.035484     0.031709
```

Step 3: Test FIFO Queuing

This lab will compare four different queuing types. The first type is the most basic, FIFO queuing.

Configure FIFO queuing on the serial interface on R1. Recall that disabling all other queuing strategies on an interface will enable FIFO queuing.

Notice that the scenario the authors have designed overpowers all of the queuing mechanisms implemented because there is simply much more traffic than the bandwidth of the serial link. If you had this ratio of legitimate traffic to bandwidth in a production network, then queuing strategies would not solve the problem. It would be necessary to obtain additional bandwidth.

Step 4: Test Weighted Fair Queuing

Enable WFQ on the serial interface. Run the NQR streams again using **nqr start send** and compare the results of the **show** commands.

Is there a significant difference between the statistics using WFQ and FIFO in this scenario?

The streams from NQR are generated in something similar to a round-robin fashion with the same number of packets for each stream. The result is that many of the same packets will be forwarded by WFQ as by FIFO, but this is only by the construction of the streams on TrafGen. In real networks, many traffic patterns are bursty, unlike this simulation. To understand what is meant by bursty traffic patterns, think of loading a web page. You type in a URL and there is a burst of traffic as the text and the graphics load. Then while you read the web page, there is no additional traffic being sent across the network. Then you click on a link, and another burst of traffic traverses the network.

What effect does the function of the NQR generator have on your results?

Provide a circumstance in which you would expect a different result from FIFO?

Step 5: Test Custom Queuing

Configure custom queuing (CQ) on R1's serial interface. Place each traffic stream in its own queue but do not customize any parameters of it. (The port numbers configured for the NQR streams are TCP ports 23, 21, 119, 22, and 6000) Run the NQR streams and compare results as you did before.

Contrast the results for the CQ test with those of the previous queuing strategies:

Try making one of the queues have a size of 10000. How does this affect all of the traffic flows?

Step 6: Test Priority Queuing

Configure priority queuing (PQ) on R1 on the serial interface facing R2. Assign one of the application protocols in use to the high priority queue, one to the medium queue, one to the normal queue, and make the low priority queue the default queue. Run the NQR streams and compare results as you did before.

How does the packet loss with PQ compare to that of previous queuing strategies?

What would happen if you put all the streams in the high priority queue?

Final Configurations

```
R1# show run
!
hostname R1
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 ip address 172.16.12.1 255.255.255.0
 priority-group 1
 clock rate 800000
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
queue-list 1 protocol ip 1 tcp telnet
queue-list 1 protocol ip 2 tcp ftp
queue-list 1 protocol ip 3 tcp nntp
queue-list 1 protocol ip 4 tcp 22
queue-list 1 default 5
queue-list 1 queue 1 byte-count 10000
priority-list 1 protocol ip high tcp telnet
priority-list 1 protocol ip medium tcp ftp
priority-list 1 protocol ip normal tcp 22
priority-list 1 default low
```



```
!
end
```

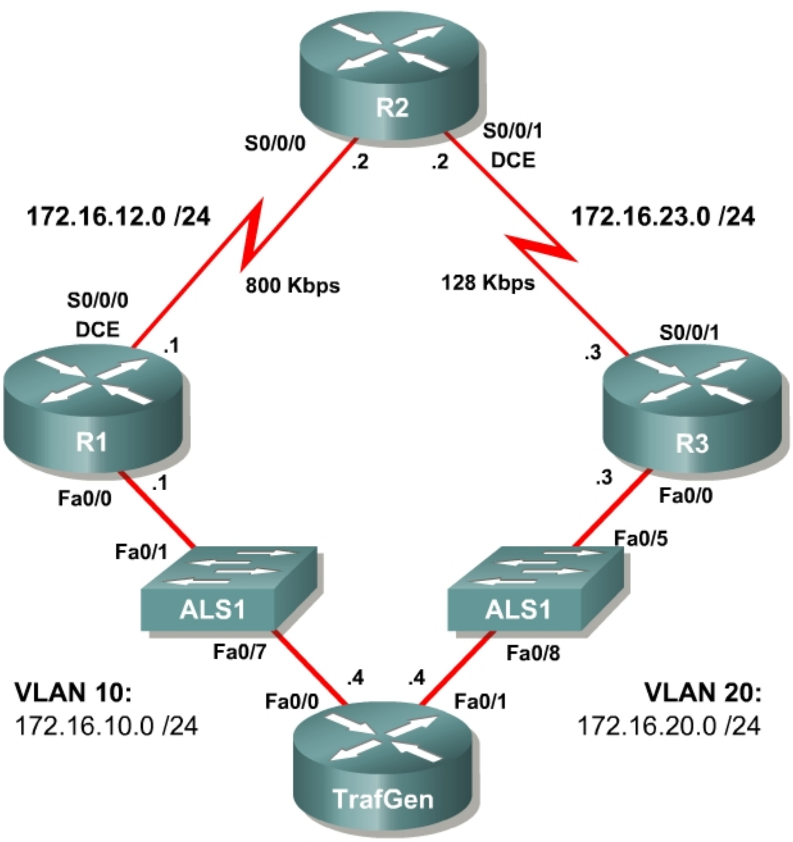
```
R2# show run
!
hostname R2
!
interface FastEthernet0/0
 ip address 172.16.20.2 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 ip address 172.16.12.2 255.255.255.0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
end
```


Lab 4.5 Class-based Queuing and NBAR

Learning Objectives

- Utilize NBAR for protocol detection
- Mark IP Precedence
- Allocate bandwidth using the Modular QoS Command-Line Interface
- Configure CBWFQ and LLQ queuing strategies

Topology Diagram



Scenario

In this lab, you will implement classification using Network-based Application Recognition (NBAR) and the Modular QoS CLI (MQC) to configure quality of service (QoS) on R1 and R2. You will configure both class-based marking and class-based queuing algorithms.

Preparation

This lab uses the Basic Pagent Configuration for TrafGen and the switch to generate and facilitate lab traffic in a stream from TrafGen to R1 to R2. Prior to beginning this lab, configure TrafGen (R4) and the switch according to the Basic Pagent Configuration in Lab 3.1: Preparing for QoS. You can accomplish this on R4 by loading the *basic-ios.cfg* file from flash memory into the NVRAM and reloading.

```
TrafGen# copy flash:basic-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

On the switch, load the *basic.cfg* file into NVRAM and reload the device.

```
Switch# copy flash:basic.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

On TrafGen, instruct TGN to load the *basic-tgn.cfg* file and to start generating traffic.

```
TrafGen> enable
TrafGen# tgn load-config
TrafGen# tgn start
```

In addition, add the Fast Ethernet 0/5 interface on the switch to VLAN 20 since R3 will be the exit point from the network topology in this lab.

```
Switch# configure terminal
Switch(config)# interface fastethernet 0/5
Switch(config-if)# switchport access vlan 20
Switch(config-if)# switchport mode access
```

Step 1: Configure the Physical Interfaces

Configure all of the physical interfaces shown in the diagram. Set the clock rate on the serial link between R1 and R2 to 800000, the clock rate of the serial link between R2 and R3 to be 128000, and use the **no shutdown** command on all interfaces. Set the informational bandwidth parameter on the serial interfaces.

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface serial 0/0/0
R1(config-if)# bandwidth 800
R1(config-if)# ip address 172.16.12.1 255.255.255.0
R1(config-if)# clock rate 800000
```



```
R1(config-if)# no shutdown

R2(config)# interface serial 0/0/0
R2(config-if)# bandwidth 800
R2(config-if)# ip address 172.16.12.2 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# interface serial 0/0/1
R2(config-if)# bandwidth 128
R2(config-if)# ip address 172.16.23.2 255.255.255.0
R2(config-if)# clock rate 128000
R2(config-if)# no shutdown

R3(config)# interface fastethernet 0/0
R3(config-if)# ip address 172.16.20.3 255.255.255.0
R3(config-if)# no shutdown
R3(config-if)# interface serial 0/0/1
R3(config-if)# bandwidth 128
R3(config-if)# ip address 172.16.23.3 255.255.255.0
R3(config-if)# no shutdown
```

Issue the **show interfaces serial 0/0/0 | include Queueing** command on R1 to verify that the queuing strategy is Weighted Fair Queuing (WFQ).

```
R1# show interface serial0/0/0 | include Queueing
  Queueing strategy: weighted fair
```

If you see “fifo” as the queuing type, use the interface-level command **fair-queue** on the serial interface.

Step 2: Configure EIGRP AS 1

Configure routing between R1, R2 and R3 using Enhanced Interior Gateway Routing Protocol (EIGRP). Include the entire 172.16.0.0/16 major network in AS 1 and disable automatic summarization.

```
R1(config)# router eigrp 1
R1(config-router)# no auto-summary
R1(config-router)# network 172.16.0.0

R2(config)# router eigrp 1
R2(config-router)# no auto-summary
R2(config-router)# network 172.16.0.0

R3(config)# router eigrp 1
R3(config-router)# no auto-summary
R3(config-router)# network 172.16.0.0
```

Verify that the number of packets counted is increasing on the outbound interface of R3. Use the **show interfaces fastethernet 0/1** command**.** Issue the command twice to make sure the number of packets output has changed. If the number is not increasing, troubleshoot Layers 1, 2, and 3 connectivity and the EIGRP topology.

Step 3: Configure NBAR Protocol Discovery

NBAR is an IOS QoS feature that allows QoS decisions to be made based on individual protocols. Access control lists (ACLs) can be used to classify traffic based on headers for Layers 1 through 4 of the OSI model. NBAR, on the other hand, allows classification based on the upper layers of the OSI model—Layers 4 through 7. Since it does not rely on TCP/UDP port numbers at Layer 4, it can be used to identify traffic from applications that have dynamic port assignments. One standard feature of NBAR, known as protocol discovery, allows you to dynamically learn which application protocols are in use on your network. NBAR Protocol Discovery can also record and display the most used protocols.

For this lab, configure NBAR Protocol Discovery on the Fast Ethernet 0/0 interface on R1. The only IP traffic leaving the interface will be EIGRP Hello packets, so the majority of packets you should expect to see will be in the inbound direction. The protocols that protocol discovery shows heavy inbound traffic for are the protocols that traffic generation was configured for. To enable protocol discovery, use the interface-level command **ip nbar protocol-discovery**.

```
R1(config)# interface fastethernet0/0
R1(config-if)# ip nbar protocol-discovery
```

After protocol discovery has been enabled for a minute or two, you can see the information it has collected by using the command **show ip nbar protocol-discovery**. This command displays statistics globally for every interface in which NBAR protocol discovery is enabled. The protocols will be ranked based on traffic usage per interface. Notice that ingress and egress traffic is separated as it is in the output of the **show interfaces** command.

```
R1# show ip nbar protocol-discovery

 FastEthernet0/0
                           Input                    Output
                           -----                    ------
   Protocol                Packet Count             Packet Count
                           Byte Count               Byte Count
                           5min Bit Rate (bps)      5min Bit Rate (bps)
                           5min Max Bit Rate (bps)  5min Max Bit Rate (bps)
   ------------------------ ------------------------ ------------------------
   ssh                     47691                    0
                           37214753                 0
                           800000                   0
                           800000                   0
   xwindows                46638                    0
                           36235048                 0
                           797000                   0
                           797000                   0
   pop3                    47549                    0
                           37165341                 0
                           796000                   0
                           796000                   0
   smtp                    47112                    0
                           36874672                 0
```



```
                         794000                   0
                         794000                   0
   http                  47099                    0
                         36687939                 0
                         791000                   0
                         791000                   0
   ntp                   44401                    0
                         34670597                 0
                         770000                   0
                         770000                   0
   ftp                   45142                    0
                         35185881                 0
                         767000                   0
                         767000                   0
   telnet                44322                    0
                         34652510                 0
                         762000                   0
                         762000                   0
   eigrp                 0                        17
                         0                        1258
                         0                        0
                         0                        0
<OUTPUT OMITTED>
```

NBAR uses a preconfigured set of port numbers, which it references during protocol discovery and normal classification operation. Issue the **show ip nbar port-map** command to view the protocol-to-port mappings. This command can also come in handy if you need to find out a well-known port number for an application and do not have access to outside resources. Existing protocol mappings can be modified and custom protocols can be defined, but those NBAR features are outside of the scope of this lab.

```
R1# show ip nbar port-map
port-map bgp                      udp 179
port-map bgp                      tcp 179
port-map bittorrent               tcp 6881 6882 6883 6884 6885 6886 6887 6888
6889
port-map citrix                   udp 1604
port-map citrix                   tcp 1494
port-map cuseeme                  udp 7648 7649 24032
port-map cuseeme                  tcp 7648 7649
port-map dhcp                     udp 67 68
port-map directconnect            tcp 411 412 413
port-map dns                      udp 53
port-map dns                      tcp 53
port-map edonkey                  tcp 4662
port-map exchange                 tcp 135
port-map fasttrack                tcp 1214
port-map finger                   tcp 79
port-map ftp                      tcp 21
port-map gnutella                 udp 6346 6347 6348
port-map gnutella                 tcp 6346 6347 6348 6349 6355 5634
port-map gopher                   udp 70
port-map gopher                   tcp 70
port-map h323                     udp 1300 1718 1719 1720 11720
port-map h323                     tcp 1300 1718 1719 1720 11000 – 11999
<OUTPUT OMITTED>
```

According to best QoS practices, where should packets be marked?

What is a trust boundary in terms of classification and marking?

Step 4: Classify and Mark Packets

The Modular QoS CLI (MQC) allows someone to create QoS policies on a router in a modular and easy-to-understand format. When creating QoS policies using MQC, there are normally three configuration tasks:

1. Define traffic classes and the method of classification. Classes of traffic are defined in class maps using match statements. The match criterion can be an access list, NBAR-recognized protocol, QoS marking, packet size, and so forth.

2. Create a QoS policy to provision network resources for any traffic classes created in Step 1. A QoS policy maps QoS actions, such as marking, queuing, shaping, policing, or compression, to selected classes.

3. Finally, the policy is applied to an interface directionally, in either the inbound or outbound direction.

Certain policy-map commands can only be applied in a specific direction. For instance, queuing strategies can only be applied in the outbound policies. The router sends an error message to the console if a queuing policy is applied to an interface in the inbound direction, because this is an impossible configuration option.

On R1, you will create a QoS policy to mark an IP Precedence based on the application-layer protocol of the packets. The 3-bit IP Precedence field is part of the legacy Type of Service (ToS) byte on IP packets. Internet standards later converted this byte to the differentiated services (DiffServ) byte which contained the 6-bit differentiated services code point (DSCP) field. The three bits of the IP Precedence field map to the three high-order bits of the DSCP field for backwards-compatibility. For instance, WFQ does not look at the three low-order bits in the DSCP field, but does set weights for each flow based on the three high-order bits of the ToS/DS byte that are used for the IP Precedence

You will apply this QoS policy outbound on R1’s Serial 0/0/0 interface.

Begin by implementing the first task: classification. Create traffic classes using NBAR for protocol recognition.

Class-maps are defined with the global configuration command **class-map** [*match-type*] *name*. The optional *match-type* argument can be set to either **match-any** or the default, **match-all**. This argument defines whether all of the successive match statements must be met in order for traffic to be classified into this class, or if only one is necessary.

Once in the class-map configuration mode, matching criteria can be defined with the **match** *criteria* command. To view all the possibilities of what can be matched on, use the **?** command. Choose to use NBAR for classification using the **match protocol** *name* command.

Create three traffic classes:

Critical: EIGRP or Network Time Protocol (NTP) traffic. These protocols are used for network control.

Interactive: Telnet, SSH, and XWindows traffic. These protocols are used for remote administration.

Web: HTTP, POP3, and SMTP traffic. These protocols are used for web and email access.

When creating these traffic classes, should you use the **match-any** or the **match-all** keyword?

The classes created must match with the match-any mode so that any of the protocols listed can be matched. Obviously, it would be impossible for a packet to be two protocols at once.

```
R1(config)# class-map match-any critical
R1(config-cmap)# match ?
  access-group         Access group
  any                  Any packets
  class-map            Class map
  cos                  IEEE 802.1Q/ISL class of service/user priority values
  destination-address  Destination address
  discard-class        Discard behavior identifier
  dscp                 Match DSCP in IP(v4) and IPv6 packets
  flow                 Flow based QoS parameters
  fr-de                Match on Frame-relay DE bit
  fr-dlci              Match on fr-dlci
  input-interface      Select an input interface to match
  ip                   IP specific values
  mpls                 Multi Protocol Label Switching specific values
  not                  Negate this match result
  packet               Layer 3 Packet length
  precedence           Match Precedence in IP(v4) and IPv6 packets
```



```
  protocol             Protocol
  qos-group            Qos-group
  source-address       Source address
  vlan                 VLANs to match
R1(config-cmap)# match protocol eigrp
R1(config-cmap)# match protocol ntp
R1(config-cmap)# class-map match-any interactive
R1(config-cmap)# match protocol telnet
R1(config-cmap)# match protocol ssh
R1(config-cmap)# match protocol xwindows
R1(config-cmap)# class-map match-any web
R1(config-cmap)# match protocol http
R1(config-cmap)# match protocol pop3
R1(config-cmap)# match protocol smtp
```

You can verify created class-maps with the command **show class-map**.

```
R1# show class-map
 Class Map match-any critical (id 1)
   Match protocol eigrp
   Match protocol ntp

 Class Map match-any class-default (id 0)
   Match any

 Class Map match-any interactive (id 2)
   Match protocol telnet
   Match protocol ssh
   Match protocol xwindows

 Class Map match-any web (id 3)
   Match protocol http
   Match protocol pop3
   Match protocol smtp
```

The next task will be to define the QoS policy in a policy map. Create a policy map in global configuration mode using the **policy-map** *name* command. Segment the policy map by traffic class by issuing the **class** *name* command. The names of the classes will be the same as the class maps you created above. Additionally, there is the built-in class "class-default," which matches any traffic not included in any other class.

```
R1(config)# policy-map markingpolicy
```

At the class configuration prompt, you can use various commands that will affect traffic of that class (use **?** to see what is available). To modify packets, use the command **set** *property value*. Create a new policy named "markingpolicy" and set the IP Precedence for matched packets as follows:

Critical: Set the IP Precedence to Network Control, represented by the value 7.

Interactive: Set the IP Precedence to Critical, represented by the value 5.

Web: Set the IP Precedence to Flash, represented by the value 3.

All other traffic: Set the IP Precedence of all other traffic to Routine, represented by the value 0. This value is the default value for IP Precedence.

There are different names for each value (these can be found out with the **?** command, and this is shown in the following output for reference).

```
R1(config-pmap)# class critical
R1(config-pmap-c)# set precedence ?
  <0-7>           Precedence value
  cos             Set packet precedence from L2 COS
  critical        Set packets with critical precedence (5)
  flash           Set packets with flash precedence (3)
  flash-override  Set packets with flash override precedence (4)
  immediate       Set packets with immediate precedence (2)
  internet        Set packets with internetwork control precedence (6)
  network         Set packets with network control precedence (7)
  priority        Set packets with priority precedence (1)
  qos-group       Set packet precedence from QoS Group.
  routine         Set packets with routine precedence (0)

R1(config-pmap-c)# set precedence 7
R1(config-pmap-c)# class interactive
R1(config-pmap-c)# set precedence 5
R1(config-pmap-c)# class web
R1(config-pmap-c)# set precedence 3
R1(config-pmap-c)# class class-default
R1(config-pmap-c)# set precedence 0
```

Verify the policy map configuration using the **show policy-map** command.

```
R1# show policy-map
  Policy Map markingpolicy
    Class critical
      set precedence 7
    Class interactive
      set precedence 5
    Class web
      set precedence 3
    Class class-default
      set precedence 1
```

Finally, apply the configuration outbound towards R2 with the interface-level command **service-policy** *direction name*.

```
R1(config)# interface serial 0/0/0
R1(config-if)# service-policy output markingpolicy
```

Once a policy map is applied to an interface, you can use an extended form of the **show policy-map** command by issuing the **show policy-map interface** *interface-name* command. This will give you detailed information and statistics on policy maps applied to an interface.

```
R1# show policy-map interface serial0/0/0
 Serial0/0/0

  Service-policy output: markingpolicy

    Class-map: critical (match-any)
```



```
      13822 packets, 10617832 bytes
      5 minute offered rate 264000 bps, drop rate 0 bps
      Match: protocol eigrp
        5 packets, 320 bytes
        5 minute rate 0 bps
      Match: protocol ntp
        13817 packets, 10617512 bytes
        5 minute rate 264000 bps
      QoS Set
        precedence 7
          Packets marked 13822

    Class-map: interactive (match-any)
      44974 packets, 34630670 bytes
      5 minute offered rate 830000 bps, drop rate 0 bps
      Match: protocol telnet
        15300 packets, 11765411 bytes
        5 minute rate 289000 bps
      Match: protocol ssh
        14451 packets, 11209788 bytes
        5 minute rate 270000 bps
      Match: protocol xwindows
        15223 packets, 11655471 bytes
        5 minute rate 282000 bps
      QoS Set
        precedence 5
          Packets marked 44984

    Class-map: web (match-any)
      44600 packets, 34404320 bytes
      5 minute offered rate 857000 bps, drop rate 0 bps
      Match: protocol http
        13688 packets, 10530109 bytes
        5 minute rate 269000 bps
      Match: protocol pop3
        14513 packets, 11240708 bytes
        5 minute rate 290000 bps
      Match: protocol smtp
        16399 packets, 12633503 bytes
        5 minute rate 312000 bps
      QoS Set
        precedence 3
          Packets marked 44620

    Class-map: class-default (match-any)
      13745 packets, 10547088 bytes
      5 minute offered rate 261000 bps, drop rate 0 bps
      Match: any
      QoS Set
        precedence 0
          Packets marked 13743
```

If a BGP packet with an IP precedence marking of 3 enters the Fast Ethernet 0/0 interface on R1 and is destined for R2, into which traffic class will the packet be classified?

What IP precedence will the packet be assigned at the egress port?

Step 5: Shape Traffic and Queue with CBWFQ and LLQ

One of the QoS actions that can be performed in a policy map is shaping. Shaping limits traffic for a traffic class to a specific rate and buffers excess traffic. Policing, a related concept drops the excess traffic. Thus, the purpose of shaping is to buffer traffic so that more traffic is sent than if you policed at the same rate because not only will the traffic conforming to the policy be sent, but also buffered excess traffic when permitted.

Policing and shaping can each be configured within a policy map as a QoS action for a specific traffic class, or you can nest policy maps to create an aggregate shaper or policer. Multiple QoS actions can be taken on a specific class of traffic so you could use shaping in conjunction with marking or compression, or various other actions. Keep this in mind for the remaining labs

The first task in creating the QoS policy is to enumerate classes. This time, use uncreative names such as “prec7” and “prec5” for packets with IP Precedences 7 and 5, respectively. Create classes like this for IP Precedences 0, 3, 5, and 7—the in Module 4.

In this circumstance, however, you will view the class-based shapers in conjunction with low-latency queuing (LLQ). There are two class-based queuing tools, class-based weighted fair queuing (CBWFQ) and low-latency queuing (LLQ). CBWFQ is similar to custom queuing (CQ) in that it provisions an average amount or percent of bandwidth to a traffic class. However, the classification mechanism in class-based tools is much more powerful because it can also use NBAR to discover application protocols and even application protocol parameters, such as the URL in an HTTP request. LLQ is a simple improvement on CBWFQ, adding the ability to designate some classes as priority traffic and ensure that they are sent before others.

On R2, create a policy map to be applied on its Serial 0/0/1 interface. This policy map will be used to shape traffic based on markings by R1.possibilities for marking from the last step. To match on IP Precedence in a class definition, use the **match precedence** *precedence* command, where the *precedence* argument is the value or representative name. You must reclassify and mark EIGRP packets because each of the EIGRP packets is link-local traffic and the EIGRP packets which you marked on ingress at R1 were not sent to R2. The new packets for the link between R1 and R2 must now be classified by an access list or NBAR. However, any NTP packets traversing the link will already be marked with IP precedence 7. You should to treat EIGRP and NTP packets in the same traffic class for consistency.

Would you use the **match-all** or **match-any** keyword when creating the “prec7” class map? Explain.

Create the class map as follows.

```
R2(config)# class-map prec0
R2(config-cmap)# match precedence 0
R2(config-cmap)# class-map prec3
R2(config-cmap)# match precedence 3
R2(config-cmap)# class-map prec5
R2(config-cmap)# match precedence 5
R2(config-cmap)# class-map match-any prec7
R2(config-cmap)# match precedence 7
R2(config-cmap)# match protocol eigrp
```

Next, create the QoS policy to shape and queue the traffic. The syntax for entering the policy map and per-class configuration will be the same as above. However, rather than changing packet properties, we will set up low-latency queuing (LLQ) for the interface. LLQ is a variant of class-based weighted fair queuing (CBWFQ). Configuring CBWFQ involves assigning each traffic class dedicated bandwidth, either through exact bandwidth amounts or relative percentage amounts. LLQ is the configured the same way, except that one or more traffic classes are designated as priority traffic and assigned to an expedite queue. All traffic that enters the expedite queue up to the bandwidth limit will be sent as soon as possible, preempting traffic from non-priority classes.

While you configure either CBWFQ or LLQ, you can allocate a certain bandwidth for a traffic class, using the **bandwidth** *rate* command, where *rate* is a bandwidth amount in kilobits per second. Alternatively, use the **bandwidth percentage** *percent* command to allocate a percentage of bandwidth, where 100 percent of the bandwidth is set by the informational bandwidth parameter that you configured in Step 1.

For LLQ solely, issue the **priority** *rate* command or the **priority percentage** *percent* command in policy map configuration mode. These commands have the same arguments, which have the same effect as the **bandwidth** commands, except that they designate that queue as the priority queue.

Create a policy named “llqpolicy” on R2. The policy should allocate 10 percent of traffic to the “prec7” traffic class, 15 percent to the “prec5” traffic class, 30 percent to the “prec3” traffic class, and 20 percent to the “prec0” traffic class. Expedite traffic that falls into the “prec7” traffic class. Also, select weighted fair-queuing as the queuing method in the default traffic class with the **fair-queue** command.

```
R2(config)# policy-map llqpolicy
R2(config-pmap)# class prec7
R2(config-pmap-c)# priority percent 10
R2(config-pmap-c)# class prec5
R2(config-pmap-c)# bandwidth percent 15
R2(config-pmap-c)# class prec3
R2(config-pmap-c)# bandwidth percent 30
R2(config-pmap-c)# class prec0
R2(config-pmap-c)# bandwidth percent 20
R2(config-pmap-c)# class class-default
R2(config-pmap-c)# fair-queue
```

Verify your QoS policy configuration using the **show policy-map** command. Notice that the priority queue is a variant on the regular queues.

```
R2# show policy-map
  Policy Map llqpolicy
    Class prec7
      Strict Priority
      Bandwidth 10 (%)
    Class prec5
      Bandwidth 15 (%) Max Threshold 64 (packets)
    Class prec3
      Bandwidth 30 (%) Max Threshold 64 (packets)
    Class prec0
      Bandwidth 20 (%) Max Threshold 64 (packets)
    Class class-default
      Flow based Fair Queueing
      Bandwidth 0 (kbps) Max Threshold 64 (packets)
```

What traffic types would usually belong in a priority queue in a production environment?

Use the same **service-policy** command from earlier to apply this policy map to the Serial 0/0/1 interface on R2 in an outbound direction.

```
R2(config)# interface serial 0/0/1
R2(config-if)# service-policy output llqpolicy
```

Verify using the interface-specific version of **show policy-map**.

```
R2# show policy-map interface serial0/0/1
 Serial0/0/1

  Service-policy output: llqpolicy
```



```
Class-map: prec7 (match-any)
  3995 packets, 3387767 bytes
  5 minute offered rate 81000 bps, drop rate 80000 bps
  Match:  precedence 7
    3941 packets, 3384319 bytes
    5 minute rate 81000 bps
  Match: protocol eigrp
    54 packets, 3448 bytes
    5 minute rate 0 bps
  Queueing
    Strict Priority
    Output Queue: Conversation 40
    Bandwidth 10 (%)
    Bandwidth 12 (kbps) Burst 300 (Bytes)
    (pkts matched/bytes matched) 3947/3384695
    (total drops/bytes drops) 3524/3314514

Class-map: prec5 (match-all)
  8378 packets, 7165609 bytes
  5 minute offered rate 165000 bps, drop rate 146000 bps
  Match:  precedence 5
  Queueing
    Output Queue: Conversation 41
    Bandwidth 15 (%)
    Bandwidth 19 (kbps)Max Threshold 64 (packets)
    (pkts matched/bytes matched) 8378/7165609
    (depth/total drops/no-buffer drops) 64/7459/0

Class-map: prec3 (match-all)
  10295 packets, 8813462 bytes
  5 minute offered rate 197000 bps, drop rate 163000 bps
  Match:  precedence 3
  Queueing
    Output Queue: Conversation 42
    Bandwidth 30 (%)
    Bandwidth 38 (kbps)Max Threshold 64 (packets)
    (pkts matched/bytes matched) 10293/8810571
    (depth/total drops/no-buffer drops) 64/8500/0

Class-map: prec0 (match-all)
  3239 packets, 2830395 bytes
  5 minute offered rate 73000 bps, drop rate 52000 bps
  Match:  precedence 0
  Queueing
    Output Queue: Conversation 43
    Bandwidth 20 (%)
    Bandwidth 25 (kbps)Max Threshold 64 (packets)
    (pkts matched/bytes matched) 3239/2830395
    (depth/total drops/no-buffer drops) 60/1988/0

Class-map: class-default (match-any)
  26 packets, 1524 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: any
  Queueing
    Flow Based Fair Queueing
    Maximum Number of Hashed Queues 32
    (total queued/total drops/no-buffer drops) 0/0/0
```


Challenge: Verifying IP Precedence

The topic of IP accounting is outside the scope of this curriculum. However, it is a useful tool for the verification of a marking policy. Issue the **ip accounting precedence** *direction* command in interface configuration mode to enable IP accounting on an interface. Apply this command on R3 for the Serial 0/0/1 interface that shows incoming markings from R2. View the accounting records for IP precedence by issuing the **show interfaces precedence** command.

```
R3(config)# interface serial0/0/1
R3(config-if)# ip accounting precedence input

R3# show interface precedence
Serial0/0/1
  Input
    Precedence 0:  10 packets, 5121 bytes
    Precedence 1:  230 packets, 85385 bytes
    Precedence 3:  193 packets, 127000 bytes
    Precedence 5:  88 packets, 62727 bytes
    Precedence 6:  5 packets, 320 bytes
    Precedence 7:  148 packets, 16984 bytes
```

Can you think of another simple way to count packets with each IP Precedence marking? You do not need to actually implement it. HINT: Think access lists.

Final Configurations

```
R1# show run
hostname R1
!
class-map match-any critical
 match protocol eigrp
 match protocol ntp
class-map match-any interactive
 match protocol telnet
 match protocol ssh
 match protocol xwindows
class-map match-any web
 match protocol http
 match protocol pop3
 match protocol smtp
!
policy-map markingpolicy
 class critical
  set precedence 7
 class interactive
  set precedence 5
 class web
  set precedence 3
 class class-default
```



```
  set precedence 0
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 ip nbar protocol-discovery
 no shutdown
!
interface Serial0/0/0
 ip address 172.16.12.1 255.255.255.0
 clock rate 800000
 service-policy output markingpolicy
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
end
```

```
R2# show run
hostname R2
!
class-map match-all prec5
 match  precedence 5
class-map match-any prec7
 match  precedence 7
 match protocol eigrp
class-map match-all prec0
 match  precedence 0
class-map match-all prec3
 match  precedence 3
!
policy-map llqpolicy
 class prec7
  priority percent 10
 class prec5
  bandwidth percent 15
 class prec3
  bandwidth percent 30
 class prec0
  bandwidth percent 20
 class class-default
  fair-queue
!
interface Serial0/0/0
 ip address 172.16.12.2 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 bandwidth 128
 ip address 172.16.23.2 255.255.255.0
 clock rate 128000
 service-policy output llqpolicy
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
end
```

```
R3# show run
hostname R3
!
```



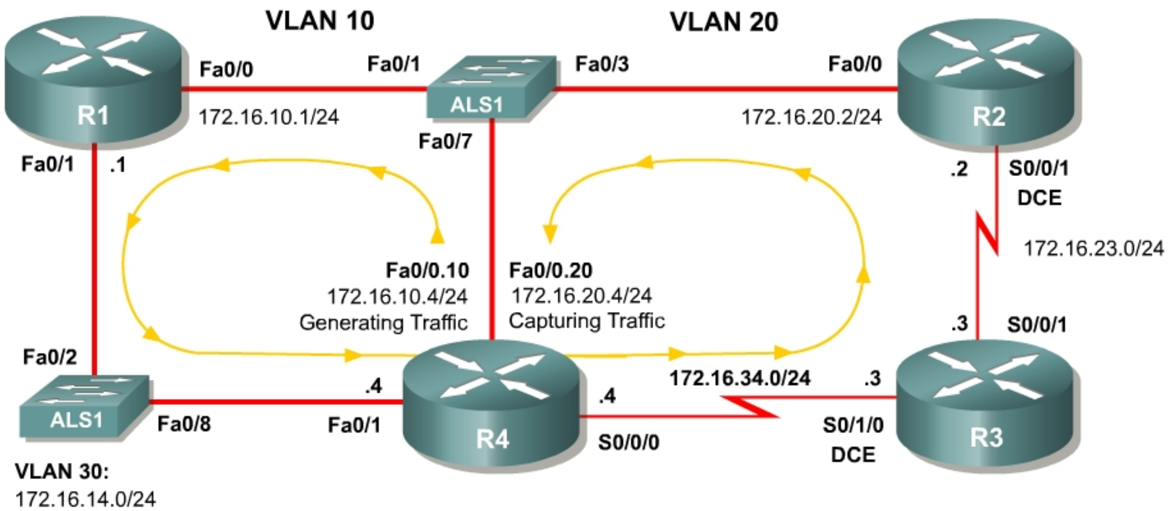
```
interface FastEthernet0/1
 ip address 172.16.20.3 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 ip address 172.16.23.3 255.255.255.0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
end
```


Lab 4.6 Class-based Marking, Shaping, and Policing

Learning Objectives

- Mark packets with DSCP values
- Implement class-based TCP Header Compression
- Configure class-based traffic shaping and policing
- Create and apply nested service policies

Topology Diagram



Scenario

In this lab, you will implement classification using network-based application recognition (NBAR) using the Modular QoS CLI (MQC) to configure quality of service on R1 and R2. You will configure class-based marking, shaping, and policing mechanisms.

You should complete Lab 4.5 before beginning this lab because this lab will build on the concepts of NBAR and marking that you configured in that scenario.

Preparation

This lab relies on the Advanced Pagent Configuration, which you should have created in Lab 3.1: Preparing for QoS.

Prior to beginning this lab, configure R4 and the switch according to the Advanced Pagent Configuration. You may easily accomplish this on R4 by loading the *advanced-ios.cfg* file from flash memory into the NVRAM, and reloading.

```
TrafGen# copy flash:advanced-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

On the switch, load the *advanced.cfg* file into NVRAM and reload the device.

```
ALS1# copy flash:advanced.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
ALS1# reload
Proceed with reload? [confirm]
```

Next, instruct TGN to load the *advanced-tgn.cfg* file. At the end of Step 1, you will begin generating TGN traffic.

```
TrafGen# tgn load-config advanced-tgn.cfg
```

Step 1: Configure the Physical Interfaces

Configure all of the physical interfaces shown in the diagram. Set the clock rate on both serial links to 800000 bits per second and use the **no shutdown** command on all necessary interfaces. Set the informational bandwidth parameter appropriately on the serial interfaces.

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface fastethernet 0/1
R1(config-if)# ip address 172.16.14.1 255.255.255.0
R1(config-if)# no shutdown

R2(config)# interface serial 0/0/1
R2(config-if)# bandwidth 800
R2(config-if)# ip address 172.16.23.2 255.255.255.0
R2(config-if)# clockrate 800000
R2(config-if)# no shutdown
R2(config-if)# interface fastethernet 0/0
R2(config-if)# ip address 172.16.20.2 255.255.255.0
R2(config-if)# no shutdown

R3(config)# interface serial 0/0/1
R3(config-if)# bandwidth 800
R3(config-if)# ip address 172.16.23.3 255.255.255.0
R3(config-if)# no shutdown
R3(config-if)# interface serial 0/1/0
R3(config-if)# bandwidth 800
R3(config-if)# ip address 172.16.34.3 255.255.255.0
R3(config-if)# clockrate 800000
R3(config-if)# no shutdown

R4(config)# interface fastethernet 0/1
R4(config-if)# ip address 172.16.14.4 255.255.255.0
```



```
R4(config-if)# no shutdown
R4(config-if)# interface serial 0/0/0
R3(config-if)# bandwidth 800
R4(config-if)# ip address 172.16.34.4 255.255.255.0
R4(config-if)# no shutdown
```

Now that R4 can reach R1 172.16.10.1 address via ARP, begin generating TGN traffic.

```
TrafGen# tgn start
```

Step 2: Configure Routing

Establish adjacencies for routing with Open Shortest Path First (OSPF). Include all connected subnets within the 172.16.0.0/16 major network for all four routers.

```
R1(config)# router ospf 1
R1(config-router)# network 172.16.0.0 0.0.255.255 area 0

R2(config)# router ospf 1
R2(config-router)# network 172.16.0.0 0.0.255.255 area 0

R3(config)# router ospf 1
R3(config-router)# network 172.16.0.0 0.0.255.255 area 0

R4(config)# router ospf 1
R4(config-router)# network 172.16.0.0 0.0.255.255 area 0
```

Step 3: Mark Packets with DSCP

Various Internet Engineering Task Force Request for Comments (IETF RFCs) have outlined a set of quality of service (QoS) per-hop behaviors (PHBs). These RFCs define a marking scheme as well as a set of actions or preferences to be followed at each hop as that data packet traverses the routed path. These RFCs build on the redefinition of the markable byte in the IP header from type of service (ToS) to differentiated services (DiffServ). These standardized PHBs define marking scheme to set six bits in the DiffServ Code Point (DSCP) field.

According to the PHB RFCs, a DSCP marking is slightly different than IP Precedence, in that it includes the queuing treatment and drop probability. Since the DiffServ byte overlaps the legacy ToS byte in an IP packet, DSCP values are backwards-compatible in networks or QoS tools that rely solely on IP Precedence. You can mark IP packets with two different types of DSCP markings: Expedited Forwarding (EF) for priority traffic (such as voice packets), and Assured Forwarding (AF). Simply marking traffic correctly does not configure the QoS tools to implement the various PHBs. However, markings with standardized meanings can drastically improve the understanding of QoS in a network.

There are no classes of EF traffic, but the RFCs define multiple classes within the AF marking. The names for the AF classes follow the pattern AF*xy*, where *x* and *y* are each small integral numbers. The *x* value represents the traffic class, while the *y* value represents the drop probability within that traffic class. There are four defined traffic classes numbered 1 through 4 and three drop priorities numbered 1 through 3. The larger the drop priority, the more likely the packet is to be dropped. For instance, you can configure weighted random early detection (WRED) to drop packets based on DSCP values.

For this scenario, R1 will classify via NBAR and mark packets with the EF and AF DSCP markings. All QoS actions will be performed within the MQC, so you will need to create traffic classes on each router. For more information on NBAR or MQC, consult the Lab 4.5: Class-based Queuing and NBAR.

To set a DSCP value, use the policy-map class configuration sub-prompt command **set dscp** *value*. Notice the available values shown in the output below.

```
R1(config-pmap-c)# set dscp ?
  <0-63>     Differentiated services codepoint value
  af11       Match packets with AF11 dscp (001010)
  af12       Match packets with AF12 dscp (001100)
  af13       Match packets with AF13 dscp (001110)
  af21       Match packets with AF21 dscp (010010)
  af22       Match packets with AF22 dscp (010100)
  af23       Match packets with AF23 dscp (010110)
  af31       Match packets with AF31 dscp (011010)
  af32       Match packets with AF32 dscp (011100)
  af33       Match packets with AF33 dscp (011110)
  af41       Match packets with AF41 dscp (100010)
  af42       Match packets with AF42 dscp (100100)
  af43       Match packets with AF43 dscp (100110)
  cos        Set packet DSCP from L2 COS
  cs1        Match packets with CS1(precedence 1) dscp (001000)
  cs2        Match packets with CS2(precedence 2) dscp (010000)
  cs3        Match packets with CS3(precedence 3) dscp (011000)
  cs4        Match packets with CS4(precedence 4) dscp (100000)
  cs5        Match packets with CS5(precedence 5) dscp (101000)
  cs6        Match packets with CS6(precedence 6) dscp (110000)
  cs7        Match packets with CS7(precedence 7) dscp (111000)
  default    Match packets with default dscp (000000)
  ef         Match packets with EF dscp (101110)
  qos-group  Set packet dscp from QoS Group.
```

Classify traffic on R1 as follows:

Create three traffic classes:

Critical: OSPF or Network Time Protocol (NTP) traffic. These protocols are used for network control. Mark with DSCP value EF.

Interactive: Telnet, SSH, and X-Windows traffic. These protocols are used for remote administration. Mark with DSCP value AF41.

Web: HTTP, POP3, and SMTP traffic. These protocols are used for web and e-mail access. Mark with DSCP value AF32.


```
R1(config)# class-map match-any critical
R1(config-cmap)# match protocol ospf
R1(config-cmap)# match protocol ntp
R1(config-cmap)# class-map match-any interactive
R1(config-cmap)# match protocol telnet
R1(config-cmap)# match protocol ssh
R1(config-cmap)# match protocol xwindows
R1(config-cmap)# class-map match-any web
R1(config-cmap)# match protocol http
R1(config-cmap)# match protocol pop3
R1(config-cmap)# match protocol smtp
```

Mark all other traffic with the default DSCP of 0.

Create the QoS policy map named “markingpolicy” and apply it outbound towards R4 on the Fast Ethernet 0/1 interface.

```
R1(config)# policy-map markingpolicy
R1(config-pmap)# class critical
R1(config-pmap-c)# set dscp ef
R1(config-pmap-c)# class interactive
R1(config-pmap-c)# set dscp af41
R1(config-pmap-c)# class web
R1(config-pmap-c)# set dscp af32
R1(config-pmap-c)# class class-default
R1(config-pmap-c)# set dscp default
R1(config-pmap-c)# interface fastethernet0/1
R1(config-if)# service-policy output markingpolicy
```

Verify the QoS configuration with the **show policy-map** command. Also, verify that the marking strategy is actively marking traffic with the **show policy-map interface** *interface* command.

```
R1# show policy-map
  Policy Map markingpolicy
    Class critical
      set dscp ef
    Class interactive
      set dscp af41
    Class web
      set dscp af32
    Class class-default
      set dscp default

R1# show policy-map interface fastethernet0/1
 FastEthernet0/1

  Service-policy output: markingpolicy

    Class-map: critical (match-any)
      242695 packets, 186052247 bytes
      5 minute offered rate 2475000 bps, drop rate 0 bps
      Match: protocol ospf
        108 packets, 7992 bytes
        5 minute rate 0 bps
      Match: protocol ntp
        242587 packets, 186044255 bytes
        5 minute rate 2475000 bps
      QoS Set
        dscp ef
```



```
          Packets marked 242695
<OUTPUT OMITTED>
```

Why would a network administrator decide to use IP Precedence over DSCP, or vice-versa?

Step 4: Configuring Class-Based Shaping

Traffic shaping is a QoS tool that allows you to define an average or peak rate at which traffic will be sent at an egress interface. Excess traffic is queued for sending later.

Observe the following rules when shaping or policing traffic:

1. At OSI Layer 1, data can only be sent at the clock rate (access rate) of the medium.

2. At OSI Layer 2, frames can be sent to approximate variable rates up to the Layer 1 clock rate by interchanging sending frames and restricting the sending of frames. In other words, traffic must be sent in bursts of data at exactly the access rate within each time interval to shape or police traffic at a specific rate.

Shaping and policing allow you to either allow the Cisco IOS to determine the amount of traffic to send within each time interval or to specify the number of bytes in the **shape** or **police** commands.

Shaping may be configured on a per-interface basis with Generic Traffic Shaping (GTS), or in a per-class basis through the MQC. Additionally, for Frame Relay networks which operate based on the concept of virtual circuits (VCs), Frame Relay Traffic Shaping (FRTS) can even be configured on a per-VC basis. In this scenario, you will use the MQC to configure Class-Based Traffic Shaping (CBTS) and simulate the function of GTS using CBTS in the Step 5.

In this step, shape all traffic traveling from R4 to R3 across the serial link to a peak rate. Create a policy map and classify traffic only into the default class; then shape peak egress rate of the default class on R4. This method of using one traffic class within the policy map to shape traffic can effectively simulate the function of GTS when you apply the policy map to an interface. Configure the peak traffic rate for a class, using the **shape peak** *rate* command. Use a peak traffic rate of 400 kbps. You can also configure the burst values more granularly, but this is beyond the scope of this lab.


```
R4(config)# policy-map shapingpolicy
R4(config-pmap)# class class-default
R4(config-pmap-c)# shape peak 400000
R4(config-pmap-c)# interface serial0/0/0
R4(config-if)# service-policy output shapingpolicy
```

Verify the configuration using the **show** commands for policy-maps.

```
R4# show policy-map
  Policy Map shapingpolicy
    Class class-default
      Traffic Shaping
         Peak Rate Traffic Shaping
         CIR 400000 (bps) Max. Buffers Limit 1000 (Packets)

R4# show policy-map interface serial0/0/0
 Serial0/0/0

  Service-policy output: shapingpolicy

    Class-map: class-default (match-any)
      546427 packets, 418135512 bytes
      5 minute offered rate 7644000 bps, drop rate 7092000 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           800000/400000    2500   10000     10000     25        2500

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      96        46540     24706516  46536     24703845  yes
```

The generated traffic is dense enough to completely saturate the serial link and/or the shaping profile, so you cannot see the function of the burst values; however, you can see that shaping is active and that packets have been delayed in transmission on account of that shaping.

What happens to the DSCP markings on IP packets traversing the serial link from R4 to R3 if no other traffic classes are referenced within the policy map?

Step 5: Configure Nested Service Policies

When you begin to create more complex QoS policies, you may find the need to apply a named policy-map inside of a class in another policy-map. You noted before that only the default class was used in the shaping policy in Step 4.

One possible scenario in which this would be necessary is if you want to apply granularity in marking, queuing, or shaping packets in distinct traffic classes but want to apply an aggregate shaper or policer to all of the traffic exiting the interface. Apply the differentiated actions in a single policy map. Then, set the shaping action in the default class in another policy map and apply the first policy map as an MQC action within the second policy map.

Use the policy map you configured in Step 4 as the outer policy map which will be applied directly to the interface. Create a new policy map to be used inside the outer policy map. Shape the individual classes using the inner policy map and shape the aggregate over all of the traffic classes in the outer policy map.

Create another policy (with appropriate classes) as shown below that shapes EF traffic to 40kbps, AF41 traffic should get 80kpbs, and AF32 traffic should get shaped to 120kbps. Apply this new policy inside the class configuration of the policy created in Step 4 using the **service-policy** *name* command.

```
R4(config)# class-map ef
R4(config-cmap)# match dscp ef
R4(config-cmap)# class-map af41
R4(config-cmap)# match dscp af41
R4(config-cmap)# class-map af32
R4(config-cmap)# match dscp af32
R4(config-cmap)# policy-map innerpolicy
R4(config-pmap)# class ef
R4(config-pmap-c)# shape peak 40000
R4(config-pmap-c)# class af41
R4(config-pmap-c)# shape peak 80000
R4(config-pmap-c)# class af32
R4(config-pmap-c)# shape peak 120000
R4(config-pmap-c)# policy-map shapingpolicy
R4(config-pmap)# class class-default
R4(config-pmap-c)# service-policy innerpolicy
```

Verify with the **show policy-map** command and the **show policy-map interface serial 0/0/0** command.

```
R4# show policy-map
  Policy Map shapingpolicy
    Class class-default
      Traffic Shaping
         Peak Rate Traffic Shaping
         CIR 400000 (bps) Max. Buffers Limit 1000 (Packets)
      service-policy innerpolicy

  Policy Map innerpolicy
    Class ef
      Traffic Shaping
         Peak Rate Traffic Shaping
         CIR 40000 (bps) Max. Buffers Limit 1000 (Packets)
    Class af41
      Traffic Shaping
         Peak Rate Traffic Shaping
         CIR 80000 (bps) Max. Buffers Limit 1000 (Packets)
    Class af32
      Traffic Shaping
         Peak Rate Traffic Shaping
         CIR 120000 (bps) Max. Buffers Limit 1000 (Packets)
```



```
R4# show policy-map interface serial0/0/0
 Serial0/0/0

  Service-policy output: shapingpolicy

    Class-map: class-default (match-any)
      492271 packets, 376494434 bytes
      5 minute offered rate 6900000 bps, drop rate 509000 bps
      Match: any
      Traffic Shaping
           Target/Average   Byte   Sustain   Excess    Interval  Increment
             Rate           Limit  bits/int  bits/int  (ms)      (bytes)
           800000/400000    2500   10000     10000     25        2500

        Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
        Active Depth                         Delayed   Delayed   Active
        -      42        24271     17196294  23348     16930349  yes

      Service-policy : innerpolicy

        Class-map: ef (match-all)
          62585 packets, 47610351 bytes
          5 minute offered rate 905000 bps, drop rate 0 bps
          Match:  dscp ef (46)
          Traffic Shaping
               Target/Average   Byte   Sustain   Excess    Interval  Increment
                 Rate           Limit  bits/int  bits/int  (ms)      (bytes)
                80000/40000     2000   8000      8000      200       2000

            Adapt  Queue     Packets   Bytes     Packets   Bytes     Shaping
            Active Depth                         Delayed   Delayed   Active
            -      64        2140      1647406   2135      1644763   yes
<OUTPUT OMITTED>
```

Step 6: Configure Traffic Policing

The difference between shaping traffic and policing traffic is that shapers attempt to smooth out a traffic profile whereas policers merely force the traffic to conform to a certain rate without buffering the excess. Policers drop excess packets and do not carry traffic from one interval to the next.

Create a new policy map to police traffic passing from R3 to R2. Police the total rate of egress traffic exiting R3's Serial 0/0/1 interface to 400 kbps.

Police the default class to the specified rate by issuing the **police rate** *rate type* command. You may also set up more granular parameters for the policer to use by issuing the **?** character.

```
R3(config)# policy-map policingpolicy
R3(config-pmap)# class class-default
R3(config-pmap-c)# police rate 400000 bps
R3(config-pmap-c-police)# interface serial0/0/1
R3(config-if)# service-policy output policingpolicy
```

Verify with the usual commands. Notice that some of the details of policing, such as the burst size, have been set up automatically since we did not specify them.


```
R3# show policy-map
  Policy Map policingpolicy
    Class class-default
     police rate 400000 bps burst 12500 bytes
       conform-action transmit
       exceed-action drop

R3# show policy-map interface serial0/0/1
 Serial0/0/1

  Service-policy output: policingpolicy

    Class-map: class-default (match-any)
      9702 packets, 6764207 bytes
      5 minute offered rate 158000 bps, drop rate 44811000 bps
      Match: any
      police:
          rate 400000 bps, burst 12500 bytes
        conformed 5912 packets, 3113901 bytes; actions:
          transmit
        exceeded 3768 packets, 3648918 bytes; actions:
          drop
        conformed 79000 bps, exceed 89000 bps
```

Step 7: Configure Class-Based TCP Header Compression

In Lab 4.3: Configuring TCP Header Compression, you configured TCP header compression on an entire interface. In the MQC, you can configure TCP and RTP header compression as a QoS action for specific traffic classes.

Issue the **compression header ip** *type* command, where *type* is either the **tcp** or **rtp** keyword. Configure TCP header compression on R4 for only AF32 traffic heading towards R3 using the existing policy-maps. For more information on header compression, consult the Lab 4.3.

```
R4(config)# policy-map innerpolicy
R4(config-pmap)# class af32
R4(config-pmap-c)# compression header ip tcp
```

If this was actual TCP traffic and not spoofed traffic, you would see packets being compressed. Because the TCP headers are not all being created naturally, some elements of the TCP header are incompressible. Notice that in the output of the **show policy-map** command no headers have been compressed. The traffic that is being generated is not legitimate TCP traffic so it will not be compressed.

```
R4# show policy-map interface
  Policy Map shapingpolicy
    Class class-default
      Traffic Shaping
         Peak Rate Traffic Shaping
         CIR 400000 (bps) Max. Buffers Limit 1000 (Packets)
      service-policy innerpolicy

  Policy Map innerpolicy
    Class ef
      Traffic Shaping
```



```
        Average Rate Traffic Shaping
        CIR 40000 (bps) Max. Buffers Limit 1000 (Packets)
    Class af41
      Traffic Shaping
         Average Rate Traffic Shaping
         CIR 80000 (bps) Max. Buffers Limit 1000 (Packets)
    Class af32
      Traffic Shaping
         Average Rate Traffic Shaping
         CIR 120000 (bps) Max. Buffers Limit 1000 (Packets)
      compress:
          header ip tcp
```

How could you create compressible TCP packets given the current topology?

Implement your solution and verify that packets are being compressed.

Final Configurations

```
R1# show run
!
hostname R1
!
class-map match-any critical
 match protocol ospf
 match protocol ntp
class-map match-any interactive
 match protocol telnet
 match protocol ssh
 match protocol xwindows
class-map match-any web
 match protocol http
 match protocol pop3
 match protocol smtp
!
policy-map markingpolicy
 class critical
  set dscp ef
 class interactive
  set dscp af41
 class web
  set dscp af32
 class class-default
  set dscp default
!
interface FastEthernet0/0
```



```
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface FastEthernet0/1
 ip address 172.16.14.1 255.255.255.0
 service-policy output markingpolicy
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

```
R2# show run
!
hostname R2
!
interface FastEthernet0/0
 ip address 172.16.20.2 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 ip address 172.16.23.2 255.255.255.0
 clock rate 800000
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

```
R3# show run
!
hostname R3
!
policy-map policingpolicy
 class class-default
  police rate 400000 bps
!
interface Serial0/0/1
 ip address 172.16.23.3 255.255.255.0
 service-policy output policingpolicy
 no shutdown
!
interface Serial0/1/0
 ip address 172.16.34.3 255.255.255.0
 clockrate 800000
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
line vty 0 4
 password cisco
 login
!
end
```

Pagent-related commands are removed from R4's output. Only commands related to this lab are shown.

```
R4# show run
```



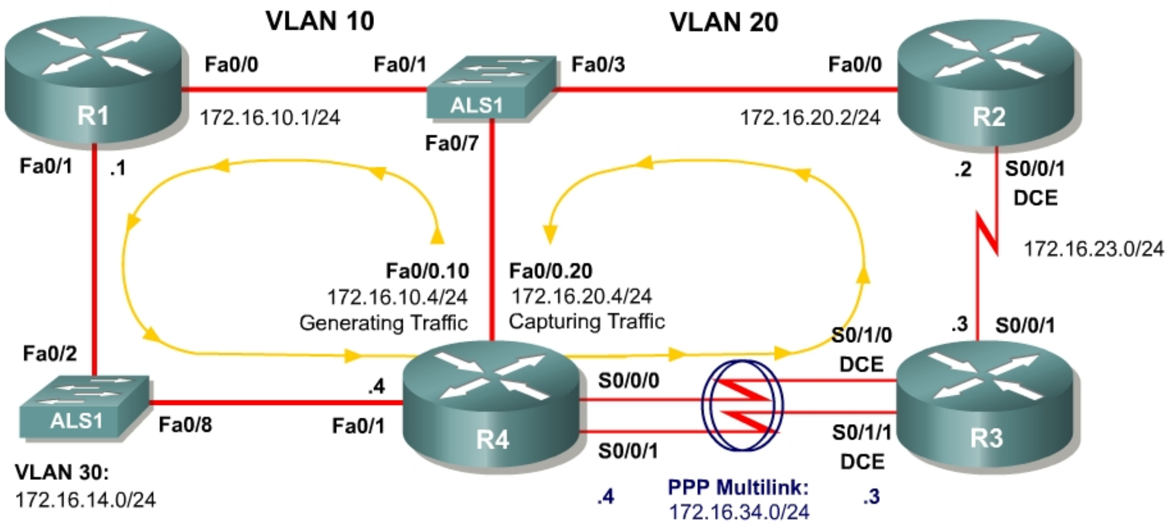
```
!
hostname R4
!
class-map match-all af41
 match  dscp af41
class-map match-all ef
 match  dscp ef
class-map match-all af32
 match  dscp af32
!
policy-map innerpolicy
 class ef
  shape average 40000
 class af41
  shape average 80000
 class af32
  shape average 120000
   compress header ip tcp
policy-map shapingpolicy
 class class-default
  shape peak 400000
  service-policy innerpolicy
!
interface FastEthernet0/1
 ip address 172.16.14.4 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 ip address 172.16.34.4 255.255.255.0
 service-policy output shapingpolicy
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```


Lab 4.7 WAN QoS Tools

Learning Objectives

- Configure Multilink PPP
- Configure Multilink PPP Link Fragmentation and Interleaving
- Configure Generic Traffic Shaping
- Configure Committed Access Rate policing

Topology Diagram



Scenario

In this lab, you will configure Generic Traffic Shaping (GTS) and Committed Access Rate (CAR) policing over Wide Area Network (WAN) serial connections. These tools are generally used on WAN connections to shape or police the entire traffic flow exiting an interface.

In this scenario, you will also configure Multilink PPP and the Link Fragmentation and Interleaving (LFI) feature.

Preparation

This lab relies on the Advanced Pagent Configuration which you should have created in Lab 3.2: Preparing for QoS.

Prior to beginning this lab, configure R4 and the switch according to the Advanced Pagent Configuration. You may easily accomplish this on R4 by loading the *advanced-ios.cfg* file from flash memory into the NVRAM, and reloading.


```
R4# copy flash:advanced-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
R4# reload
Proceed with reload? [confirm]
```

On the switch, load the *advanced.cfg* file into NVRAM and reload the device.

```
ALS1# copy flash:advanced.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
ALS1# reload
Proceed with reload? [confirm]
```

Next, instruct TGN to load the *advanced-tgn.cfg* file. At the end of Step 1, you will begin generating TGN traffic.

```
R4# tgn load-config advanced-tgn.cfg
```

Step 1: Configure the Physical Interfaces

Configure all of the physical interfaces shown in the diagram, except for the two serial links between R3 and R4. You will configure these two serial links in Step 2.

Set the clock rate on the serial link between R2 and R3 to 64 kbps and use the **no shutdown** command on all interfaces. Set the informational bandwidth parameter appropriately on the R2-R3 serial interfaces.

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface fastethernet 0/1
R1(config-if)# ip address 172.16.14.1 255.255.255.0
R1(config-if)# no shutdown

R2(config)# interface serial 0/0/1
R2(config-if)# bandwidth 64
R2(config-if)# ip address 172.16.23.2 255.255.255.0
R2(config-if)# clockrate 64000
R2(config-if)# no shutdown
R2(config-if)# interface fastethernet 0/0
R2(config-if)# ip address 172.16.20.2 255.255.255.0
R2(config-if)# no shutdown

R3(config)# interface serial 0/0/1
R3(config-if)# bandwidth 64
R3(config-if)# ip address 172.16.23.3 255.255.255.0
R3(config-if)# no shutdown

R4(config)# interface fastethernet 0/1
R4(config-if)# ip address 172.16.14.4 255.255.255.0
R4(config-if)# no shutdown
```


Now that R4 can reach R1 172.16.10.1 address via ARP, begin generating TGN traffic.

```
R4# tgn start
```

Step 2: Configure Multilink PPP

Multilink PPP is a PPP feature that allows multiple physical connections to be logically bound together to make a logical link across underlying serial connections encapsulated with PPP. The multilink PPP interface regards its bandwidth as the aggregate of the individual PPP connections.

For this lab, use multilink PPP to aggregate the two serial links between R3 and R4. They will be set up to be 64 kbps links individually, but their multilink logical connection will be 128 kbps.

First, configure the physical interfaces, Serial 0/1/0 and Serial 0/1/1 on R3 and Serial 0/0/0 and Serial 0/0/1 on R4. Set the clock rate on the DCE interfaces to 64 kbps and assign the informational bandwidth parameter appropriately. You will notice later that the multilink interface's informational bandwidth parameter is the sum of the active physical interface bandwidths as calculated from the individual bandwidth parameters.

Next, set up the interfaces to use PPP as the Layer 2 encapsulation with the **encapsulation ppp** command. Enable PPP multilink on each interface with the **ppp multilink** command and configure each interface to participate in PPP multilink group 1 with the **ppp multilink group** *number* command. Bring up the interfaces with the **no shutdown** command. Do not configure any IP addresses on the physical interfaces since they will solely operate at Layer 2.

```
R3(config)# interface serial 0/1/0
R3(config-if)# clockrate 64000
R3(config-if)# bandwidth 64
R3(config-if)# encapsulation ppp
R3(config-if)# ppp multilink
R3(config-if)# ppp multilink group 1
R3(config-if)# no shutdown
R3(config-if)# interface serial 0/1/1
R3(config-if)# clockrate 64000
R3(config-if)# bandwidth 64
R3(config-if)# encapsulation ppp
R3(config-if)# ppp multilink
R3(config-if)# ppp multilink group 1
R3(config-if)# no shutdown

R4(config)# interface serial 0/0/0
R4(config-if)# bandwidth 64
R4(config-if)# encapsulation ppp
R4(config-if)# ppp multilink
R4(config-if)# ppp multilink group 1
R4(config-if)# no shutdown
R4(config-if)# interface serial 0/0/1
R4(config-if)# bandwidth 64
R4(config-if)# encapsulation ppp
```



```
R4(config-if)# ppp multilink
R4(config-if)# ppp multilink group 1
R4(config-if)# no shutdown
```

Issue the **interface multilink** *number* command in global configuration mode to enter configuration mode for the multilink interface. Since you are using group number 1, configure the multilink interface with number 1. Assign the IP address shown in the diagram to the multilink interface.

```
R3(config)# interface multilink 1
R3(config-if)# ip address 172.16.34.3 255.255.255.0

R4(config)# interface multilink 1
R4(config-if)# ip address 172.16.34.4 255.255.255.0
```

Verify that you can **ping** across the link. If not, troubleshoot.

```
R3# ping 172.16.34.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.34.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/18/20 ms

R4# ping 172.16.34.3

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.34.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/18/20 ms
```

To look at PPP multilink statistics, use the PPP-specific command **show ppp multilink**. The bandwidth shown in this output is the sum of the individual link bandwidths. The output below varies slightly between the routers because they are running different IOS versions.

```
R3# show ppp multilink

Multilink1, bundle name is R4
  Endpoint discriminator is R4
  Bundle up for 00:03:29, total bandwidth 128, load 1/255
  Receive buffer limit 24000 bytes, frag timeout 1500 ms
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 5 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x2C received sequence, 0x2D sent sequence
  Member links: 2 active, 0 inactive (max not set, min not set)
    Se0/1/0, since 00:26:36
    Se0/1/1, since 00:26:22
No inactive multilink interfaces

R4# show ppp multilink

Multilink1
  Bundle name: R3
  Remote Endpoint Discriminator: [1] R3
  Local Endpoint Discriminator: [1] R4
  Bundle up for 00:03:35, total bandwidth 128, load 1/255
  Receive buffer limit 24000 bytes, frag timeout 1500 ms
```



```
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 1 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x2D received sequence, 0x2C sent sequence
  Member links: 2 active, 0 inactive (max not set, min not set)
    Se0/0/0, since 00:26:42
    Se0/0/1, since 00:26:28
No inactive multilink interfaces
```

Issue the generic **show interfaces** *interface* command to view multilink interface information. The bandwidth shown in this output is the aggregate of the active serial interfaces that you have assigned to this multilink group.

```
R3# show interfaces multilink 1
Multilink1 is up, line protocol is up
  Hardware is multilink group interface
  Internet address is 172.16.34.3/24
  MTU 1500 bytes, BW 128 Kbit, DLY 100000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, LCP Open, multilink Open
  Open: IPCP, CDPCP, loopback not set
  Keepalive set (10 sec)
  DTR is pulsed for 2 seconds on reset
  Last input 00:00:34, output never, output hang never
  Last clearing of "show interface" counters 00:06:55
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     28 packets input, 4168 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     28 packets output, 4626 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```

```
R4# show interfaces multilink 1
Multilink1 is up, line protocol is up
  Hardware is multilink group interface
  Internet address is 172.16.34.4/24
  MTU 1500 bytes, BW 128 Kbit, DLY 100000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation PPP, LCP Open, multilink Open
  Open: IPCP, CDPCP, loopback not set
  Keepalive set (10 sec)
  DTR is pulsed for 2 seconds on reset
  Last input 00:00:33, output never, output hang never
  Last clearing of "show interface" counters 00:07:38
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     29 packets input, 4606 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     29 packets output, 4846 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```


Notice that the queuing strategy is first-in, first-out (FIFO) on the logical interfaces. Normally, the default queuing strategy on a serial interface with the same speed would be weighted fair queuing (WFQ).

What is another type of interface that would benefit from being bundled in PPP?

From a conceptual perspective, what other types of logical bundling can occur in a network? Give at least two examples.

Step 3: Configure Multilink PPP LFI

Link Fragmentation and Interleaving (LFI) allows the interfaces to fragment large packets down to a set amount in order to minimize the serialization delay between the time high-priority packets enter the hardware queue (FIFO) and the time they are sent. For instance, in voice applications, where delay and jitter are the top quality of service considerations, it is important that voice packets encounter minimal delay especially on low-speed serial interfaces where there is a large serialization delay.

Once packets have been fragmented, the LFI mechanism must also allow fragments of packets to be transmitted non-consecutively. For instance, voice packets must be allowed to be sent between fragments of large packets.

Shut down the multilink interface to prevent link flapping while you configure LFI. Next, change the queuing strategy on the multilink interface from FIFO to weighted fair queuing (WFQ) with the **fair-queue** command in interface configuration mode. Set the interleaving fragment delay with the **ppp multilink fragment delay** *milliseconds* command. Reduce the maximum delay to 15 ms from the default 30 ms. This delay setting controls the maximum size to which packets must be fragmented, attempting to avoid negative results in delay-sensitive applications.

Enable MLPPP interleaving with the **ppp multilink interleave** command. Finally, bring the interface back up.


```
R3(config)# interface multilink 1
R3(config-if)# shutdown
R3(config-if)# fair-queue
R3(config-if)# ppp multilink fragment delay 15
R3(config-if)# ppp multilink interleave
R3(config-if)# no shutdown

R4(config)# interface multilink 1
R4(config-if)# shutdown
R4(config-if)# fair-queue
R4(config-if)# ppp multilink fragment delay 15
R4(config-if)# ppp multilink interleave
R4(config-if)# no shutdown
```

Issue the **show ppp multilink** command to view the LFI configuration.

```
R3# show ppp multilink

Multilink1, bundle name is R4
  Endpoint discriminator is R4
  Bundle up for 00:00:48, total bandwidth 128, load 1/255
  Receive buffer limit 24000 bytes, frag timeout 1500 ms
  Interleaving enabled
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 3 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0xA received sequence, 0xA sent sequence
  Member links: 2 active, 0 inactive (max not set, min not set)
    Se0/1/0, since 00:01:03, 120 weight, 112 frag size
    Se0/1/1, since 00:01:03, 120 weight, 112 frag size
No inactive multilink interfaces

R4# show ppp multilink

Multilink1
  Bundle name: R3
  Remote Endpoint Discriminator: [1] R3
  Local Endpoint Discriminator: [1] R4
  Bundle up for 00:05:19, total bandwidth 128, load 1/255
  Receive buffer limit 24000 bytes, frag timeout 1500 ms
  Interleaving enabled
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 6 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x19 received sequence, 0x19 sent sequence
  Member links: 2 active, 0 inactive (max not set, min not set)
    Se0/0/0, since 00:05:34, 120 weight, 112 frag size
    Se0/0/1, since 00:05:34, 120 weight, 112 frag size
No inactive multilink interfaces
```

Step 4: Configure Routing

Establish adjacencies for routing with Open Shortest Path First (OSPF). Include all connected subnets within the 172.16.0.0/16 major network for all four routers.

```
R1(config)# router ospf 1
R1(config-router)# network 172.16.0.0 0.0.255.255 area 0

R2(config)# router ospf 1
R2(config-router)# network 172.16.0.0 0.0.255.255 area 0
```



```
R3(config)# router ospf 1
R3(config-router)# network 172.16.0.0 0.0.255.255 area 0

R4(config)# router ospf 1
R4(config-router)# network 172.16.0.0 0.0.255.255 area 0
```

Which interface does the adjacency between R3 and R4 form on?

Step 5: Configure Generic Traffic Shaping

In Lab 4.6: Class-based Marking, Shaping, and Policing, you configured traffic shaping using the Modular QoS command-line (CLI) interface (MQC). Shaping can be configured on a per-interface basis by the use of Generic Traffic Shaping (GTS), which you will configure in this lab. Generic traffic shaping is considered a legacy QoS feature. In most modern networks, you would use the MQC version of traffic shaping instead. However, it is useful to configure GTS both pedagogically as well as to demonstrate traffic shaping outside of the MQC. All of the configuration for GTS can be accomplished with the use of the **traffic-shape** command in interface configuration mode.

Imagine that R3 is owned by an ISP. You have added another 64 kbps serial link from R3 to R4 to the multilink group. However, according to your traffic contract, the ISP is only responsible to forward traffic from you at a committed information rate (CIR) of 128 kbps over this PPP multilink interface. Any excess traffic may be dropped by the ISP without warning.

Understanding that your excess traffic may be dropped, you wish to minimize the effect any policing in the provider network by configuring traffic shaping at the exit to your network, R4’s multilink PPP interface.

Configure traffic shaping on R4’s multilink interface towards R3 and shape the flow of traffic to a rate of 128 kbps. Issue the **traffic-shape rate** *rate* command in interface configuration mode. Set the *rate* argument to 128 kbps. The traffic will be buffered in software by the traffic-shaping.

```
R4(config)# interface multilink 1
R4(config-if)# traffic-shape rate 128000
```

Verify traffic shaping with the **show traffic-shape** and **show traffic-shape statistics** commands. The former command shows statically configured options while the latter command displays dynamically captured statistics.

```
R4# show traffic-shape
```



```
Interface   Mu1
       Access Target    Byte   Sustain   Excess    Interval  Increment Adapt
VC     List   Rate      Limit  bits/int  bits/int  (ms)      (bytes)   Active
-             128000    1984   7936      7936      62        992       -

R4# show traffic-shape statistics
                  Acc. Queue Packets   Bytes     Packets   Bytes     Shaping
I/F               List Depth                     Delayed   Delayed   Active
Mu1                     75    19524     7279630   19500     7272037   yes
```

Step 6: Configure Committed Access Rate Policing

Traffic policing is similar to shaping. The difference is, while shaping tries to smooth out a traffic profile, policing merely forces the traffic to conform to a certain rate, without buffering it. The picture below illustrates the difference (taken from cisco.com).

Describe a situation in which you would use both traffic shaping and policing but not on the same interface:

Like shaping, policing can be configured either using the MQC to configure class-based policing or on a per-interface basis with Committed Access Rate (CAR) policing. You configure CAR on an interface by setting a policing rate with the **rate-limit** command.

Set R3's Serial 0/0/1 interface to police egress traffic to 56 kbps with a normal burst size of 1500 bytes and a maximum burst size of 4000 bytes. Issue the **rate-limit** *direction bps normal-burst maxmium-burst* **conform-action** *action* **exceed-action** *action* command. When packets conform to the policy, send them by using the **continue** keyword. When packets do not, **drop** them.

This command may cause the Open Shortest Path First (OSPF) adjacency between R2 and R3 to "flap" (go down and then back up) periodically, because some of the OSPF hello packets get dropped through CAR, despite WFQ on the interface.

```
R3(config)# interface serial 0/0/1
R3(config-if)# rate-limit output 56000 1500 4000 conform-action continue
exceed-action drop
```

Verify with the command **show interfaces rate-limit**.

```
R3# show interfaces rate-limit
Serial0/0/1
  Output
    matches: all traffic
      params:  56000 bps, 1500 limit, 4000 extended limit
```



```
conformed 17433 packets, 5992721 bytes; action: continue
exceeded 14032 packets, 6137014 bytes; action: drop
last packet: 16ms ago, current burst: 2580 bytes
last cleared 00:14:27 ago, conformed 55000 bps, exceeded 56000 bps
```

Final Configurations

```
R1# show run
!
hostname R1
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface FastEthernet0/1
 ip address 172.16.14.1 255.255.255.0
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end

R2# show run
!
hostname R2
!
interface FastEthernet0/0
 ip address 172.16.20.2 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 ip address 172.16.23.2 255.255.255.0
 clock rate 64000
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end

R3# show run
!
hostname R3
!
interface Multilink1
 ip address 172.16.34.3 255.255.255.0
 fair-queue 64 16 0
 ppp multilink
 ppp multilink fragment delay 15
 ppp multilink interleave
 ppp multilink group 1
!
interface Serial0/0/1
 ip address 172.16.23.3 255.255.255.0
 rate-limit output 56000 1500 4000 conform-action continue exceed-action drop
 no shutdown
!
interface Serial0/1/0
 bandwidth 64
 no ip address
```



```
 encapsulation ppp
 clock rate 64000
 ppp multilink
 ppp multilink group 1
 no shutdown
!
interface Serial0/1/1
 bandwidth 64
 no ip address
 encapsulation ppp
 clock rate 64000
 ppp multilink
 ppp multilink group 1
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

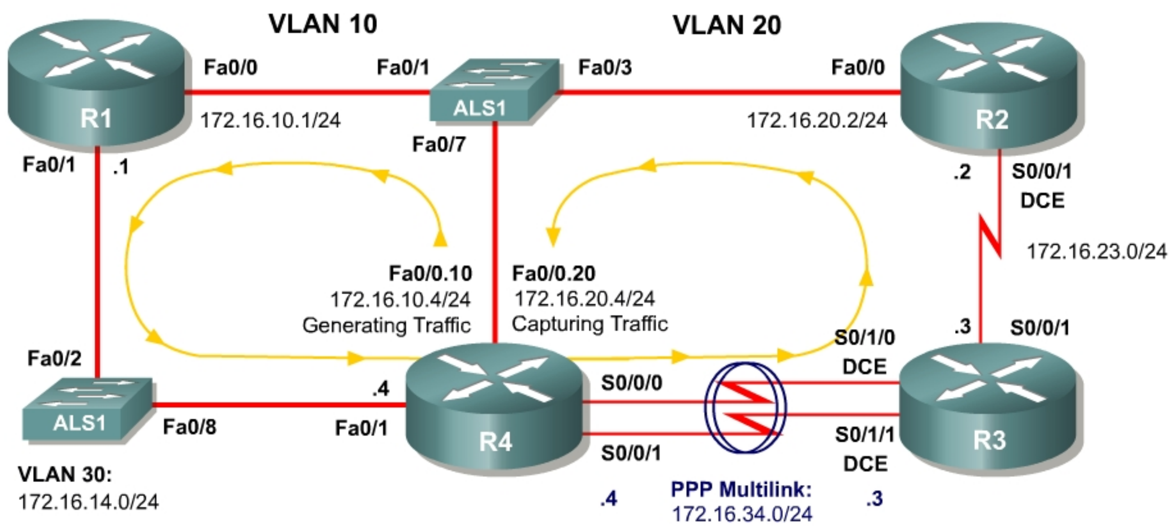
```
R4# show run
!
hostname R4
!
interface Multilink1
 ip address 172.16.34.4 255.255.255.0
 fair-queue 64 16 0
 traffic-shape rate 128000 7936 7936 1000
 ppp multilink
 ppp multilink interleave
 ppp multilink group 1
 ppp multilink fragment delay 15
!
interface FastEthernet0/1
 ip address 172.16.14.4 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 bandwidth 64
 no ip address
 encapsulation ppp
 ppp multilink
 ppp multilink group 1
 no shutdown
!
interface Serial0/0/1
 bandwidth 64
 no ip address
 encapsulation ppp
 ppp multilink
 ppp multilink group 1
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```


Lab 4.8 Shaping and Policing

Learning Objectives

- Use shaping to avoid the effects of policing

Topology Diagram



Scenario

In this lab, you will explore how traffic shaping interacts with traffic policing.

This lab will use the NQR tool from the Pagent toolset to observe delay and jitter statistics as you implement your solutions. You will investigate how different shaping and policing affect packet delay. If you have extra time to complete this lab, do not hesitate to extend this scenario to more configurations than simply those given here.

Typically, commands and command output will only be shown if they have not been implemented in preceding Module 4 labs, so it is highly recommended that you complete Labs 4.1 through 4.7 to ensure knowledge of the queuing, shaping, and policing strategies and their configurations.

Preparation

This lab relies on the Advanced Pagent Configuration which you should have created in Lab 3.1: Preparing for QoS.

Prior to beginning this lab, configure R4 and the switch according to the Advanced Pagent Configuration. You may easily accomplish this on R4 by loading the *advanced-ios.cfg* file from flash memory into the NVRAM, and reloading.

```
R4# copy flash:advanced-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
R4# reload
Proceed with reload? [confirm]
```

On the switch, load the *advanced.cfg* file into NVRAM and reload the device.

```
ALS1# copy flash:advanced.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
ALS1# reload
Proceed with reload? [confirm]
```

Unlike Labs 4.6 and 4.7, this lab will use the NQR tool in the Pagent toolset rather than the TGN traffic generator. Do not load the TGN traffic generator configuration.

Step 1: Configure Physical Interfaces and Routing

1. Configure all IP addresses shown in the diagram and use a clockrate of 800 kbps on all serial links. On the serial interfaces, set the informational bandwidth appropriately.

2. Bind the serial links between R3 and R4 in a PPP multilink. Do not configure Link Fragmentation and Interleaving (LFI) on the multilink interface.

3. Configure OSPF to route for all networks shown in the diagram.

4. Make sure that the outbound queuing method for R3's serial interface facing R2 is WFQ.

Step 2: Configure NQR on R4

The NQR tool in the Pagent toolset can assist network administrators in discovering delay and jitter statistics for traffic traversing their network. Enter NQR configuration mode by issuing the **nqr** command from the privilege EXEC prompt.

Copy and paste the configuration shown below into NQR on R4. This configuration will simulate two traffic streams: a constant high-bandwidth stream and a bursty, lower-bandwidth stream concurrent with it. Please see appendix A for the NETLAB compatible version.


```
fastethernet0/0
add tcp
send 2000
rate 150
length random 200 to 1000
datalink ios-dependent fastethernet0/0.10
l2-arp-for 172.16.10.1
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 21
fastethernet0/0.20 ios-dependent capture
add clone-of 1
l4-dest 23
send 500
rate 100
burst on
burst duration on 1000
burst duration off 3000
```

The NQR configuration here sends a controlled amount of packets—2000 for the larger stream, 500 for the smaller stream—and will stop when all packets are sent.

To begin NQR testing, issue either the **start send** command in NQR configuration mode or the **nqr start send** command from privileged EXEC mode. Time will pass, and then the router will inform you when all packets have been sent. There is no need to stop the streams since they will stop on their own.

Finally, issue the **show pkt-seq-drop-stats**, **show delay**, and **show jitter** NQR commands to display drop/resequencing, delay, and jitter statistics, respectively. Example output is shown below, although this type of output will not be shown again later in the lab. Record all statistics by copying and pasting them into a text editor such as Notepad. Record a baseline reading for your current topology.

```
R4(NQR:OFF,Fa0/0:2/2)# start send
R4(NQR:SEND,Fa0/0:2/2)#

   Send process complete.

R4(NQR:WAIT,Fa0/0:2/2)#
R4(NQR:OFF,Fa0/0:2/2)# show pkt-seq-drop-stats

Summary of packet sequence/drop stats of traffic streams
  ts#   template interface      sent     recvd   dropped   out-of-seq  max-seq
  1     TCP      Fa0/0.10*      2000      1919        81           37      568
  2     TCP      Fa0/0.10*       500       500         0            0      500

R4(NQR:OFF,Fa0/0:2/2)# show delay-stats

Summary of delay-stats of traffic streams
 ts#    template interface    min-delay    max-delay    avg-delay  stdev-delay
  1     TCP      Fa0/0.10*     0.004364     0.580043     0.238835     0.143506
  2     TCP      Fa0/0.10*     0.004390     0.273886     0.098115     0.077852

R4(NQR:OFF,Fa0/0:2/2)# show jitter-stats
```



```
Summary of jitter-stats of traffic streams
 ts#    template interface   min-jitter   max-jitter   avg-jitter stdev-jitter
  1     TCP      Fa0/0.10*     0.000033     0.367644     0.116765     0.083715
  2     TCP      Fa0/0.10*     0.000370     0.156045     0.066655     0.040675
```

Notice that packets are even dropped when no policing or shaping is configured because congestion occurred with only default queuing tools in place.

Step 3: Configure Traffic Policing

On R3, police egress traffic toward R2 to a rate of 700 kbps. Configure this either on a per-interface basis or using a policy-map to police the default class.

Then, run the NQR test again and record and compare statistics with the baseline statistics you captured in Step 2.

Run NQR again, record all statistics, and then compare NQR statistics.

How did these packet drop statistics compare to the earlier ones?

Identify where packet drops occurred in the topology using the **show interfaces** command.

Step 4: Configure Traffic Shaping

Configure R4 to shape traffic exiting the multilink interface. Shape the traffic down to the same rate that you are using to police traffic on R3. Use either the class-based method by shaping the default class or using the Generic Traffic Shaping on the multilink interface.

Run NQR again, record all statistics, and then compare NQR statistics.

How would shaping engender fewer packet drops even if the policing rate was not changed?

To what real-life scenario is this situation similar?

Final Configurations

```
R1# show run
```



```
!
hostname R1
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface FastEthernet0/1
 ip address 172.16.14.1 255.255.255.0
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

R2# **show run**

```
!
hostname R2
!
interface FastEthernet0/0
 ip address 172.16.20.2 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 ip address 172.16.23.2 255.255.255.0
 clock rate 800000
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

R3# **show run**

```
!
hostname R3
!
policy-map mypolicy
 class class-default
   police 700000
!
interface Multilink1
 ip address 172.16.34.3 255.255.255.0
 ppp multilink
 ppp multilink group 1
!
interface Serial0/0/1
 ip address 172.16.23.3 255.255.255.0
 service-policy output mypolicy
 no shutdown
!
interface Serial0/1/0
 bandwidth 800
 no ip address
 encapsulation ppp
 clock rate 800000
 ppp multilink
 ppp multilink group 1
 no shutdown
!
interface Serial0/1/1
 bandwidth 800
```



```
 no ip address
 encapsulation ppp
 clock rate 800000
 ppp multilink
 ppp multilink group 1
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

```
R4# show run
!
hostname R4
!
policy-map mypolicy
 class class-default
  shape peak 700000
!
interface Multilink1
 ip address 172.16.34.4 255.255.255.0
 ppp multilink
 ppp multilink group 1
 service-policy output mypolicy
!
interface FastEthernet0/1
 ip address 172.16.14.4 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 bandwidth 800
 ip address 172.16.34.4 255.255.255.0
 encapsulation ppp
 ppp multilink
 ppp multilink group 1
 no shutdown
!
interface Serial0/0/1
 bandwidth 800
 no ip address
 encapsulation ppp
 ppp multilink
 ppp multilink group 1
 no shutdown
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
!
end
```

Appendix A: NetLab-compatible NQR Configuration

NQR Configuration on R4

```
fastethernet0/0
add tcp
send 2000
rate 150
length random 200 to 1000
l2-dest $R1 Fa0/0's MAC$
```



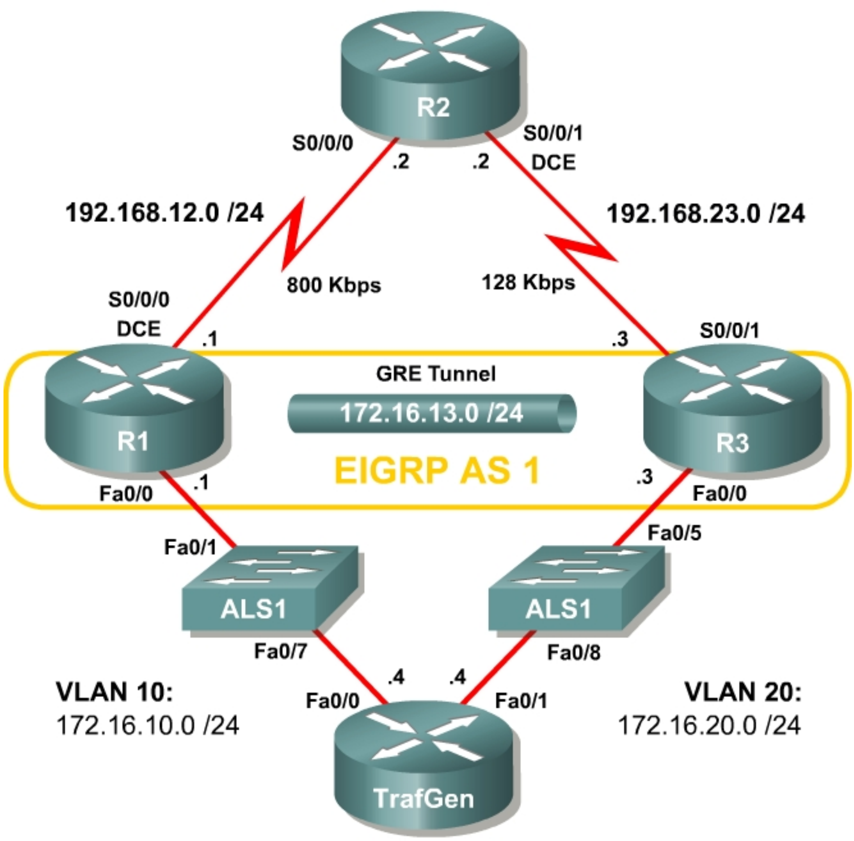
```
l3-src 172.16.10.4
l3-dest 172.16.20.4
l4-dest 21
fastethernet0/0 capture
add clone-of 1
l4-dest 23
send 500
rate 100
burst on
burst duration on 1000
burst duration off 3000
```


Lab 4.9 QoS Pre-classify

Learning Objectives

- Configure a GRE tunnel
- Configure QoS pre-classify
- Verify QoS pre-classify operation

Topology Diagram



Scenario

Weighted fair queuing (WFQ) allows routers to determine the ordering of packets for transmission on the basis of the flow or conversation into which a packet falls. A flow is defined by the source and destination addresses and port numbers, the transport protocol, and the IP Precedence value.

Both Generic Routing Encapsulation (GRE) and IPsec tunnels copy a packet's markable type of service/differentiated services (ToS/DiffServ) byte from the inner header to the outer header during encapsulation. Flow-based tools all along the tunnel's path will be able to view the IP Precedence or differentiated services code point (DSCP) marking. However, WFQ manages the allocation of network bandwidth by classifying traffic into prioritized flows, and dividing the network bandwidth fairly between those flows. Along the majority of the tunnel path, the only information able to be used to classify traffic will be the ToS/DiffServ byte.

However, at the tunnel endpoints you can make more intelligent decisions about the prioritization of packets because you have access to the inner packets before you encapsulate them with another IP header.

This scenario will guide you through implementing the QoS pre-classify feature to ensure that flow-based tools can make more intelligent decisions in provisioning bandwidth for tunneled flows.

Preparation

This lab uses the Basic Pagent Configuration for TrafGen and the switch to generate and facilitate lab traffic in a stream from TrafGen to R1 to R2. Prior to beginning this lab, configure TrafGen (R4) and the switch according to the Basic Pagent Configuration in Lab 3.1: Preparing for QoS. You can accomplish this easily on R4 by loading the *basic-ios.cfg* file from flash memory into the NVRAM, and reloading.

```
TrafGen# copy flash:basic-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

On the switch, load the *basic.cfg* file into NVRAM and reload the device.

```
Switch# copy flash:basic.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

On TrafGen, instruct TGN to load the *basic-tgn.cfg* file and to start generating traffic.

```
TrafGen# tgn load-config basic-tgn.cfg
TrafGen# tgn start
```

In addition, add the Fast Ethernet 0/5 interface on the switch to VLAN 20 since R3 will be the exit point from the network topology in this lab.

```
Switch# configure terminal
Switch(config)# interface fastethernet 0/5
Switch(config-if)# switchport access vlan 20
Switch(config-if)# switchport mode access
```


Step 1: Configure the Physical Interfaces

Configure all of the physical interfaces shown in the diagram. Set the clock rate on the serial link between R1 and R2 to 800000, the clock rate of the serial link between R2 and R3 to be 128000, and use the **no shutdown** command on all interfaces. Set the informational bandwidth parameter on the serial interfaces.

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface serial 0/0/0
R1(config-if)# bandwidth 800
R1(config-if)# ip address 192.168.12.1 255.255.255.0
R1(config-if)# clock rate 800000
R1(config-if)# no shutdown

R2(config)# interface serial 0/0/0
R2(config-if)# bandwidth 800
R2(config-if)# ip address 192.168.12.2 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# interface serial 0/0/1
R2(config-if)# bandwidth 128
R2(config-if)# ip address 192.168.23.2 255.255.255.0
R2(config-if)# clock rate 128000
R2(config-if)# no shutdown

R3(config)# interface fastethernet 0/0
R3(config-if)# ip address 172.16.20.3 255.255.255.0
R3(config-if)# no shutdown
R3(config-if)# interface serial 0/0/1
R3(config-if)# bandwidth 128
R3(config-if)# ip address 192.168.23.3 255.255.255.0
R3(config-if)# no shutdown
```

Issue the **show interfaces serial 0/0/0 | include Queueing** command on R1 to verify that the queuing strategy is WFQ.

```
R1# show interface serial0/0/0 | include Queueing
  Queueing strategy: weighted fair
```

If you see "fifo" as the queuing type, use the interface-level **fair-queue** command on the serial interface to change the queuing strategy to WFQ.

Step 2: Configure Static Routing

Configure R1 and R3 with default routes towards R2.

```
R1(config)# ip route 0.0.0.0 0.0.0.0 192.168.12.2

R3(config)# ip route 0.0.0.0 0.0.0.0 192.168.23.2
```

To which destination networks will R2 be able to forward IP traffic?

Does R2 have any knowledge of how to route to the 172.16.0.0/16 major network?

Step 3: Configure the GRE Tunnel

Your company currently maintains a GRE tunnel through the ISP router R2 terminating at R1 and R3. Create the tunnel interfaces on both R1 and R3 and use the addresses in the 192.168.0.0/16 address range as the endpoints of the tunnel. Use IP addresses in the 172.16.23.0/24 subnet as the addressing for the tunnel interfaces themselves. R2 does not need to have routing information for the network addresses you use in your private network (172.16.0.0/16).

Create a GRE tunnel interface, by issuing the **interface tunnel** *number* command to enter interface configuration mode for the tunnel interface. The tunnel interface number is only locally significant; however, for simplicity, use tunnel interface number 0 on both R1 and R3. Next, configure addressing for the tunnel interface itself with the **ip address** *address mask* command, just like you would do on any other interface. Finally, assign a source and destination address for the tunnel with the **tunnel source** *address* and **tunnel destination** *address* commands, respectively. The tunnel source can alternatively be specified by interface.

Tunneled traffic will be first sent to the other end of the GRE tunnel before being forwarded to its destination. Tunneling accomplishes this function by encapsulating packets with an outer IP header with the source and destination addresses supplied with the two previous commands. You do not need to configure a tunnel mode because the default tunnel mode is GRE. For more information on configuring GRE tunnels, reference the ISCW Lab 3.2: Configuring GRE Tunnels.

```
R1(config)# interface tunnel 0
R1(config-if)# tunnel source serial 0/0/0
R1(config-if)# tunnel destination 192.168.23.3
R1(config-if)# ip address 172.16.13.1 255.255.255.0

R3(config)# interface tunnel 0
R3(config-if)# tunnel source serial 0/0/1
R3(config-if)# tunnel destination 192.168.12.1
R3(config-if)# ip address 172.16.13.3 255.255.255.0
```

Verify that you can **ping** across the tunnel to the other side. If you can do this, you have successfully set up the tunnel.

```
R1# ping 172.16.13.3
```



```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.13.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/28/44 ms

R3# ping 172.16.13.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.13.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 12/24/36 ms
```

Step 4: Configure Routing

Configure routing between R1 and R3 using Enhanced Interior Gateway Routing Protocol (EIGRP). Include the entire 172.16.0.0/16 major network in AS 1 and disable automatic summarization.

```
R1(config)# router eigrp 1
R1(config-router)# no auto-summary
R1(config-router)# network 172.16.0.0

R3(config)# router eigrp 1
R3(config-router)# no auto-summary
R3(config-router)# network 172.16.0.0
```

Verify that the number of packets counted is increasing on the outbound interface of R3 using the **show interfaces fastethernet 0/1** command. Issue the command twice to make sure the number of packets output has changed. If the number is not increasing, troubleshoot Layer 1, 2, and 3 connectivity and the EIGRP topology.

If a tunnel's queuing strategy is first-in, first-out (FIFO), you may experience extreme delays in sending EIGRP packets over your tunnel. Remember that all of the traffic generated by Pagent is attempting to traverse the link as well and may cause delays in sending the EIGRP hellos.

Step 5: Enable the QoS Pre-classify Feature

On R1, issue the **show queue** *interface* command to view the open conversations exiting the Serial 0/0/0 interface.

```
R1 show queue serial 0/0/0
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 5798913
  Queueing strategy: weighted fair
  Output queue: 64/1000/64/5798913 (size/max total/threshold/drops)
     Conversations  1/10/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 64/32384/2643514/0/0
  Conversation 38, linktype: ip, length: 991
  source: 192.168.12.1, destination: 192.168.23.3, id: 0x45C8, ttl: 255, prot:
47
```


Notice there is only one conversation. The protocol number at the end is 47, which is GRE—the default tunnel encapsulation.

Why is there only one conversation listed, despite multiple traffic flows coming out of TrafGen?

Does GRE copy the inner IP header's ToS/DiffServ byte to the encapsulating IP header?

On the basis of your answer to the previous question and given the current configuration of the network, what is the maximum number of tunneled GRE conversations that could be seen in the output of the **show queue serial 0/0/0** command?

QoS pre-classify allows traffic to be classified by the physical interface's flow-based queuing strategy before being encapsulated so that the physical interface's network bandwidth can be fairly distributed amongst distinct tunneled flows, and not only those tunneled flows that will be based on IP Precedence. This ensures that a disproportionate amount of tunneled traffic is not dropped or significantly delayed at the physical interface.

Enable the QoS pre-classify feature by issuing the **qos pre-classify** command in interface configuration mode for the tunnel interfaces.

```
R1(config)# interface tunnel 0
R1(config-if)# qos pre-classify

R3(config)# interface tunnel 0
R3(config-if)# qos pre-classify
```

Now, try looking at the queue contents of the serial interface.

```
R1# show queue serial0/0/0
```



```
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 8512802
  Queueing strategy: weighted fair
  Output queue: 69/1000/64/8512802 (size/max total/threshold/drops)
     Conversations  7/11/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1158 kilobits/sec

  (depth/weight/total drops/no-buffer drops/interleaves) 12/32384/437229/0/0
  Conversation 43, linktype: ip, length: 641
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 6000

  (depth/weight/total drops/no-buffer drops/interleaves) 6/32384/436558/0/0
  Conversation 189, linktype: ip, length: 129
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 25

  (depth/weight/total drops/no-buffer drops/interleaves) 8/32384/436129/0/0
  Conversation 244, linktype: ip, length: 158
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 80

  (depth/weight/total drops/no-buffer drops/interleaves) 7/32384/437819/0/0
  Conversation 31, linktype: ip, length: 899
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 123

  (depth/weight/total drops/no-buffer drops/interleaves) 8/32384/441933/0/0
  Conversation 187, linktype: ip, length: 606
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 23

  (depth/weight/total drops/no-buffer drops/interleaves) 11/32384/445293/0/0
  Conversation 18, linktype: ip, length: 1003
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 110

  (depth/weight/total drops/no-buffer drops/interleaves) 17/32384/3157/0/0
  Conversation 164, linktype: ip, length: 1504
  source: 172.16.10.4, destination: 172.16.20.4, id: 0x0000, ttl: 59,
  TOS: 0 prot: 6, source port 0, destination port 6000
```

What is different about this output compared to the **show interfaces serial 0/0/0** output shown at the beginning of Step 5?

Final Configurations

```
R1# show run
!
hostname R1
!
interface Tunnel0
 ip address 172.16.13.1 255.255.255.0
```



```
 qos pre-classify
 tunnel source Serial0/0/0
 tunnel destination 192.168.23.3
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 ip address 192.168.12.1 255.255.255.0
 fair-queue
 clock rate 800000
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
ip route 0.0.0.0 0.0.0.0 192.168.12.2
!
end
```

```
R2# show run
!
hostname R2
!
interface Serial0/0/0
 ip address 192.168.12.2 255.255.255.0
 fair-queue
 no shutdown
!
interface Serial0/0/1
 ip address 192.168.23.2 255.255.255.0
 clock rate 800000
 no shutdown
!
end
```

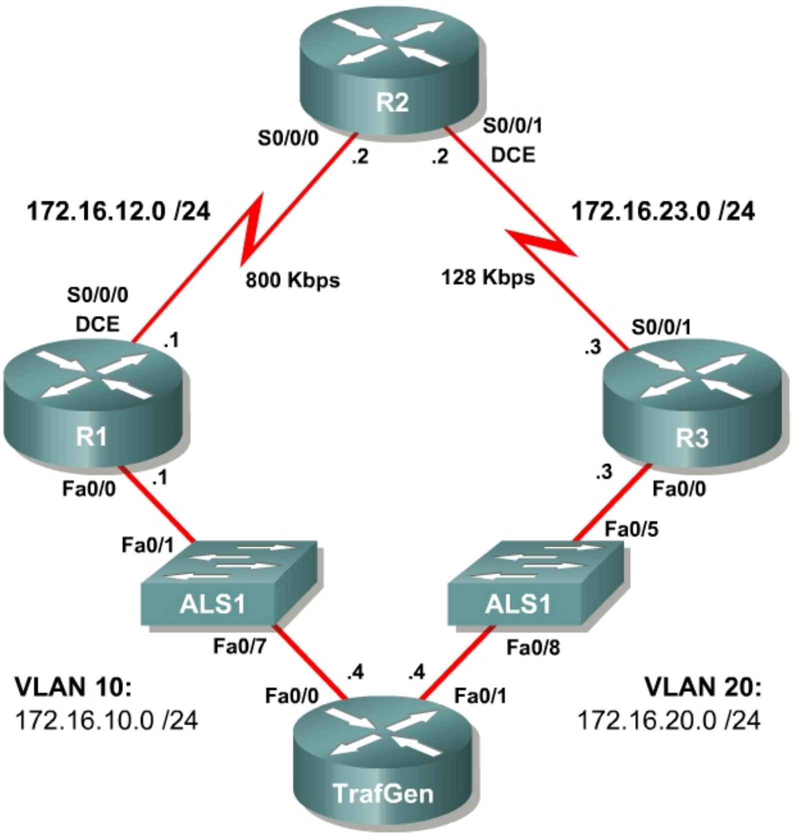
```
R3# show run
!
hostname R3
!
interface Tunnel0
 ip address 172.16.13.3 255.255.255.0
 qos pre-classify
 tunnel source Serial0/0/1
 tunnel destination 192.168.12.1
!
interface FastEthernet0/1
 ip address 172.16.20.3 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 ip address 192.168.23.3 255.255.255.0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
ip route 0.0.0.0 0.0.0.0 192.168.23.2
!
end
```


Lab 5.1 AutoQoS

Learning Objectives

- Configure AutoQoS Discovery
- Configure AutoQoS
- Verify AutoQoS behavior

Topology Diagram



Scenario

In this lab, you will configure AutoQoS, a Cisco QoS solution for simple, scalable QoS deployments. For this lab you are required to use a Pagent IOS image on TrafGen to generate lab traffic.

Preparation

This lab uses the Basic Pagent Configuration for TrafGen and the Switch to generate and facilitate lab traffic in a stream from TrafGen to R1 to R2. Prior to beginning this lab, configure TrafGen (R4) and the switch according to the

Basic Pagent Configuration in Lab 3.1: Preparing for QoS. You may simply accomplish this on R4 by loading the *basic-ios.cfg* file from Flash memory into the NVRAM, and reloading.

```
TrafGen# copy flash:basic-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

Next, instruct TGN to load the *basic-tgn.cfg* file and to start generating traffic.

```
TrafGen> enable
TrafGen# tgn load-config
TrafGen# tgn start
```

On the switch, load the *basic.cfg* file into NVRAM and reload the device.

```
ALS1# copy flash:basic.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
ALS1# reload
Proceed with reload? [confirm]
```

In addition, add the Fast Ethernet 0/5 interface on the switch to VLAN 20 since R3 will be the exit point from the network topology in this lab.

```
ALS1# configure terminal
ALS1(config)# interface fastethernet 0/5
ALS1(config-if)# switchport access vlan 20
ALS1(config-if)# switchport mode access
```

Step 1: Configure the Physical Interfaces

Configure all of the physical interfaces shown in the topology diagram. Set the clock rate on the serial link between R1 and R2 to 800 Kbps and the clock rate of the serial link between R2 and R3 to 128 Kbps; use the **no shutdown** command on all interfaces. Set the informational bandwidth parameter appropriately on the serial interfaces.

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface serial 0/0/0
R1(config-if)# bandwidth 800
R1(config-if)# ip address 172.16.12.1 255.255.255.0
R1(config-if)# clock rate 800000
R1(config-if)# no shutdown

R2(config)# interface serial 0/0/0
R2(config-if)# bandwidth 800
R2(config-if)# ip address 172.16.12.2 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# interface serial 0/0/1
R2(config-if)# bandwidth 128
```



```
R2(config-if)# ip address 172.16.23.2 255.255.255.0
R2(config-if)# clock rate 128000
R2(config-if)# no shutdown

R3(config)# interface fastethernet 0/0
R3(config-if)# ip address 172.16.20.3 255.255.255.0
R3(config-if)# no shutdown
R3(config-if)# interface serial 0/0/1
R3(config-if)# bandwidth 128
R3(config-if)# ip address 172.16.23.3 255.255.255.0
R3(config-if)# no shutdown
```

Note: If you do not use the basic-ios.cfg and basic-tgn.cfg files, enter these commands on R4 to configure it for traffic generation.

```
TrafGen(config)#interface fastethernet 0/0
TrafGen(config-if)# ip address 172.16.10.4 255.255.255.0
TrafGen(config-if)# no shutdown
TrafGen(config-if)# interface fastethernet 0/1
TrafGen(config-if)# ip address 172.16.20.4
TrafGen(config-if)# no shutdown
```

From global configuration mode on TrafGen, enter TGN configuration mode:

```
TrafGen# tgn
TrafGen(TGN:OFF<Fa0/0:none)#
```

Enter (or copy and paste) the following commands at the prompt. Note that you will need to enter the MAC address of R1's FastEthernet 0/0 interface in the highlighted field.

```
fastethernet 0/0
add tcp
rate 1000
L2-dest [enter MAC address of R1 Fa0/0]
L3-src 172.16.10.4
L3-dest 172.16.20.4
L4-dest 23
length random 16 to 1500
burst on
burst duration off 1000 to 2000
burst duration on 1000 to 3000
add fastethernet0/0 1
l4-dest 80
data ascii 0 GET /index.html HTTP/1.1
add fastethernet0/0 1
l4-dest 21
add fastethernet0/0 1
l4-dest 123
add fastethernet0/0 1
l4-dest 110
add fastethernet0/0 1
l4-dest 25
add fastethernet0/0 1
l4-dest 22
add fastethernet0/0 1
l4-dest 6000
```



```
!
end
```

Start generating traffic by entering the “start” command at the TGN prompt:

```
TrafGen(TGN:ON,Fa0/0:8/8)# start
```

Step 2: Configure EIGRP AS 1

Configure routing between R1, R2 and R3 using Enhanced Interior Gateway Router Protocol (EIGRP). Include the entire 172.16.0.0/16 major network in AS 1 and disable automatic summarization.

```
R1(config)# router eigrp 1
R1(config-router)# no auto-summary
R1(config-router)# network 172.16.0.0

R2(config)# router eigrp 1
R2(config-router)# no auto-summary
R2(config-router)# network 172.16.0.0

R3(config)# router eigrp 1
R3(config-router)# no auto-summary
R3(config-router)# network 172.16.0.0
```

Verify that the number of packets counted is increasing on the outbound interface of R3 using the **show interfaces fastethernet 0/1** command. Issue the command twice to make sure the number of packets output has changed. If the number is not increasing, troubleshoot Layer 1, 2, and 3 connectivity and the EIGRP configurations.

Step 3: Configure AutoQoS

AutoQoS is an IOS feature that observes traffic patterns on an interface via Network-based Application Recognition (NBAR) and generates appropriate class-based QoS policies based on observed traffic patterns.

You must initiate AutoQoS in a discovery phase in which the application observes traffic on an interface. You may decide to observe traffic over a significant period of time to ensure that all types of traffic have been accounted for.

Then, you must instruct AutoQoS to create QoS policies. The policies that AutoQoS creates can both mark traffic and implement various traffic shaping mechanisms. For more information on NBAR and the MQC, consult Lab 4.5: Class-based Queuing and NBAR.

Configure AutoQoS on R1’s Serial 0/0/0 interface so that the application can observe traffic passing through R1 toward R2. Begin the discovery phase of

AutoQoS by applying the **auto discovery qos** command in interface configuration mode.

```
R1(config)# interface serial 0/0/0
R1(config-if)# auto discovery qos
```

The router may not respond to input for a few moments while AutoQoS starts.

Let auto discovery run for a few minutes, and then peruse the traffic profile and suggested policy using the **show auto discovery qos** command. Your output may vary, as the results from this command are dynamically generated based on the traffic patterns observed.

```
R1# show auto discovery qos
Serial0/0/0
 AutoQoS Discovery enabled for applications
 Discovery up time: 2 minutes, 26 seconds
 AutoQoS Class information:
 Class Voice:
  No data found.
 Class Interactive Video:
  No data found.
 Class Signaling:
  No data found.
 Class Streaming Video:
  No data found.
 Class Transactional:
  Recommended Minimum Bandwidth: 10635 Kbps/688% (AverageRate)
  Detected applications and data:
  Application/       AverageRate        PeakRate          Total
  Protocol           (kbps/%)           (kbps/%)          (bytes)
  -----------        -----------        --------          ------------
  telnet             3640/235           4235/274          66441515
  ssh                3536/229           4359/282          64545226
  xwindows           3459/224           3863/250          63133333
 Class Bulk:
  Recommended Minimum Bandwidth: 10568 Kbps/684% (AverageRate)
  Detected applications and data:
  Application/       AverageRate        PeakRate          Total
  Protocol           (kbps/%)           (kbps/%)          (bytes)
  -----------        -----------        --------          ------------
  ftp                3564/230           4110/266          65052327
  smtp               3522/228           4086/264          64278471
  pop3               3482/225           4314/279          63556253
 Class Scavenger:
  No data found.
 Class Management:
  No data found.
 Class Routing:
  Recommended Minimum Bandwidth: 0 Kbps/0% (AverageRate)
  Detected applications and data:
  Application/       AverageRate        PeakRate          Total
  Protocol           (kbps/%)           (kbps/%)          (bytes)
  -----------        -----------        --------          ------------
  eigrp              0/0                0/0               1984
 Class Best Effort:
  Current Bandwidth Estimation: 6953 Kbps/450% (AverageRate)
  Detected applications and data:
  Application/       AverageRate        PeakRate          Total
  Protocol           (kbps/%)           (kbps/%)          (bytes)
```



```
  -----------        -----------        --------          ------------
  ntp                3510/227           4127/267          64072875
  http               3443/222           4159/269          62848166

Suggested AutoQoS Policy for the current uptime:
 !
 class-map match-any AutoQoS-Transactional-Se0/0/0
  match protocol telnet
  match protocol ssh
  match protocol xwindows
 !
 class-map match-any AutoQoS-Bulk-Se0/0/0
  match protocol ftp
  match protocol smtp
  match protocol pop3
 !
 policy-map AutoQoS-Policy-Se0/0/0
  class AutoQoS-Transactional-Se0/0/0
   bandwidth remaining percent 49
   random-detect dscp-based
   set dscp af21
  class AutoQoS-Bulk-Se0/0/0
   bandwidth remaining percent 49
   random-detect dscp-based
   set dscp af11
  class class-default
   fair-queue
```

There are a few observations you can make about this output. Besides the details of the statistics gathered, you can see that it separates traffic into classes based on function and latency requirements. At the end of the output, a suggested traffic policy is created. If the traffic generated by the traffic generator was different or more extensive, you might see other classes being utilized, with their own entries in the policy.

How many traffic classes has AutoQoS derived from the observed patterns?

Is this how you would also classify traffic generated by the Pagent router if you were to implement the suggested QoS policy on the command line? Explain.

What does the DSCP marking AF11 indicate?

What does the differentiated services code point (DSCP) marking AF21 indicate?

Are these markings locally significant to the router or globally significant over the entire routed path?

How much bandwidth do you expect to be allocated to the transactional and buik traffic classes respectively?

Although auto discovery uses NBAR for protocol recognition, it does not actually configure NBAR protocol discovery on the interface. You can verify this by looking at the running configuration for the serial interface.

```
R1# show run interface serial 0/0/0
Building configuration...

Current configuration : 107 bytes
!
interface Serial0/0/0
 ip address 172.16.12.1 255.255.255.0
 auto discovery qos
 clock rate 800000
end
```

Issue the **auto qos** command in interface configuration mode to implement the current AutoQoS-recommended configuration. This command requires AutoQoS' auto discovery to already be active.

```
R1(config)# interface serial0/0/0
R1(config-if)# auto qos
```

Verify the configuration that AutoQoS has applied by issuing the **show auto qos** command.


```
R1# show auto qos
 !
 policy-map AutoQoS-Policy-Se0/0/0
  class AutoQoS-Transactional-Se0/0/0
   bandwidth remaining percent 49
   random-detect dscp-based
   set dscp af21
  class AutoQoS-Bulk-Se0/0/0
   bandwidth remaining percent 49
   random-detect dscp-based
   set dscp af11
  class class-default
   fair-queue
 !
 class-map match-any AutoQoS-Transactional-Se0/0/0
  match protocol ssh
  match protocol telnet
  match protocol xwindows
 !
 class-map match-any AutoQoS-Bulk-Se0/0/0
  match protocol ftp
  match protocol smtp
  match protocol pop3

Serial0/0/0 -
 !
 interface Serial0/0/0
  service-policy output AutoQoS-Policy-Se0/0/0
```

Which queuing tool does the policy generated on router R1 represent?

Thus, when you issue the **auto qos** command, AutoQoS immediately generates the MQC configuration and applies it to the interface. Verify the statistics on the policy map using the **show policy-map interface serial 0/0/0** command.

```
R1# show policy-map interface serial 0/0/0
 Serial0/0/0

  Service-policy output: AutoQoS-Policy-Se0/0/0

    Class-map: AutoQoS-Transactional-Se0/0/0 (match-any)
      24415 packets, 19366297 bytes
      5 minute offered rate 194000 bps, drop rate 187000 bps
      Match: protocol ssh
        8564 packets, 6637316 bytes
        5 minute rate 69000 bps
      Match: protocol xwindows
        8758 packets, 7046646 bytes
        5 minute rate 77000 bps
      Match: protocol telnet
        7093 packets, 5682335 bytes
        5 minute rate 53000 bps
```



```
      Queueing
        Output Queue: Conversation 265
        Bandwidth remaining 49 (%)
        (pkts matched/bytes matched) 24564/19497687
        (depth/total drops/no-buffer drops) 41/23580/0
         exponential weight: 9
         mean queue depth: 41

 dscp    Transmitted      Random drop      Tail drop    Minimum Maximum  Mark
         pkts/bytes       pkts/bytes       pkts/bytes    thresh  thresh  prob
 af11       0/0              0/0              0/0           32      40  1/10
 af12       0/0              0/0              0/0           28      40  1/10
 af13       0/0              0/0              0/0           24      40  1/10
 af21     985/788284       145/117412      23486/18634727   32      40  1/10
 af22       0/0              0/0              0/0           28      40  1/10
 af23       0/0              0/0              0/0           24      40  1/10
 af31       0/0              0/0              0/0           32      40  1/10
 af32       0/0              0/0              0/0           28      40  1/10
 af33       0/0              0/0              0/0           24      40  1/10
 af41       0/0              0/0              0/0           32      40  1/10
 af42       0/0              0/0              0/0           28      40  1/10
 af43       0/0              0/0              0/0           24      40  1/10
  cs1       0/0              0/0              0/0           22      40  1/10
  cs2       0/0              0/0              0/0           24      40  1/10
  cs3       0/0              0/0              0/0           26      40  1/10
  cs4       0/0              0/0              0/0           28      40  1/10
  cs5       0/0              0/0              0/0           30      40  1/10
  cs6       0/0              0/0              0/0           32      40  1/10
  cs7       0/0              0/0              0/0           34      40  1/10
   ef       0/0              0/0              0/0           36      40  1/10
 rsvp       0/0              0/0              0/0           36      40  1/10
default     0/0              0/0              0/0           20      40  1/10

      QoS Set
        dscp af21
          Packets marked 24769

    Class-map: AutoQoS-Bulk-Se0/0/0 (match-any)
      25530 packets, 19973981 bytes
      5 minute offered rate 200000 bps, drop rate 192000 bps
      Match: protocol pop3
        7795 packets, 6150162 bytes
        5 minute rate 66000 bps
      Match: protocol smtp
        9381 packets, 7226367 bytes
        5 minute rate 67000 bps
      Match: protocol ftp
        8354 packets, 6597452 bytes
        5 minute rate 72000 bps
      Queueing
        Output Queue: Conversation 266
        Bandwidth remaining 49 (%)
        (pkts matched/bytes matched) 25847/20236550
        (depth/total drops/no-buffer drops) 41/24769/0
         exponential weight: 9
         mean queue depth: 41

 dscp    Transmitted      Random drop      Tail drop    Minimum Maximum  Mark
         pkts/bytes       pkts/bytes       pkts/bytes    thresh  thresh  prob
 af11    1090/869842       246/196528      24536/19186281   32      40  1/10
 af12       0/0              0/0              0/0           28      40  1/10
 af13       0/0              0/0              0/0           24      40  1/10
 af21       0/0              0/0              0/0           32      40  1/10
```



```
  af22      0/0               0/0              0/0            28       40  1/10
  af23      0/0               0/0              0/0            24       40  1/10
  af31      0/0               0/0              0/0            32       40  1/10
  af32      0/0               0/0              0/0            28       40  1/10
  af33      0/0               0/0              0/0            24       40  1/10
  af41      0/0               0/0              0/0            32       40  1/10
  af42      0/0               0/0              0/0            28       40  1/10
  af43      0/0               0/0              0/0            24       40  1/10
   cs1      0/0               0/0              0/0            22       40  1/10
   cs2      0/0               0/0              0/0            24       40  1/10
   cs3      0/0               0/0              0/0            26       40  1/10
   cs4      0/0               0/0              0/0            28       40  1/10
   cs5      0/0               0/0              0/0            30       40  1/10
   cs6      0/0               0/0              0/0            32       40  1/10
   cs7      0/0               0/0              0/0            34       40  1/10
    ef      0/0               0/0              0/0            36       40  1/10
  rsvp      0/0               0/0              0/0            36       40  1/10
default     0/0               0/0              0/0            20       40  1/10

      QoS Set
        dscp af11
          Packets marked 25975

    Class-map: class-default (match-any)
      16903 packets, 13301976 bytes
      5 minute offered rate 130000 bps, drop rate 128000 bps
      Match: any
      Queueing
        Flow Based Fair Queueing
        Maximum Number of Hashed Queues 256
        (total queued/total drops/no-buffer drops) 115/17584/0
```

Why is the auto discovery step separate from the actual implementation of AutoQoS?

Step 4: Configure AutoQoS with DSCP

In the previous step, you configured AutoQoS with a base configuration that classified traffic based on protocols. The configuration marked the packets with various DSCP values in addition to configuring CBWFQ. AutoQoS in an enterprise deployment can be configured to trust DSCP values from other routers and make QoS decisions based on those values.

Describe the efficiency of enabling AutoQoS on all routers in your network, but not configuring AutoQoS to trust markings from other routers:

Modify the **auto discovery qos** command with the **trust** keyword on on R2's Serial 0/0/0 interface.

```
R2(config)# interface serial 0/0/1
R2(config-if)# auto discovery qos trust
```

Wait a few minutes for auto discovery to capture statistics. Then, use the **show auto discovery qos** command to view the traffic patterns that AutoQoS has observed.

```
R2# show auto discovery qos
Serial0/0/1
 AutoQoS Discovery enabled for trusted DSCP
 Discovery up time: 9 minutes, 23 seconds
 AutoQoS Class information:
 Class Voice:
  No data found.
 Class Interactive Video:
  No data found.
 Class Signaling:
  No data found.
 Class Streaming Video:
  No data found.
 Class Transactional:
  Recommended Minimum Bandwidth: 397 Kbps/25% (AverageRate)
  Detected DSCPs and data:
  DSCP value         AverageRate        PeakRate           Total
                     (kbps/%)           (kbps/%)           (bytes)
  -----------        -----------        --------           ------------
  18/af21            397/25             475/30             27986160
 Class Bulk:
  Recommended Minimum Bandwidth: 394 Kbps/25% (AverageRate)
  Detected DSCPs and data:
  DSCP value         AverageRate        PeakRate           Total
                     (kbps/%)           (kbps/%)           (bytes)
  -----------        -----------        --------           ------------
  10/af11            394/25             478/30             27770932
 Class Scavenger:
  No data found.
 Class Management:
  No data found.
 Class Routing:
  No data found.
 Class Best Effort:
  Current Bandwidth Estimation: 0 Kbps/0% (AverageRate)
  Detected DSCPs and data:
  DSCP value         AverageRate        PeakRate           Total
                     (kbps/%)           (kbps/%)           (bytes)
  -----------        -----------        --------           ------------
  0/default          0/0                3/<1               54449

Suggested AutoQoS Policy for the current uptime:
 !
 class-map match-any AutoQoS-Transactional-Trust
  match ip dscp af21
  match ip dscp af22
  match ip dscp af23
 !
 class-map match-any AutoQoS-Bulk-Trust
  match ip dscp af11
  match ip dscp af12
```



```
  match ip dscp af13
 !
 policy-map AutoQoS-Policy-Se0/0/1-Trust
  class AutoQoS-Transactional-Trust
   bandwidth remaining percent 25
   random-detect dscp-based
  class AutoQoS-Bulk-Trust
   bandwidth remaining percent 25
   random-detect dscp-based
  class class-default
   fair-queue
```

Notice that the output is very similar to the output in the previous step. However, this time, the statistics are based on DSCP values, not individual applications. Enable AutoQoS on the interface.

```
R2(config)# interface serial0/0/1
R2(config-if)# auto qos
```

Verify using the command **show auto qos**.

```
R2# show auto qos
 !
 policy-map AutoQoS-Policy-Se0/0/1-Trust
  class AutoQoS-Transactional-Trust
   bandwidth remaining percent 25
   random-detect dscp-based
  class AutoQoS-Bulk-Trust
   bandwidth remaining percent 25
   random-detect dscp-based
  class class-default
   fair-queue
 !
 class-map match-any AutoQoS-Bulk-Trust
  match ip dscp af11
  match ip dscp af12
  match ip dscp af13
 !
 class-map match-any AutoQoS-Transactional-Trust
  match ip dscp af21
  match ip dscp af22
  match ip dscp af23

Serial0/0/1 -
 !
 interface Serial0/0/1
  service-policy output AutoQoS-Policy-Se0/0/1-Trust
```

Final Configurations

```
R1# show run
!
hostname R1
!
policy-map AutoQoS-Policy-Se0/0/0
 class AutoQoS-Transactional-Se0/0/0
  bandwidth remaining percent 49
  random-detect dscp-based
  set dscp af21
 class AutoQoS-Bulk-Se0/0/0
```



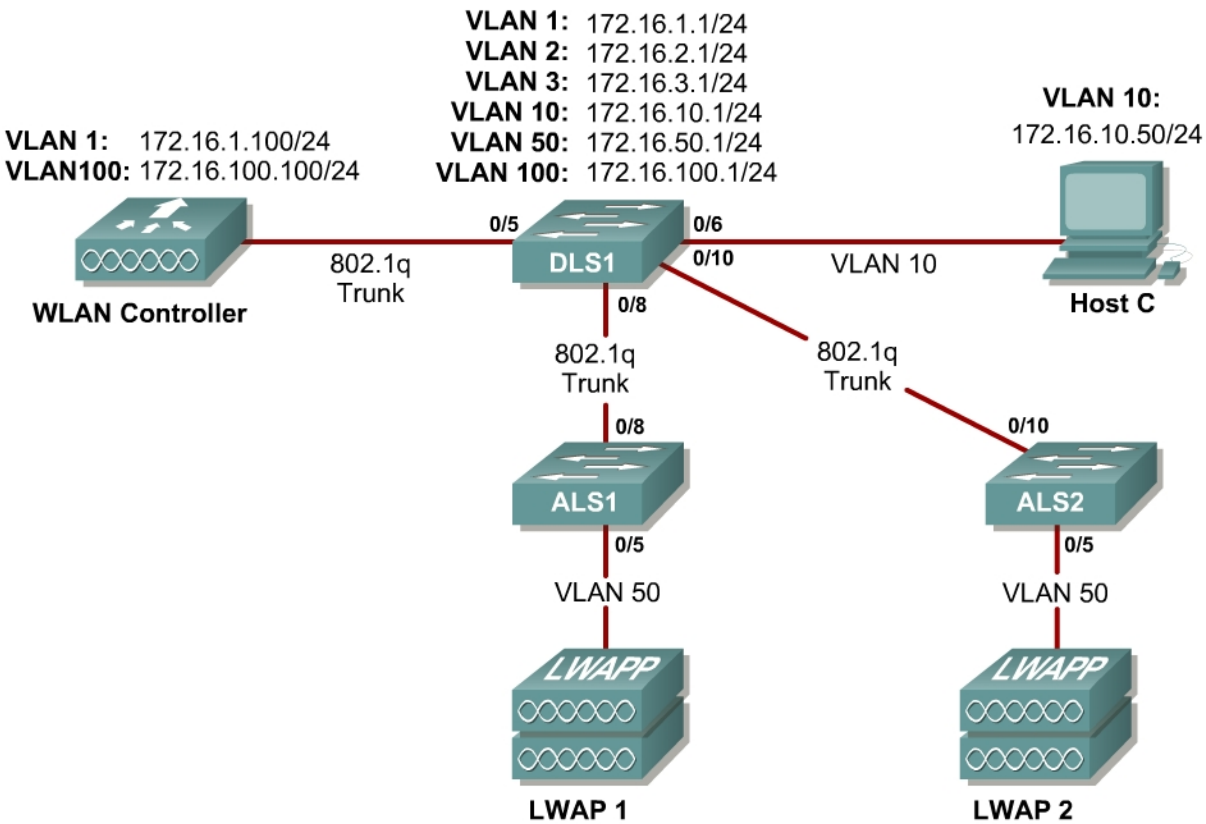
```
  bandwidth remaining percent 49
  random-detect dscp-based
  set dscp af11
 class class-default
  fair-queue
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface Serial0/0/0
 ip address 172.16.12.1 255.255.255.0
 auto qos
 auto discovery qos
 clock rate 800000
 service-policy output AutoQoS-Policy-Se0/0/0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
end

R2# show run
!
hostname R2
!
policy-map AutoQoS-Policy-Se0/0/1-Trust
 class AutoQoS-Transactional-Trust
  bandwidth remaining percent 25
  random-detect dscp-based
 class AutoQoS-Bulk-Trust
  bandwidth remaining percent 25
  random-detect dscp-based
 class class-default
  fair-queue
!
interface Serial0/0/0
 ip address 172.16.12.2 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 ip address 172.16.23.2 255.255.255.0
 auto qos
 auto discovery qos trust
 clock rate 800000
 service-policy output AutoQoS-Policy-Se0/0/1-Trust
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
end

R3# show run
!
hostname R3
!
interface FastEthernet0/1
 ip address 172.16.20.3 255.255.255.0
 no shutdown
```



```
!
interface Serial0/0/1
 ip address 172.16.23.3 255.255.255.0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
end
```


Lab 6.1a Configuring a WLAN Controller

Topology Diagram



Scenario

In the next two labs, you will configure a wireless solution involving a WLAN controller, two lightweight wireless access points, and a switched wired network. You will configure a WLAN controller to broadcast SSIDs from the lightweight wireless access points. If you have a wireless client nearby, connect to the WLANs and access devices from the inside of your pod to verify your configuration of the controller and access points.

Note: It is required that you upgrade the WLC firmware image to 4.0.206.0 or higher in order to accomplish this lab.

Step 1

Erase the startup-config file and delete the vlan.dat file from each switch. On the WLAN controller, use the **clear controller** command followed by the **reset system** command to reset them.

Step 2

Explanation of VLANs:
VLAN 1 – This VLAN is the management VLAN for the WLC
VLAN 2 and VLAN 3 – These VLANs are for hosts in the WLANs
VLAN 10 – The host is in this VLAN
VLAN 50 – The APs are in this VLAN
VLAN 100 – The AP-manager interface of the WLC is in this VLAN

Set up DLS1 as a VTP server, and ALS1 and ALS2 as clients. Put them in VTP domain CISCO. Set up the switch-to-switch links shown in the diagram as 802.1q trunks. Add VLANs 2, 3, 10, 50, and 100 to DLS1.

```
DLS1(config)# vtp mode server
DLS1(config)# vtp domain CISCO
DLS1(config)# vlan 2,3,10,50,100
DLS1(config-vlan)# interface fastethernet0/8
DLS1(config-if)# switchport trunk encapsulation dot1q
DLS1(config-if)# switchport mode trunk
DLS1(config-if)# interface fastethernet0/10
DLS1(config-if)# switchport trunk encapsulation dot1q
DLS1(config-if)# switchport mode trunk

ALS1(config)# vtp mode client
ALS1(config)# vtp domain CISCO
ALS1(config)# interface fastethernet0/8
ALS1(config-if)# switchport mode trunk

ALS2(config)# vtp mode client
ALS2(config)# vtp domain CISCO
ALS2(config)# interface fastethernet0/10
ALS2(config-if)# switchport mode trunk
```

Verify that VTP traffic has passed between the switch by comparing the non-zero VTP configuration revision between switches with the **show vtp status** command.

```
DLS1# show vtp status
VTP Version                     : 2
Configuration Revision          : 1
Maximum VLANs supported locally : 1005
Number of existing VLANs        : 10
VTP Operating Mode              : Server
VTP Domain Name                 : CISCO
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x6A 0x6B 0xCA 0x3C 0xF0 0x45 0x87 0xAC
Configuration last modified by 0.0.0.0 at 3-1-93 00:02:01
Local updater ID is 0.0.0.0 (no valid interface found)

ALS1# show vtp status
VTP Version                     : 2
Configuration Revision          : 1
Maximum VLANs supported locally : 255
Number of existing VLANs        : 10
VTP Operating Mode              : Client
```



```
VTP Domain Name                 : CISCO
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x6A 0x6B 0xCA 0x3C 0xF0 0x45 0x87 0xAC
Configuration last modified by 0.0.0.0 at 3-1-93 00:02:01

ALS2# show vtp status
VTP Version                     : 2
Configuration Revision          : 1
Maximum VLANs supported locally : 255
Number of existing VLANs        : 10
VTP Operating Mode              : Client
VTP Domain Name                 : CISCO
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x6A 0x6B 0xCA 0x3C 0xF0 0x45 0x87 0xAC
Configuration last modified by 0.0.0.0 at 3-1-93 00:02:01
```

Step 3

Configure all the switched virtual interfaces (SVIs) shown in the diagram for DLS1.

```
DLS1(config)# interface vlan 1
DLS1(config-if)# ip address 172.16.1.1 255.255.255.0
DLS1(config-if)# interface vlan 2
DLS1(config-if)# ip address 172.16.2.1 255.255.255.0
DLS1(config-if)# interface vlan 3
DLS1(config-if)# ip address 172.16.3.1 255.255.255.0
DLS1(config-if)# interface vlan 10
DLS1(config-if)# ip address 172.16.10.1 255.255.255.0
DLS1(config-if)# interface vlan 50
DLS1(config-if)# ip address 172.16.50.1 255.255.255.0
DLS1(config-if)# interface vlan 100
DLS1(config-if)# ip address 172.16.100.1 255.255.255.0
```

Step 4

DHCP gives out dynamic IP addresses on a subnet to network devices or hosts rather than statically setting the addresses. This is useful when dealing with lightweight access points, which usually do not have an initial configuration. The WLAN controller that the lightweight wireless access point associates with defines the configuration. A lightweight access point can dynamically receive an IP address and then communicate over IP with the WLAN controller. In this scenario, you will also use it to assign IP addresses to hosts that connect to the WLANs.

First, set up DLS1 to exclude the first 150 addresses from each subnet from DHCP to avoid conflicts with static IP addresses by using the global configuration command **ip dhcp excluded-address** *low-address* [*high-address*].

```
DLS1(config)# ip dhcp excluded-address 172.16.1.1 172.16.1.150
DLS1(config)# ip dhcp excluded-address 172.16.2.1 172.16.2.150
DLS1(config)# ip dhcp excluded-address 172.16.3.1 172.16.3.150
```



```
DLS1(config)# ip dhcp excluded-address 172.16.10.1 172.16.10.150

DLS1(config)# ip dhcp excluded-address 172.16.50.1 172.16.50.150
DLS1(config)# ip dhcp excluded-address 172.16.100.1 172.16.100.150
```

To advertise on different subnets, create DHCP pools with the **ip dhcp pool** *name* command. After a pool is configured for a certain subnet, the IOS DHCP server processes requests on that subnet, because it is enabled by default. From the DHCP pool prompt, set the network and mask to use with the **network** *address /mask* command. Set a default gateway with the **default-router** *address* command.

VLAN 50 also uses the **option** command, which allows you to specify a DHCP option. In this case, option 43 is specified (a vendor-specific option), which gives the lightweight wireless access points the IP address of the WLAN controller AP Manager interface. It is specified in a hexadecimal TLV (type, length, value) format. F1 is the hardcoded type of option, 04 represents the length of the value (an IP address is 4 octets), and AC106464 is the hexadecimal representation of 172.16.100.100, which is going to be the AP manager address of the WLAN controller. DHCP option 60 specifies the identifier that access points will use in DHCP. This lab was written using Cisco Aironet 1240 series access points. If you are using a different access point series, consult http://www.cisco.com/univercd/cc/td/doc/product/wireless/aero1500/1500hig5/1500_axg.htm.

```
DLS1(config)# ip dhcp pool pool1
DLS1(dhcp-config)# network 172.16.1.0 /24
DLS1(dhcp-config)# default-router 172.16.1.1
DLS1(dhcp-config)# ip dhcp pool pool2
DLS1(dhcp-config)# network 172.16.2.0 /24
DLS1(dhcp-config)# default-router 172.16.2.1
DLS1(dhcp-config)# ip dhcp pool pool3
DLS1(dhcp-config)# network 172.16.3.0 /24
DLS1(dhcp-config)# default-router 172.16.3.1
DLS1(dhcp-config)# ip dhcp pool pool10
DLS1(dhcp-config)# network 172.16.10.0 /24
DLS1(dhcp-config)# default-router 172.16.10.1
DLS1(dhcp-config)# ip dhcp pool pool50
DLS1(dhcp-config)# network 172.16.50.0 /24
DLS1(dhcp-config)# default-router 172.16.50.1
DLS1(dhcp-config)# option 43 hex f104ac106464
DLS1(dhcp-config)# option 60 ascii "Cisco AP c1240"
DLS1(dhcp-config)# ip dhcp pool pool100
DLS1(dhcp-config)# network 172.16.100.0 /24
DLS1(dhcp-config)# default-router 172.16.100.1
```

Step 5

On all three switches, configure each access point’s switchport with the **spanning-tree portfast** command so that each access point receives an IP address from DHCP immediately, thereby avoiding spanning-tree delays. Use VLAN 100 as the AP Manager interface for the WLAN controller. All control and data traffic between the controller and the lightweight wireless access points passes over this VLAN to this interface. Configure the ports going to the lightweight wireless access points in VLAN 50. DLS1 will route the traffic between the VLANs. Configure the interface on DLS1 that connects to the WLAN controller as an 802.1q trunk.

```
DLS1(config)# interface fastethernet0/5
DLS1(config-if)# switchport trunk encapsulation dot1q
DLS1(config-if)# switchport mode trunk

ALS1(config)# interface fastethernet0/5
ALS1(config-if)# switchport mode access
ALS1(config-if)# switchport access vlan 50
ALS1(config-if)# spanning-tree portfast

ALS2(config)# interface fastethernet0/5
ALS2(config-if)# switchport mode access
ALS2(config-if)# switchport access vlan 50
ALS2(config-if)# spanning-tree portfast
```

Step 6

You have a PC running Microsoft Windows attached to DLS1. First, configure the switchport facing the host to be in VLAN 10.

```
DLS1(config)# interface fastethernet0/6
DLS1(config-if)# switchport mode access
DLS1(config-if)# switchport access vlan 10
DLS1(config-if)# spanning-tree portfast
```

Next, configure the host with an IP address in VLAN 10, which will later be used to access the HTTP web interface of the WLAN controller.

In the **Control Panel**, select **Network Connections**.

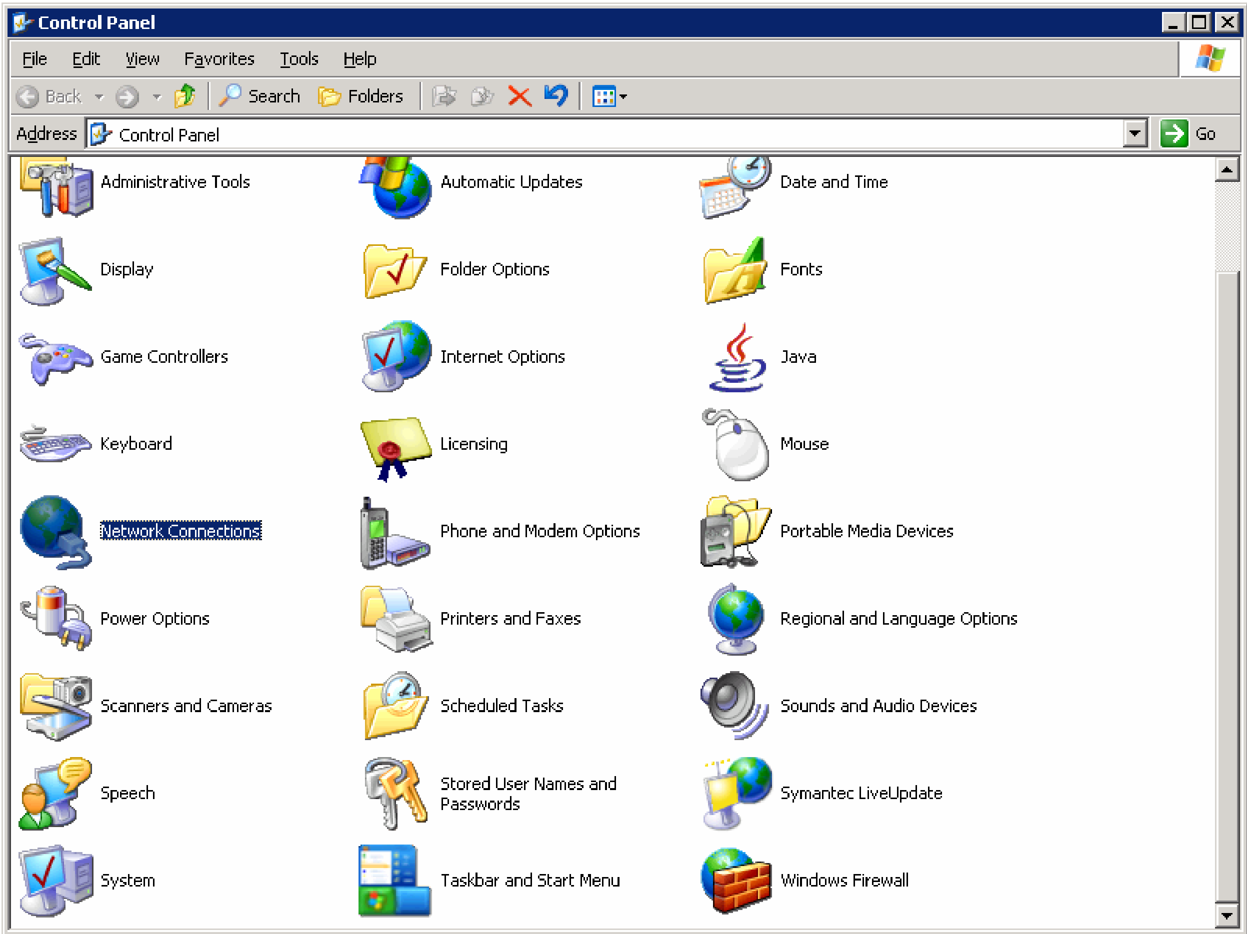


Figure 5-1: Microsoft Windows Control Panel

Right-click on the LAN interface that connects to DLS1, and select **Properties**. Select **Internet Protocol (TCP/IP)** and then click the **Properties** button.

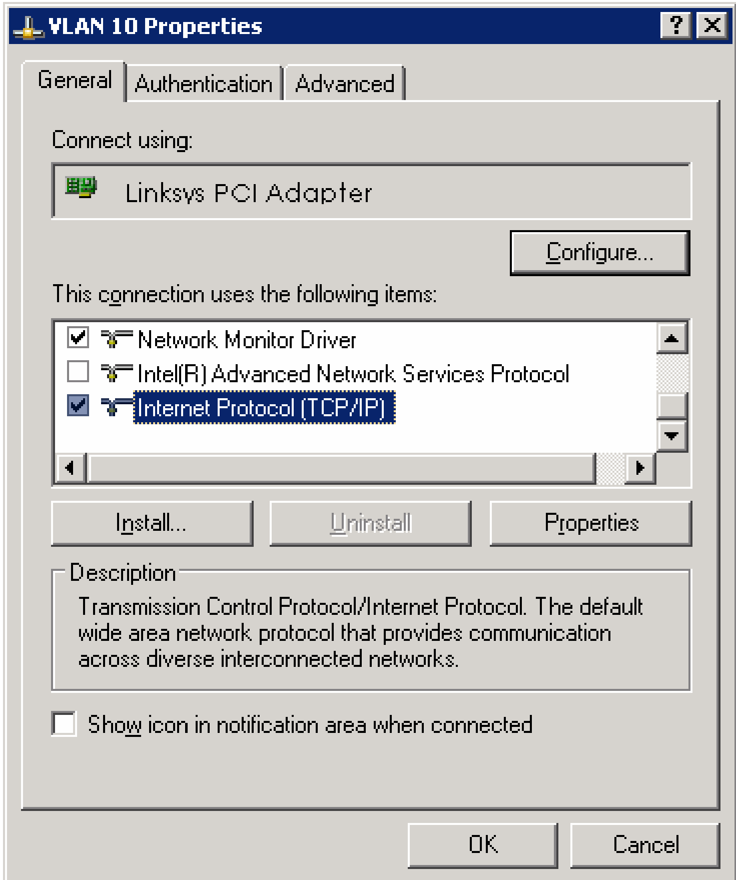


Figure 5-2: Modify the Properties for Interface on VLAN 10

Finally, configure the IP address shown in the diagram on the interface.

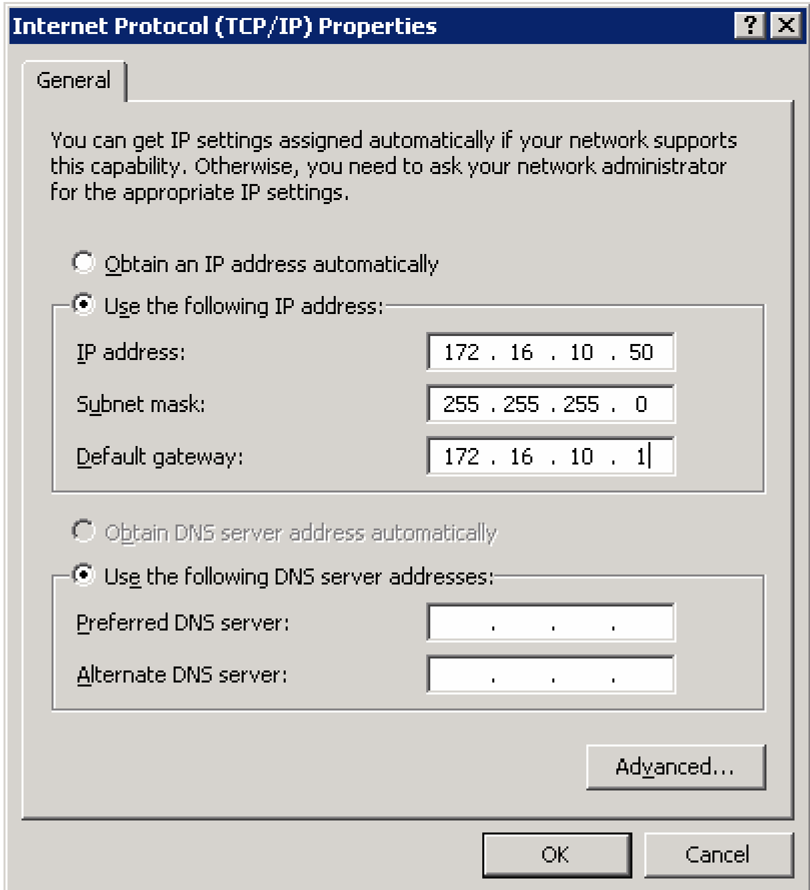


Figure 5-3: Configure IP Address, Subnet, and Gateway

Click **OK** to apply the TCP/IP settings, and then again to exit the configuration dialog box. From the Start Menu, click **Run**. Issue the **cmd** command and press the Return key. At the Windows command-line prompt, ping DLS1's VLAN 10 interface. You should receive responses. If you do not, troubleshoot, verifying the VLAN of the switchport and the IP address and subnet mask on each of the devices on VLAN 10.

```
C:\Documents and Settings\Administrator> ping 172.16.10.1

Pinging 172.16.10.1 with 32 bytes of data:

Reply from 172.16.10.1: bytes=32 time=1ms TTL=255
Reply from 172.16.10.1: bytes=32 time<1ms TTL=255
Reply from 172.16.10.1: bytes=32 time<1ms TTL=255
Reply from 172.16.10.1: bytes=32 time<1ms TTL=255

Ping statistics for 172.16.10.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```


Step 7

Enable IP routing on DLS1. This lets DLS1 route between all subnets shown in the diagram. DLS1 can effectively route between all the VLANs configured because it has an SVI in each subnet. Each IP subnet is shown in the output of the **show ip route** command issued on DLS1.

```
DLS1(config)# ip routing

DLS1# show ip route
Codes: C - connected, S - static, 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, su - IS-IS summary, 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

     172.16.0.0/24 is subnetted, 7 subnets
C       172.16.1.0 is directly connected, Vlan1
C       172.16.2.0 is directly connected, Vlan2
C       172.16.3.0 is directly connected, Vlan3
C       172.16.10.0 is directly connected, Vlan10
C       172.16.50.0 is directly connected, Vlan50
C       172.16.100.0 is directly connected, Vlan100
```

Step 8

When you first restart the WLAN controller, a configuration wizard prompts you to enter basic configuration attributes. You will know that you have entered the wizard interface when you see “Welcome to the Cisco Wizard Configuration Tool.” Pressing the Return key allows the default configuration options to be used. The default option will be in square brackets in the wizard prompts. If there is more than once choice in square brackets, it will be the option in capital letters.

The first prompt asks for a hostname. Use the default. Use “cisco” as both the username and password.

```
Welcome to the Cisco Wizard Configuration Tool
Use the '-' character to backup
System Name [Cisco_49:43:c0]:
Enter Administrative User Name (24 characters max): cisco
Enter Administrative Password (24 characters max): <cisco>
```

Enter the management interface information. The management interface communicates with the management workstation in VLAN 1. The interface number is 1, because this is the port trunked from the controller to the switch. The VLAN number is 0 for untagged. It is untagged because VLAN 1 is the native 802.1q VLAN, and is therefore sent untagged through 802.1q trunks.

```
Management Interface IP Address: 172.16.1.100
Management Interface Netmask: 255.255.255.0
```



```
Management Interface Default Router: 172.16.1.1
Management Interface VLAN Identifier (0 = untagged): 0
Management Interface Port Num [1 to 4]: 1
Management Interface DHCP Server IP Address: 172.16.1.1
```

Configure an interface to communicate with the lightweight access points. This will be in VLAN 100 and is tagged as such on the trunk.

```
AP Manager Interface IP Address: 172.16.100.100
AP Manager Interface Netmask: 255.255.255.0
AP Manager Interface Default Router: 172.16.100.1
AP Manager Interface VLAN Identifier (0 = untagged): 100
AP Manager Interface Port Num [1 to 4]: 1
AP Manager Interface DHCP Server (172.16.1.1): 172.16.100.1
```

Configure the virtual gateway IP address as 1.1.1.1 (this is acceptable because you are not using this for routing). The virtual gateway IP address is typically a fictitious, unassigned IP address, such as the address we are using here, to be used by Layer 3 Security and Mobility managers.

```
Virtual Gateway IP Address: 1.1.1.1
```

Configure the mobility group and network name as “ccnppod.” Allow static IP addresses by hitting enter, but do not configure a RADIUS server now.

```
Mobility/RF Group Name: ccnppod

Network Name (SSID): ccnppod
Allow Static IP Addresses [YES][no]:

Configure a RADIUS Server now? [YES][no]: no
Warning! The default WLAN security policy requires a RADIUS server.

Please see documentation for more details.
```

Use the defaults for the rest of the settings. (Hit enter on each prompt).

```
Enter Country Code (enter 'help' for a list of countries) [US]:

Enable 802.11b Network [YES][no]:
Enable 802.11a Network [YES][no]:
Enable 802.11g Network [YES][no]:
Enable Auto-RF [YES][no]:

Configuration saved!
Resetting system with new configuration...
```

Step 9

When the WLAN controller has finished restarting, log in with the username “cisco” and password “cisco.”

```
User: cisco
Password: <cisco>
```


Change the controller prompt to WLAN_CONTROLLER with the **config prompt** *name* command. Notice that the prompt changes.

```
(Cisco Controller) > config prompt WLAN_CONTROLLER

(WLAN_CONTROLLER) >
```

Enable Telnet and HTTP access to the WLAN controller. HTTPS access is enabled by default, but unsecured HTTP is not.

```
(WLAN_CONTROLLER) > config network telnet enable

(WLAN_CONTROLLER) > config network webmode enable
```

Save your configuration with the **save config** command, which is analogous to the Cisco IOS **copy run start** command.

```
(WLAN_CONTROLLER) > save config

Are you sure you want to save? (y/n) y

Configuration Saved!
```

To verify the configuration, you can issue the **show interface summary**, **show wlan summary**, and **show run-config** commands on the WLAN controller.

How is the WLAN controller's **show run-config** command different than the Cisco IOS **show running-config** command?

Final Configurations

```
DLS1# show run
hostname DLS1
!
ip routing
ip dhcp excluded-address 172.16.1.1 172.16.1.150
ip dhcp excluded-address 172.16.2.1 172.16.2.150
ip dhcp excluded-address 172.16.3.1 172.16.3.150
ip dhcp excluded-address 172.16.10.1 172.16.10.150
ip dhcp excluded-address 172.16.50.1 172.16.50.150
ip dhcp excluded-address 172.16.100.1 172.16.100.150
!
ip dhcp pool pool2
   network 172.16.2.0 255.255.255.0
   default-router 172.16.2.1
!
ip dhcp pool pool3
   network 172.16.3.0 255.255.255.0
   default-router 172.16.3.1
!
ip dhcp pool pool10
   network 172.16.10.0 255.255.255.0
   default-router 172.16.10.1
!
```



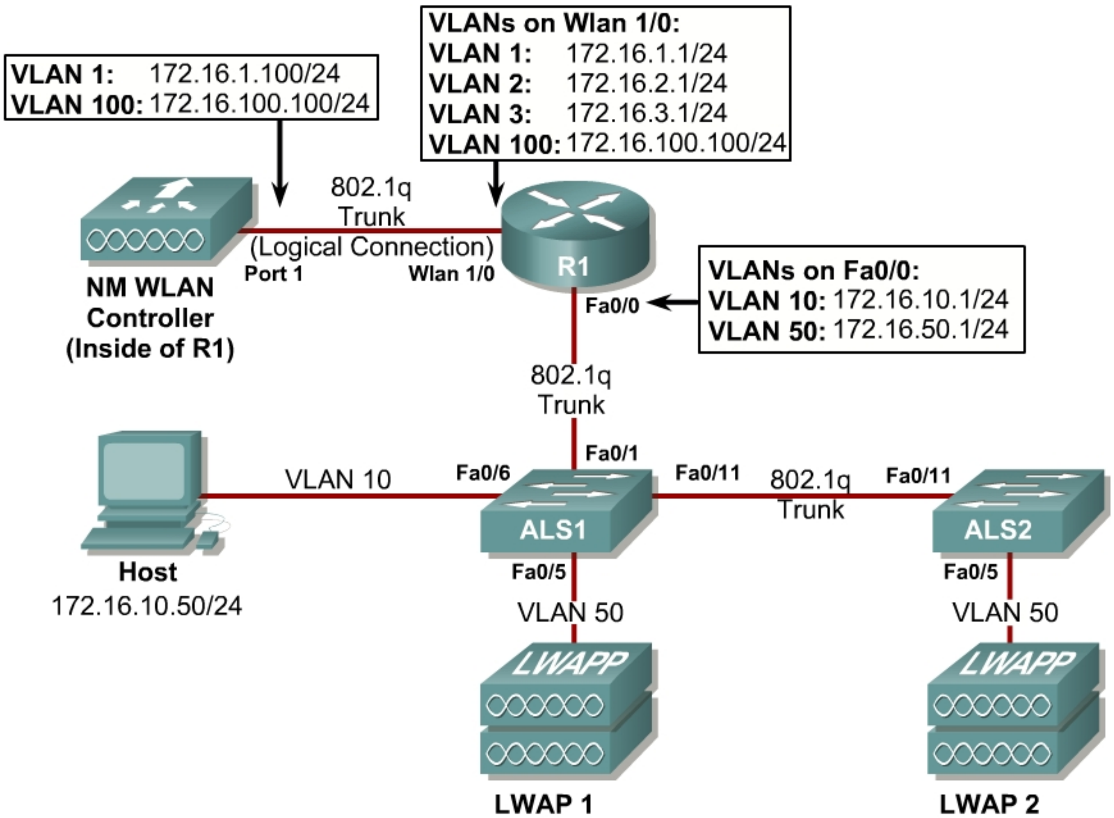
```
ip dhcp pool pool50
   network 172.16.50.0 255.255.255.0
   default-router 172.16.50.1
   option 43 hex f104ac106464
   option 60 ascii "Cisco AP c1240"
!
ip dhcp pool pool100
   network 172.16.100.0 255.255.255.0
   default-router 172.16.100.1
!
ip dhcp pool pool1
   network 172.16.1.0 255.255.255.0
   default-router 172.16.1.1
!
interface FastEthernet0/5
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/6
 switchport mode access
 switchport access vlan 10
 spanning-tree portfast
!
interface FastEthernet0/7
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface FastEthernet0/9
 switchport trunk encapsulation dot1q
 switchport mode trunk
!
interface Vlan1
 ip address 172.16.1.1 255.255.255.0
 no shutdown
!
interface Vlan2
 ip address 172.16.2.1 255.255.255.0
 no shutdown
!
interface Vlan3
 ip address 172.16.3.1 255.255.255.0
 no shutdown
!
interface Vlan10
 ip address 172.16.10.1 255.255.255.0
 no shutdown
!
interface Vlan50
 ip address 172.16.50.1 255.255.255.0
 no shutdown
!
interface Vlan100
 ip address 172.16.100.1 255.255.255.0
 no shutdown
end

ALS1# show run
hostname ALS1
!
interface FastEthernet0/5
 switchport access vlan 50
 switchport mode access
 spanning-tree portfast
```



```
!
interface FastEthernet0/7
 switchport mode trunk
end

ALS2# show run
hostname ALS2
!
interface FastEthernet0/5
 switchport access vlan 50
 switchport mode access
 spanning-tree portfast
!
interface FastEthernet0/9
 switchport mode trunk
!
end
```


Lab 6.1b Configuring a WLAN Controller

Topology Diagram



Scenario

In the next two labs, you will configure a wireless solution involving a router with a built-in WLAN controller, two lightweight wireless access points, and a switched wired network. You will configure a WLAN controller to broadcast SSIDs from the lightweight wireless access points. If you have a wireless client nearby, connect to the WLANs and access devices from the inside of your pod to verify your configuration of the controller and access points.

Note: It is required that you upgrade the NM WLC firmware image to 4.0.206.0 or higher in order to accomplish this lab.

Step 1

Erase the startup-config file and delete the vlan.dat file from each switch, and erase the startup-config file on each router. Set hostnames on all of the devices.

Step 2

Explanation of VLANs:
VLAN 1 – This VLAN is the management VLAN for the WLC
VLAN 2 and VLAN 3 – These VLANs are for hosts in the WLANs
VLAN 10 – The host is in this VLAN
VLAN 50 – The APs are in this VLAN
VLAN 100 – The AP-manager interface of the WLC is in this VLAN

Configure ALS1 and ALS2 to run VTP in transparent mode in the VTP domain "CISCO", and create VLANs 10 and 50 on them. Also, set up a trunk link between them as well as towards R1.

```
ALS1(config)# vtp mode transparent
Setting device to VTP TRANSPARENT mode.
ALS1(config)# vtp domain CISCO
Changing VTP domain name from NULL to CISCO
ALS1(config)# vlan 10,50
ALS1(config-vlan)# int fastethernet0/1
ALS1(config-if)# switchport mode trunk
ALS1(config-if)# int fastethernet0/11
ALS1(config-if)# switchport mode trunk

ALS2(config)# vtp mode transparent
Setting device to VTP TRANSPARENT mode.
ALS2(config)# vtp domain CISCO
Changing VTP domain name from NULL to CISCO
ALS2(config)# vlan 10,50
ALS2(config-if)# int fastethernet0/11
ALS2(config-if)# switchport mode trunk
```

Step 3

Configure the subinterfaces on R1 for both FastEthernet0/0 and wlan-controller1/0 ports shown in the diagram. Both will be configured as 802.1q trunks with a VLAN on each subinterface. Make sure you use the native VLAN on the physical wlan-controller1/0 interface, as you will not be able to connect to the controller unless there is an IP address on the physical interface. Don't forget to add **no shutdown** commands to both physical interfaces.

```
R1(config)# int fastethernet0/0
R1(config-if)# no shutdown
R1(config-if)# int fastethernet0/0.10
R1(config-subif)# encapsulation dot1q 10
R1(config-subif)# ip address 172.16.10.1 255.255.255.0
R1(config-subif)# int fastethernet0/0.50
R1(config-subif)# encapsulation dot1q 50
R1(config-subif)# ip address 172.16.50.1 255.255.255.0
R1(config-subif)# int wlan-controller1/0
R1(config-if)# ip address 172.16.1.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# int wlan-controller1/0.2
R1(config-subif)# encapsulation dot1q 2
```



```
If the interface doesn't support baby giant frames
maximum mtu of the interface has to be reduced by 4
bytes on both sides of the connection to properly
transmit or receive large packets. Please refer to
documentation on configuring IEEE 802.1Q vLANs.

R1(config-subif)# ip address 172.16.2.1 255.255.255.0
R1(config-subif)# int wlan-controller1/0.3
R1(config-subif)# encapsulation dot1q 3
R1(config-subif)# ip address 172.16.3.1 255.255.255.0
R1(config-subif)# int wlan-controller1/0.100
R1(config-subif)# encapsulation dot1q 100
R1(config-subif)# ip address 172.16.100.1 255.255.255.0
```

Step 4

DHCP gives out dynamic IP addresses on a subnet to network devices or hosts rather than statically setting the addresses. This is useful when dealing with lightweight access points, which usually do not have an initial configuration. The WLAN controller that the lightweight wireless access point associates with defines the configuration. A lightweight access point can dynamically receive an IP address and then communicate over IP with the WLAN controller. In this scenario, you will also use it to assign IP addresses to hosts that connect to the WLANs.

First, set up R1 to exclude the first 150 addresses from each subnet from DHCP to avoid conflicts with static IP addresses by using the global configuration command **ip dhcp excluded-address** *low-address* [*high-address*].

```
R1(config)# ip dhcp excluded-address 172.16.1.1 172.16.1.150
R1(config)# ip dhcp excluded-address 172.16.2.1 172.16.2.150
R1(config)# ip dhcp excluded-address 172.16.3.1 172.16.3.150
R1(config)# ip dhcp excluded-address 172.16.10.1 172.16.10.150
R1(config)# ip dhcp excluded-address 172.16.50.1 172.16.50.150
R1(config)# ip dhcp excluded-address 172.16.100.1 172.16.100.150
```

To advertise on different subnets, create DHCP pools with the **ip dhcp pool** *name* command. After a pool is configured for a certain subnet, the IOS DHCP server processes requests on that subnet, because it is enabled by default. From the DHCP pool prompt, set the network and mask to use with the **network** *address* /*mask* command. Set a default gateway with the **default-router** *address* command.

VLAN 50 also uses the **option** command, which allows you to specify a DHCP option. In this case, option 43 is specified (a vendor-specific option), which gives the lightweight wireless access points the IP address of the WLAN controller AP Manager interface. It is specified in a hexadecimal TLV (type, length, value) format. F1 is the hardcoded type of option, 04 represents the length of the value (an IP address is 4 octets), and AC106464 is the hexadecimal representation of 172.16.100.100, which is going to be the AP manager address of the WLAN controller. DHCP option 60 specifies the identifier that access points will use in DHCP. This lab was written using Cisco Aironet 1240 series access points. If you are using a different access point series, consult http://www.cisco.com/univercd/cc/td/doc/product/wireless/aero1500/1500hig5/1500_axg.htm.

```
R1(config)# ip dhcp pool pool1
R1(dhcp-config)# network 172.16.1.0 /24
R1(dhcp-config)# default-router 172.16.1.1
R1(dhcp-config)# ip dhcp pool pool2
R1(dhcp-config)# network 172.16.2.0 /24
R1(dhcp-config)# default-router 172.16.2.1
R1(dhcp-config)# ip dhcp pool pool3
R1(dhcp-config)# network 172.16.3.0 /24
R1(dhcp-config)# default-router 172.16.3.1
R1(dhcp-config)# ip dhcp pool pool10
R1(dhcp-config)# network 172.16.10.0 /24
R1(dhcp-config)# default-router 172.16.10.1
R1(dhcp-config)# ip dhcp pool pool50
R1(dhcp-config)# network 172.16.50.0 /24
R1(dhcp-config)# default-router 172.16.50.1
R1(dhcp-config)# option 43 hex f104ac106464
R1(dhcp-config)# option 60 ascii "Cisco AP c1240"
R1(dhcp-config)# ip dhcp pool pool100
R1(dhcp-config)# network 172.16.100.0 /24
R1(dhcp-config)# default-router 172.16.100.1
```

Step 5

On both switches, configure all access points to bypass the spanning-tree port states with the **spanning-tree portfast** command. With this command, each access point receives an IP address from DHCP immediately, without worrying about timing out from DHCP. Configure the switchports going to the lightweight wireless access points in VLAN 50. R1 will route the tunneled WLAN traffic towards the WLAN controllers AP-manager interface.

```
ALS1(config)# int fastethernet0/5
ALS1(config-if)# switchport mode access
ALS1(config-if)# switchport access vlan 50
ALS1(config-if)# spanning-tree portfast

ALS2(config)# int fastethernet0/5
ALS2(config-if)# switchport mode access
ALS2(config-if)# switchport access vlan 50
ALS2(config-if)# spanning-tree portfast
```

Step 6

You have a PC running Microsoft Windows attached to ALS1. First, configure the switchport connecting to the host in VLAN 10 with portfast. Management traffic from the host for the WLAN controller will be routed to R1 towards the management interface of the WLC.

```
ALS1(config)# int fastethernet0/6
ALS1(config-if)# switchport mode access
ALS1(config-if)# switchport access vlan 10
```



```
ALS1(config-if)# spanning-tree portfast
```

Next, configure the host with an IP address in VLAN 10, which will later be used to access the HTTP web interface of the WLAN controller later. Follow the procedure below to prepare the host to access the WLAN controller.

In the **Control Panel**, select **Network Connections**.

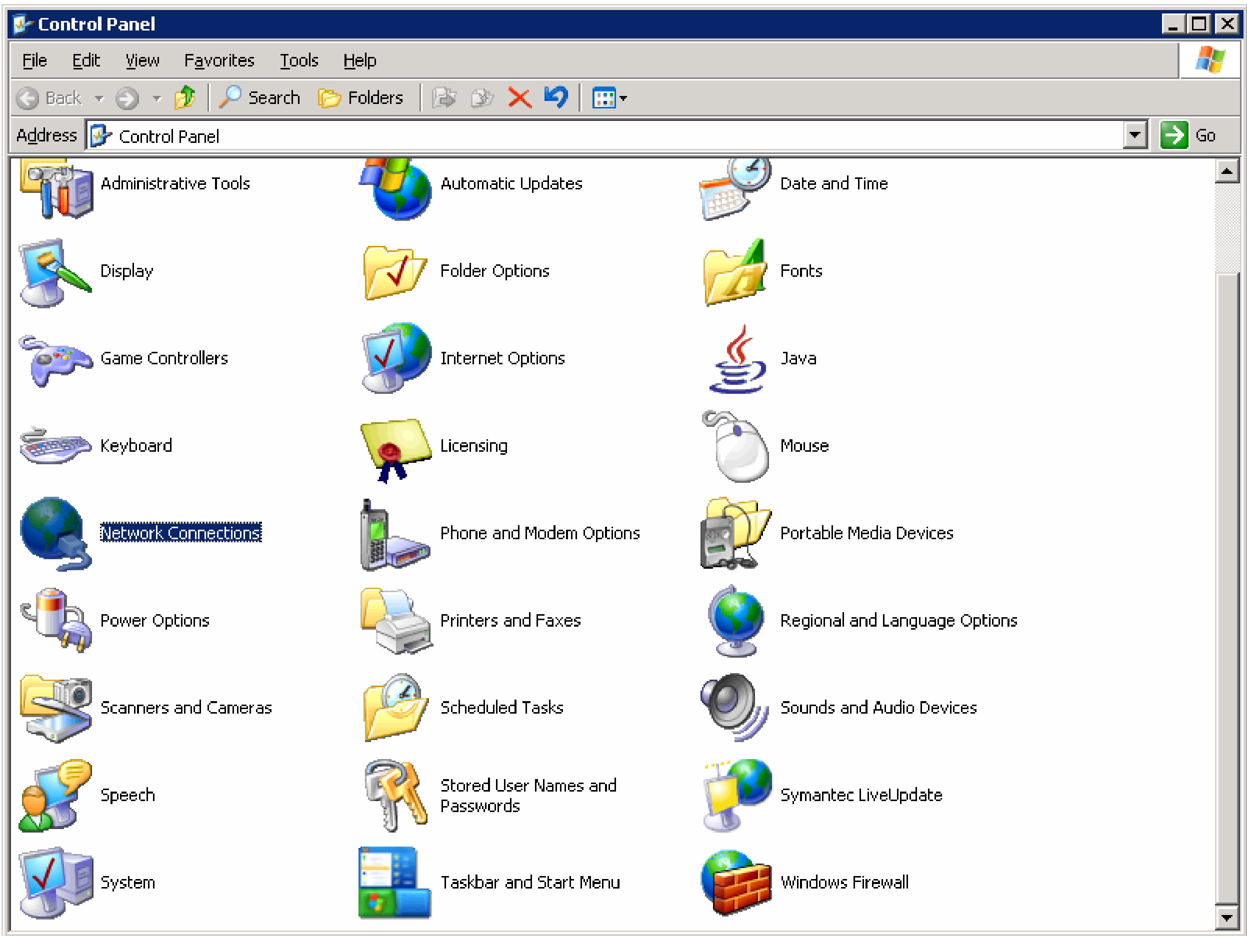


Figure 5-1: Microsoft Windows Control Panel

Right-click on the LAN interface that connects to ALS1, and select **Properties**. Select **Internet Protocol (TCP/IP)** and then click the **Properties** button.

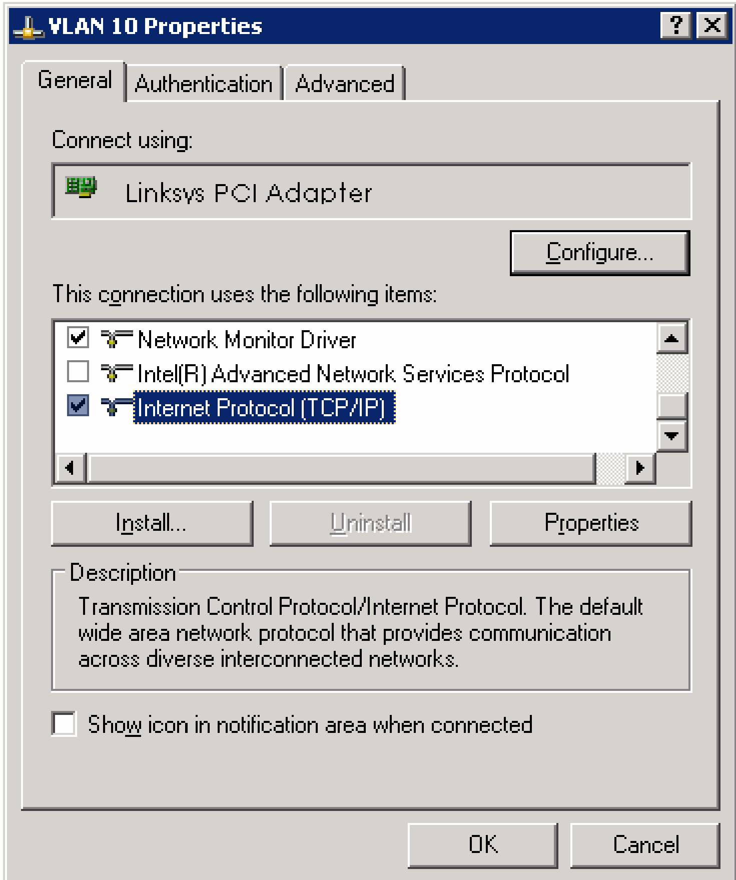


Figure 5-2: Modify the Properties for Interface on VLAN 10

Finally, configure the IP address shown in the diagram on the interface.

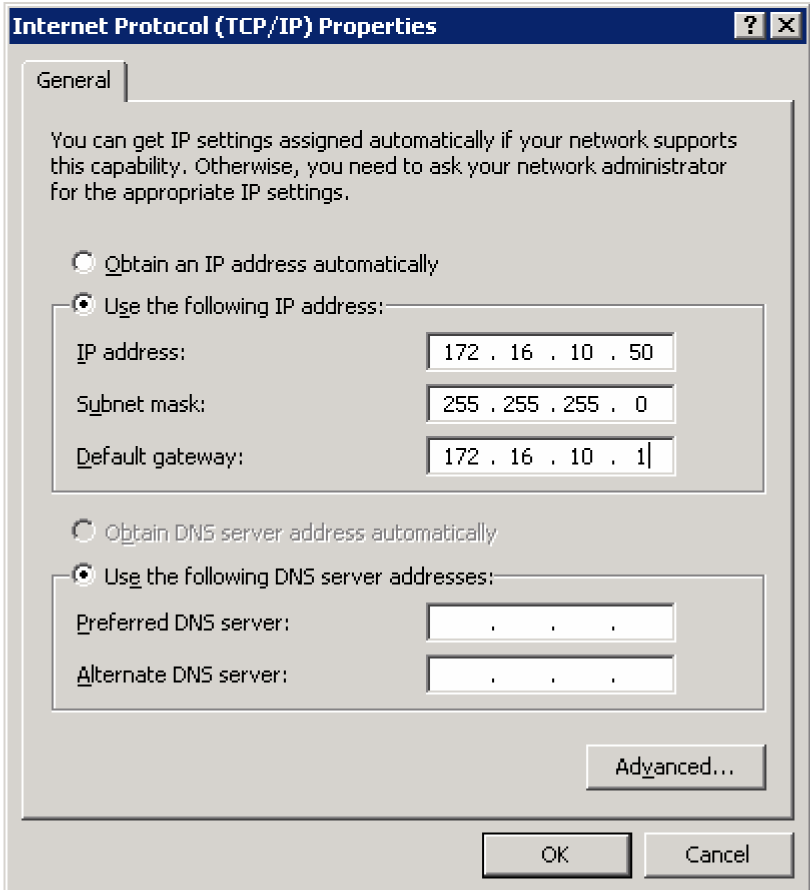


Figure 5-3: Configure IP Address, Subnet, and Gateway

Click **OK** to apply the TCP/IP settings, and then again to exit the configuration dialog box. From the Start Menu, click **Run**. Issue the **cmd** command and press the Return key. At the Windows command-line prompt, ping R1's VLAN 10 interface. You should receive responses. If you do not, troubleshoot, verifying the VLAN of the switchport and the IP address and subnet mask on each of the devices on VLAN 10.

```
C:\Documents and Settings\Administrator> ping 172.16.10.1

Pinging 172.16.10.1 with 32 bytes of data:

Reply from 172.16.10.1: bytes=32 time=1ms TTL=255
Reply from 172.16.10.1: bytes=32 time<1ms TTL=255
Reply from 172.16.10.1: bytes=32 time<1ms TTL=255
Reply from 172.16.10.1: bytes=32 time<1ms TTL=255

Ping statistics for 172.16.10.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```


Step 7

R1 will route between all subnets shown in the diagram, because it has a connected interface in each subnet. Each IP subnet is shown in the output of the **show ip route** command issued on R1.

```
R1#show ip route
Codes: C - connected, S - static, 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
       i - IS-IS, su - IS-IS summary, 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

     172.16.0.0/24 is subnetted, 6 subnets
C       172.16.50.0 is directly connected, FastEthernet0/0.50
C       172.16.10.0 is directly connected, FastEthernet0/0.10
C       172.16.1.0 is directly connected, wlan-controller1/0
C       172.16.2.0 is directly connected, wlan-controller1/0.2
C       172.16.3.0 is directly connected, wlan-controller1/0.3
C       172.16.100.0 is directly connected, wlan-controller1/0.100
```

Step 8

Now that the underlying network infrastructure is set up, you can set up the WLAN controller.

At R1's privileged exec prompt, you can control the state of the WLC inside R1. To see what types of commands you can execute, use the command **service-module *interface* ?**.

```
R1#service-module wlan-controller1/0 ?
  reload      Reload service module
  reset       Hardware reset of Service Module
  session     Service module session
  shutdown    Shutdown service module
  statistics  Service Module Statistics
  status      Service Module Information
```

After you review what you can do to the internal wlan-controller, reset it. Right after the line protocol comes back up on the controller, connect to it using the **session** argument for **service-module** as shown below.

```
R1#service-module wlan-controller1/0 reset
Use reset only to recover from shutdown or failed state
Warning: May lose data on the hard disc!
Do you want to reset?[confirm]
Trying to reset Service Module wlan-controller1/0.

R1#
*Feb 14 06:27:03.311: %LINEPROTO-5-UPDOWN: Line protocol on Interface wlan-
controller1/0, changed state to down
*Feb 14 06:27:23.311: %LINEPROTO-5-UPDOWN: Line protocol on Interface wlan-
controller1/0, changed state to up
R1#service-module wlan-controller1/0 session
```



```
Trying 172.16.1.1, 2066 ... Open
Cisco Bootloader Loading stage2...

    Cisco Bootloader (Version 4.0.206.0)


                     .o88b. d888888b .d8888.  .o88b.  .d88b.
                    d8P  Y8   `88'   88'  YP d8P  Y8 .8P  Y8.
                    8P         88    `8bo.   8P      88    88
                    8b         88      `Y8b. 8b      88    88
                    Y8b  d8   .88.   db   8D Y8b  d8 `8b  d8'
                     `Y88P' Y888888P `8888Y'  `Y88P'  `Y88P'

<OUTPUT OMITTED>
```

If you start up the WLC and it does not have a cleared configuration, you may use "Recover-Config" as the first username used to login after the NM has been restarted. If you are already at a command prompt for the WLC, use the **clear config** command followed by the **reset system** command.

Once connected to the WLAN controller with an erased configuration, a wizard starts to allow you to configure basic settings. Pressing the Return key allows the default configuration options to be used (whatever appears in square brackets will be the default, and if there are multiple entries in square brackets, the one in capital letters will be the default).

The first prompt asks for a hostname. Use the default. Use "cisco" as both the username and password.

```
Welcome to the Cisco Wizard Configuration Tool
Use the '-' character to backup
System Name [Cisco_49:43:c0]:
Enter Administrative User Name (24 characters max): cisco
Enter Administrative Password (24 characters max): <cisco>
```

Enter the management interface information. The management interface communicates with the management workstation in VLAN 1. The interface number is 1, because this is the only interface on the NM WLC (it is the logical connection to R1's wlan-controller1/0). The VLAN number is 0 for untagged. It is untagged it is the native 802.1q VLAN, and is going to be sent to the physical (non-subinterface) interface of R1.

```
Management Interface IP Address: 172.16.1.100
Management Interface Netmask: 255.255.255.0
Management Interface Default Router: 172.16.1.1
Management Interface VLAN Identifier (0 = untagged): 0
Management Interface Port Num [1]: 1
Management Interface DHCP Server IP Address: 172.16.1.1
```

Configure an interface to communicate with the lightweight access points (tunneled access point traffic will be sent here). This will be in VLAN 100 and is tagged as such on the trunk.

```
AP Manager Interface IP Address: 172.16.100.100
```



```
AP Manager Interface Netmask: 255.255.255.0
AP Manager Interface Default Router: 172.16.100.1
AP Manager Interface VLAN Identifier (0 = untagged): 100
AP Manager Interface Port Num [1]: 1
AP Manager Interface DHCP Server (172.16.1.1): 172.16.100.1
```

Configure the virtual gateway IP address as 1.1.1.1 (this is acceptable because you are not using this for routing). The virtual gateway IP address is typically a fictitious, unassigned IP address, such as the address we are using here, to be used by Layer 3 Security and Mobility managers.

```
Virtual Gateway IP Address: 1.1.1.1
```

Configure the mobility group and network name as “ccnppod.” Allow static IP addresses by hitting enter, but do not configure a RADIUS server now.

```
Mobility/RF Group Name: ccnppod

Network Name (SSID): ccnppod
Allow Static IP Addresses [YES][no]:

Configure a RADIUS Server now? [YES][no]: no
Warning! The default WLAN security policy requires a RADIUS server.

Please see documentation for more details.
```

Use the defaults for the rest of the settings by hitting enter, except for the time settings. Do not configure a time server, but do set the current time.

```
Enter Country Code (enter 'help' for a list of countries) [US]:

Enable 802.11b Network [YES][no]:
Enable 802.11a Network [YES][no]:
Enable 802.11g Network [YES][no]:
Enable Auto-RF [YES][no]:

Configure a NTP server now? [YES][no]: no
Configure the system time now? [YES][no]: yes
Enter the date in MM/DD/YY format: 02/14/07
Enter the time in HH:MM:SS format: 02:17:00

Configuration correct? If yes, system will save it and reset. [yes][NO]: yes

Configuration saved!
Resetting system with new configuration...
```

Step 9

When the WLAN controller has finished restarting, log in with the username “cisco” and password “cisco.”

```
User: cisco
Password: <cisco>
```

Change the controller prompt to WLAN_CONTROLLER with the **config prompt** *name* command. Notice that the prompt changes.


```
(Cisco Controller) > config prompt WLAN_CONTROLLER

(WLAN_CONTROLLER) >
```

Enable Telnet and HTTP access to the WLAN controller. HTTPS access is enabled by default, but unsecured HTTP is not.

```
(WLAN_CONTROLLER) > config network telnet enable

(WLAN_CONTROLLER) > config network webmode enable
```

Save your configuration with the **save config** command, which is analogous to the Cisco IOS **copy run start** command.

```
(WLAN_CONTROLLER) > save config

Are you sure you want to save? (y/n) y


Configuration Saved!
```

To verify the configuration, you can issue the **show interface summary**, **show wlan summary**, and **show run-config** commands on the WLAN controller.

How is the WLAN controller's **show run-config** command different than the Cisco IOS **show running-config** command?

Final Configuration

```
R1#show run
hostname R1
!
ip dhcp excluded-address 172.16.1.1 172.16.1.150
ip dhcp excluded-address 172.16.2.1 172.16.2.150
ip dhcp excluded-address 172.16.3.1 172.16.3.150
ip dhcp excluded-address 172.16.10.1 172.16.10.150
ip dhcp excluded-address 172.16.50.1 172.16.50.150
ip dhcp excluded-address 172.16.100.1 172.16.100.150
!
ip dhcp pool pool1
   network 172.16.1.0 255.255.255.0
   default-router 172.16.1.1
!
ip dhcp pool pool2
   network 172.16.2.0 255.255.255.0
   default-router 172.16.2.1
!
ip dhcp pool pool3
   network 172.16.3.0 255.255.255.0
   default-router 172.16.3.1
!
ip dhcp pool pool10
   network 172.16.10.0 255.255.255.0
```



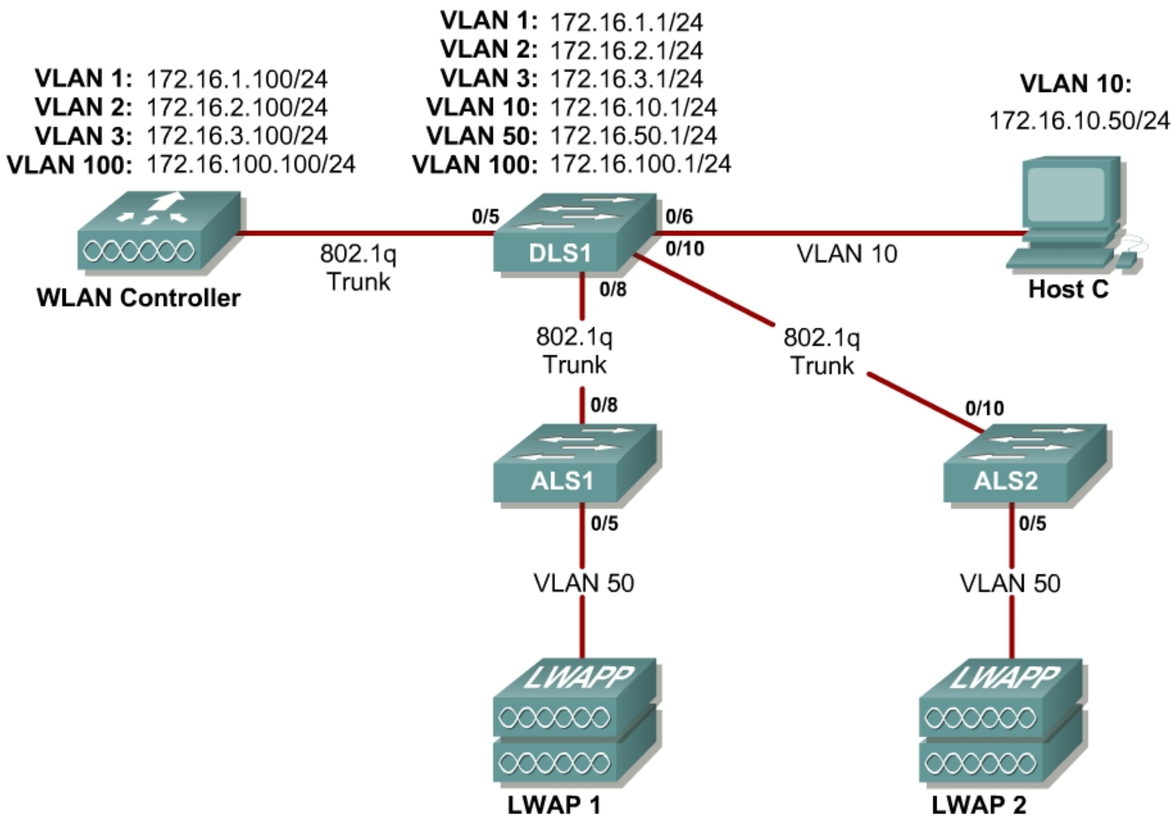
```
   default-router 172.16.10.1
!
ip dhcp pool pool50
   network 172.16.50.0 255.255.255.0
   default-router 172.16.50.1
   option 43 hex f104ac106464
   option 60 ascii "Cisco AP c1240"
!
ip dhcp pool pool100
   network 172.16.100.0 255.255.255.0
   default-router 172.16.100.1
!
interface FastEthernet0/0
 no shutdown
!
interface FastEthernet0/0.10
 encapsulation dot1Q 10
 ip address 172.16.10.1 255.255.255.0
!
interface FastEthernet0/0.50
 encapsulation dot1Q 50
 ip address 172.16.50.1 255.255.255.0
!
interface wlan-controller1/0
 ip address 172.16.1.1 255.255.255.0
 no shutdown
!
interface wlan-controller1/0.2
 encapsulation dot1Q 2
 ip address 172.16.2.1 255.255.255.0
!
interface wlan-controller1/0.3
 encapsulation dot1Q 3
 ip address 172.16.3.1 255.255.255.0
!
interface wlan-controller1/0.100
 encapsulation dot1Q 100
 ip address 172.16.100.1 255.255.255.0
end

ALS1#show run
hostname ALS1
!
vtp domain CISCO
vtp mode transparent
!
vlan 10,50
!
interface FastEthernet0/1
 switchport mode trunk
!
interface FastEthernet0/5
 switchport access vlan 50
 switchport mode access
 spanning-tree portfast
!
interface FastEthernet0/6
 switchport access vlan 10
 switchport mode access
 spanning-tree portfast
!
interface FastEthernet0/11
 switchport mode trunk
```



```
end

ALS2#show run
hostname ALS2
!
vtp domain CISCO
vtp mode transparent
!
vlan 10,50
!
interface FastEthernet0/5
 switchport access vlan 50
 switchport mode access
 spanning-tree portfast
!
interface FastEthernet0/11
 switchport mode trunk
end
```


Lab 6.2a Configuring a WLAN Controller via the Web Interface

Topology Diagram



Scenario

Continuing from the previous lab, you will now set up the WLAN controller through its web interface. Previously you configured it through the CLI.

Step 1

Set up all the switches as they were in the previous lab. Make sure that the WLAN controller and host also have the same configuration as before.

Step 2

On the host, open up Internet Explorer and go to the URL "https://172.16.1.100". This is the secure method of connecting to the management interface of the WLAN controller. You can also use "http://172.16.1.100" since we previously enabled regular insecure HTTP access in the CLI for Lab 6.1. If you connect to the secure address, you may be prompted with a security warning. Click **Yes** to accept it and you will be presented with the login screen for the WLAN controller. Click **Login** and an authentication dialog box will appear.

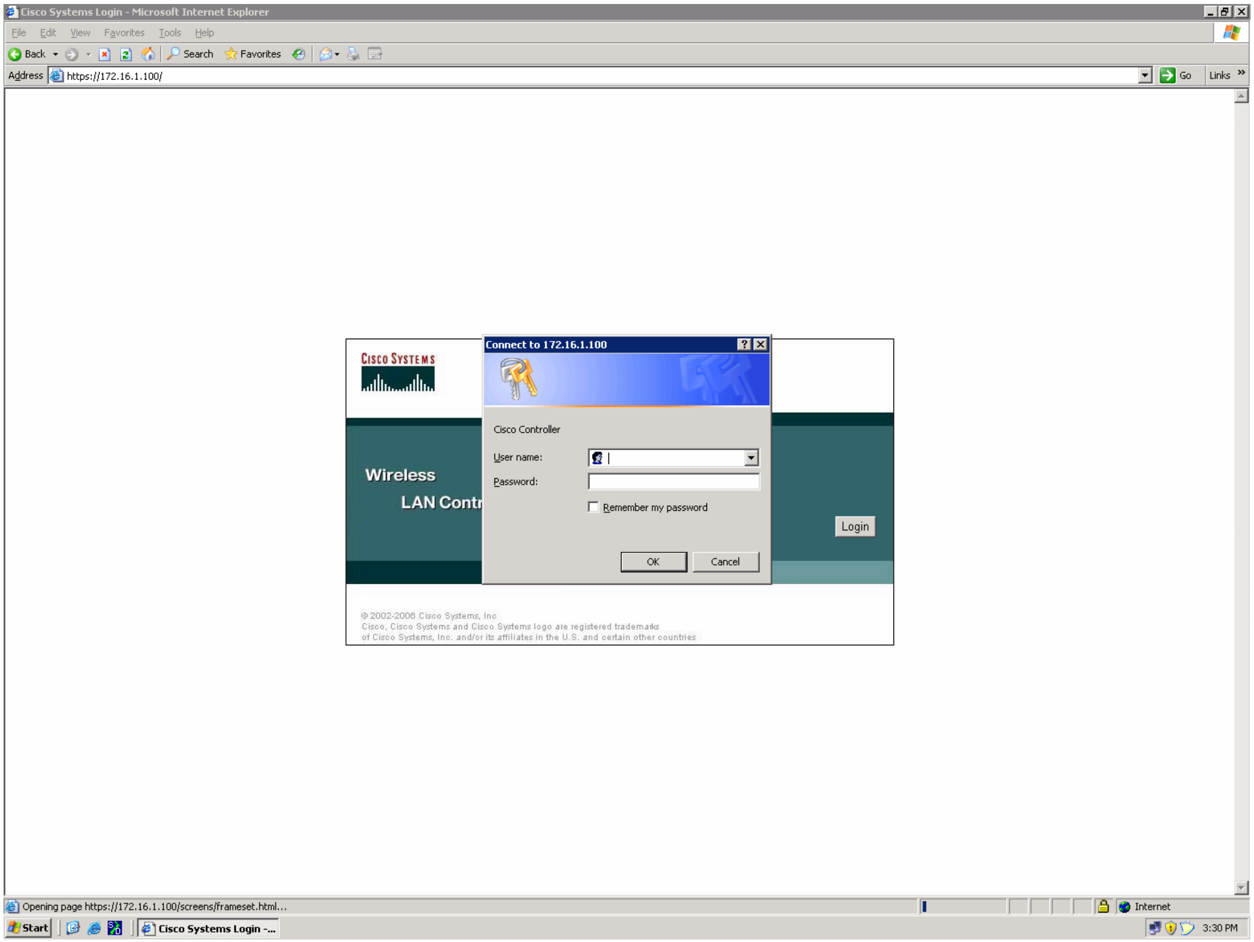


Figure 2-1: Authentication Dialog Box for WLAN Controller Web Access

Use "cisco" as both the username and password. You configured these in the previous lab. Click **OK** to get to the main page of the graphical user interface (GUI). You are then presented with the monitor page for the WLAN controller.

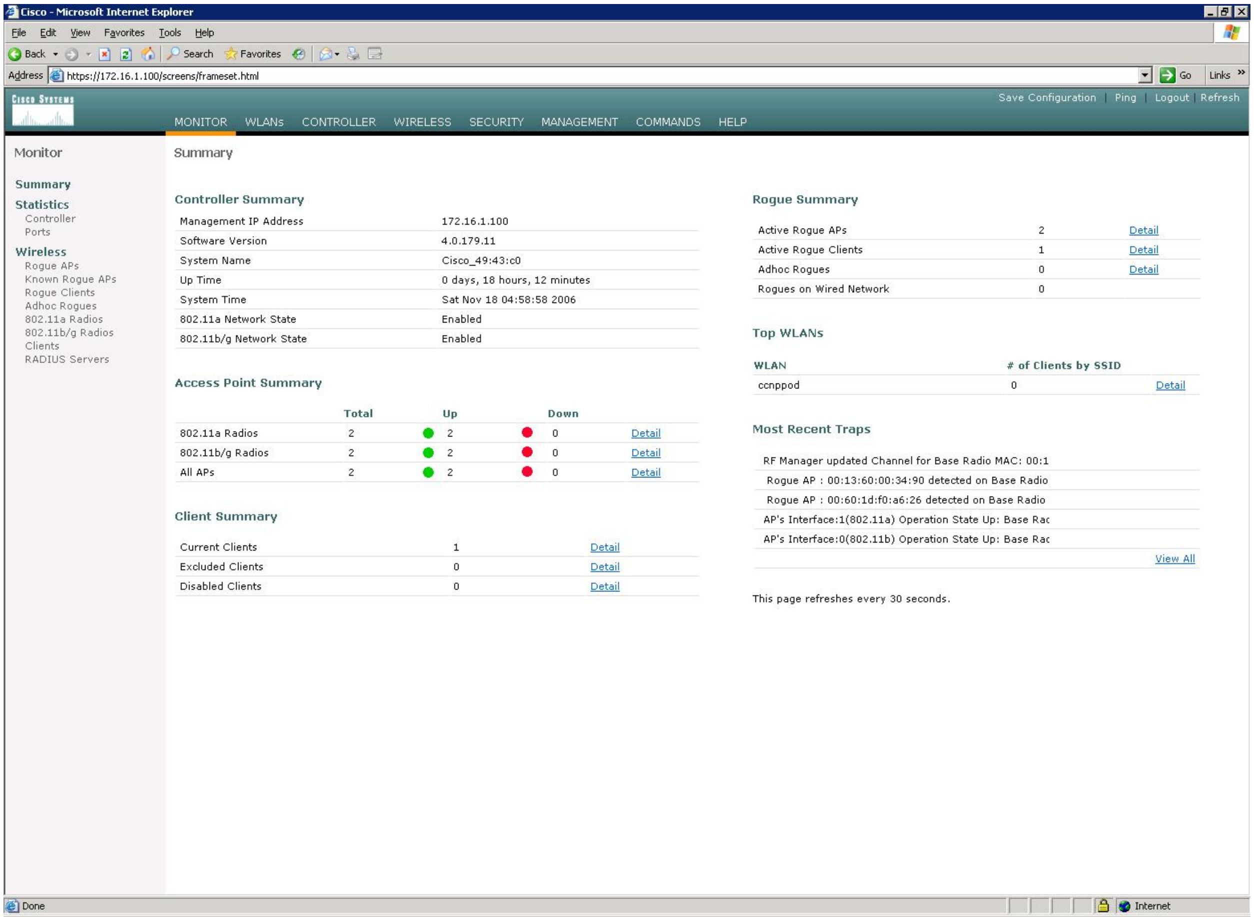


Figure 2-2: WLAN Controller Monitor Page

Make sure you see 2 access points under the “Access Point Summary” part of the page. You may also see it detecting rogue access points if your lab has other wireless networks around it; this behavior is normal. You can also see various port controller and port statistics by clicking their respective links on the left-hand menu on the screen.

Step 3

The next task in configuring WLANs is to add in the logical interfaces on the WLAN controller corresponding to VLANs 2 and 3. To do this, click the **Controller** link on the top of the web interface. Then, click **Interfaces** link on the left side bar.

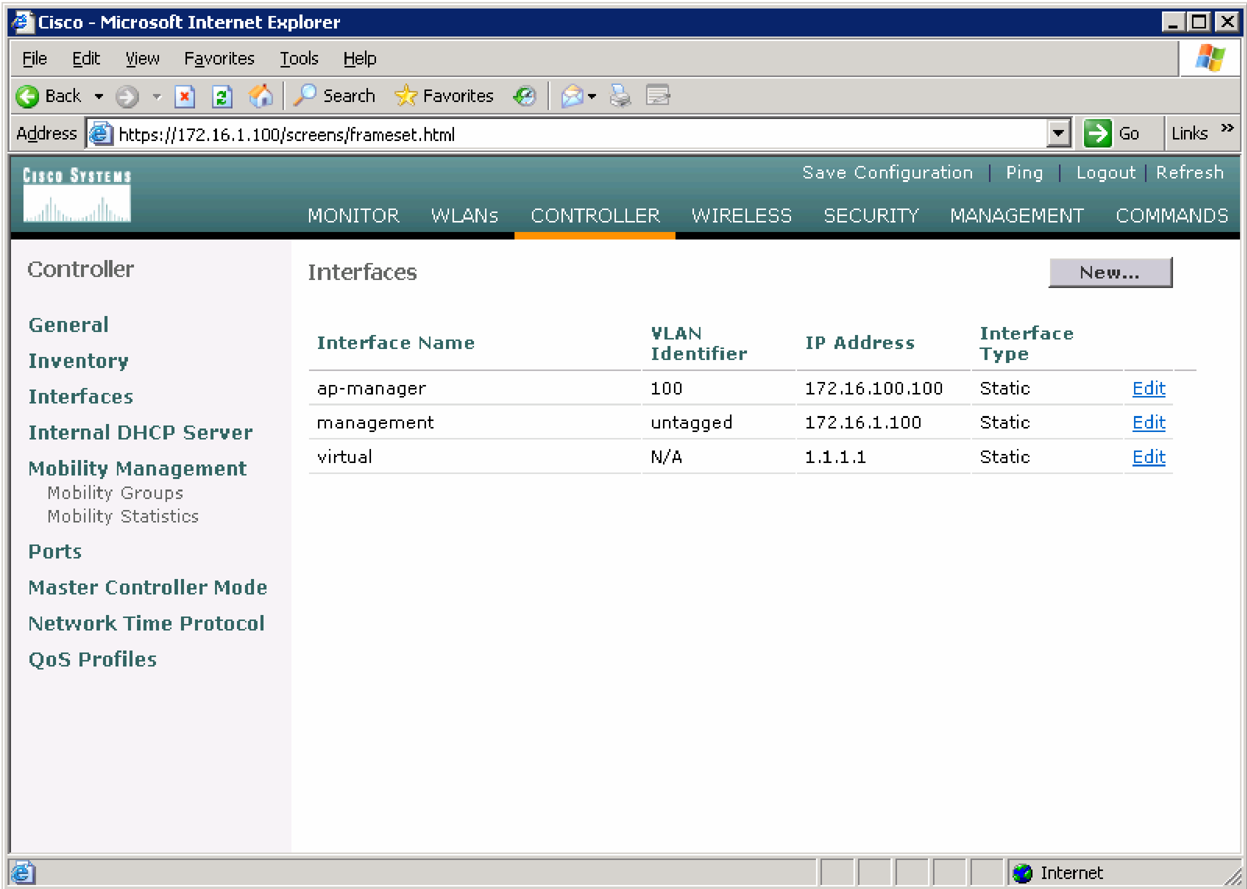


Figure 3-1: Interface Configuration Page

Click the **New...** link to create a new interface. Give the new interface a name of VLAN2 and VLAN number 2. Click **Apply** to submit the parameters.

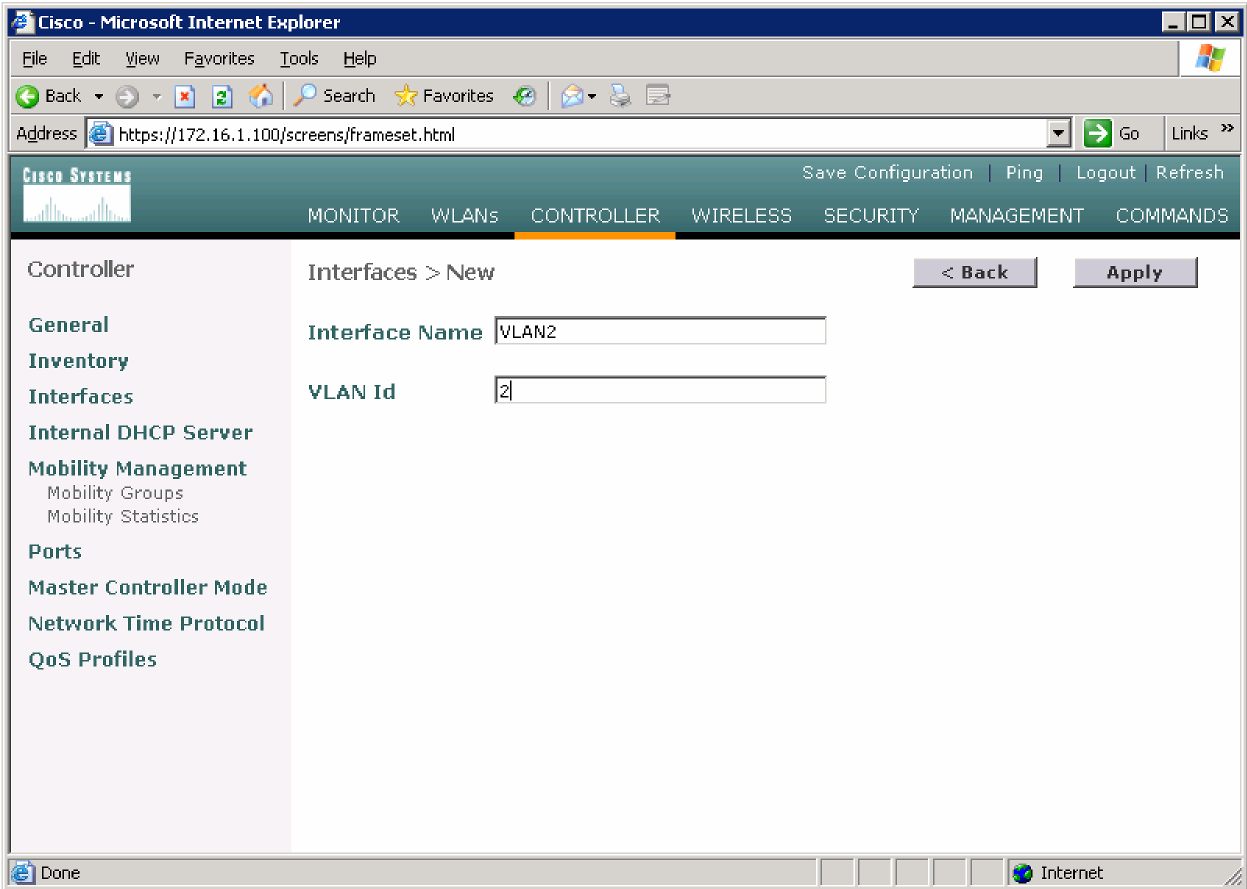


Figure 3-2: Creating a New VLAN Interface

On the next page, configure the IP address shown in the diagram. Also configure this on physical port 1, since that is the port trunked to the switch. After you have entered in all the changes, click **Apply**. Click **OK** to the warning box that comes up. This warning says that there may be a temporary connectivity loss on the APs while changes are applied.

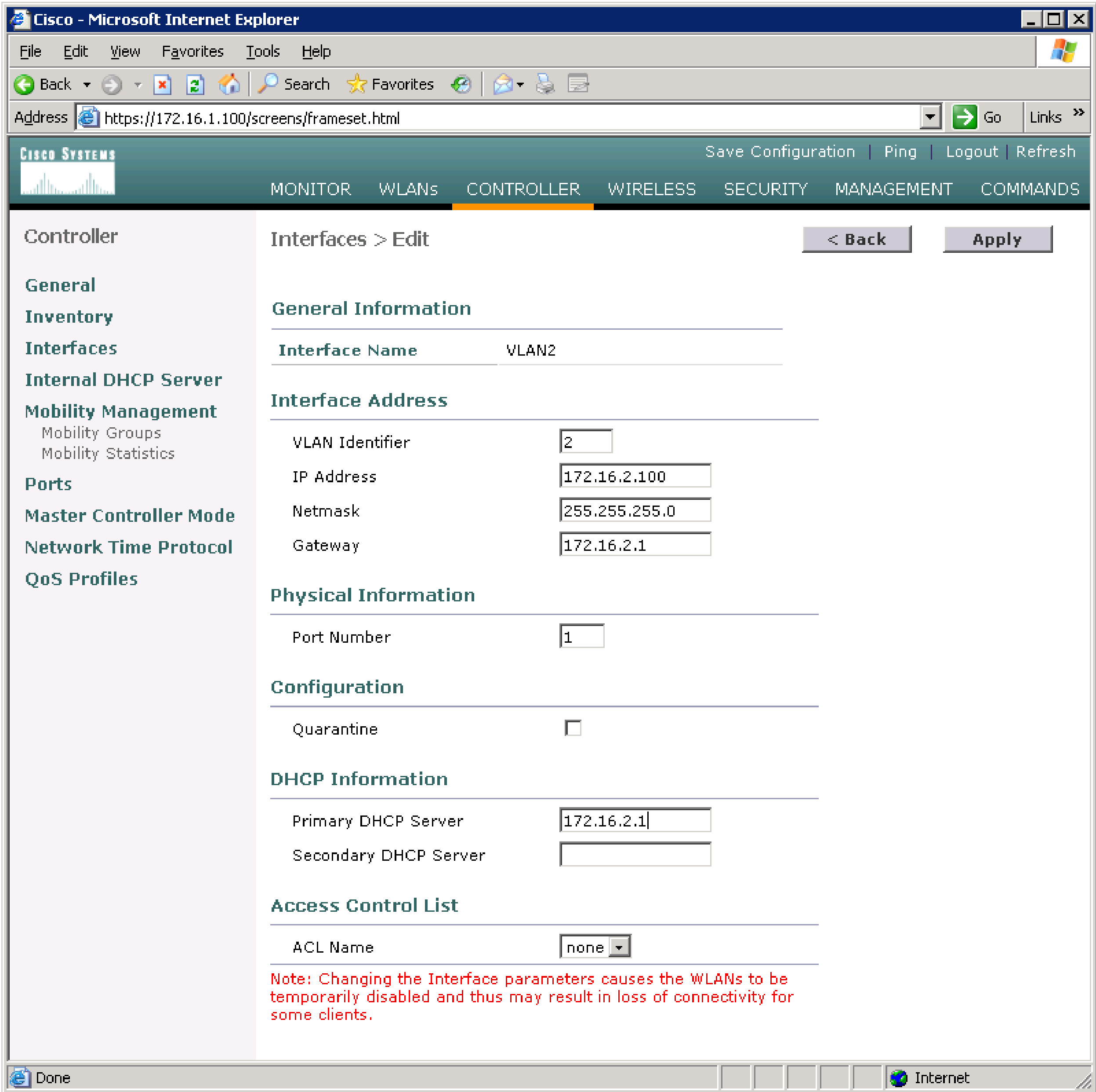


Figure 3-3: Configuring VLAN Interface Properties

The new interface should appear in the interfaces list. Do the same configuration steps for VLAN 3.

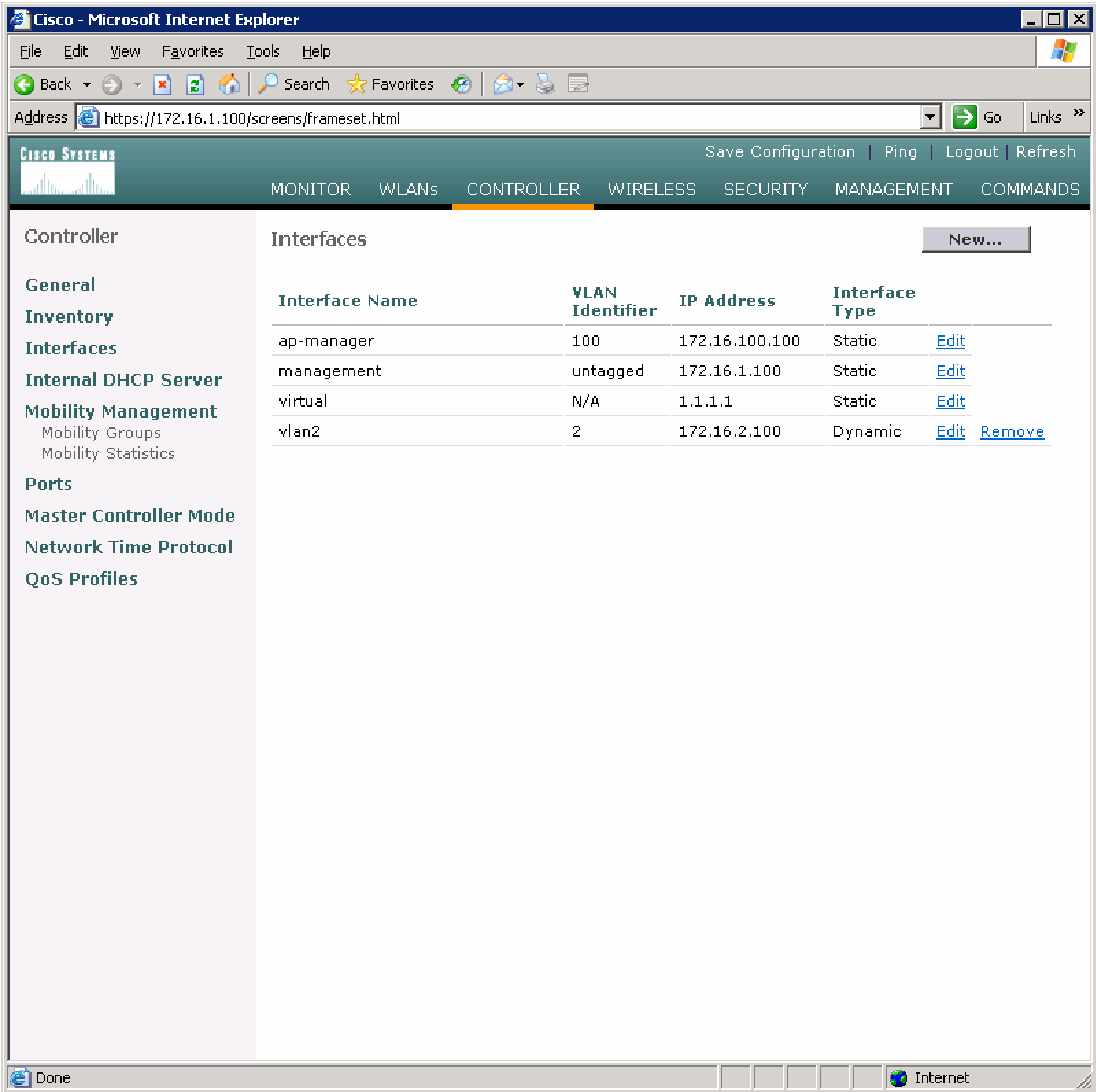


Figure 3-4: Verify Existing VLAN Interfaces

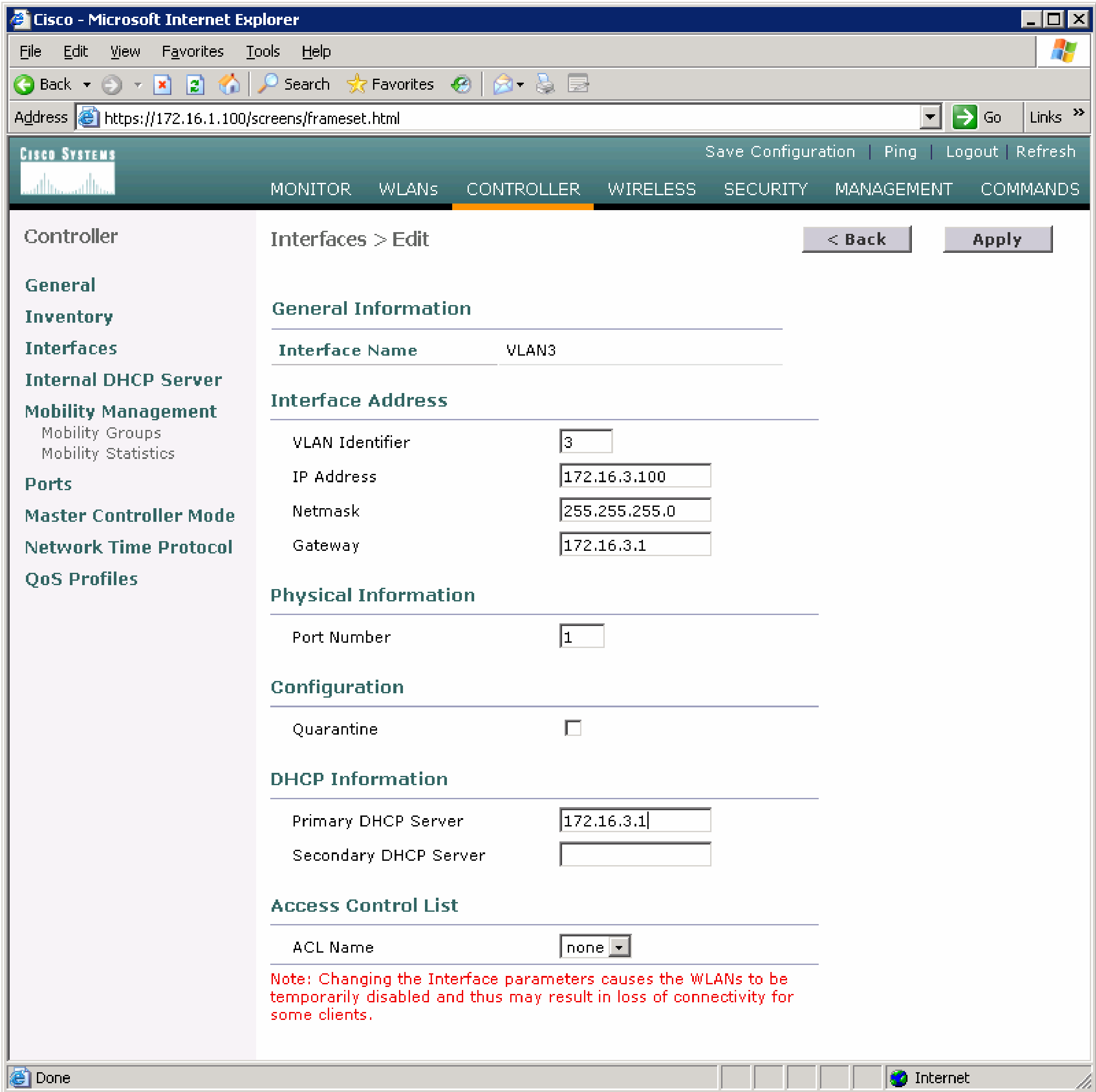


Figure 3-5: Configuring the VLAN 3 Interface

Make sure both interfaces appear in the interface table.

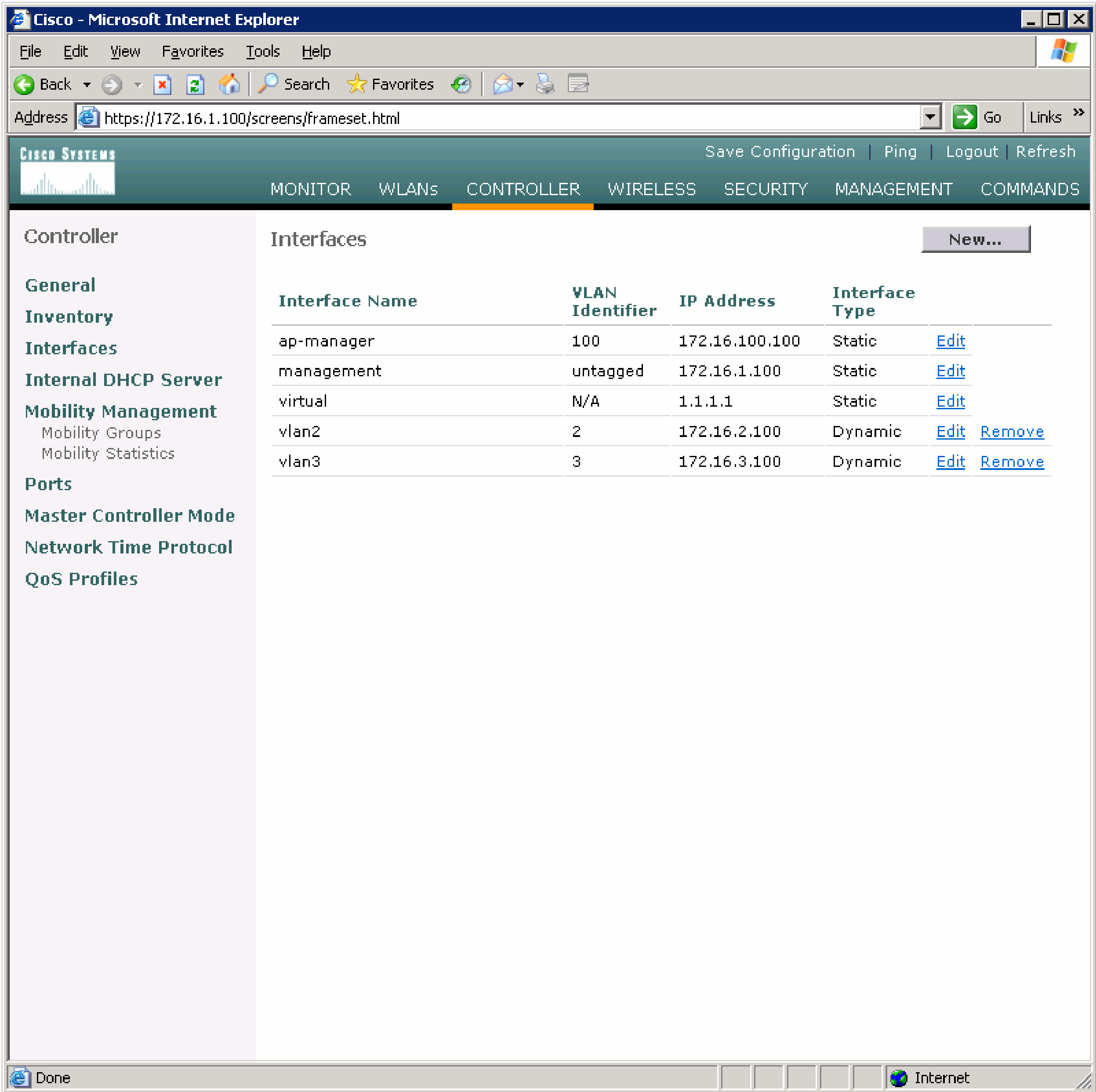


Figure 3-6: Verifying VLAN Interfaces on the WLAN Controller

Step 4

Now, you can configure the WLANs corresponding to these VLANs. To do this, first click the **WLANs** link at the top of the page. This will show you all configured WLANs.

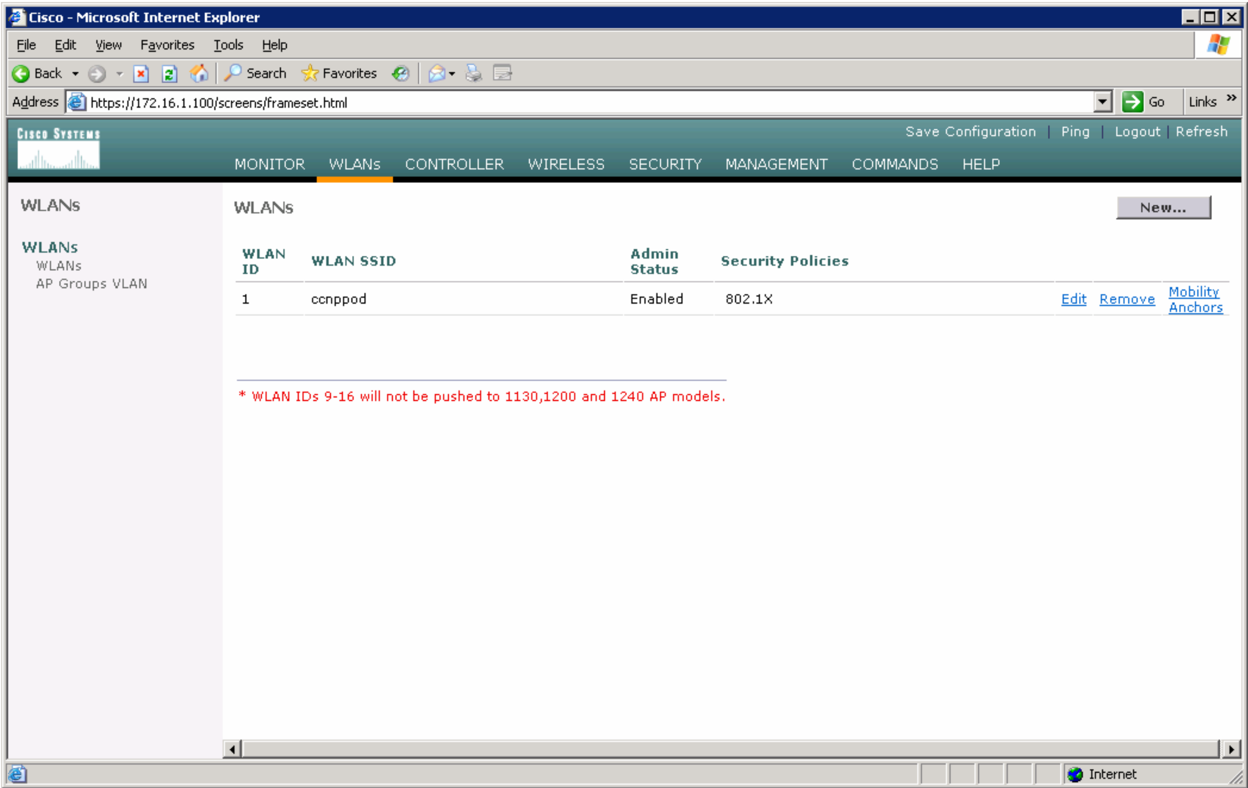


Figure 4-1: Viewing Existing WLANs

On the existing one, click **Edit** on the right of it. Remove the layer 2 security and change the interface to VLAN2. This will associate this WLAN with the correct VLAN.

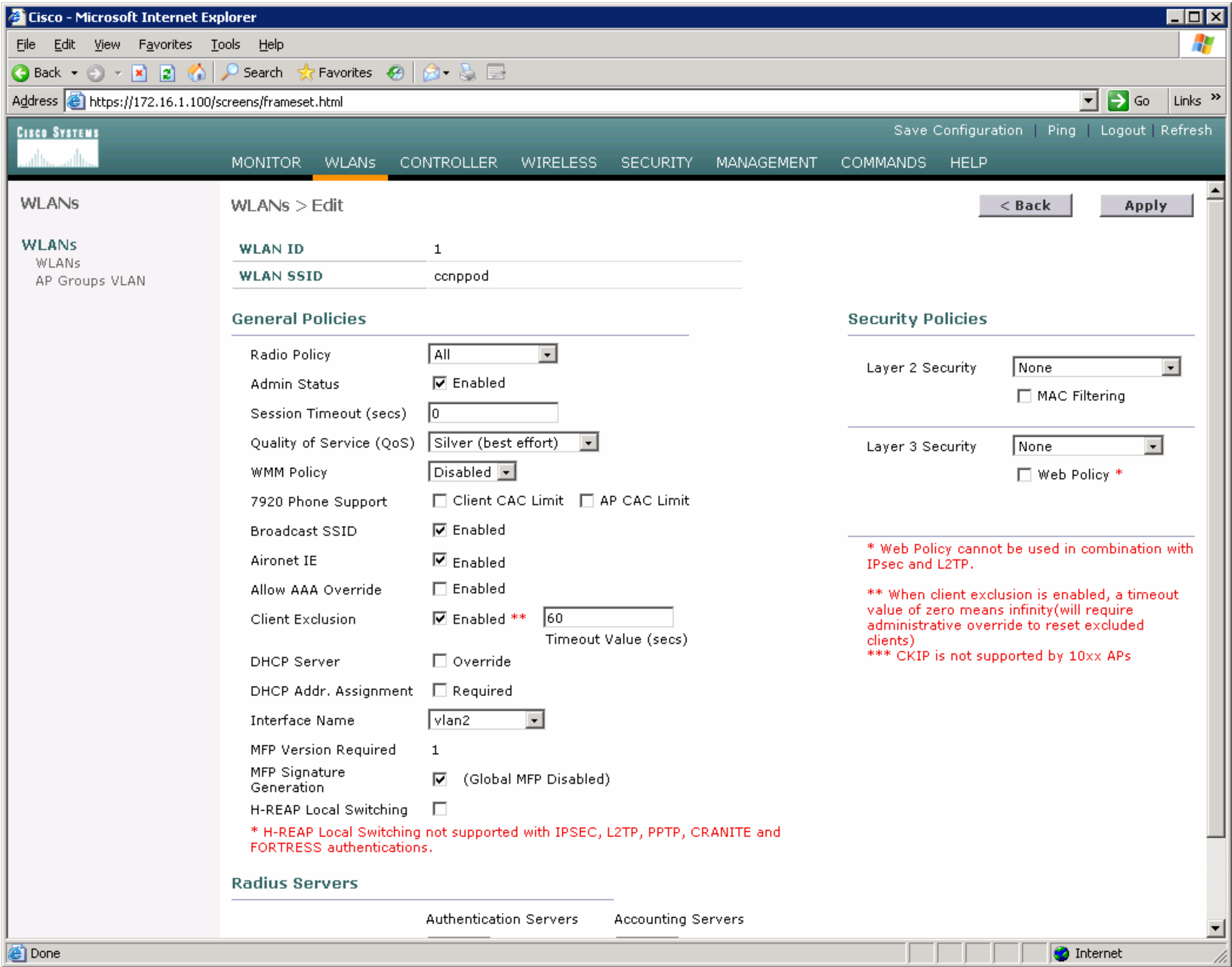


Figure 4-2: Edit the Configuration for WLAN 1

Click **Apply** and click **OK** to the warning box that comes up.

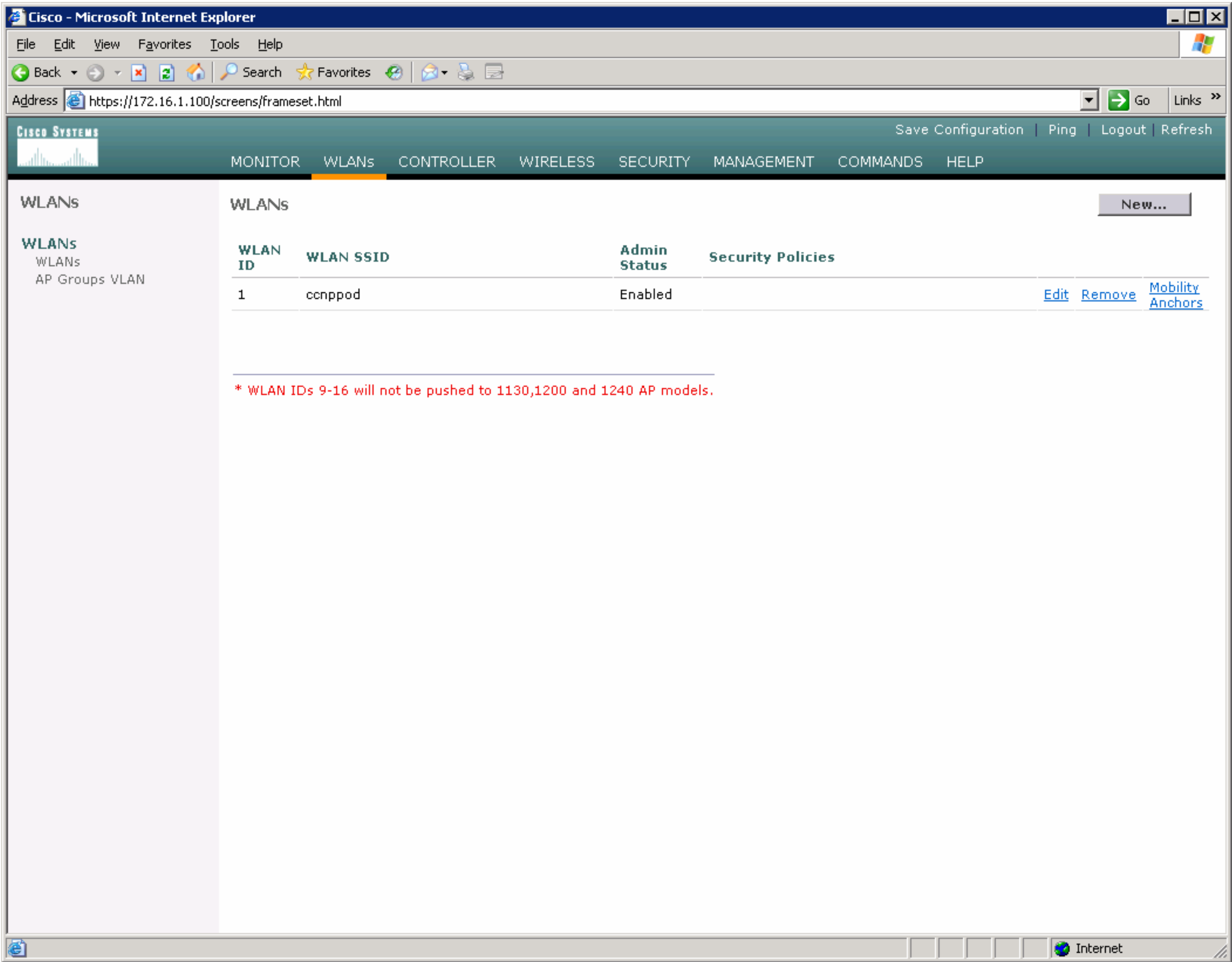


Figure 4-3: WLAN 1 without a Security Policy

Click **New...** and configure a WLAN for VLAN 3. Use the SSID “ccnplab”.

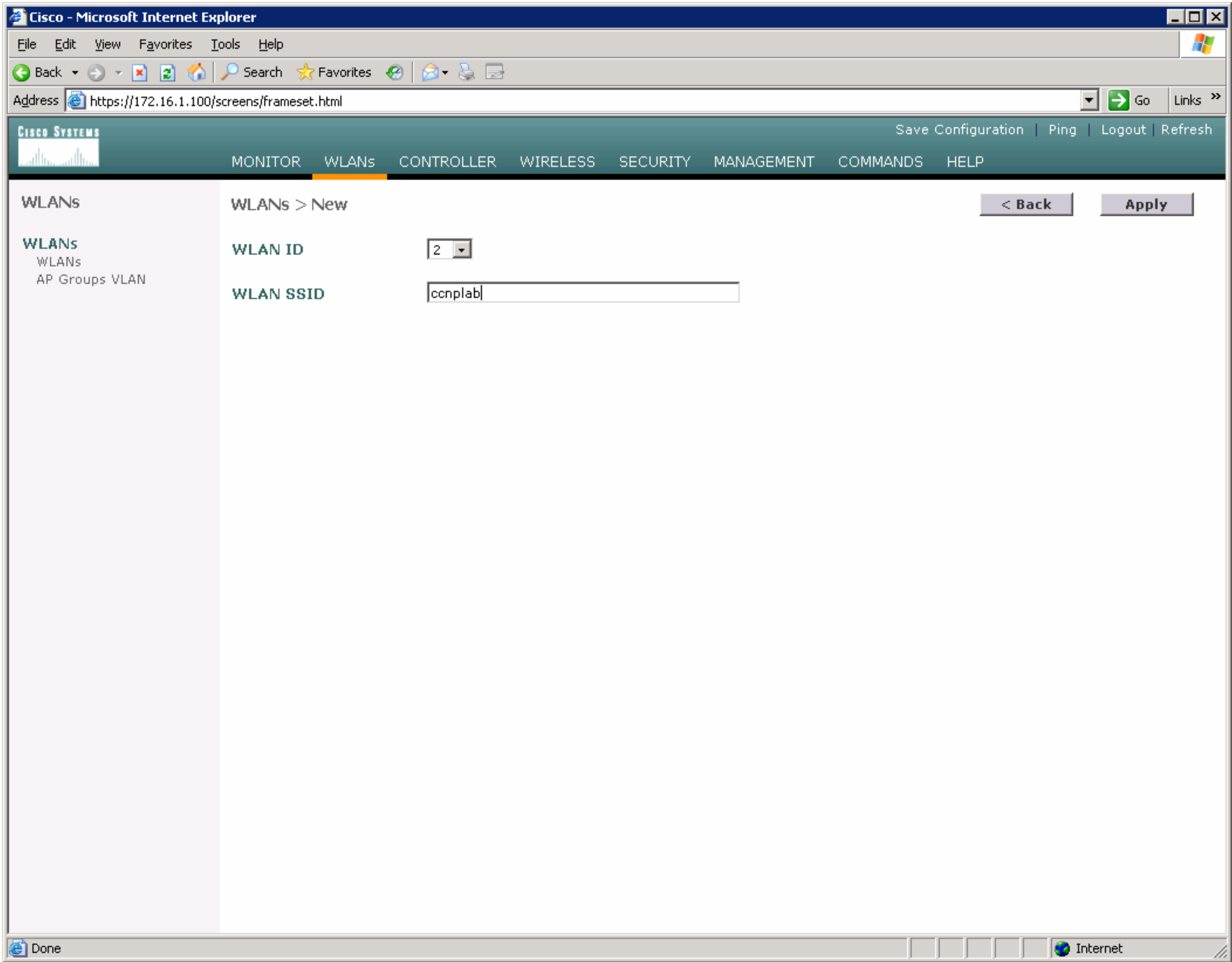


Figure 4-4: Adding a New SSID for WLAN 2

On this WLAN, configure the layer 2 security as Static WEP and use a 40 bit WEP key. Make the key index 2 and use a key of “cisco”. Also, set the administrative status of the WLAN to enabled and change the interface name to VLAN3. When you are done, click **Apply** and you should see both WLANs in the WLAN list.

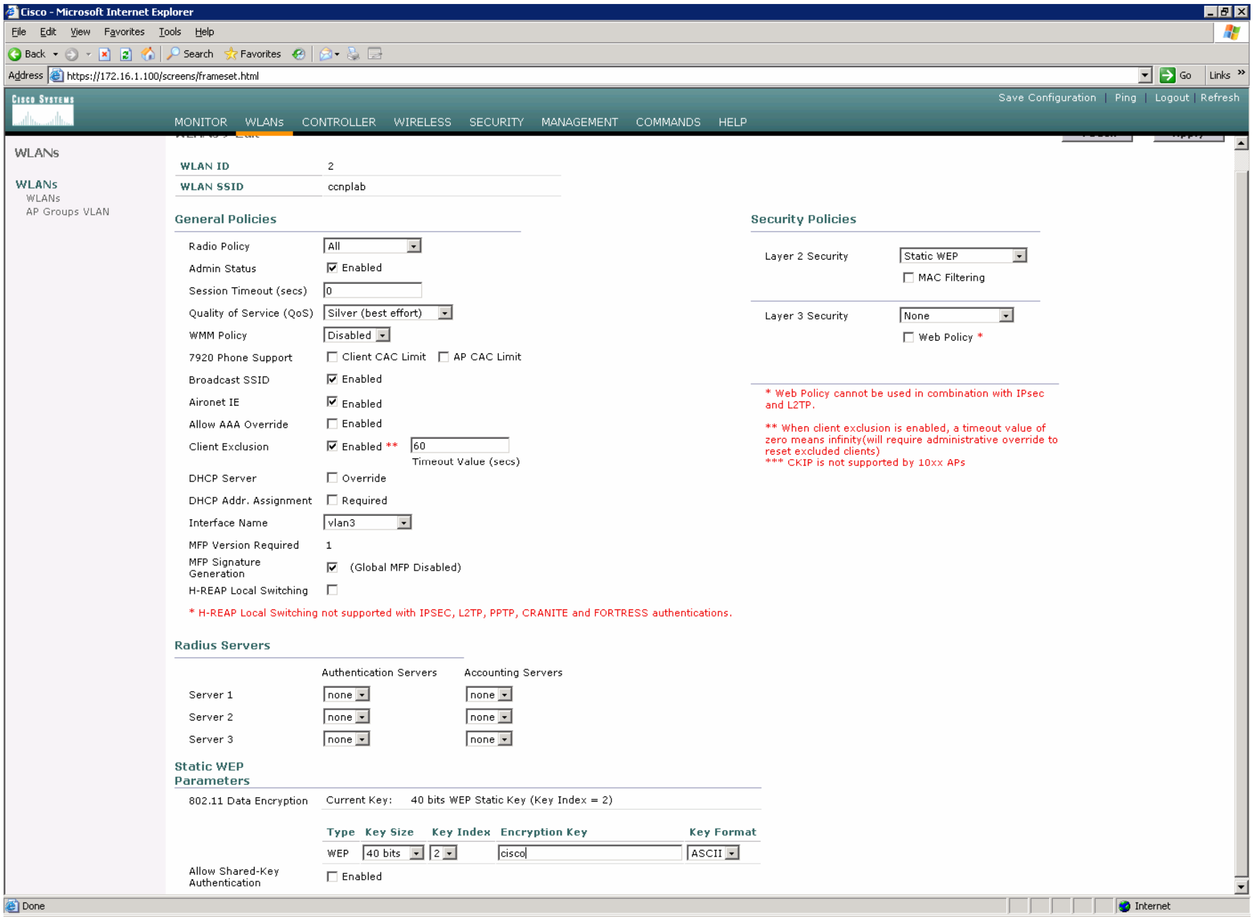


Figure 4-5: Configuring VLAN Association and Authentication for VLAN 3

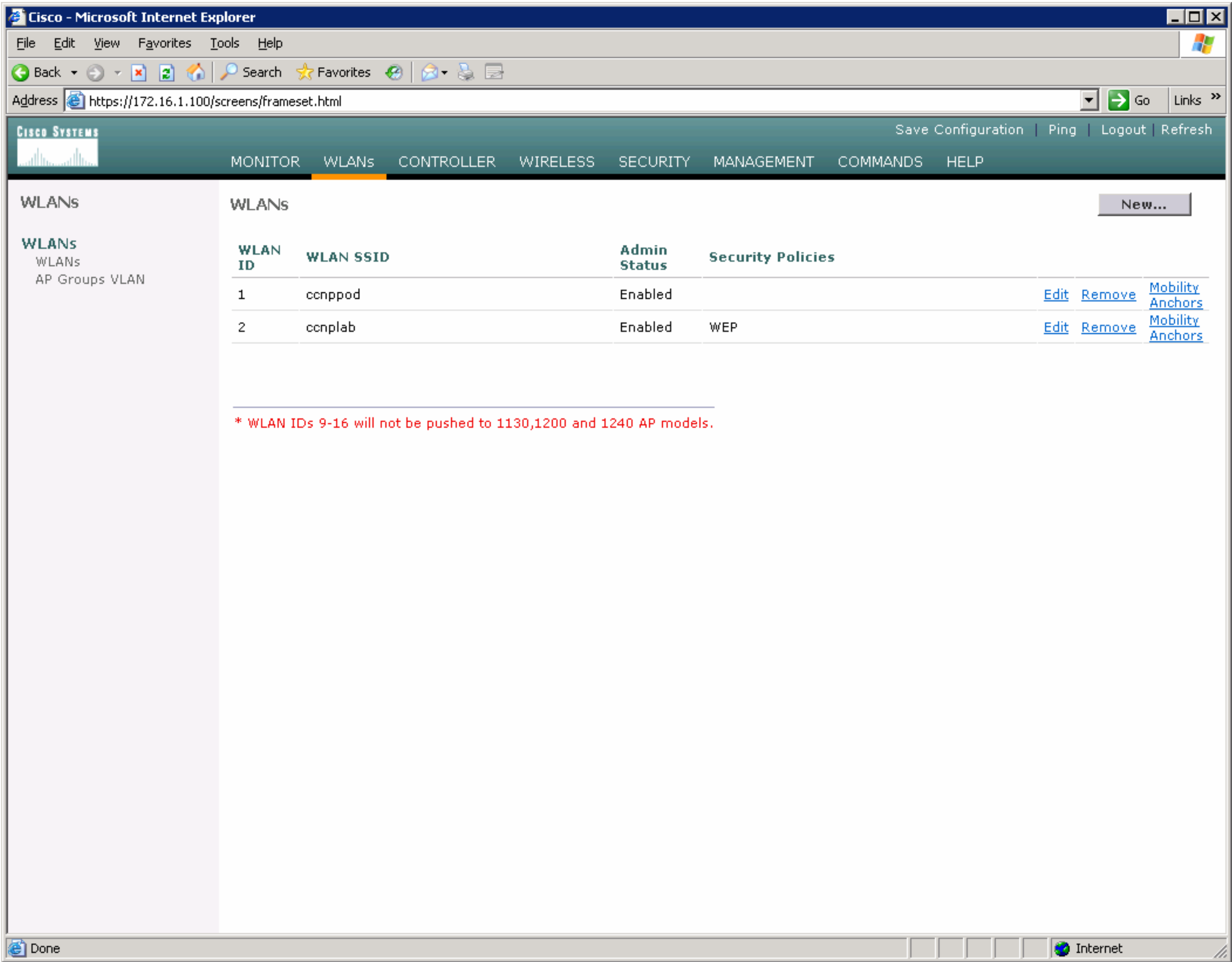
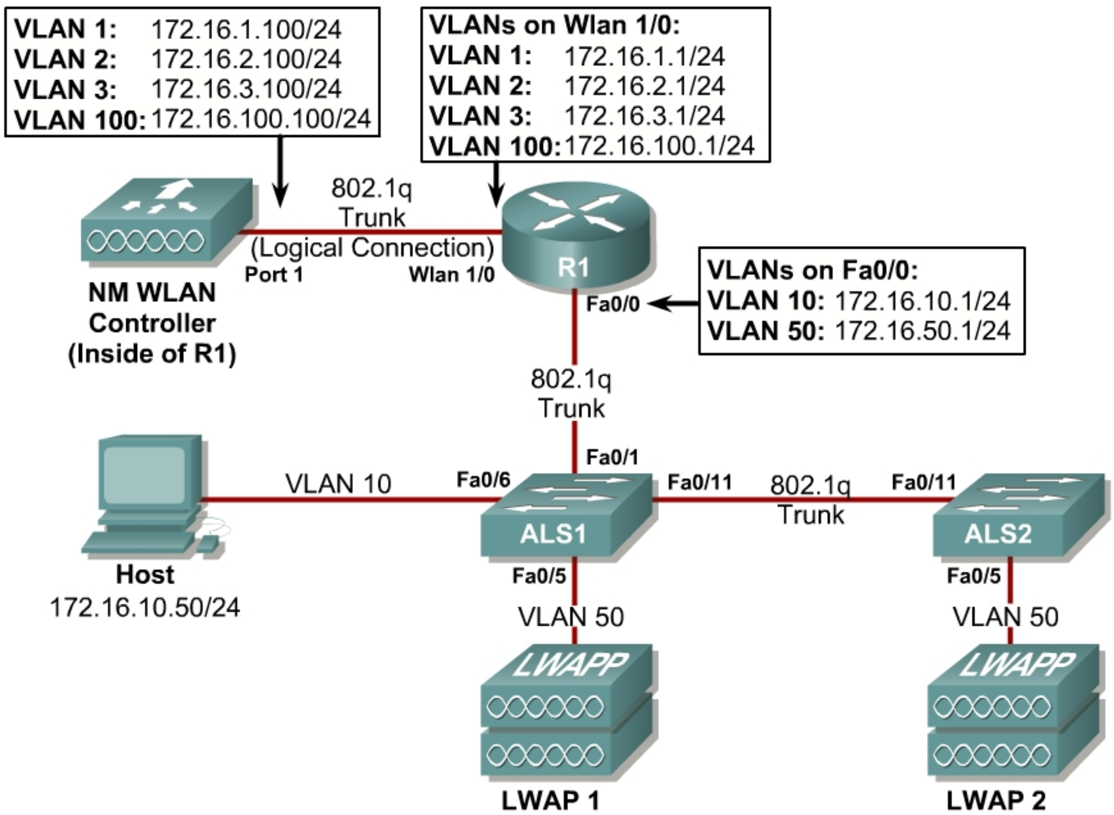


Figure 4-6: Verifying Final WLAN Configuration

At this point, if you have a computer with a wireless card installed you should be able to see both SSIDs and connect to the WLANs/VLANs associated with them. Notice that each WLAN exists in a separate subnet, because each WLAN is in a separate VLAN.

Lab 6.2b Configuring a WLAN Controller via the Web Interface

Topology Diagram



Scenario

Continuing from the previous lab, you will now set up the WLAN controller through its web interface. Previously you configured it through the CLI.

Step 1

Set up all the switches as they were in the previous lab. Make sure that the WLAN controller and host also have the same configuration as before.

Step 2

On the host, open up Internet Explorer and go to the URL "https://172.16.1.100". This is the secure method of connecting to the management interface of the WLAN controller. You can also use "http://172.16.1.100" since we previously enabled regular insecure HTTP access in the CLI for Lab 6.1. If you connect to the secure address, you may be prompted with a security warning. Click **Yes** to accept it and you will be presented with the login screen for the WLAN controller. Click **Login** and an authentication dialog box will appear.

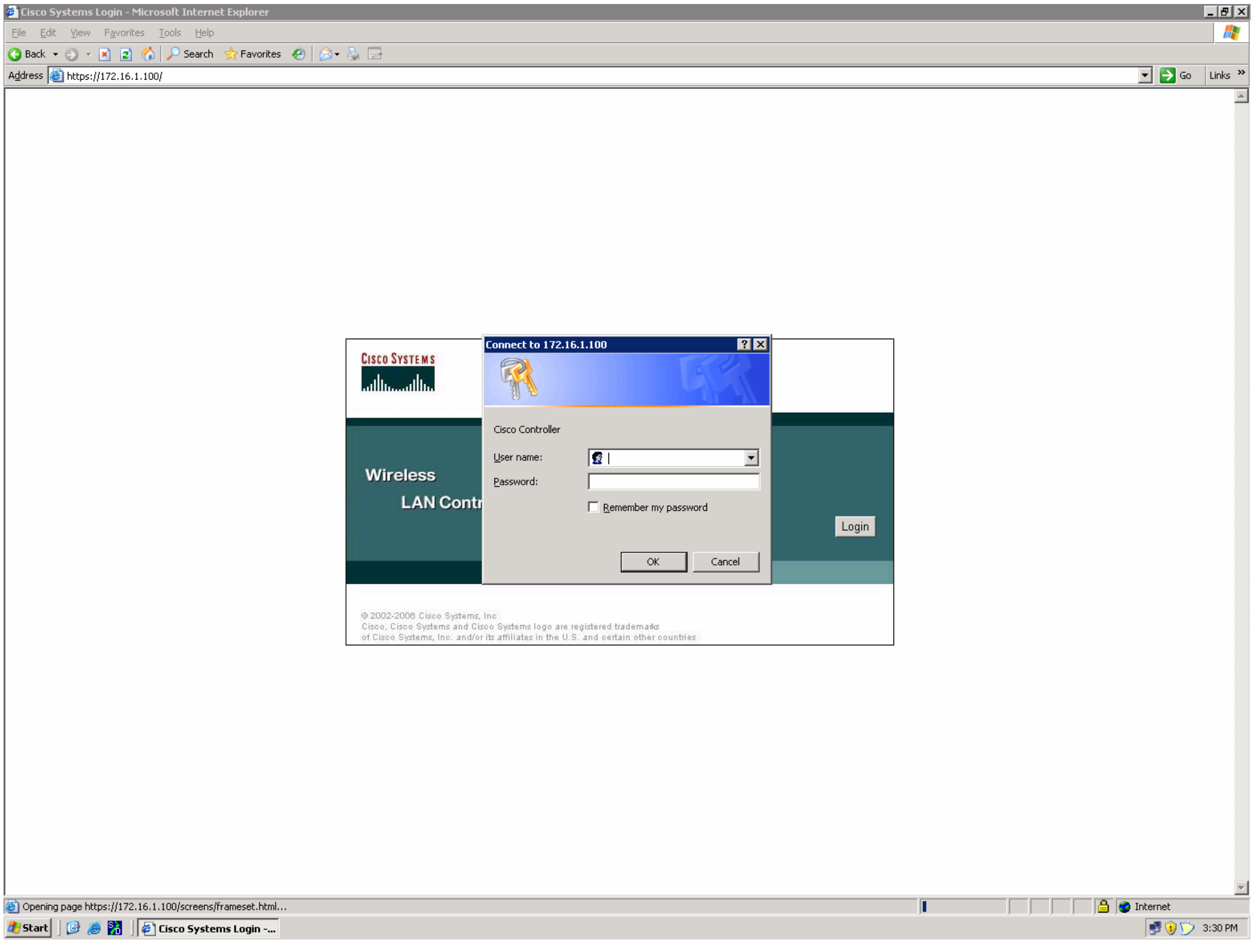


Figure 2-1: Authentication Dialog Box for WLAN Controller Web Access

Use "cisco" as both the username and password. You configured these in the previous lab. Click **OK** to get to the main page of the graphical user interface (GUI). You are then presented with the monitor page for the WLAN controller.

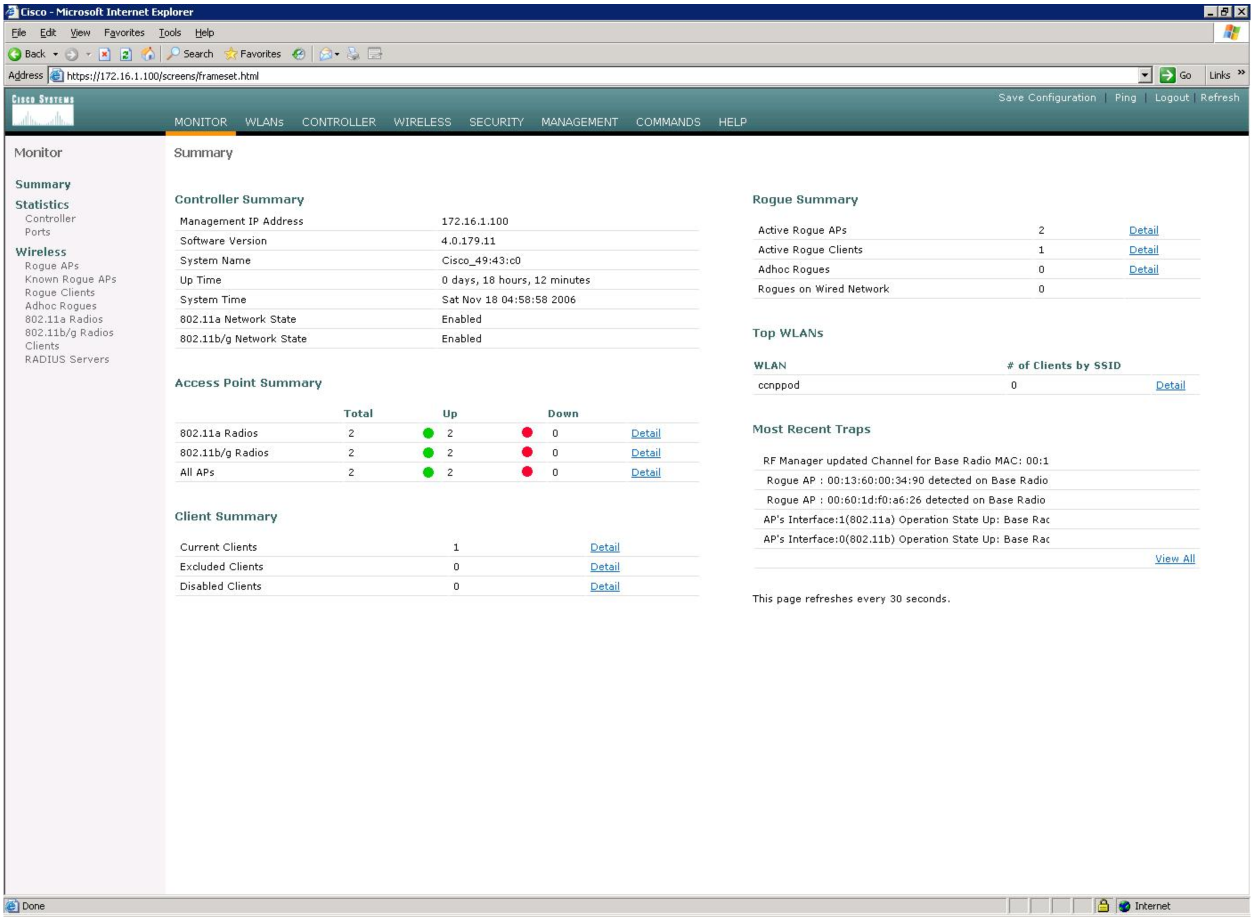


Figure 2-2: WLAN Controller Monitor Page

Make sure you see 2 access points under the “Access Point Summary” part of the page. You may also see it detecting rogue access points if your lab has other wireless networks around it; this behavior is normal. You can also see various port controller and port statistics by clicking their respective links on the left-hand menu on the screen.

Step 3

The next task in configuring WLANs is to add in the logical interfaces on the WLAN controller corresponding to VLANs 2 and 3. To do this, click the **Controller** link on the top of the web interface. Then, click **Interfaces** link on the left side bar.

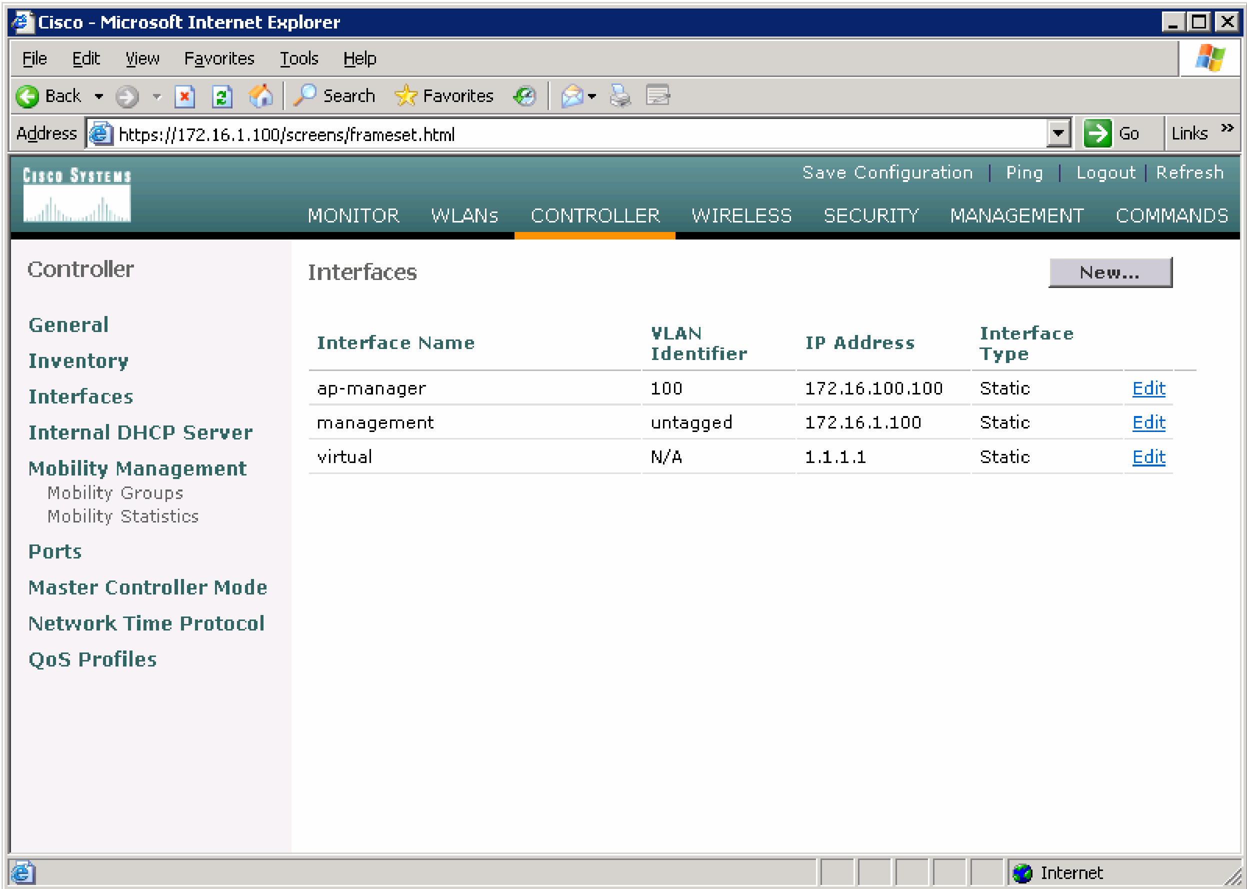


Figure 3-1: Interface Configuration Page

Click the **New...** link to create a new interface. Give the new interface a name of VLAN2 and VLAN number 2. Click **Apply** to submit the parameters.

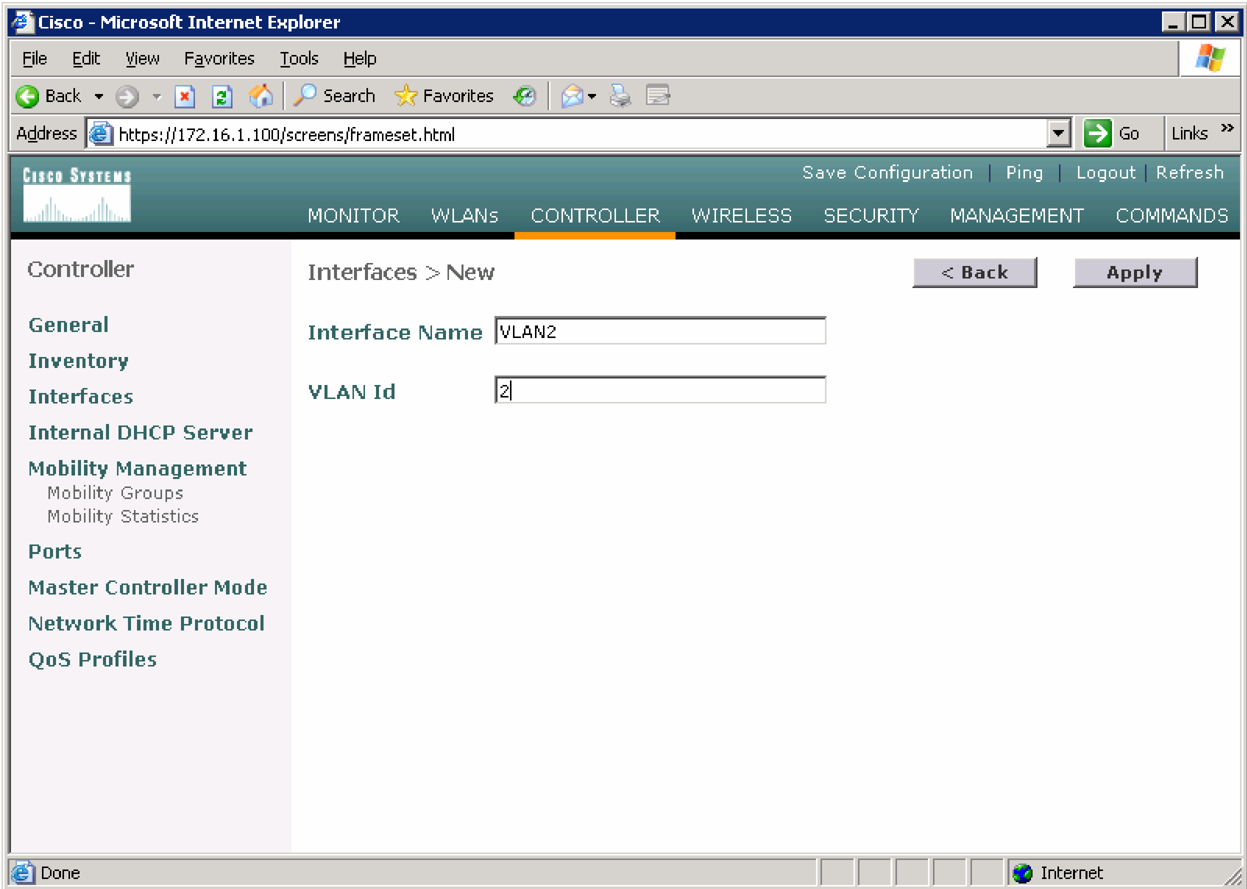


Figure 3-2: Creating a New VLAN Interface

On the next page, configure the IP address shown in the diagram. Also configure this on physical port 1, since that is the port trunked to the switch. After you have entered in all the changes, click **Apply**. Click **OK** to the warning box that comes up. This warning says that there may be a temporary connectivity loss on the APs while changes are applied.

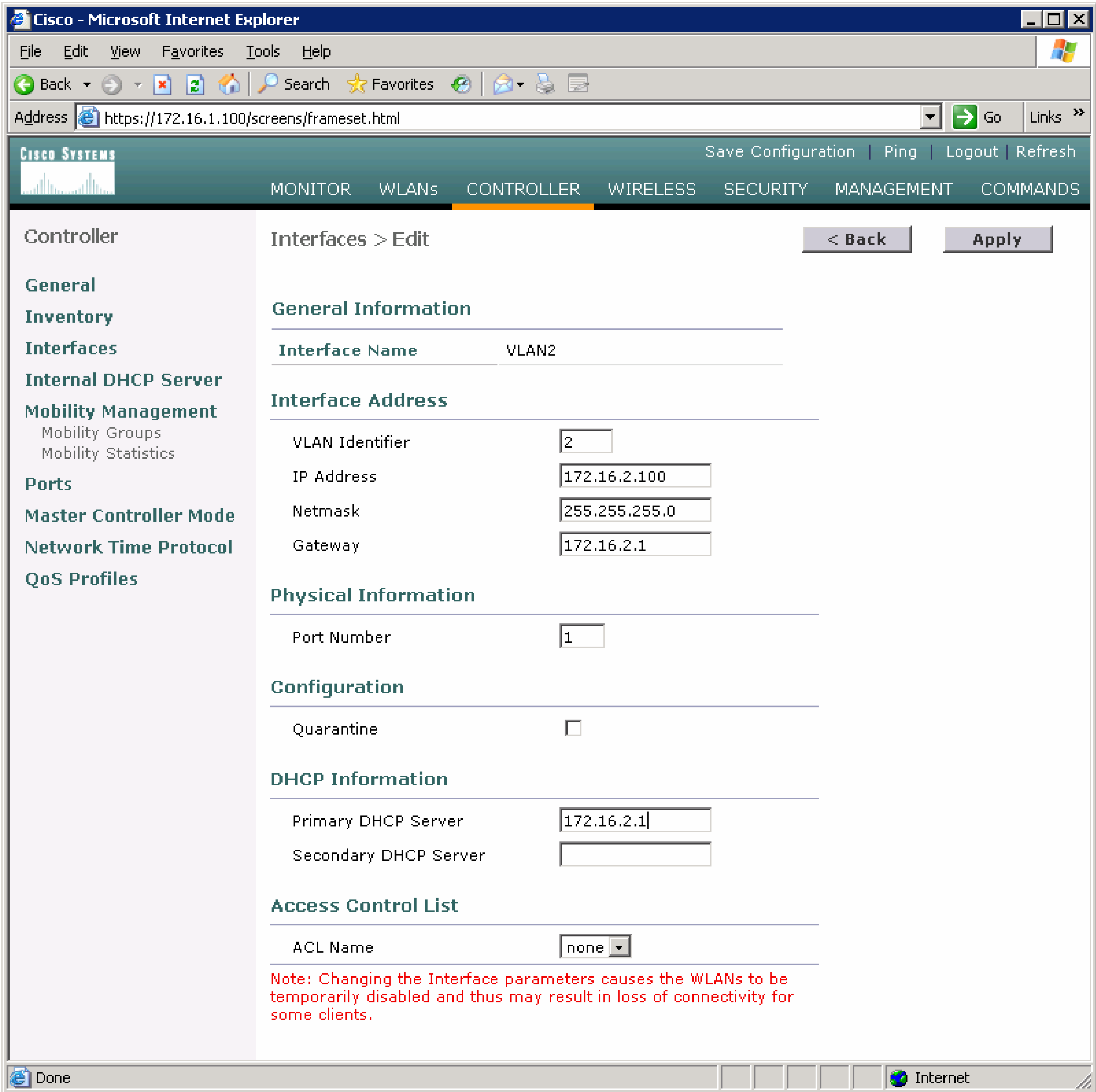


Figure 3-3: Configuring VLAN Interface Properties

The new interface should appear in the interfaces list. Do the same configuration steps for VLAN 3.

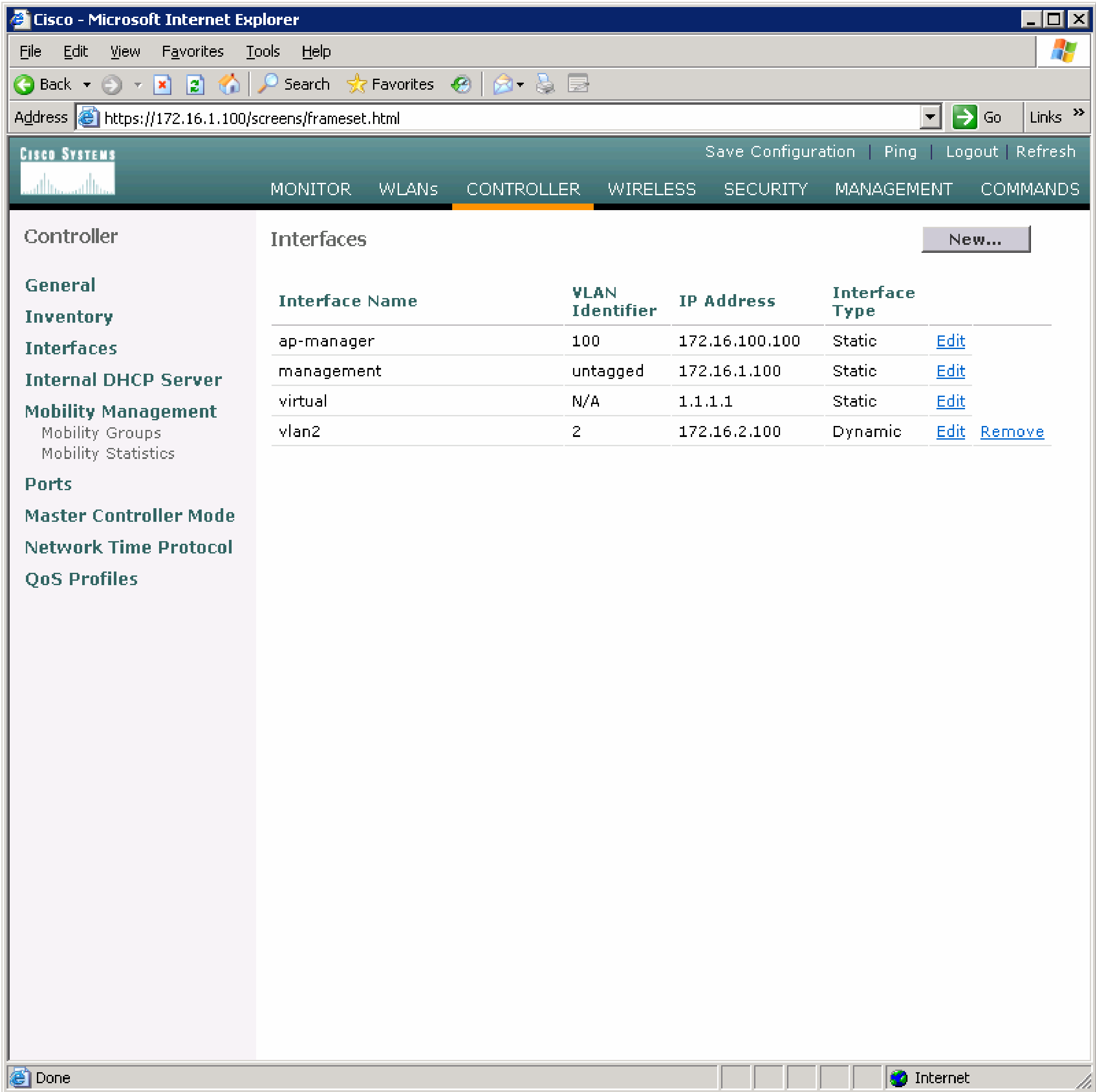


Figure 3-4: Verify Existing VLAN Interfaces

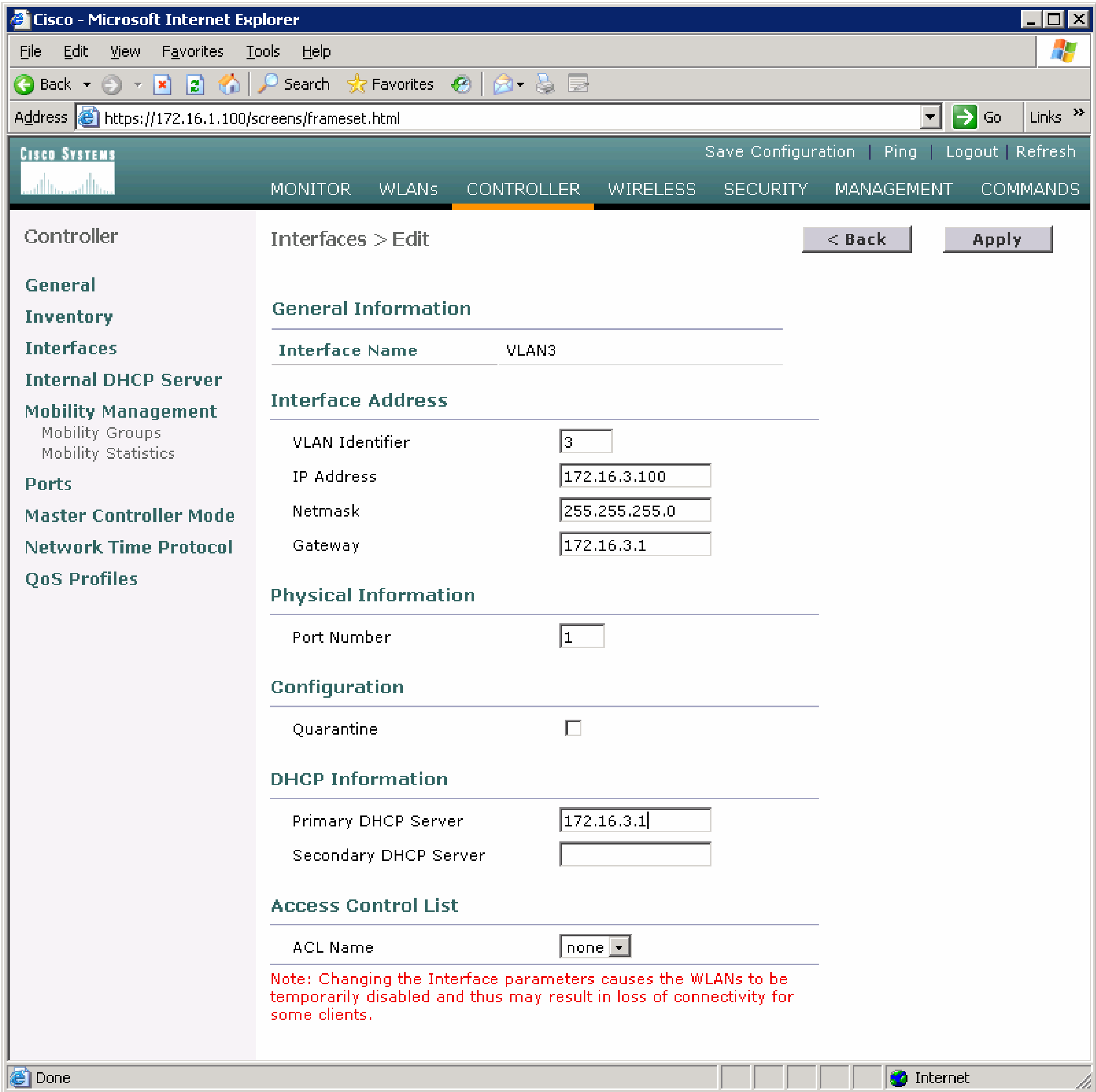


Figure 3-5: Configuring the VLAN 3 Interface

Make sure both interfaces appear in the interface table.

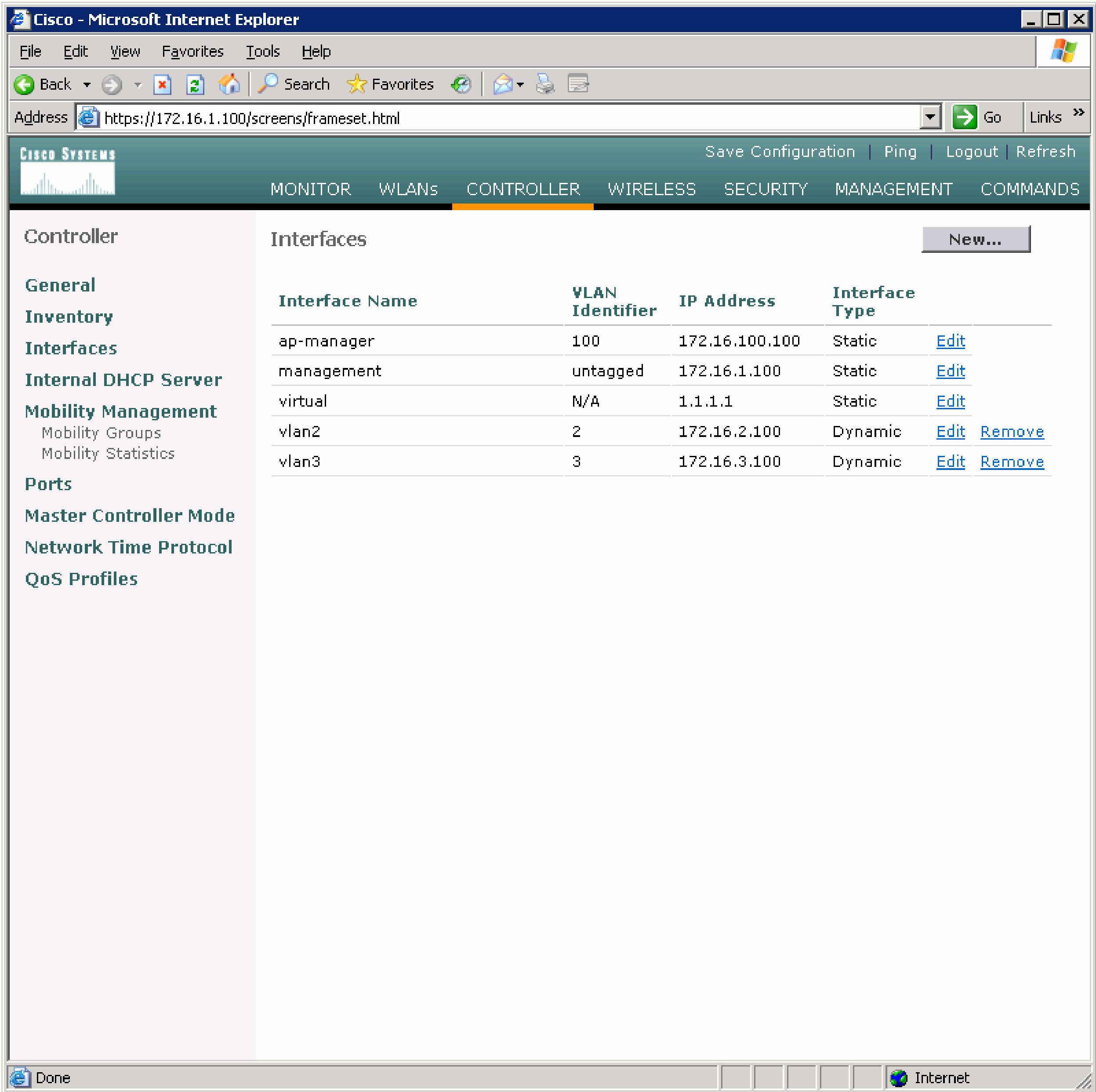


Figure 3-6: Verifying VLAN Interfaces on the WLAN Controller

Step 4

Now, you can configure the WLANs corresponding to these VLANs. To do this, first click the **WLANs** link at the top of the page. This will show you all configured WLANs.

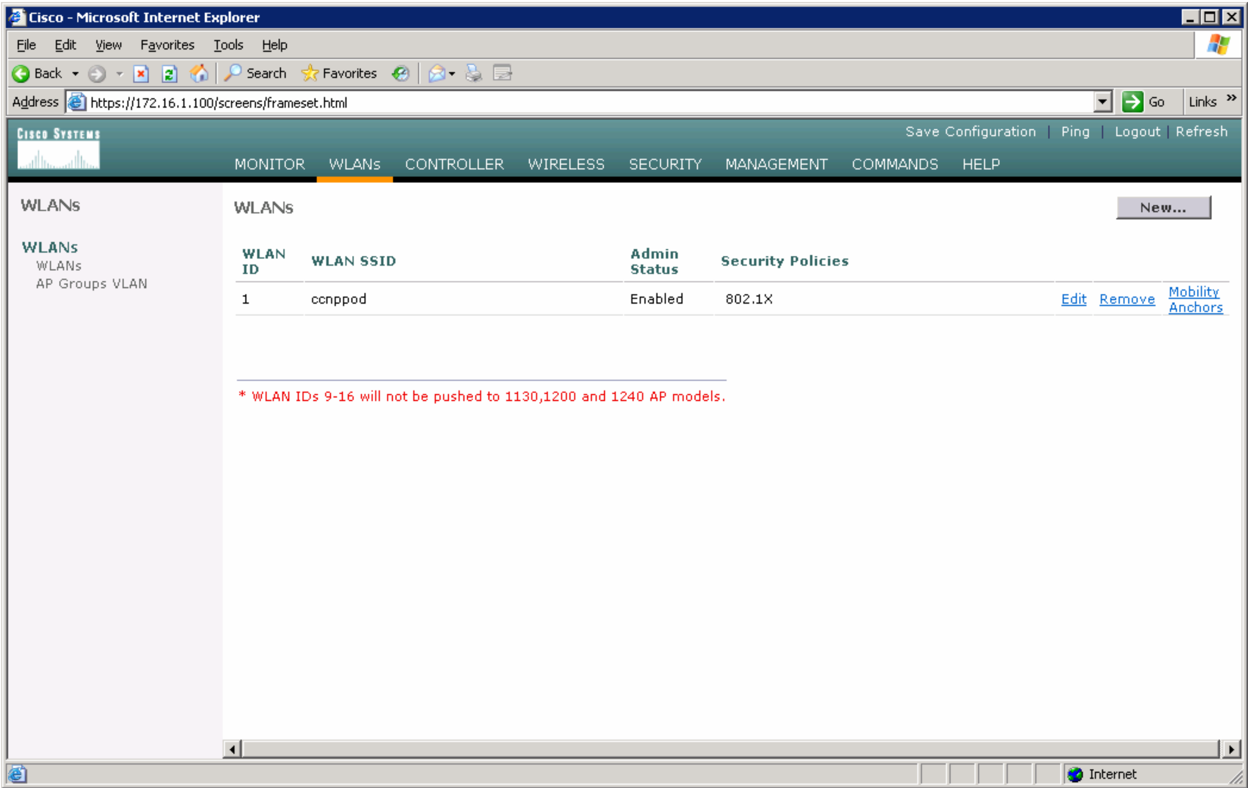


Figure 4-1: Viewing Existing WLANs

On the existing one, click **Edit** on the right of it. Remove the layer 2 security and change the interface to VLAN2. This will associate this WLAN with the correct VLAN.

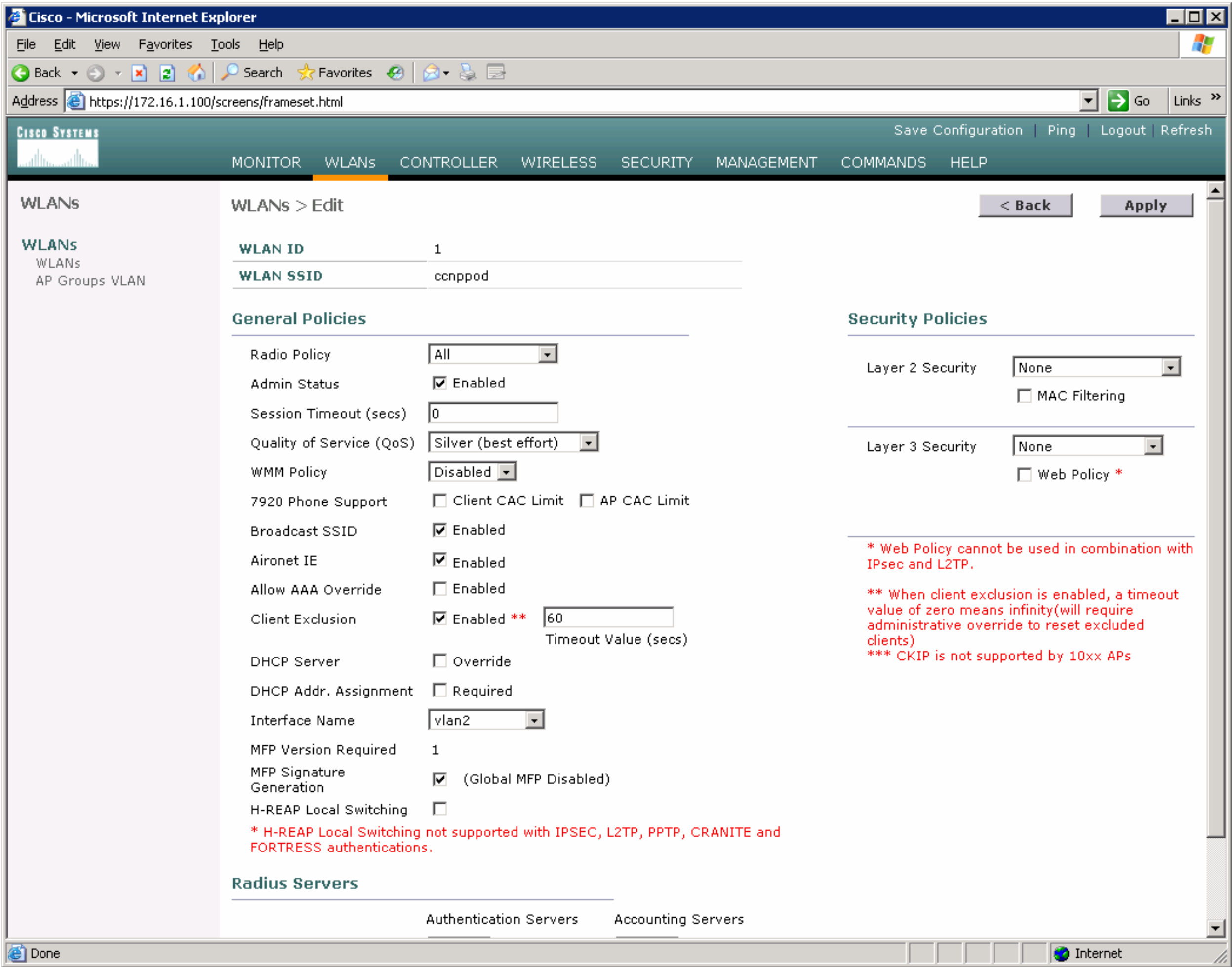


Figure 4-2: Edit the Configuration for WLAN 1

Click **Apply** and click **OK** to the warning box that comes up.

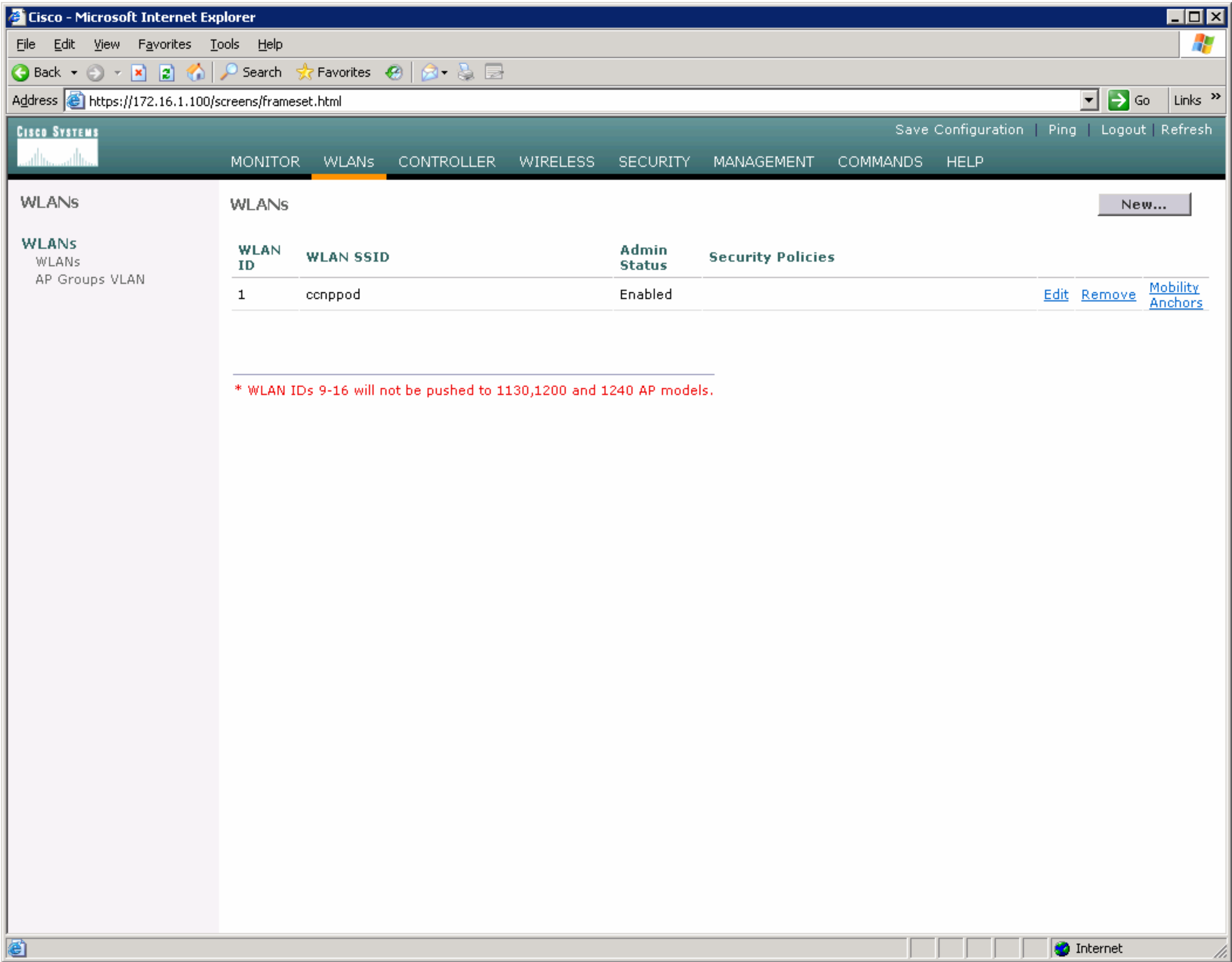


Figure 4-3: WLAN 1 without a Security Policy

Click **New...** and configure a WLAN for VLAN 3. Use the SSID “ccnplab”.

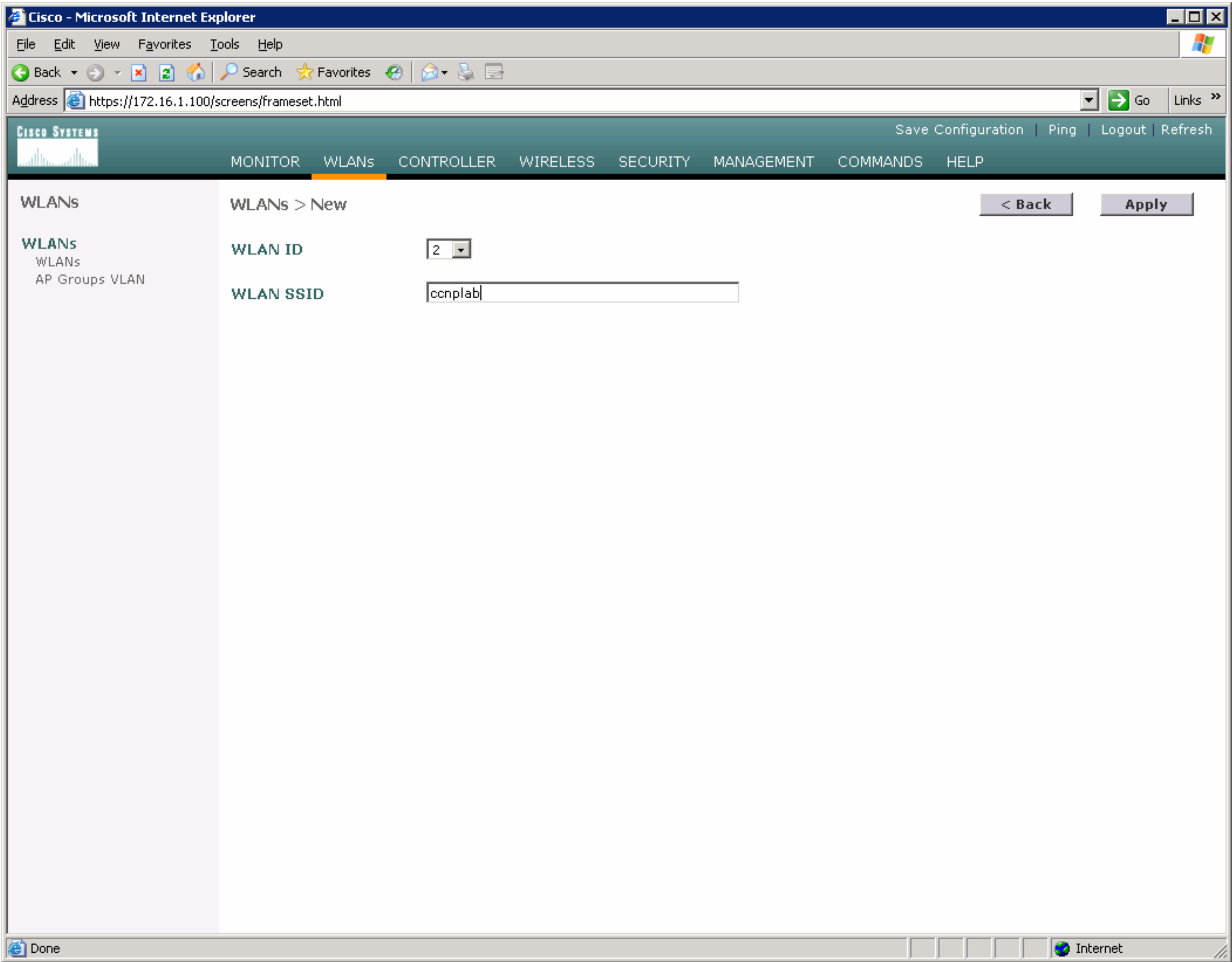


Figure 4-4: Adding a New SSID for WLAN 2

On this WLAN, configure the layer 2 security as Static WEP and use a 40 bit WEP key. Make the key index 2 and use a key of “cisco”. Also, set the administrative status of the WLAN to enabled and change the interface name to VLAN3. When you are done, click **Apply** and you should see both WLANs in the WLAN list.

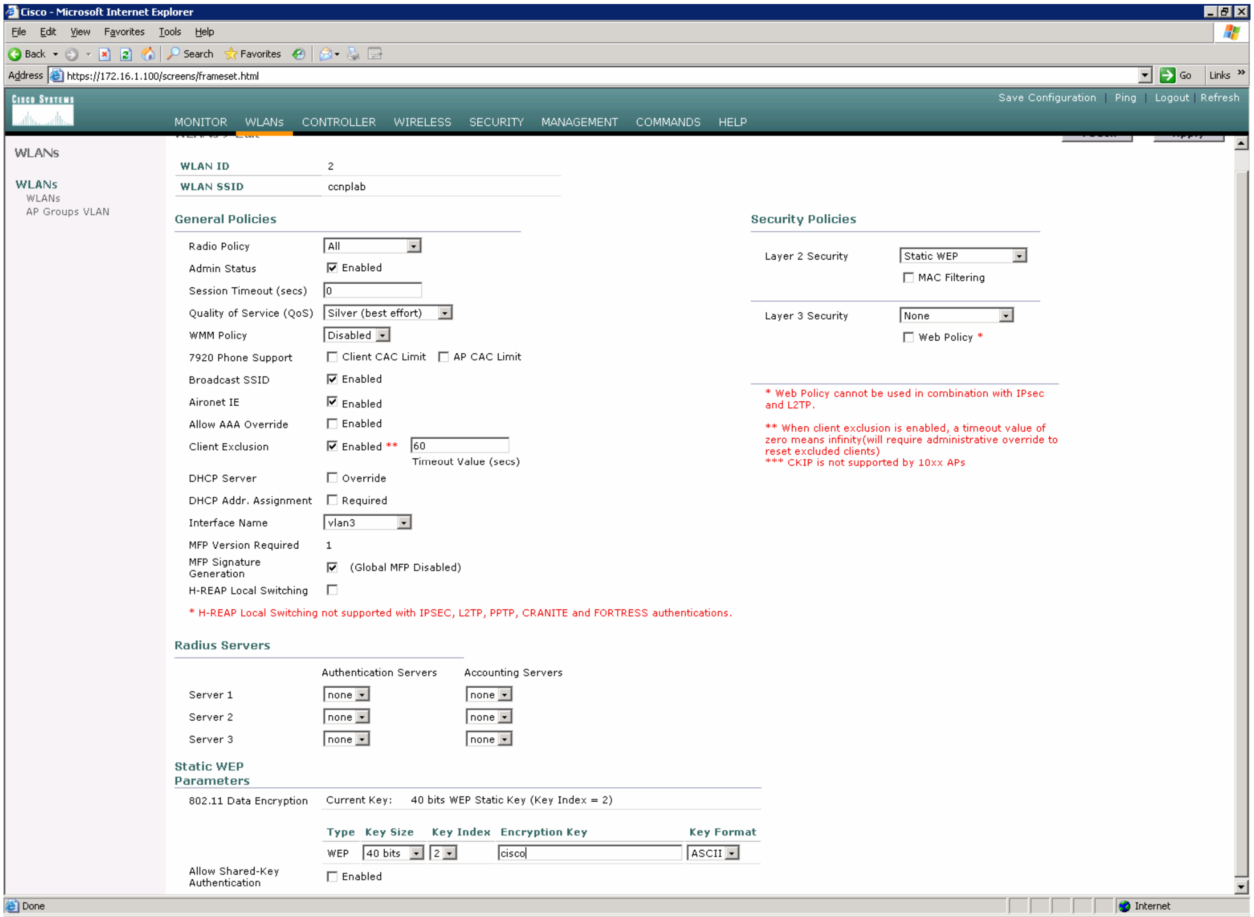


Figure 4-5: Configuring VLAN Association and Authentication for VLAN 3

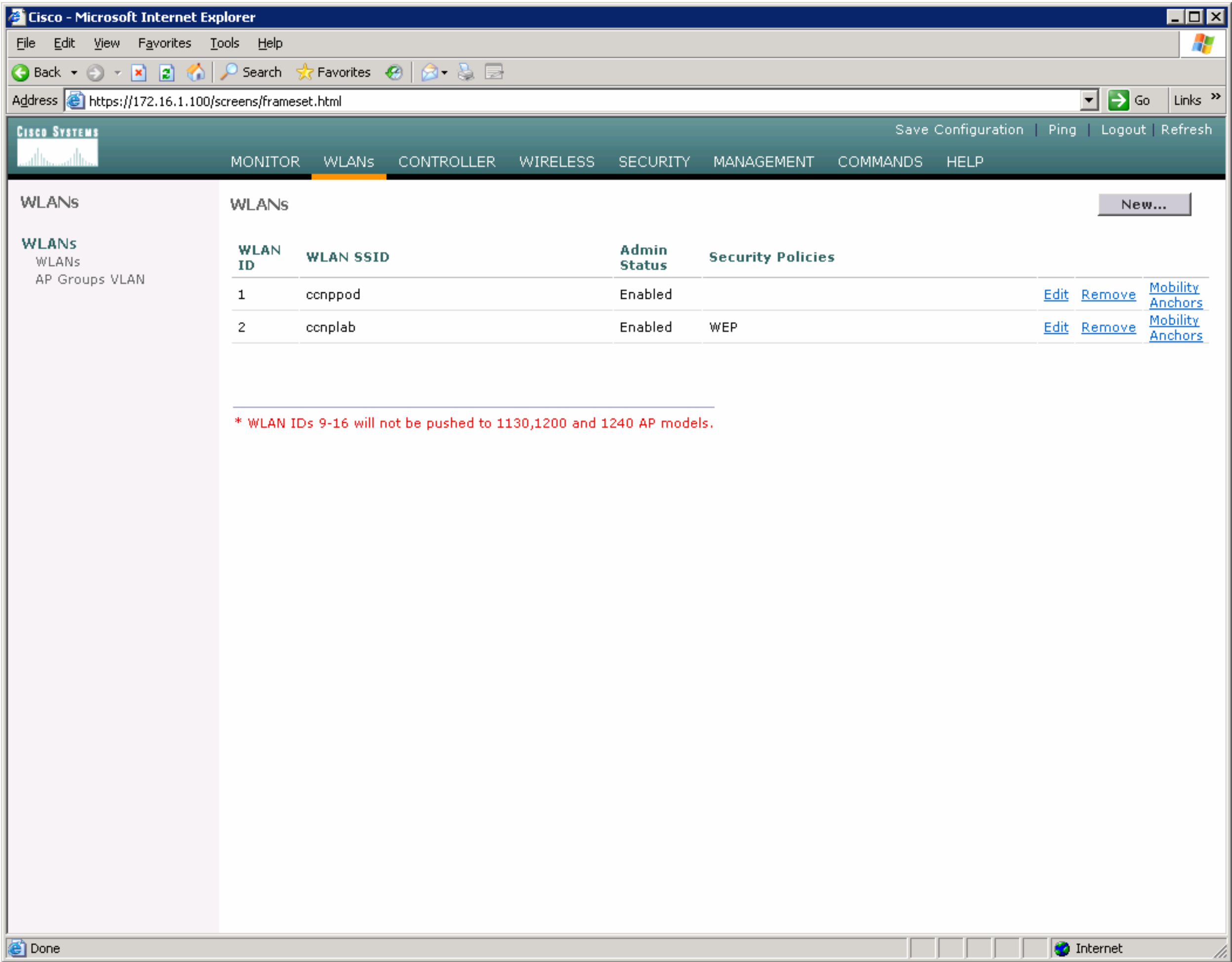
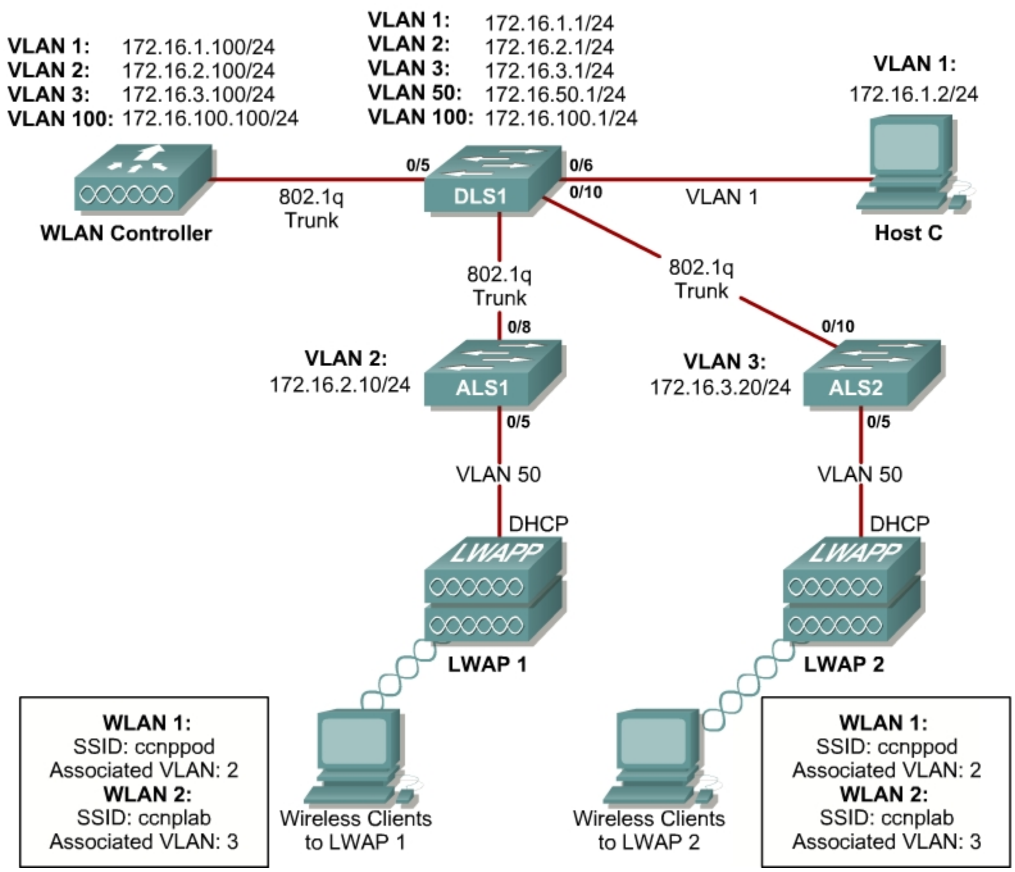


Figure 4-6: Verifying Final WLAN Configuration

At this point, if you have a computer with a wireless card installed you should be able to see both SSIDs and connect to the WLANs/VLANs associated with them. Notice that each WLAN exists in a separate subnet, because each WLAN is in a separate VLAN.

Lab 6.3 Configuring a Wireless Client

Topology Diagram



Scenario

In this lab, you will install a Cisco Aironet wireless PC card on a laptop. Then you will also configure the Cisco Aironet Desktop Utility (ADU) to connect to an access point.

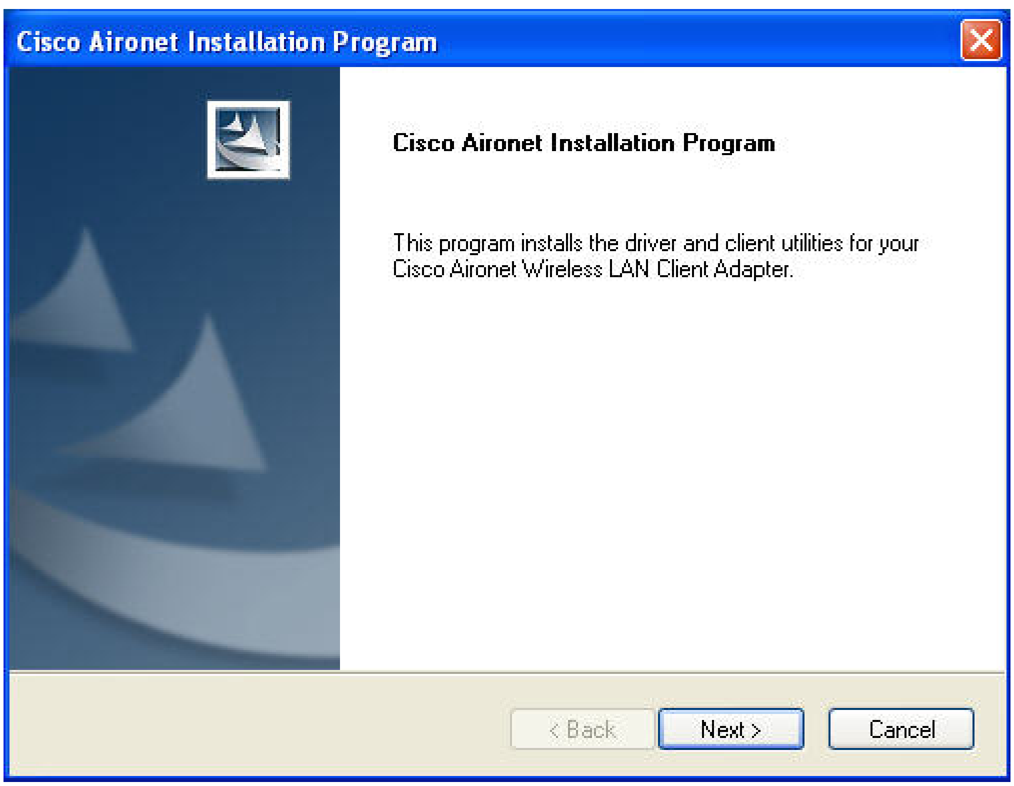
Step 1

Place the Cisco Aironet 802.11 a/b/g Wireless Adapter into an open NIC slot on your laptop.

Step 2

WinClient-802.11a-b-g-Ins-Wizard-v30

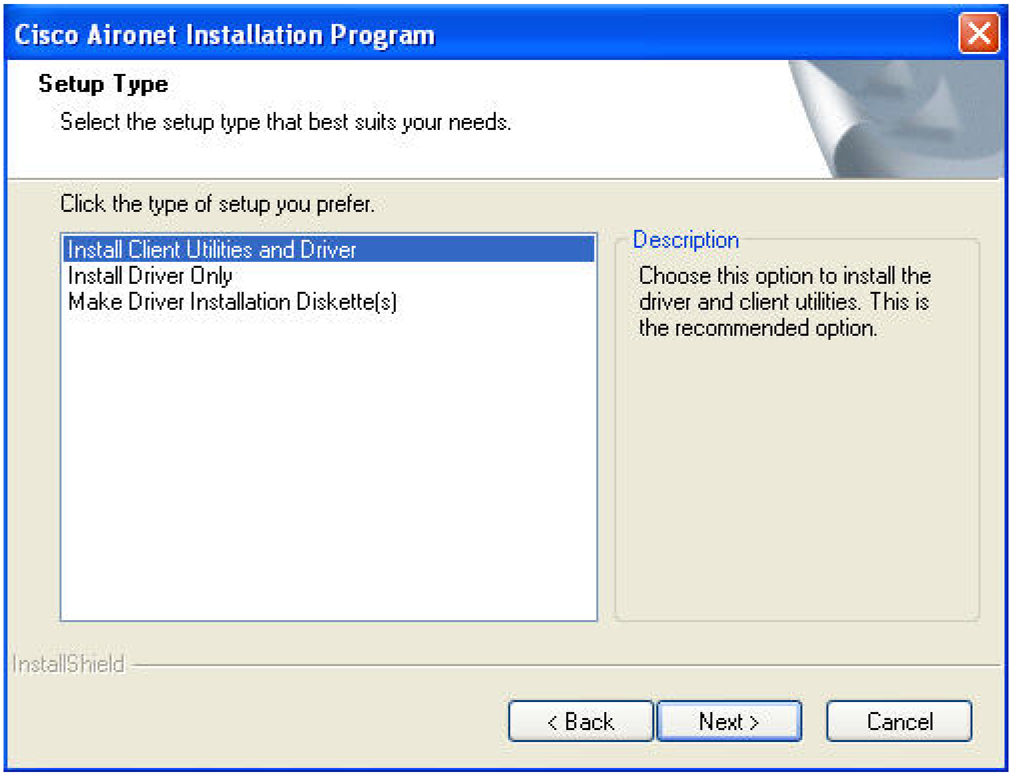
Once you have transferred the Installation Wizard software to your hard drive, double-click on it. The following is the first screen to appear.



First Page of the Cisco Aironet Installation Wizard

Step 3

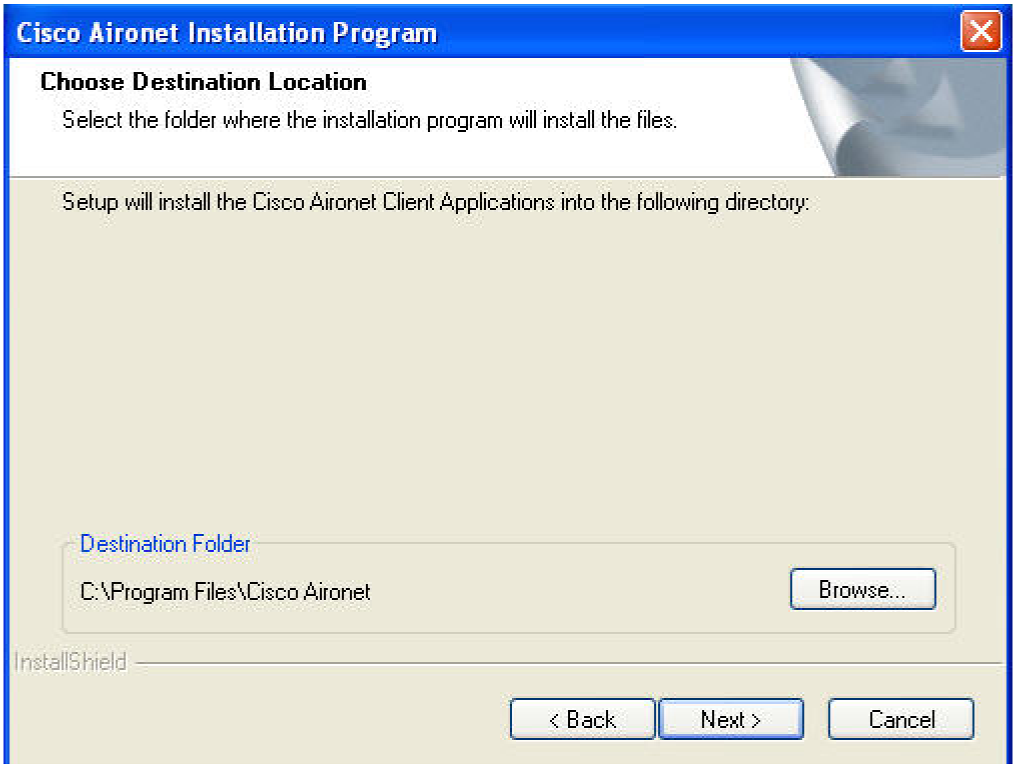
Click on Next. Then select **Install Client Utilities and Driver**. Click on **Next.**



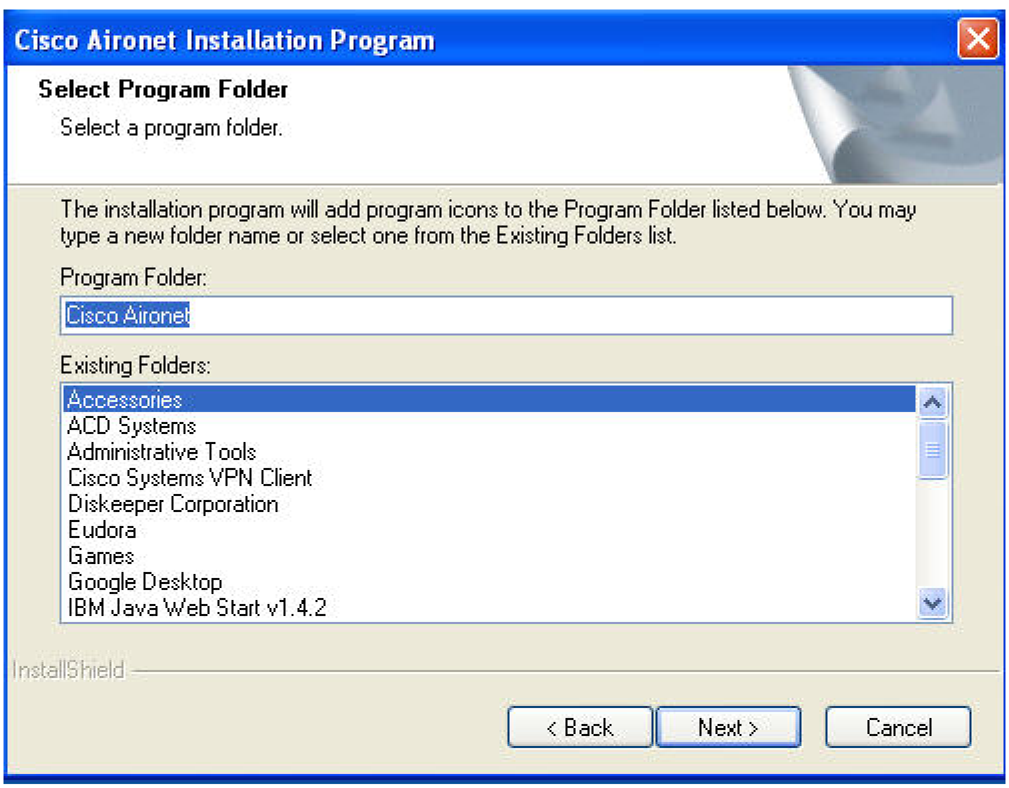
Choose Install Client Utilities and Driver

Step 4

On the next two screens, choose the default setting by clicking on **Next** unless instructed otherwise by your teacher.



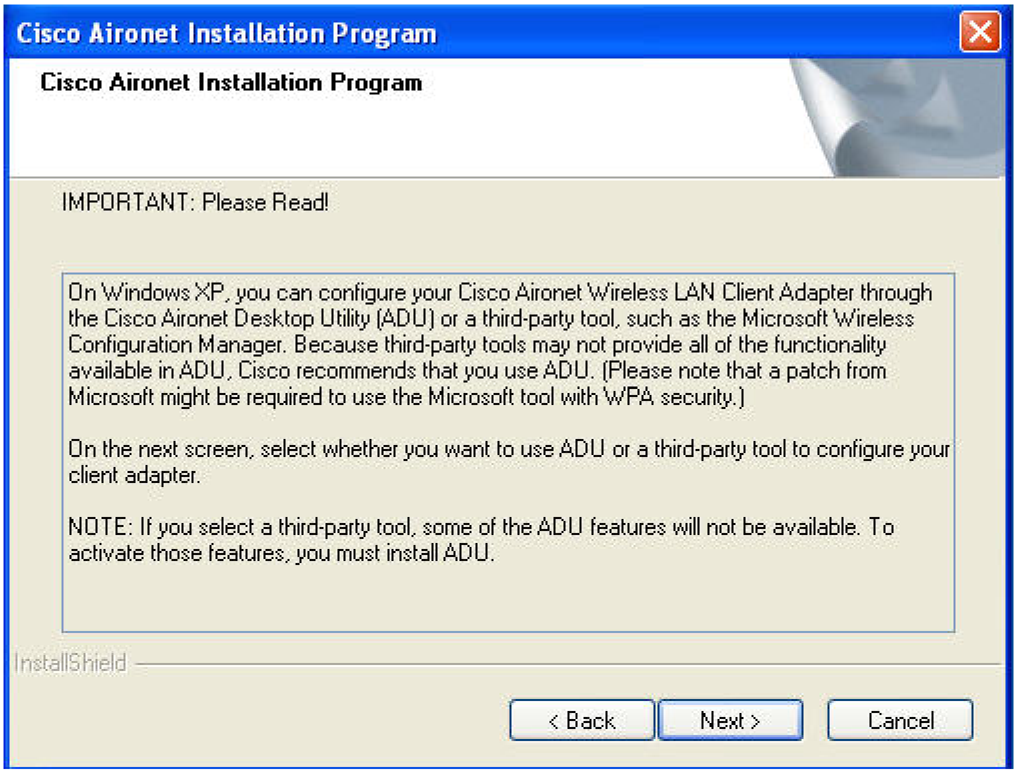
Choose Destination Location for Software Installation



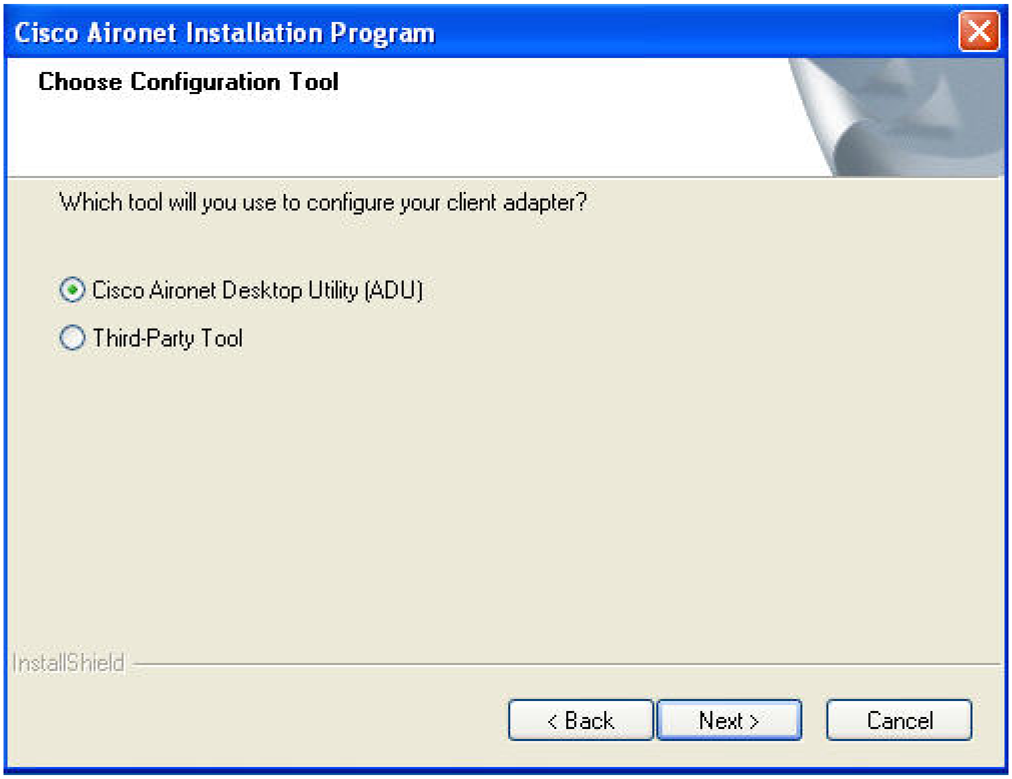
Select Program Folder for Software Installation

Step 5

If you are running Microsoft Windows XP, you get a warning about using the Cisco ADU rather than the default Microsoft Wireless Configuration Manager. After this screen, you have the option to choose between the two. Choose Cisco ADU, because it is more capable than the one from Microsoft.



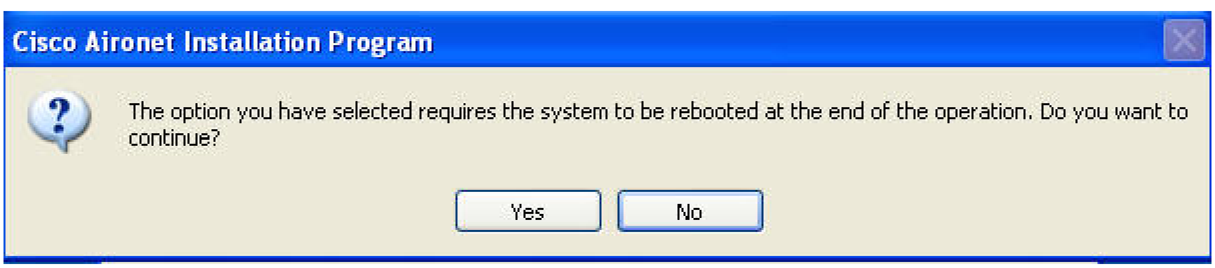
Windows XP Warning



Choose ADU as the Configuration Tool

Step 6

Click on **Yes** to reboot your system at the end of the operation. On the next screen click **OK**.

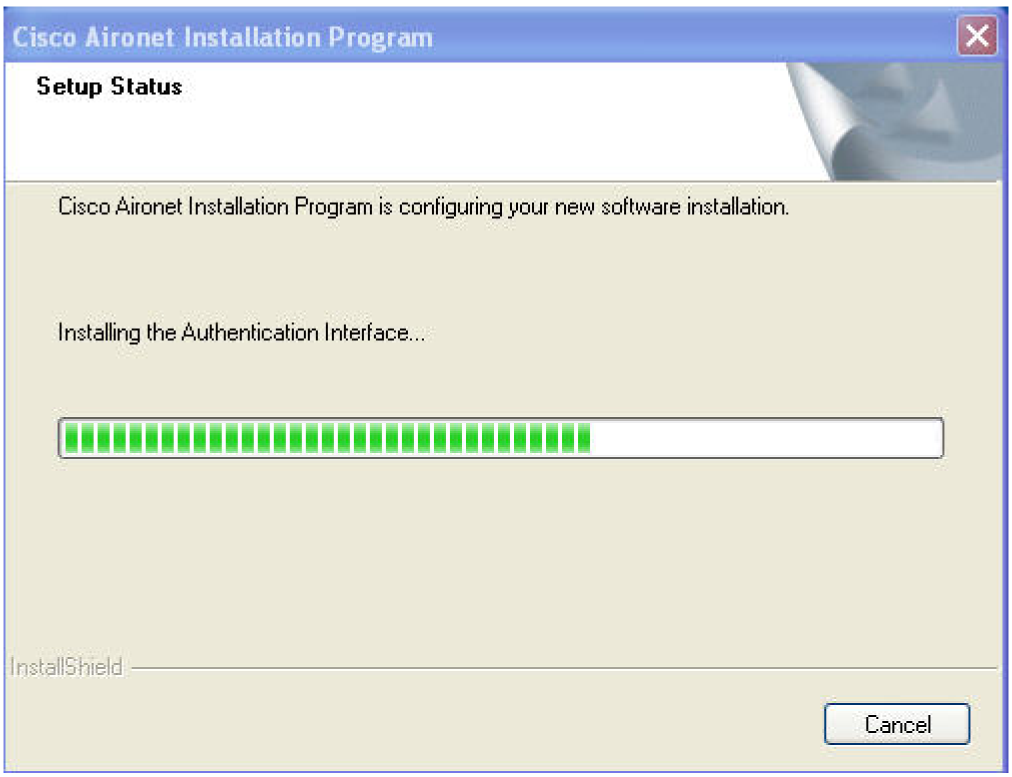


Reboot at the end of the operation

Click OK to Continue

Step 6

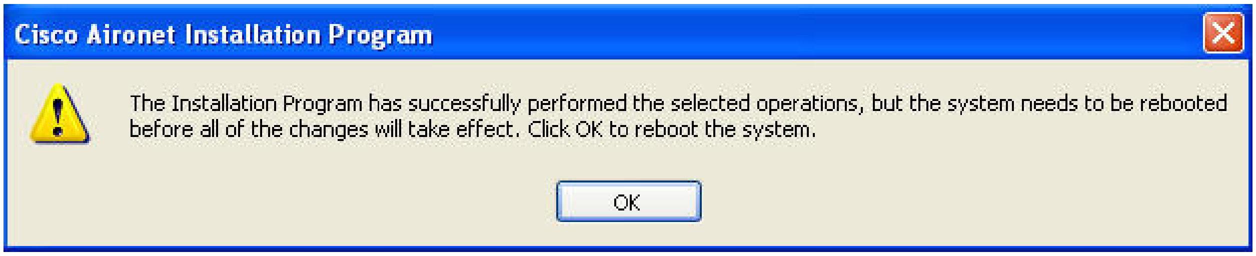
The **Setup Status** screen will show the status of the software installation.



Setup Status

Step 7

When Setup is complete, reboot the computer by clicking **OK**.



Click OK to Continue

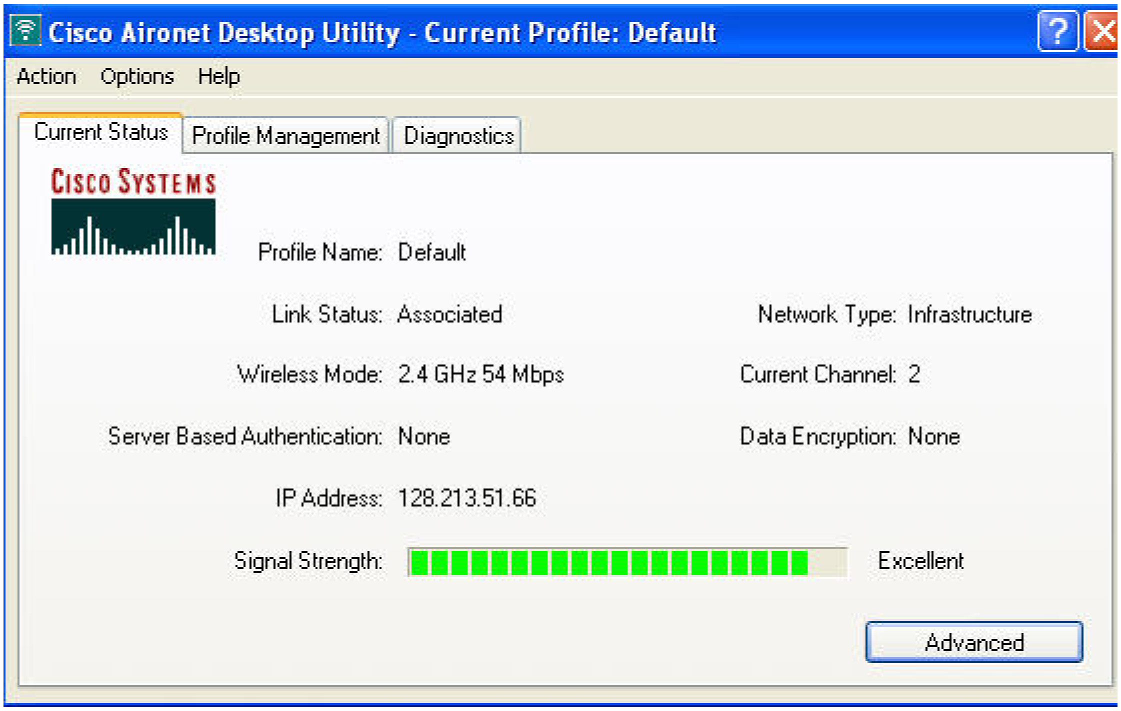
Step 8

After the computer has rebooted, click on the shortcut to the Aironet Desktop Utility (ADU).

Shortcut to Aironet Desktop Utility

Step 9

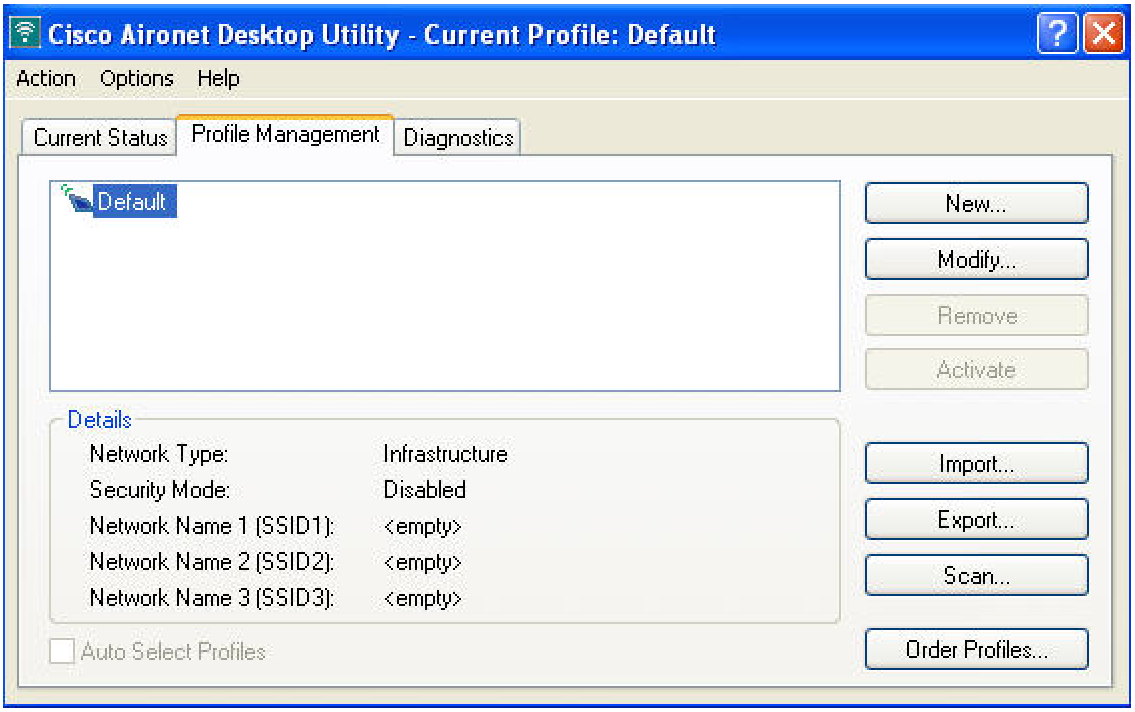
The **Current Status** screen appears by default. In the image below, the PC has found a production wireless network and associated with its access point. If your lab is close to a production wireless network, you may have a similar result. If your PC is not close to a production network, then your **Current Status** screen will look different.



Current Status Screen

Step 10

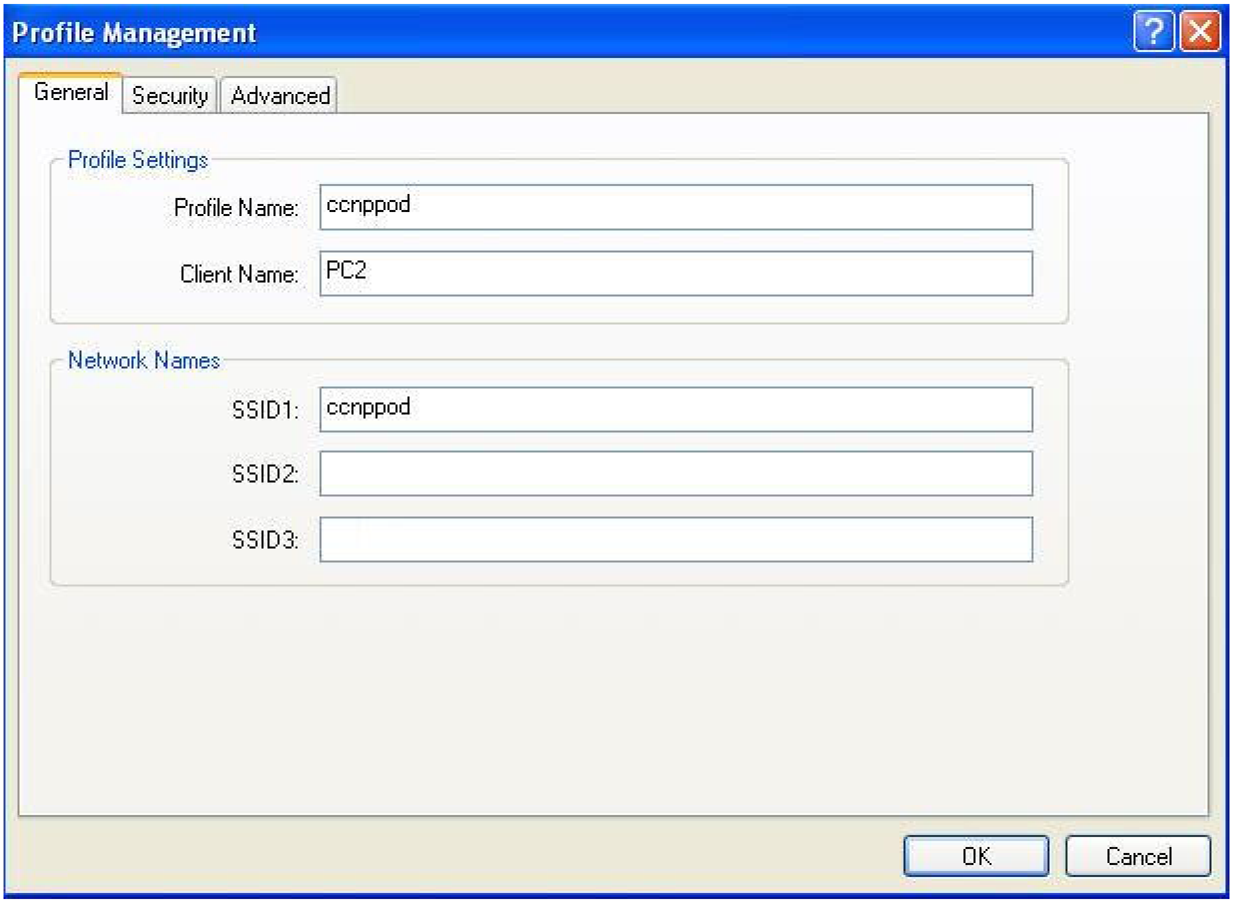
Whether or not you are connected to a production wireless network, you now want to connect to the lab network. Click on the **Profile Management** tab next to the **Current Status** tab. Then click on the **New** button in the upper right hand corner of the screen.



Profile Management Screen

Step 11

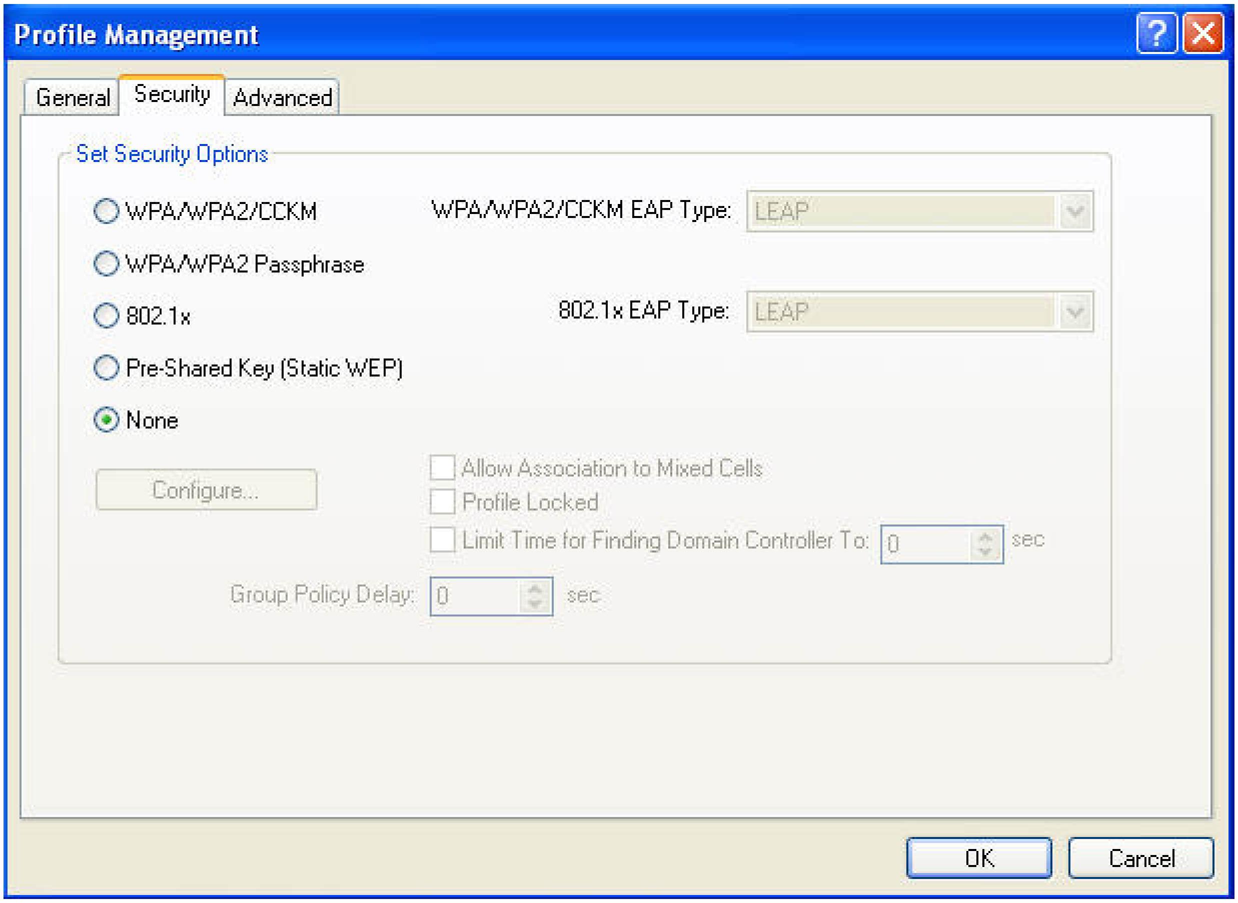
Enter the profile name “ccnppod.” Use the SSID of “ccnppod.”.



SSID configuration

Step 12

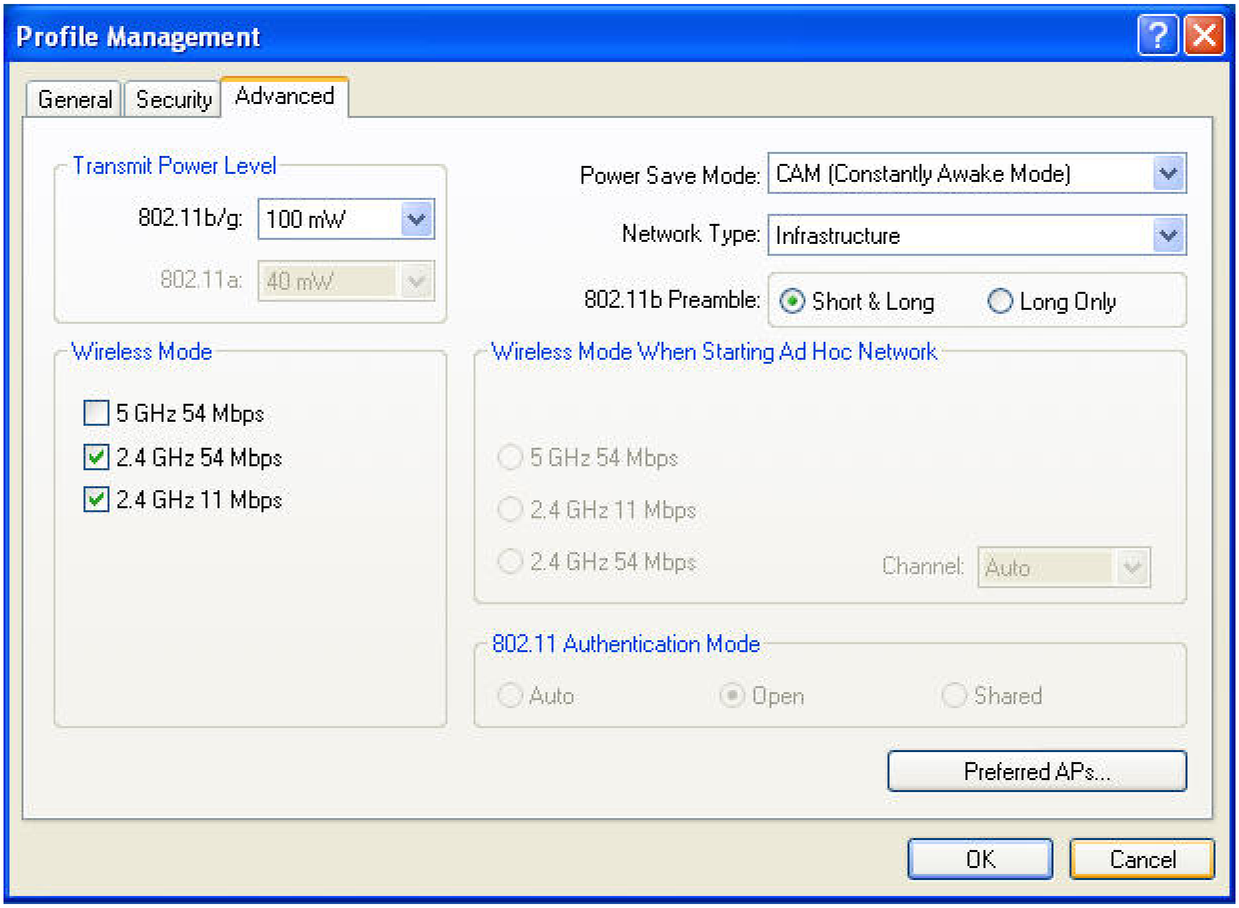
Select the **Security** tab. Select **None**.



Security Options

Step 13

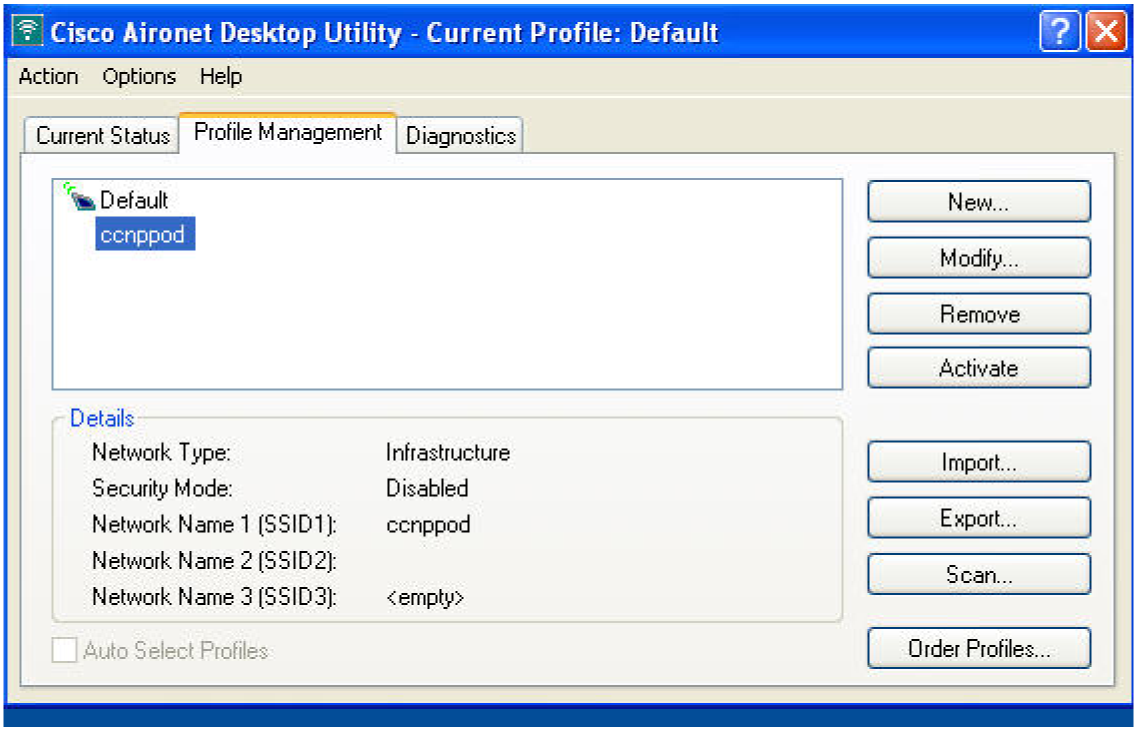
Select the **Advanced** tab. Uncheck **5GHz 54 Mbps** because you are not using 802.11a. Then click **OK**.



Advanced Configuration Options

Step 14

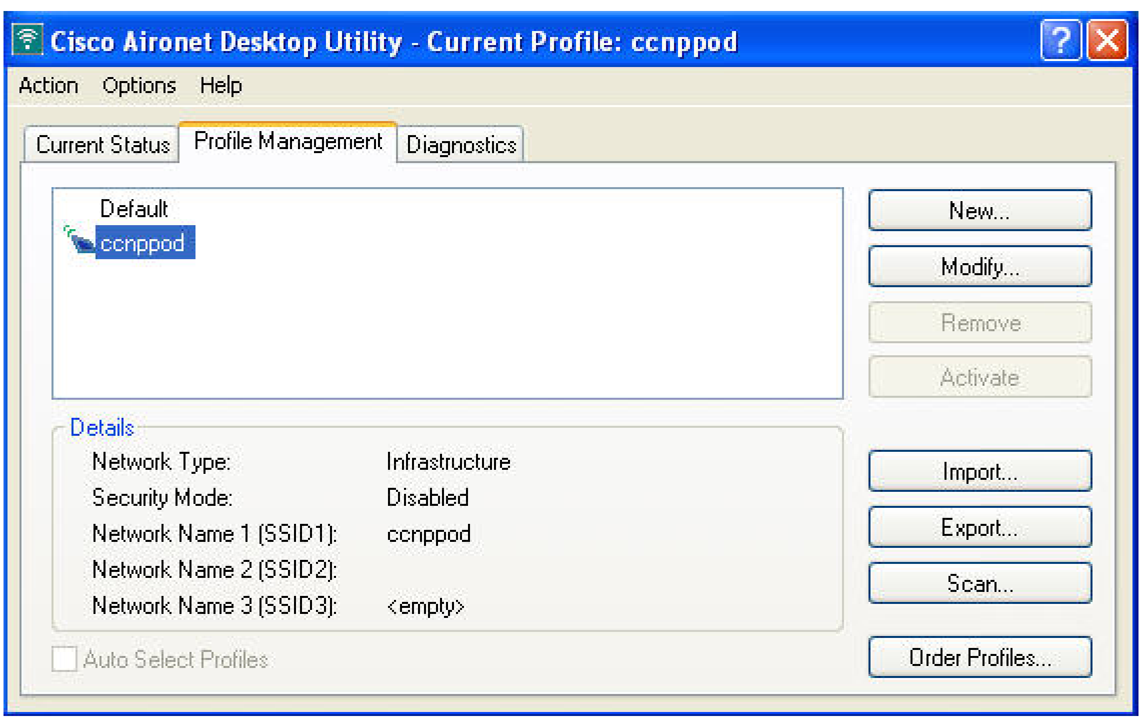
After you click on **OK**, you are returned to the **Profile Management** screen. In addition to the Default profile, there is now the ccnppod profile. Click the **Activate** button on the right hand side of the screen.



Click on the Activate Button

Step 15

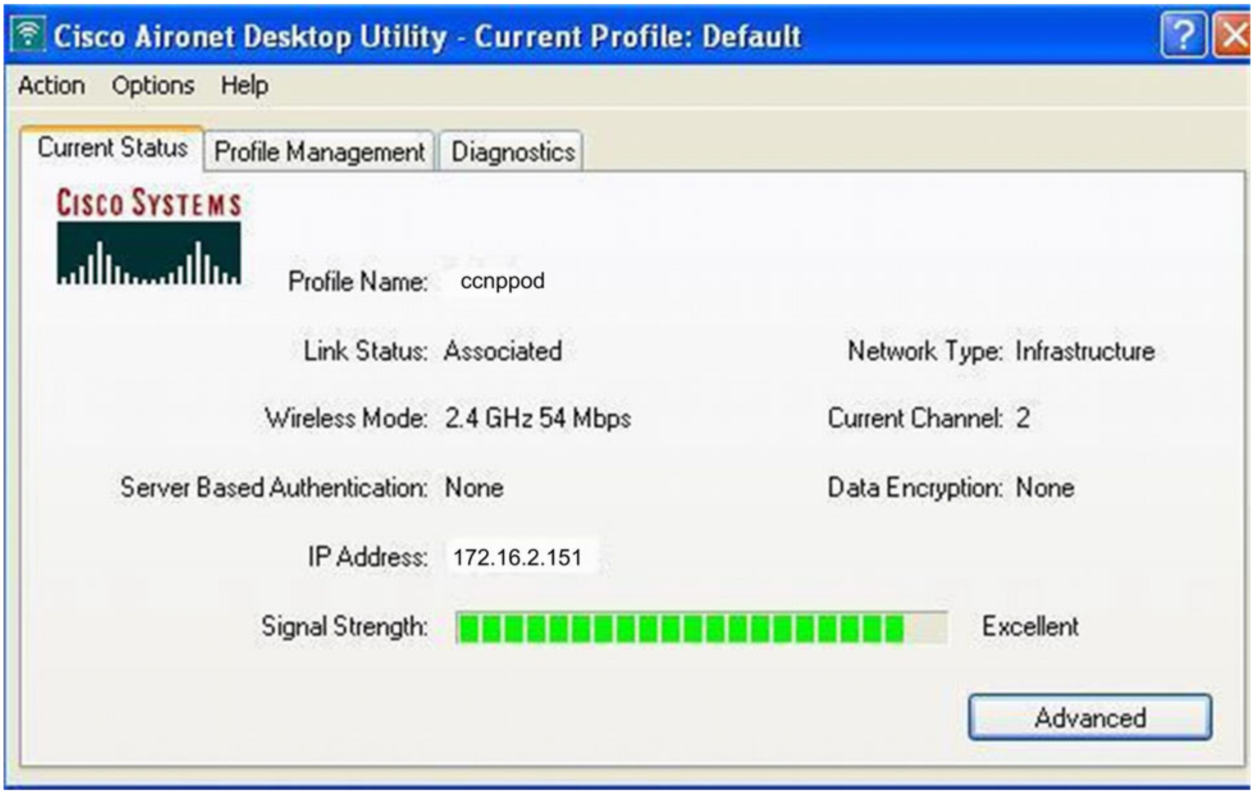
After clicking the **Activate** button, your screen will look like the image below.



ccnppod profile activated

Step 16

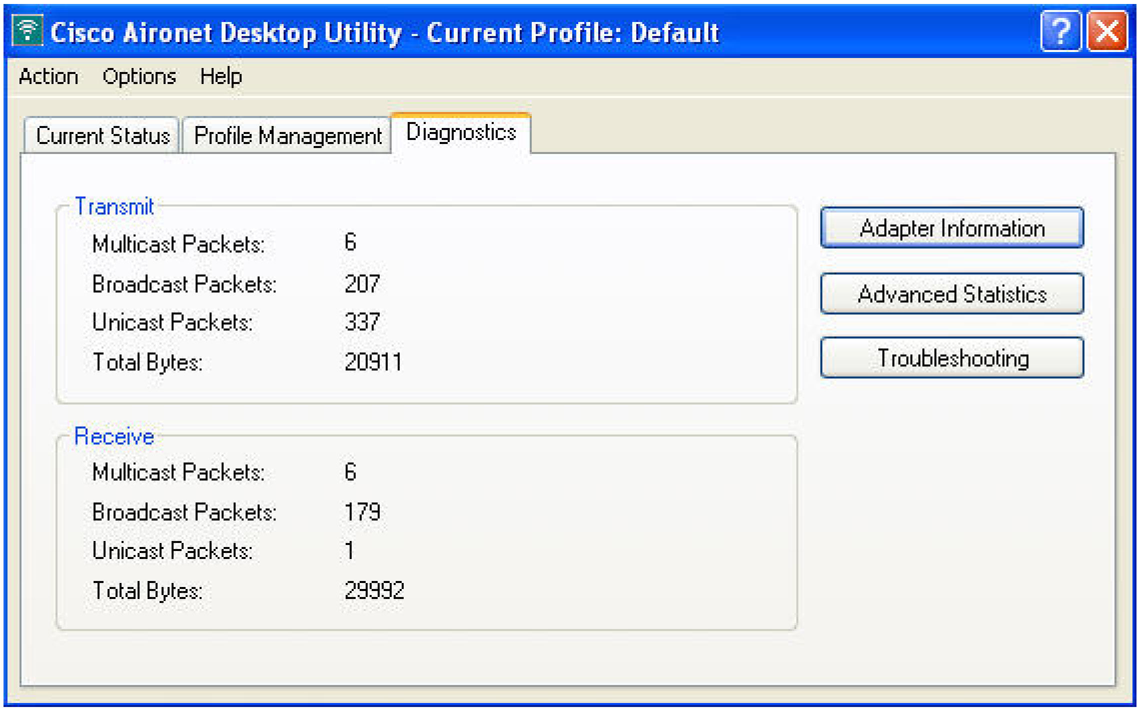
Click on the **Current Status** tab, and your screen will look similar to the image below.



Current Status of ccnppod profile

Step 17

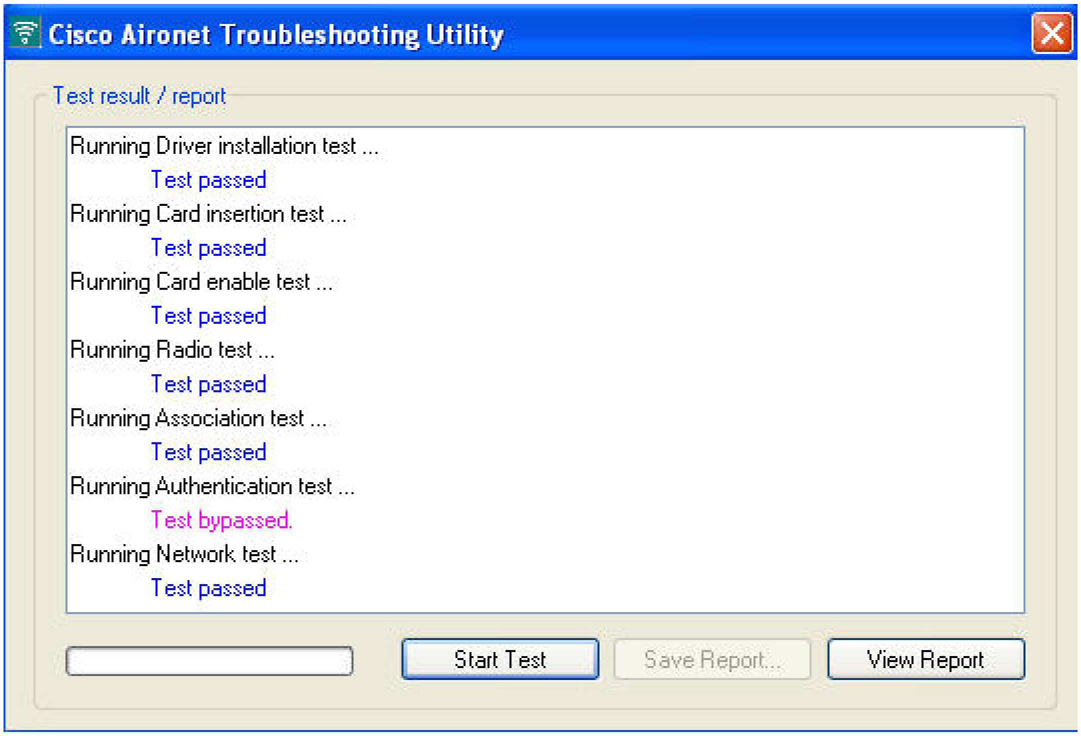
The **Diagnostics** tab shows transmit and receive data about the wireless connection.



Diagnostics Screen

Step 18

To run tests on the wireless card and see the results, click the **Troubleshooting** button.



Running Troubleshooting Tests

Lab 6.4 Configuring WPA Security with Preshared Keys

Learning Objectives

- Configure a Wireless LAN with WPA security policies using preshared keys
- Authenticate with a wireless access point with WPA security protocols

Topology Diagram

Select the appropriate diagram based upon whether you have external or internal WLAN controllers:

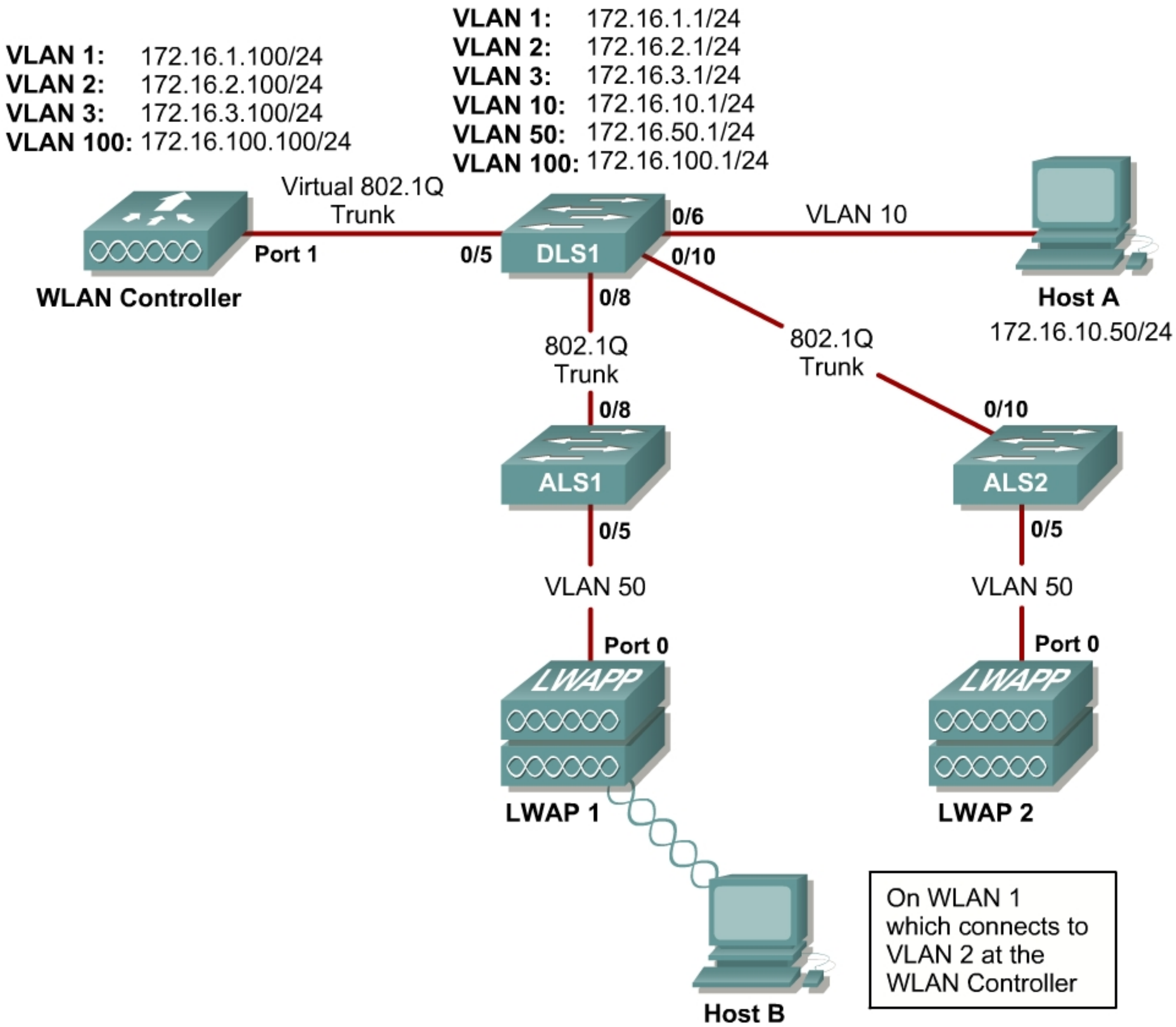


Figure 1-1: Ethernet Connectivity Diagram for Module 6, External WLAN Controller

Connectivity Diagram using a Wireless LAN Controller Network Module

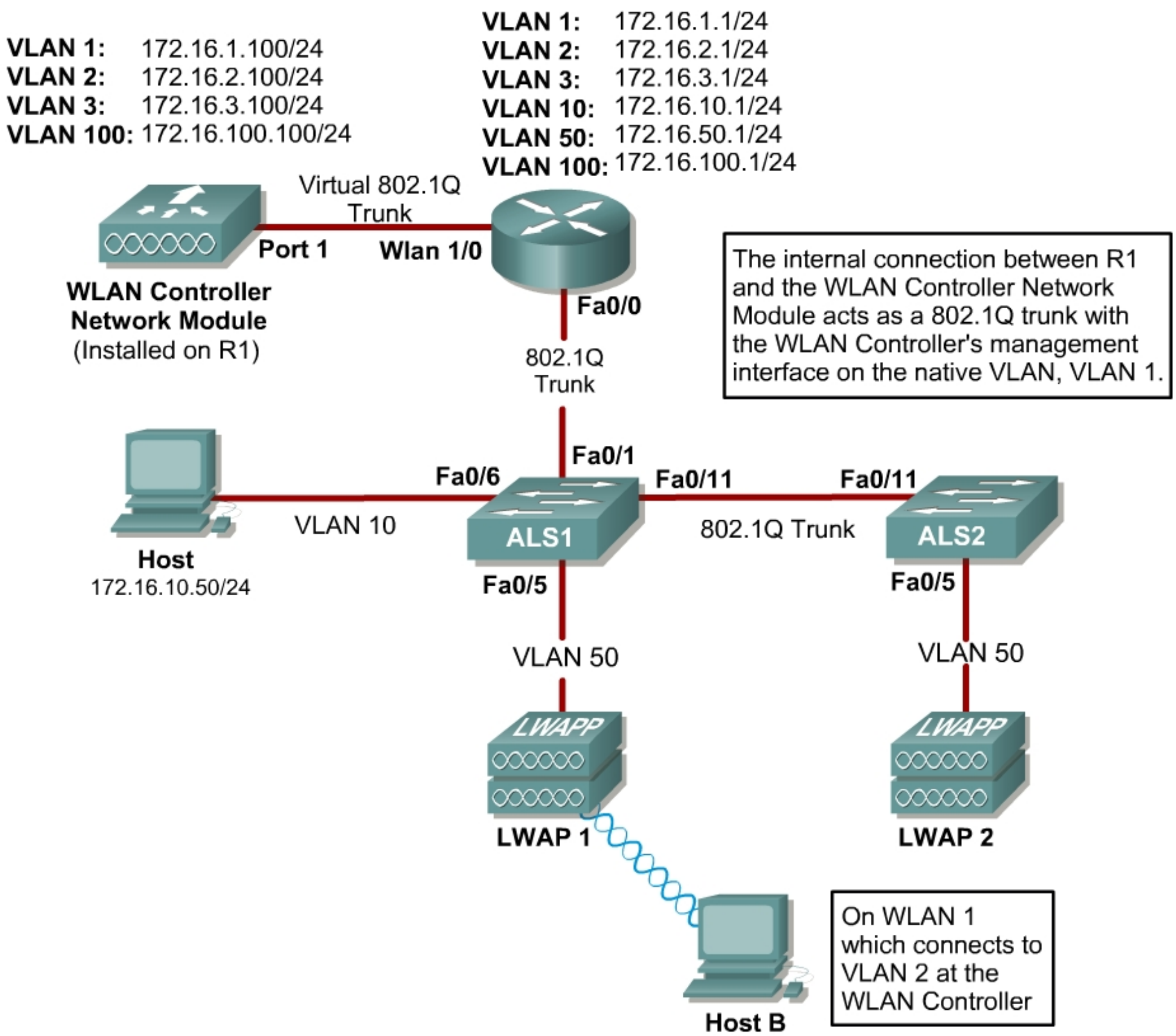


Figure 1-2: Ethernet Connectivity Diagram for Module 6, Internal WLAN Controller

Scenario

In this lab, you will configure and verify Wi-Fi Protected Access (WPA) security in a wireless environment using preshared keys.

This lab requires two separate PCs, Host A and Host B. Host A will act on VLAN 10 as the Cisco access control server (ACS) server and will also be used to configure the wireless LAN (WLAN) controller as a PC has been used to do in previous labs. Host B requires a Cisco wireless network card with the Aironet Desktop Utility installed. Host B will function as a wireless client on WLAN 1 which corresponds to VLAN 2.

You may complete this scenario using either the external WLAN controller (WLC) or the network module that resides in a router. However, you must load the final configurations from the end of Lab 6.1: Configuring a WLAN Controller.

We highly recommended that you complete Labs 6.1, 6.2, and 6.3 before attempting this lab.

Note:

This lab will only go into the details of configuring WLAN security using WPA-PSK. For more information on using the web interface of the WLC, consult Lab 6.2: Configuring a WLAN Controller via the Web Interface.

Preparation

Complete Lab 6.1 and ensure that all switches and routers, the WLAN controller, and the host are configured the way they would be at the end of Lab 6.1.

At the end of Lab 6.1, you should already have the following features configured and verified:

- VLAN connectivity
- Trunk ports
- HTTP access to the WLC
- Lightweight Access Points (LWAPs) associated with the controller

Step 1: Connect to the WLC from the Host

On Host A, open up Internet Explorer and go to the URL https://172.16.1.100. This is the secure method of connecting to the management interface of the WLAN controller. You can also use http://172.16.1.100 since we previously enabled regular insecure HTTP access in the command-line interface (CLI) for Lab 6.1. If you connect to the secure address, you may be prompted with a security warning. Click **Yes** to accept it and you will be presented with the login screen for the WLAN controller. Click **Login** and an authentication dialog box will appear.

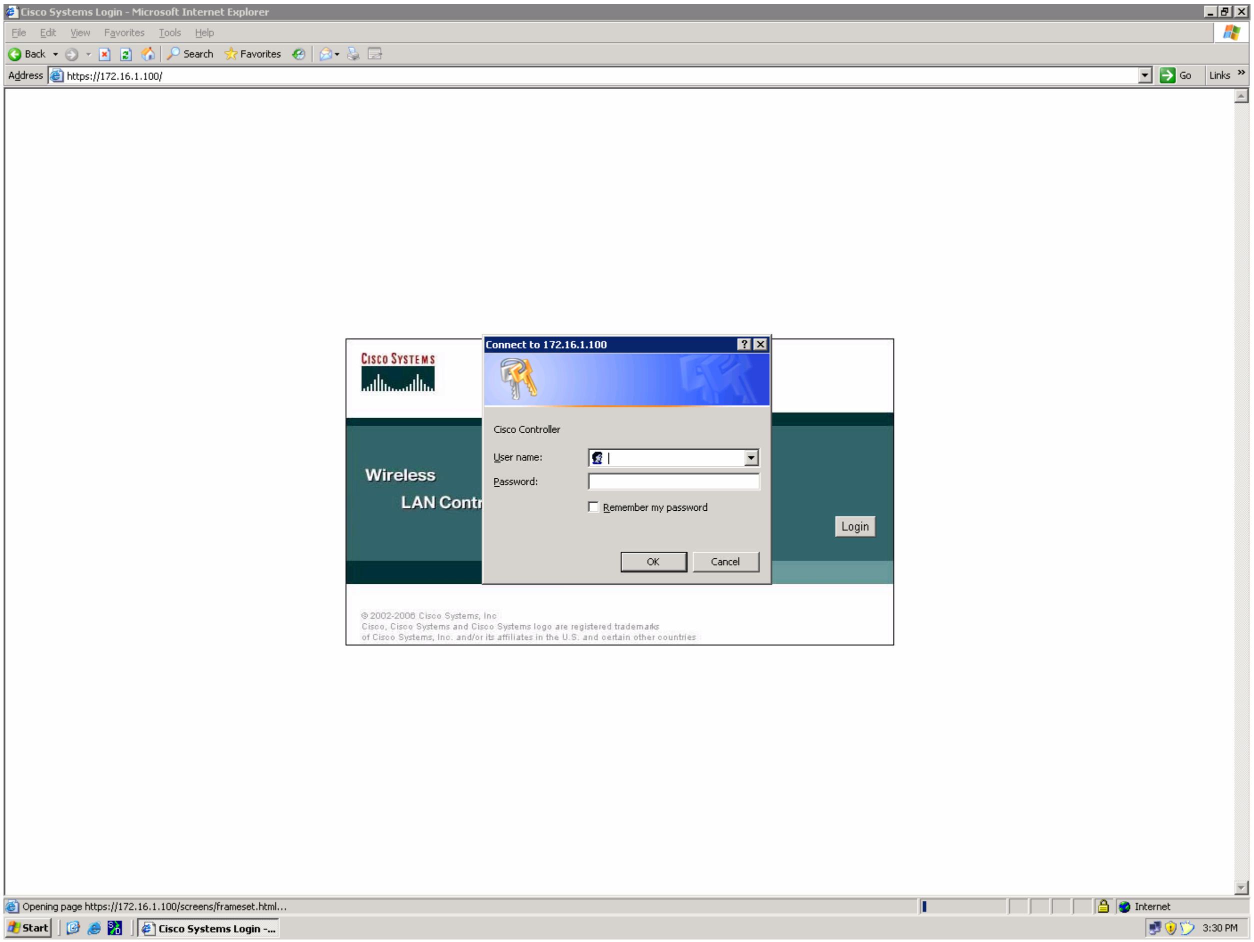


Figure 1-1: HTTP Access to the WLAN Controller

Use “cisco” as both the username and password. You configured these in the previous lab. Click **OK** to get to the main page of the graphical user interface (GUI). You are then presented with the monitor page for the WLAN controller.

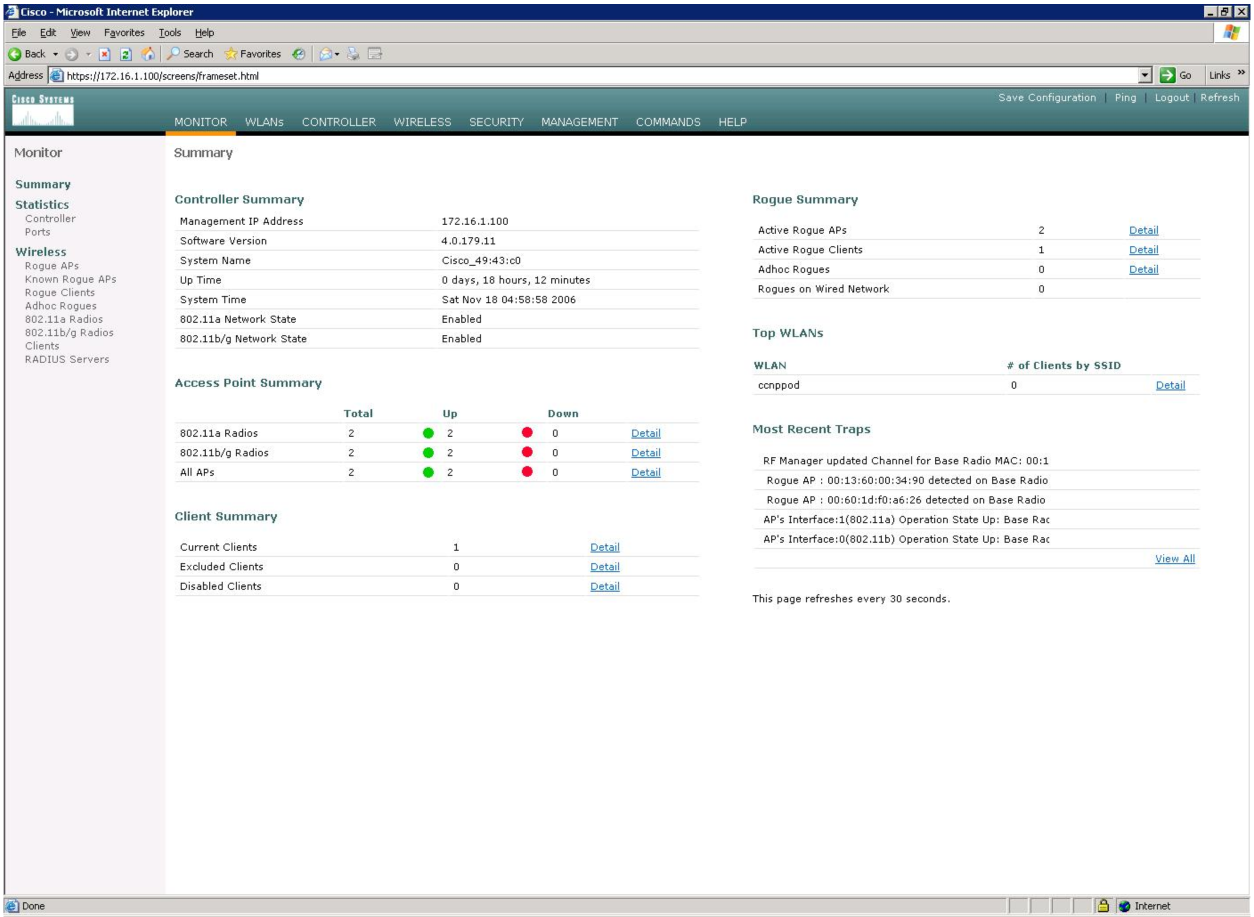


Figure 1-2: WLAN Controller Monitor Page

Make sure you see two access points under the "Access Point Summary" part of the page. If you do not, reload the LWAPs, otherwise, troubleshoot. You may also see it detecting rogue access points if your lab has other wireless networks around it; this behavior is normal. You can also see various port controller and port statistics by clicking their respective links on the left-hand menu on the screen.

Step 2: Assign a VLAN to a WLAN

Since this step is identical to steps found in Lab 6.2: Configuring a WLAN Controller via the Web Interface, we will not explain the many details of each of the configuration changes. For more information on what these changes do, reference Lab 6.2.

Click the **Controller** tab at the top of the window. Then, click **Interfaces** in the left pane. Click **New** to create a new interface.

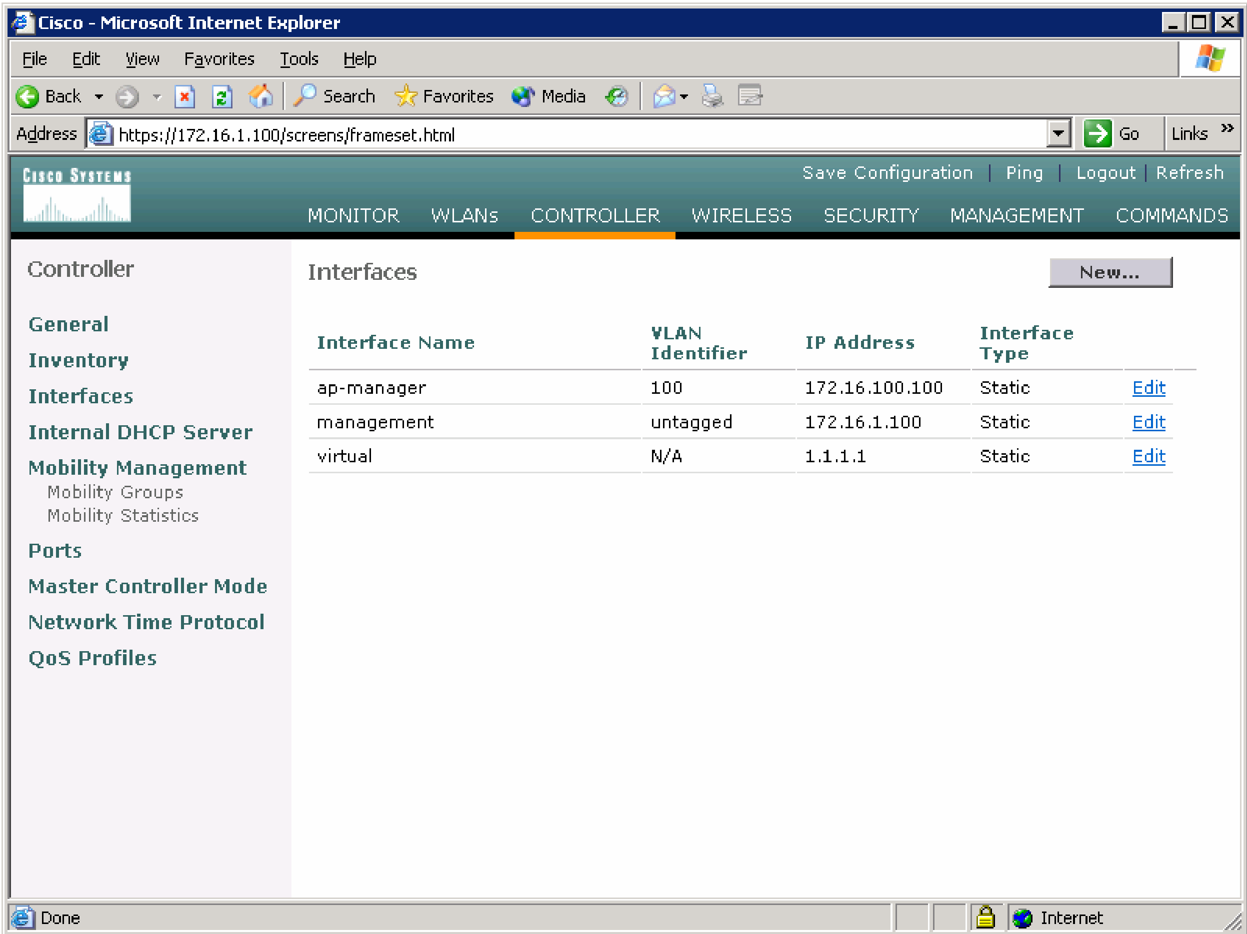


Figure 2-1: Interface Configuration Page

Name the interface “VLAN2” and assign it to 802.1Q tag 2, just like in Lab 6.2. Click **Apply** when you have completed this.

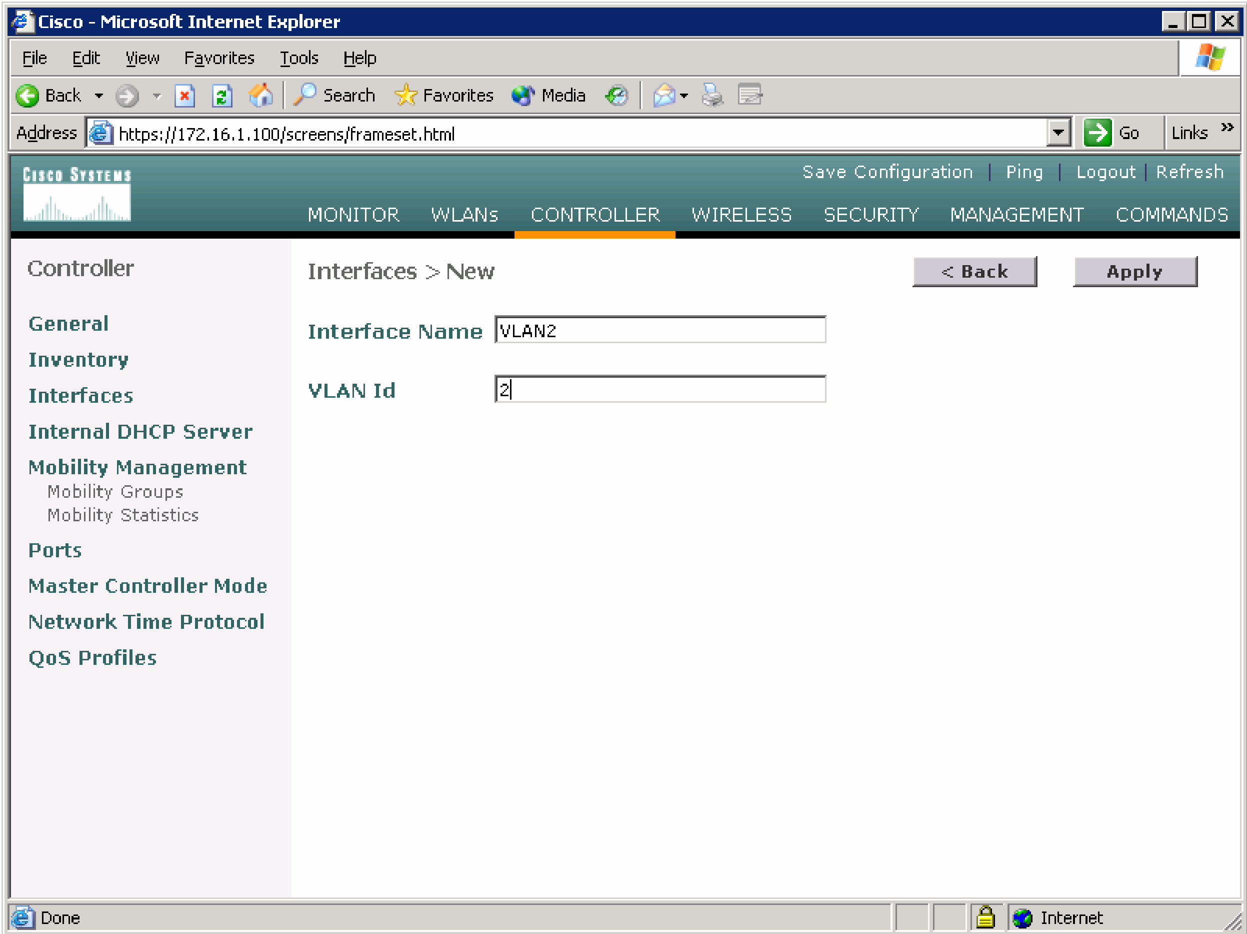


Figure 2-2: Creating a New VLAN Interface

Configure the IP address, default gateway, port number, and Dynamic Host Configuration Protocol (DHCP) server for this interface as shown in Figure 2-3, and then click **Apply**.

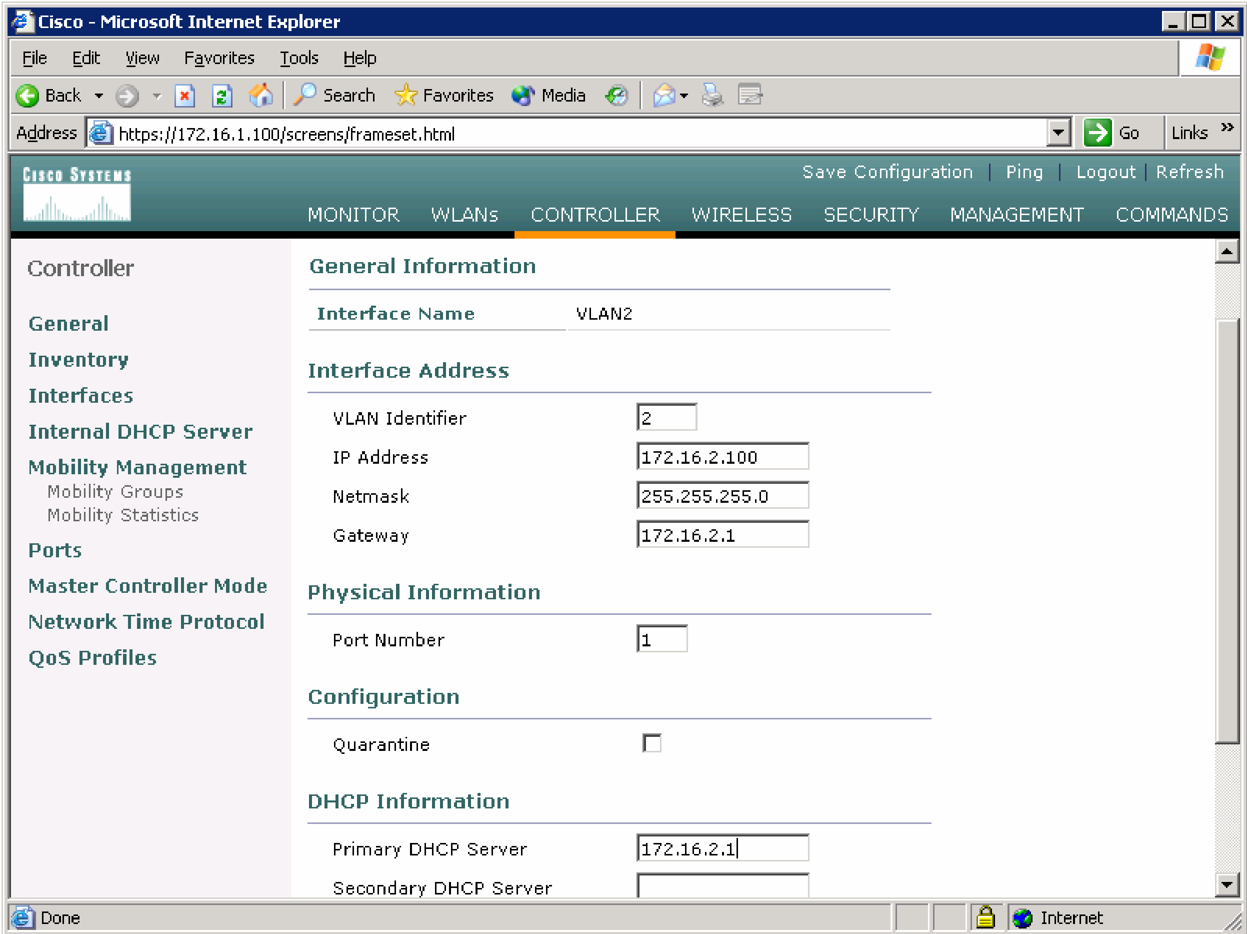


Figure 2-3: Configuring VLAN Interface Properties

Accept the warning by clicking **OK**.

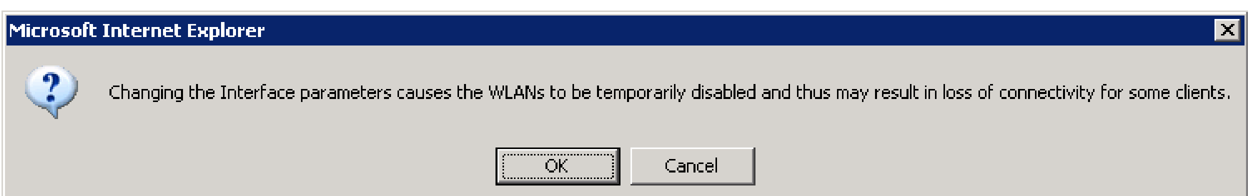


Figure 2-4: Interface Parameter Confirmation

You should see the new interface in the interface list.

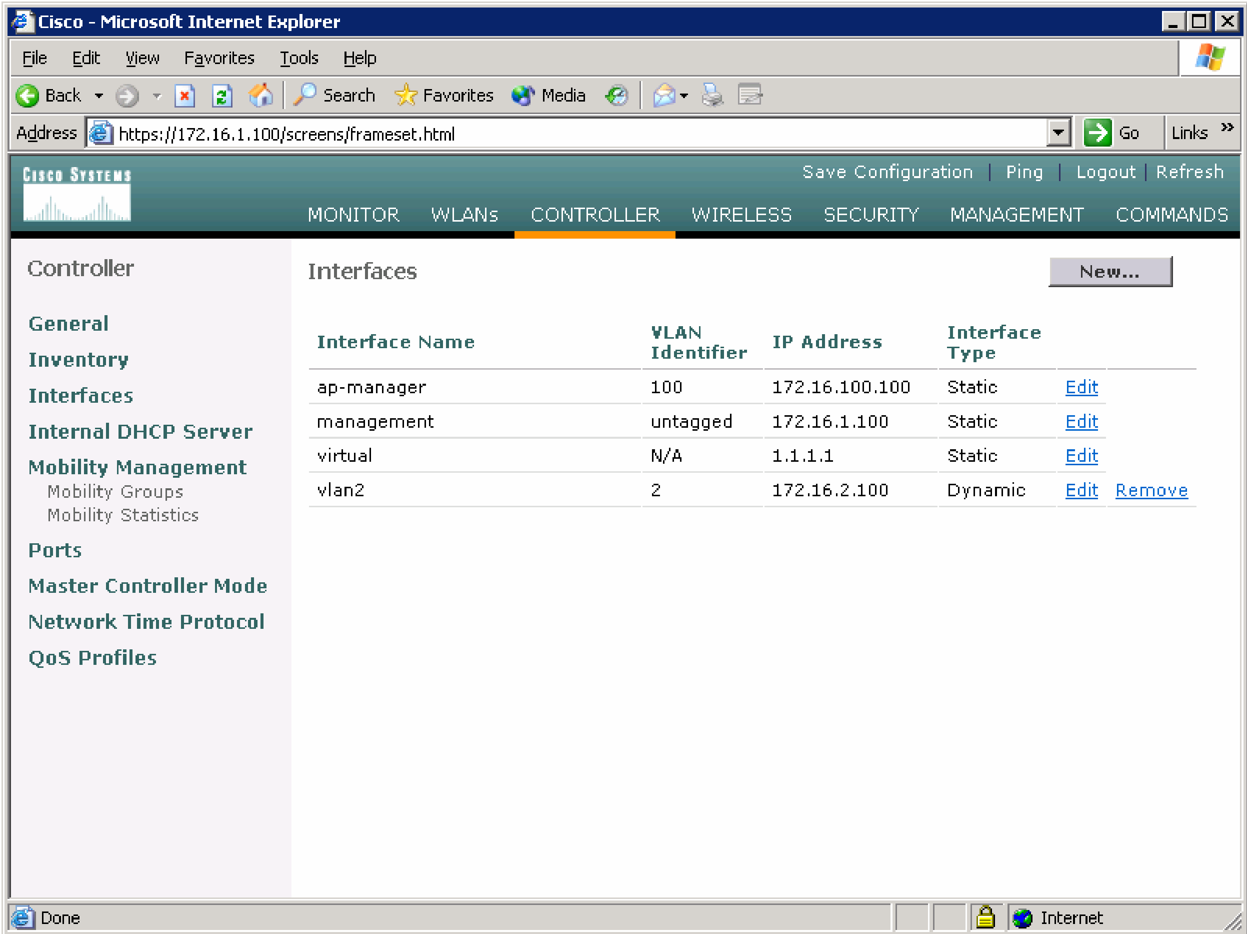


Figure 2-5: Verify Existing VLAN Interfaces

Click the **WLANs** tab at the top of the screen to view the current WLAN configuration. Click **Edit** for the WLAN shown (it is towards the right of the screen).

What is the default security policy for a WLAN? Hint: Reference Figure 2-6.

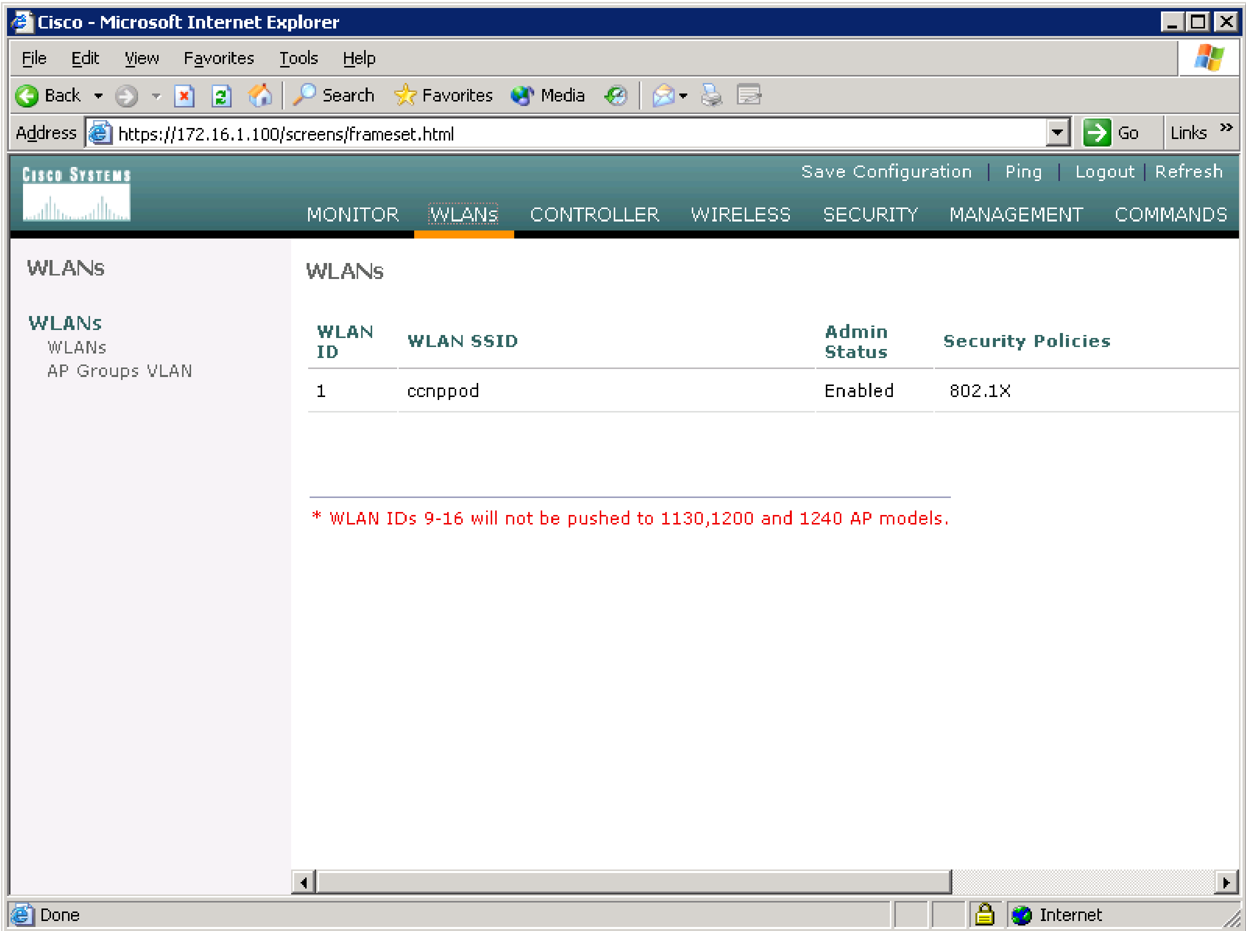


Figure 2-6: Viewing Existing WLANs with Security Policies

On the right side of the WLAN configuration page, change the layer 2 security method to **WPA1+WPA2**. Also make sure that the **Broadcast SSID** option is checked. Even though you are broadcasting the service set identifier (SSID), no clients should be able to connect until you set the security policies configured later.

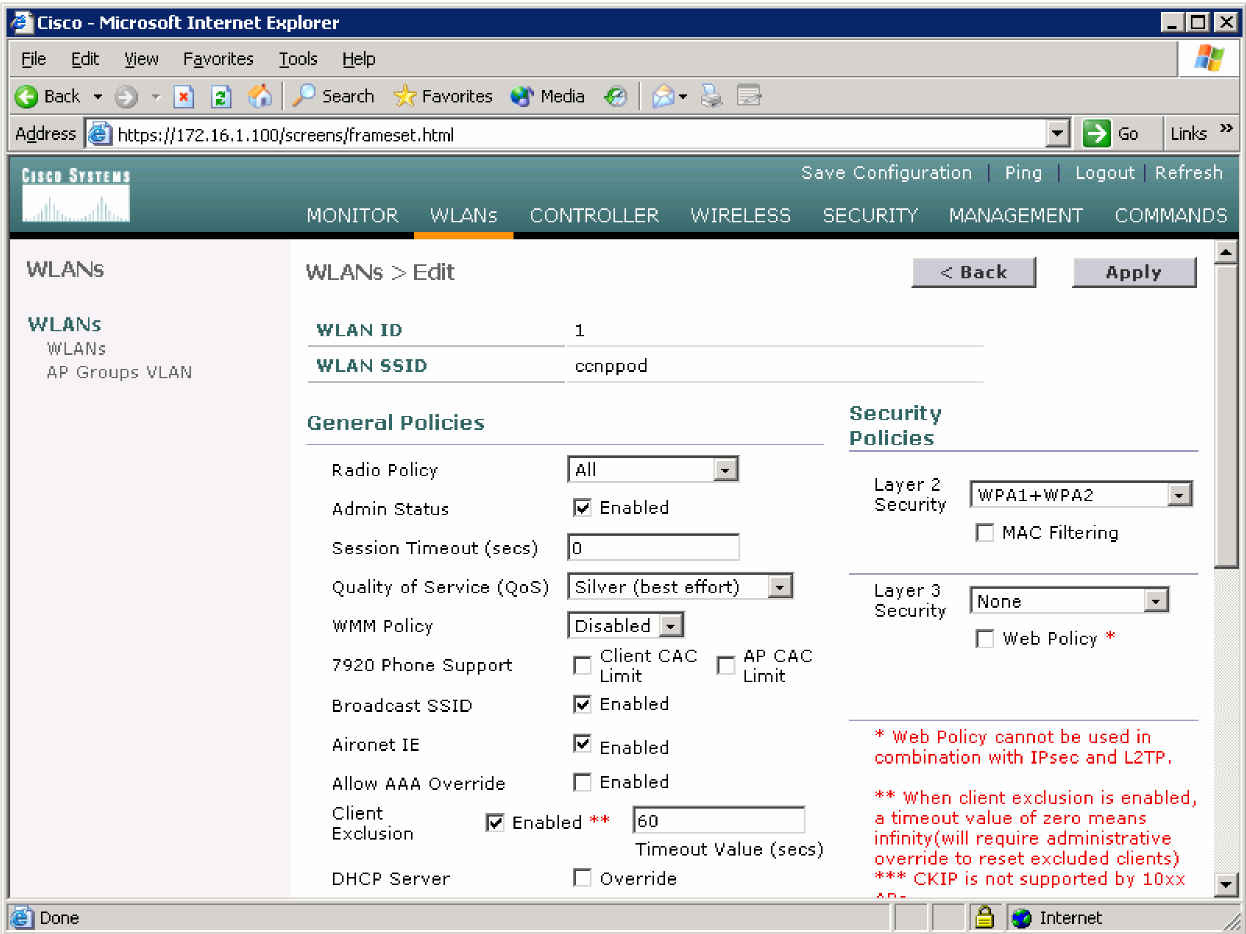


Figure 2-7: Editing the Configuration for WLAN 1

Scroll down the page and change the interface to the VLAN 2 interface created earlier.

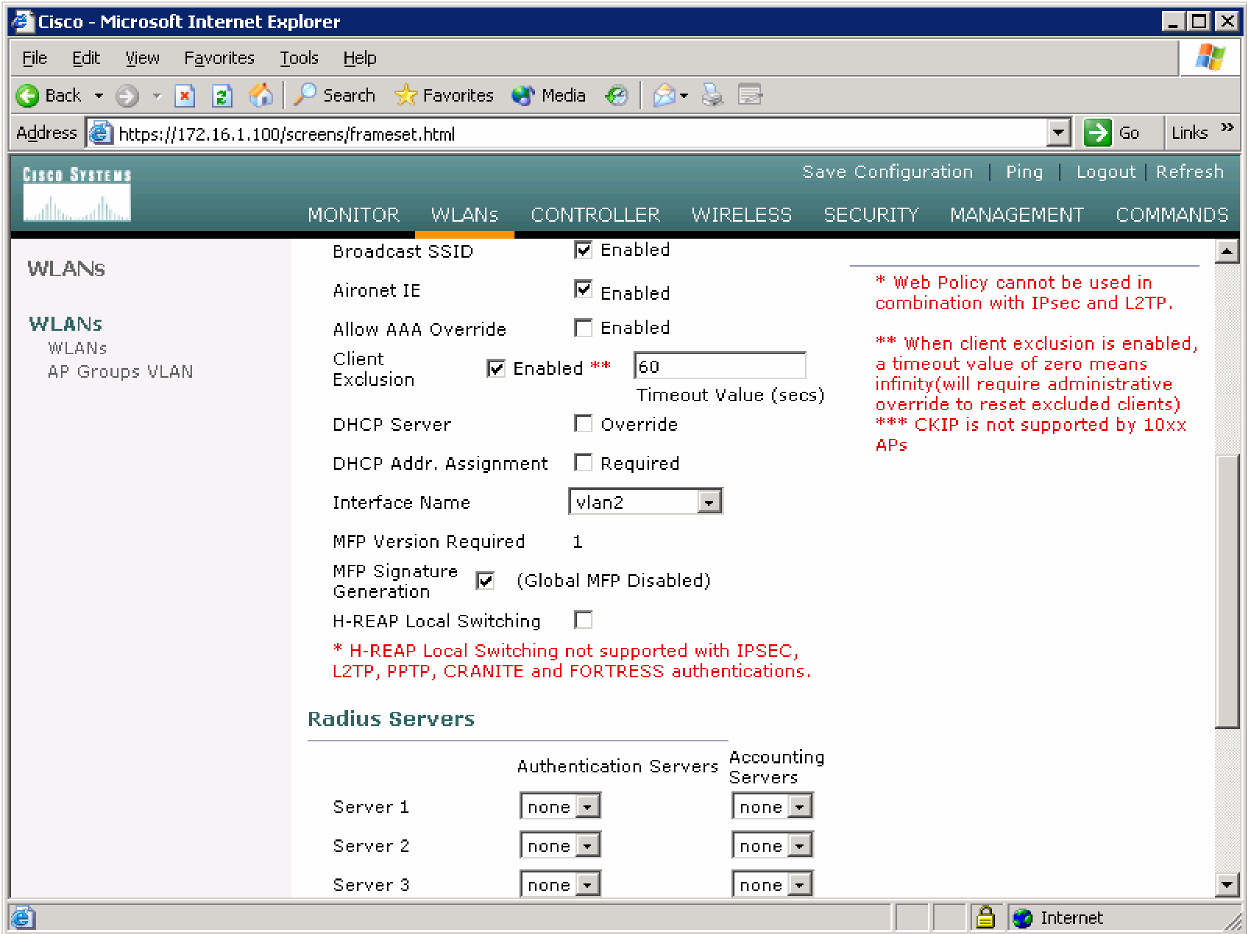


Figure 2-8: Editing the VLAN Interface Connected to WLAN 1

Use a **WPA2** policy with Advanced Encryption Standard (AES) encryption. Configure a preshared key of “password”. Click **Apply** at the top of the page when done.

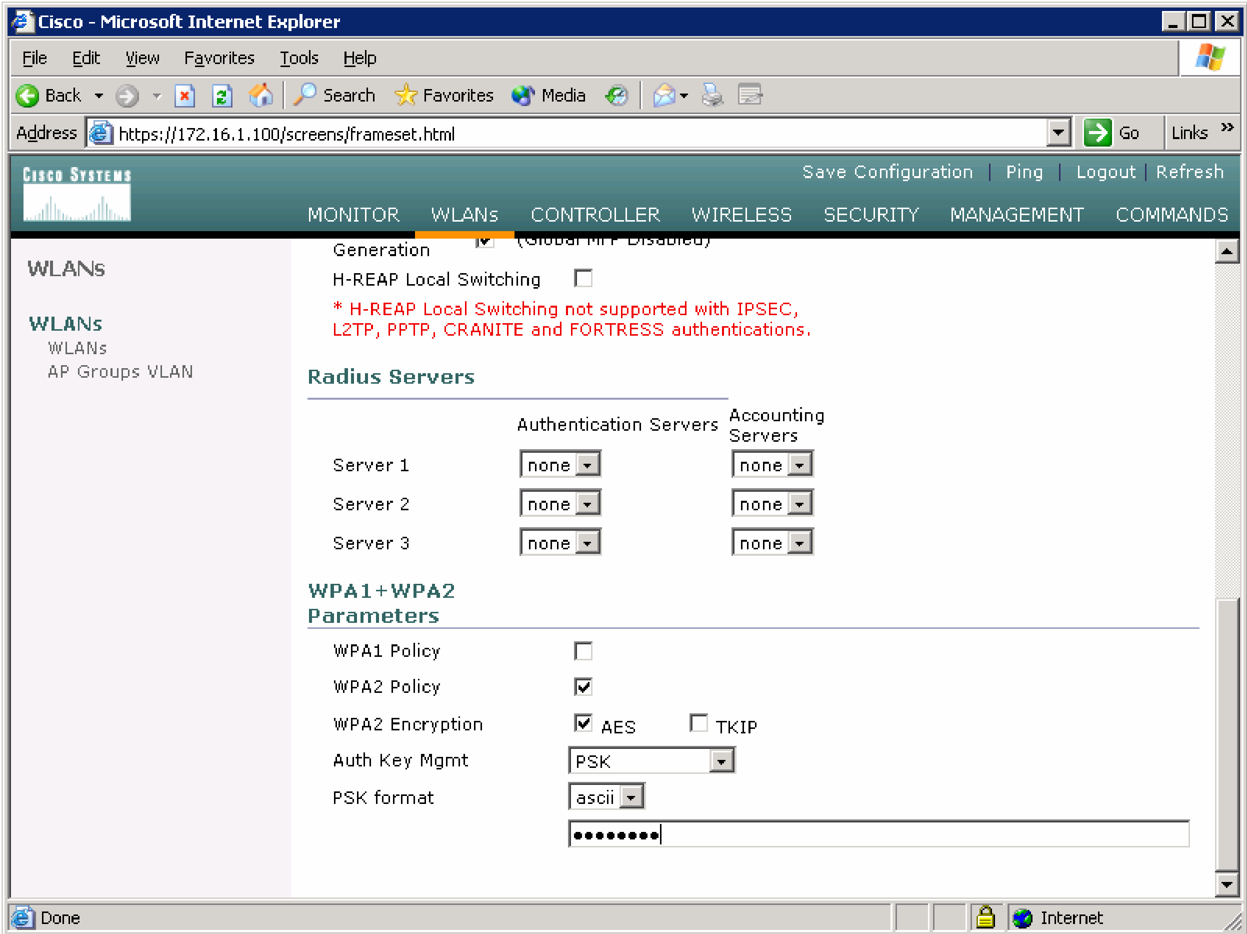


Figure 2-9: Editing the Security Policy for WLAN 1

You should be returned to the WLAN list screen with the new security method shown. Assuming that the LWAPs are associated with the WLC correctly, they should now broadcast this SSID and clients should be able to connect.

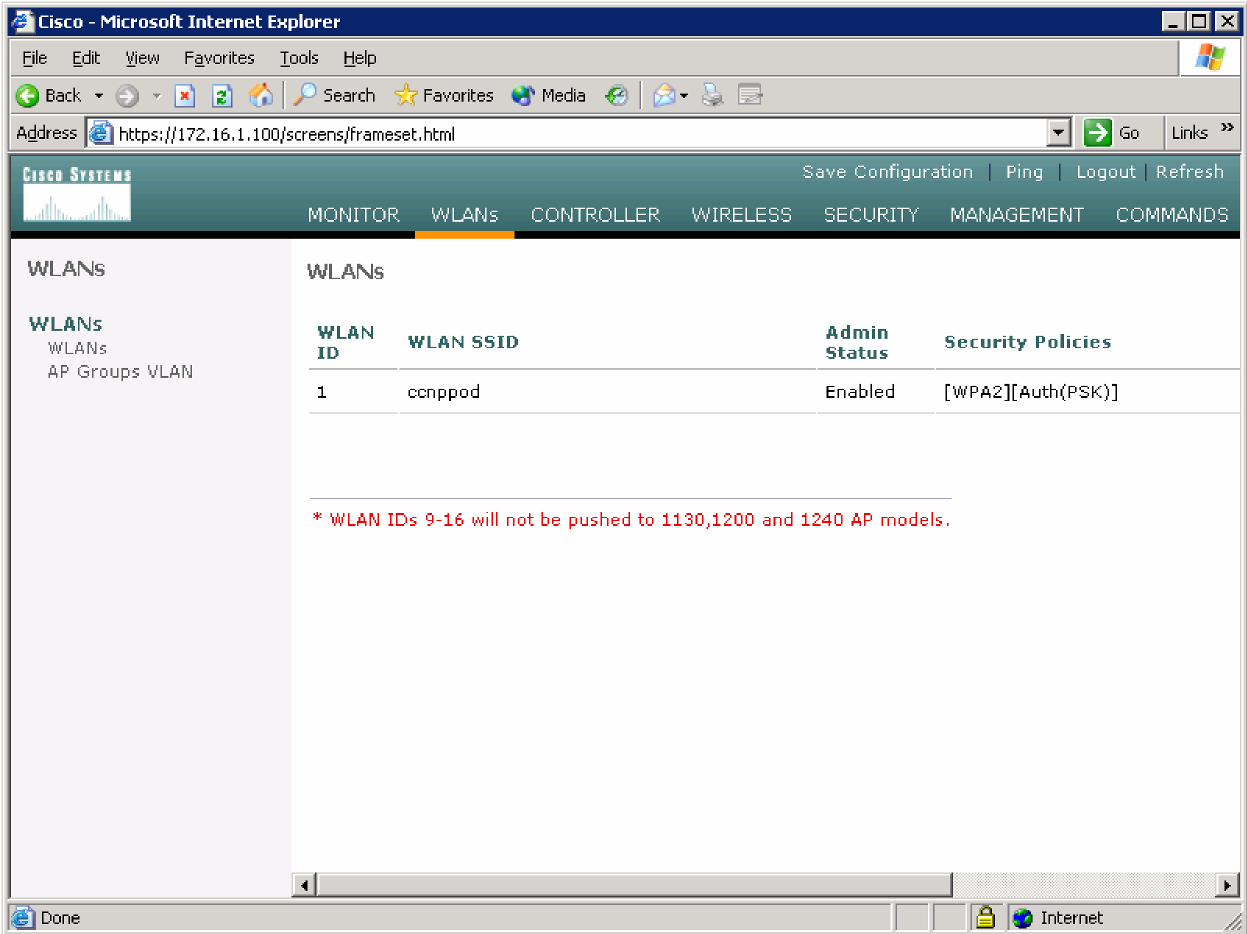


Figure 2-10: WLAN 1 with a WPA2 Security Policy

What is the benefit in configuring preshared keys as the wireless security method?

What is the downside of configuring preshared keys as the wireless security method?

Step 3: Connect to WLAN Using Cisco Aironet Desktop Utility

On Host B, open up the Cisco Aironet Desktop Utility either by the icon on the desktop or the program shortcut in the start menu. If you do not have the Cisco Aironet Desktop Utility (ADU) installed, consult Lab 6.3: Configuring a Wireless Client. Once in the ADU, click the **Profile Management** tab. Next, click **New** to make a new profile.

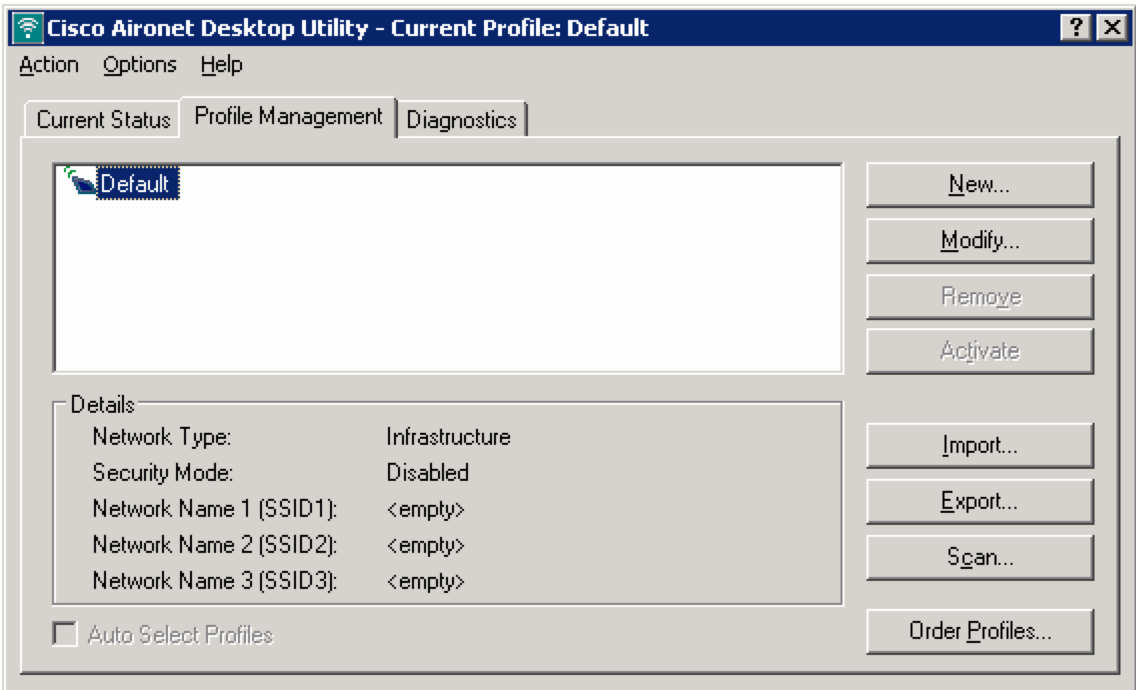


Figure 3-1: Cisco ADU Profile Management Tab

Use a profile name and SSID of “ccnppod” since this was the SSID configured in Lab 6.1. Use any client name desired. Here, “CiscoClient” is the name used.

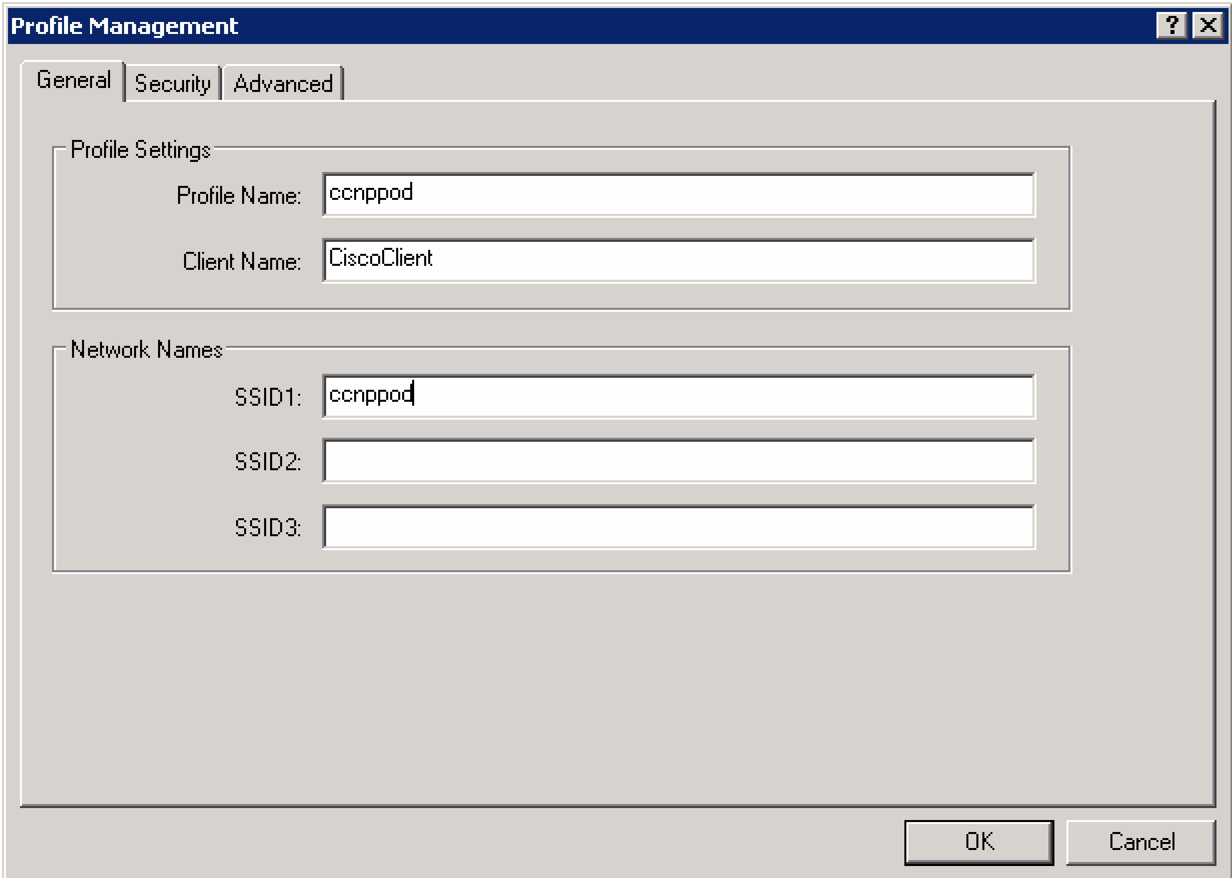


Figure 3-2: Configuring Profile Options and SSID

Click the **Security** tab and set the security type as **WPA/WPA2 Passphrase**. We are using the passphrase because we configured preshared keys rather than a more advanced method. After selecting the security method, click **Configure**.

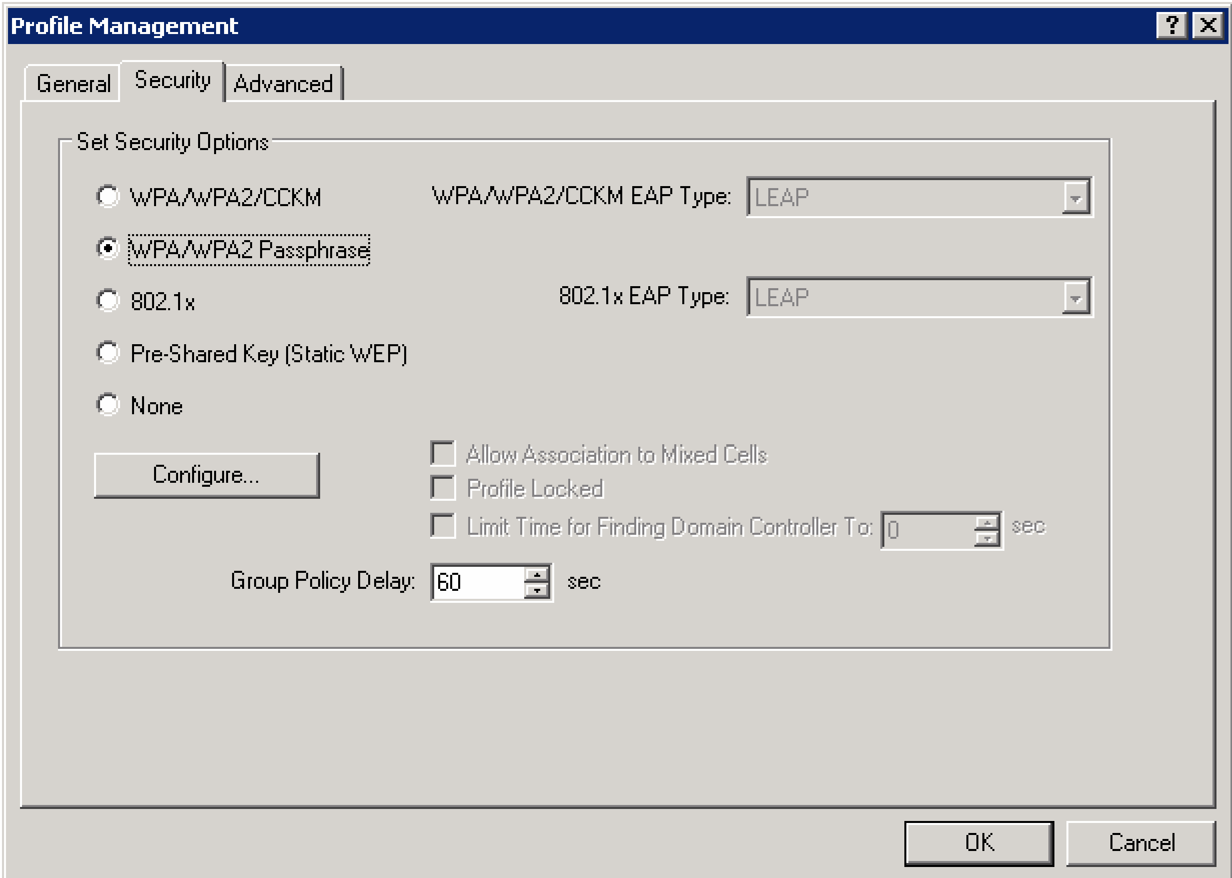


Figure 3-3: Wireless Security Options

Enter in the same password used before for WPA, which is "password," and then click **OK**.

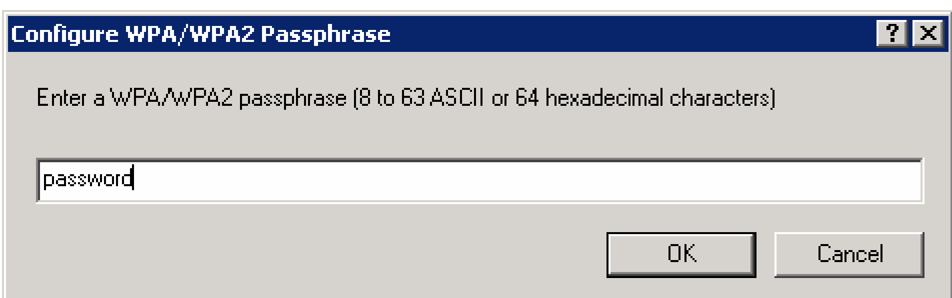


Figure 3-4: Passphrase Configuration

Select the **ccnppod** profile and click **Activate**.

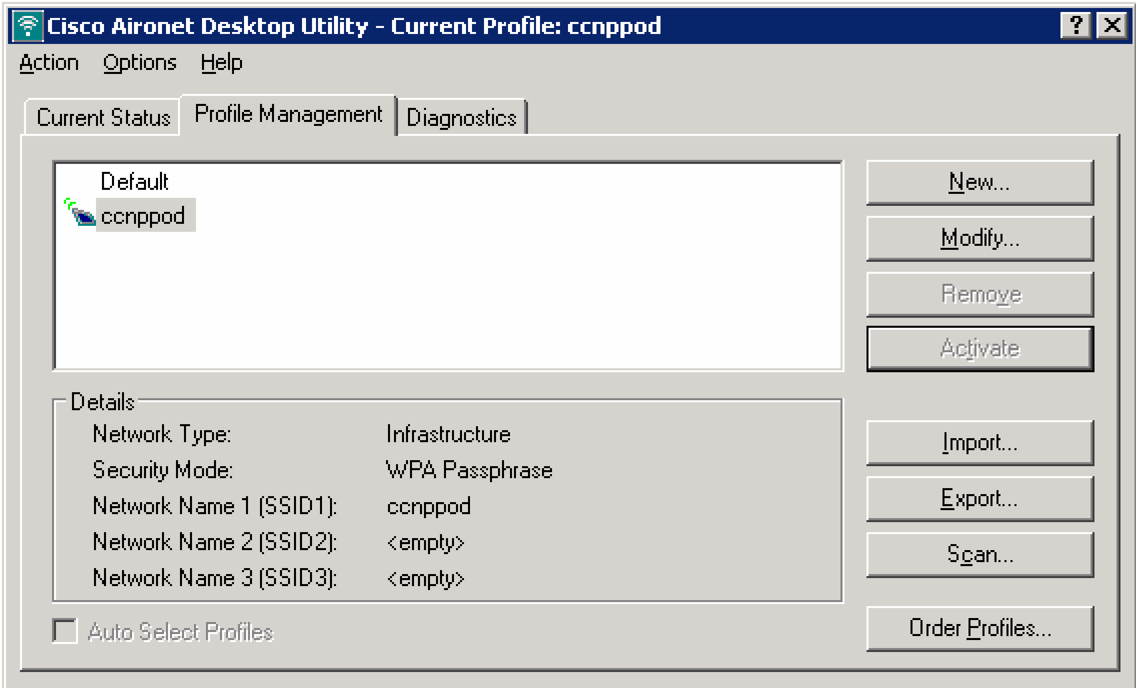


Figure 3-5: Selecting a Wireless Profile

Click the **Current Status** tab and make sure that you have received an IP address in the correct subnet. If you receive a correct IP, you have successfully configured and connected to the WLAN.

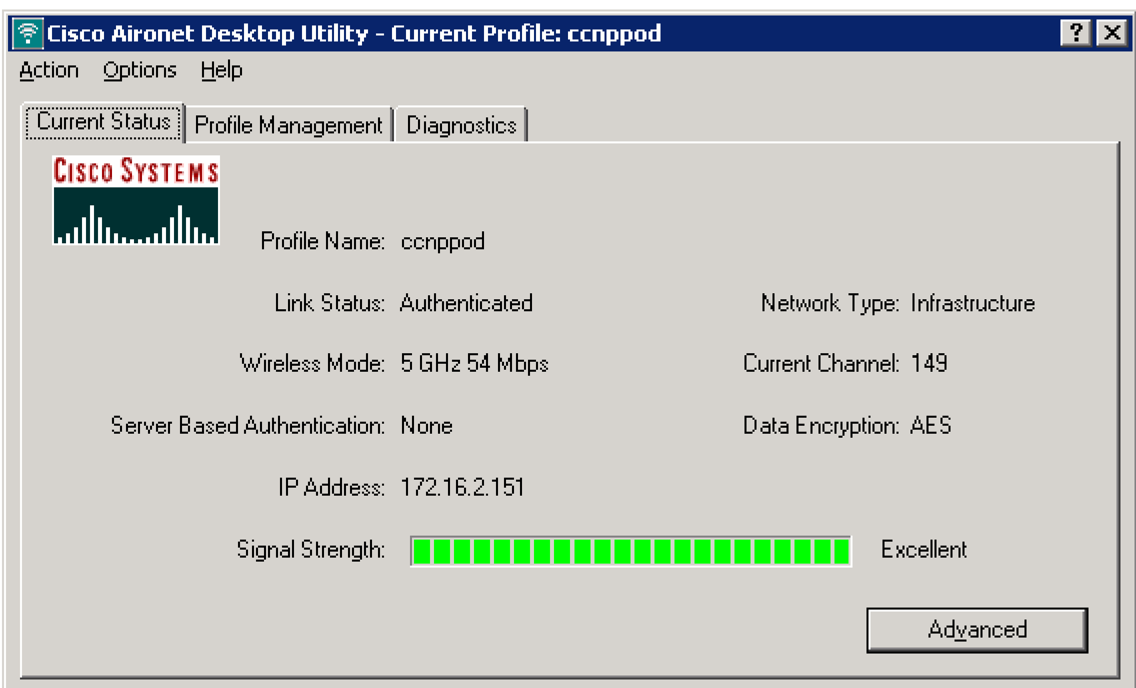


Figure 3-6: Current Wireless Profile Status

Lab 6.5 Configuring LEAP

Learning Objectives

- Install the Cisco Secure ACS server on a Windows host PC
- Configure a RADIUS server
- Configure a WLAN to use the 802.1X security protocol and LEAP
- Authenticate with an access point using 802.1X security and LEAP

Topology Diagram

Select the appropriate diagram based upon whether you have external or internal WLAN controllers:

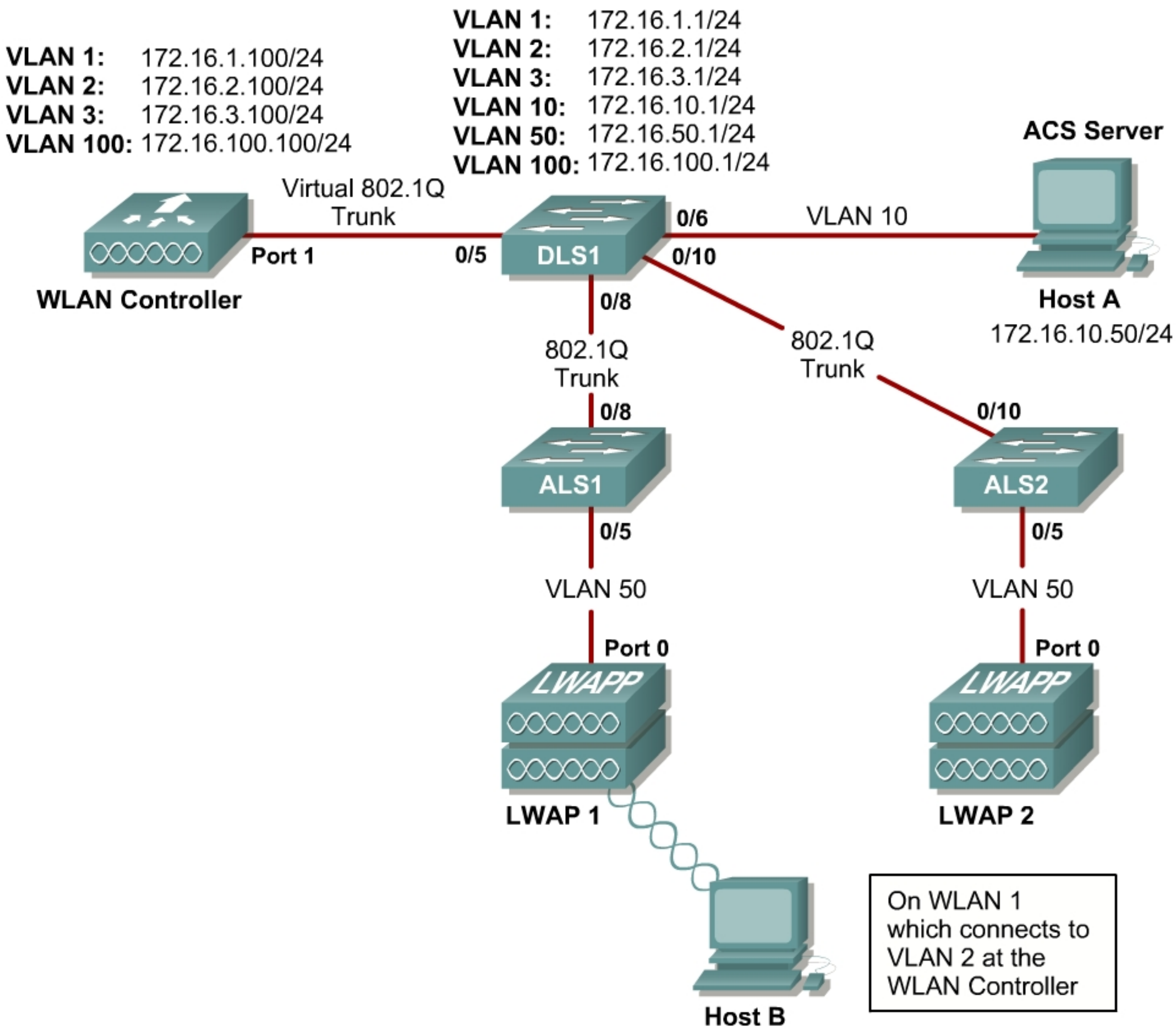


Figure 1-1: Ethernet Connectivity Diagram for Module 6, External WLAN Controller

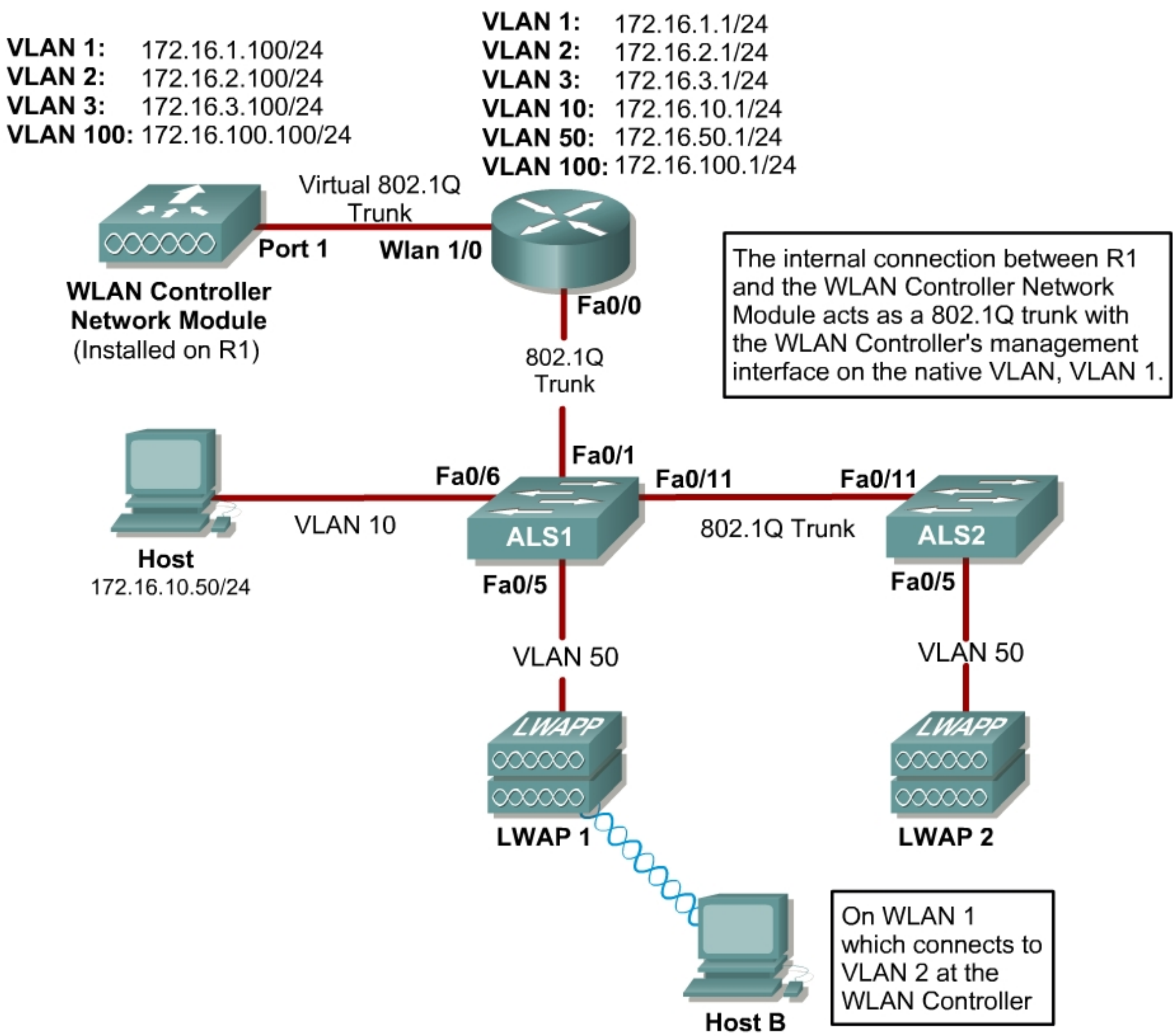


Figure 1-2: Ethernet Connectivity Diagram for Module 6, Internal WLAN Controller

Scenario

In this lab, you will configure and verify 802.1X security in a wireless environment. The 802.1X authentication protocol is built on the Extensible Authentication Protocol (EAP) and the RADIUS authentication protocol and provides per-client authentication and network admission.

This lab requires two separate PCs, Host A and Host B. Host A will act on VLAN 10 as the Cisco access control server (ACS) and will also be used to configure the wireless LAN (WLAN) controller the way a PC has been used to do in previous labs. Host B requires a Cisco wireless network card with the Aironet Desktop Utility installed. Host B will function as a wireless client on WLAN 1 which corresponds to VLAN 2.

You may complete this scenario using either the external wireless LAN controller (WLC) or the network module that resides in a router. However, you must load the final configurations from the end of Lab 6.1: Configuring a WLAN Controller.

We highly recommend that you complete Labs 6.1, 6.2, and 6.3 before attempting this lab.

Note:

This lab will only go into the details of configuring the 802.1X security protocol. For more information on using the web interface of the WLC, consult Lab 6.2: Configuring a WLAN Controller via the Web Interface.

Preparation

Complete Lab 6.1 and ensure that all switches and routers, the WLAN controller, and the host are configured the way they would be at the end of Lab 6.1.

At the end of Lab 6.1, you should already have the following features configured and verified:

- VLAN connectivity
- Trunk ports
- HTTP access to the WLC
- Lightweight Access Points (LWAPs) associated with the controller

Step 1: Install Cisco Secure ACS

If you have already installed Cisco Secure ACS on Host A, skip this step.

This step will guide you through installing the 90-day trial version of Cisco Secure ACS on Host A. After you download the trial to Host A and extract it, run Setup.exe. The installer will start.

Note: At the time of this writing, Cisco Secure ACS will only install and run on Microsoft Windows Server Editions. You will not be able to run the CiscoSecure ACS on Microsoft Windows XP.

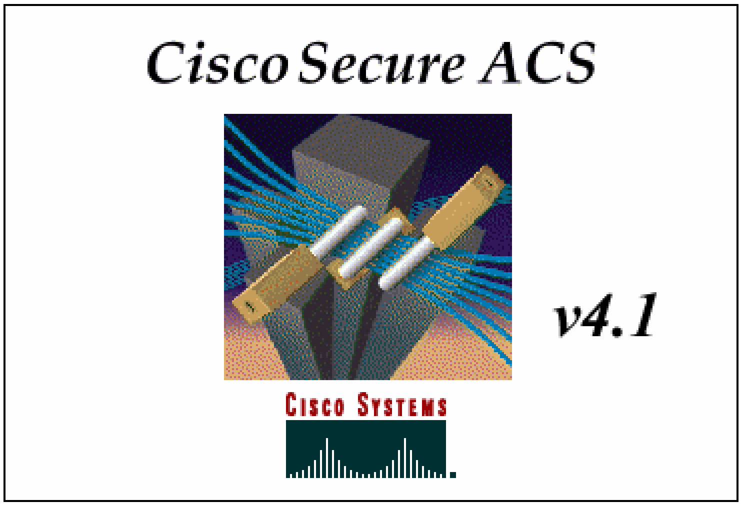


Figure 1-1: CiscoSecure ACS Splash Screen

After reading the terms of the license agreement, click **ACCEPT** to accept them.

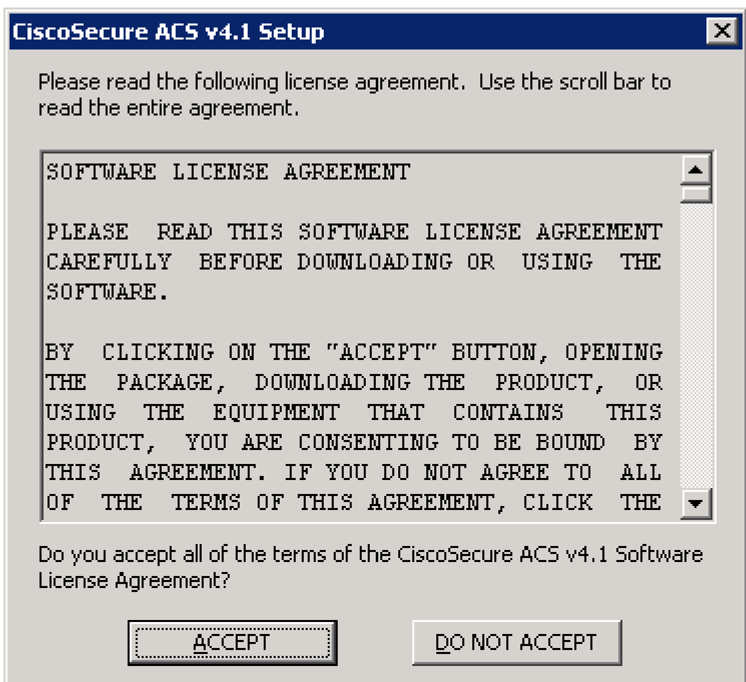


Figure 1-2: CiscoSecure ACS License Agreement

Click **Next** to continue the installation process.

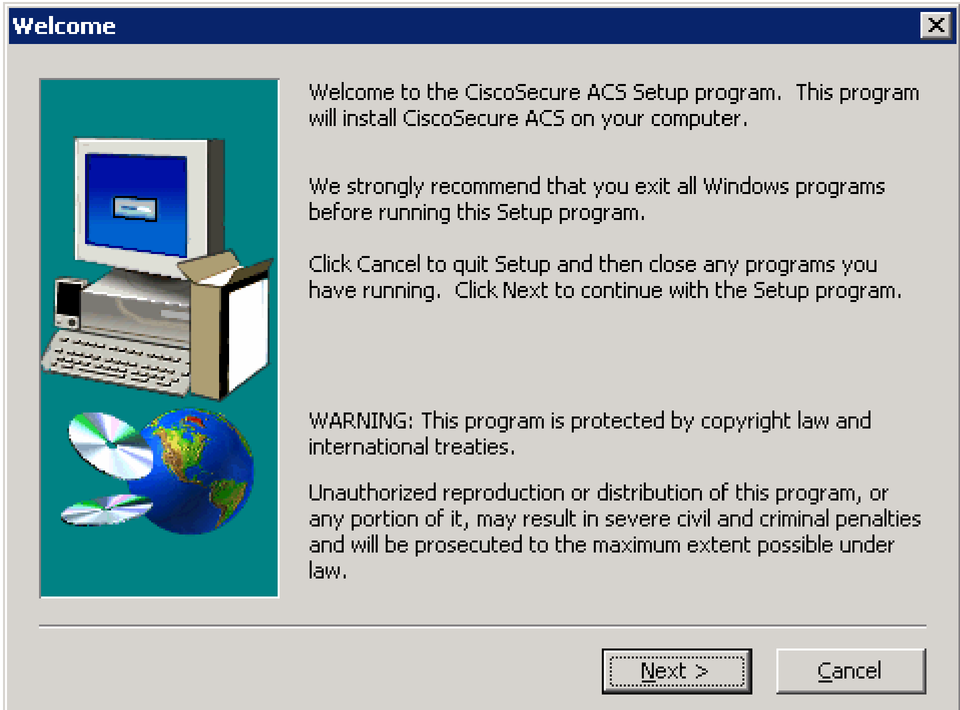


Figure 1-3: CiscoSecure ACS Installation Wizard

Verify that all of the requirements in the checklist are satisfied and check all of the options before clicking **Next** again.

Figure 1-4: CiscoSecure ACS Pre-Installation Checklist

Use the default installation folder and click **Next**.

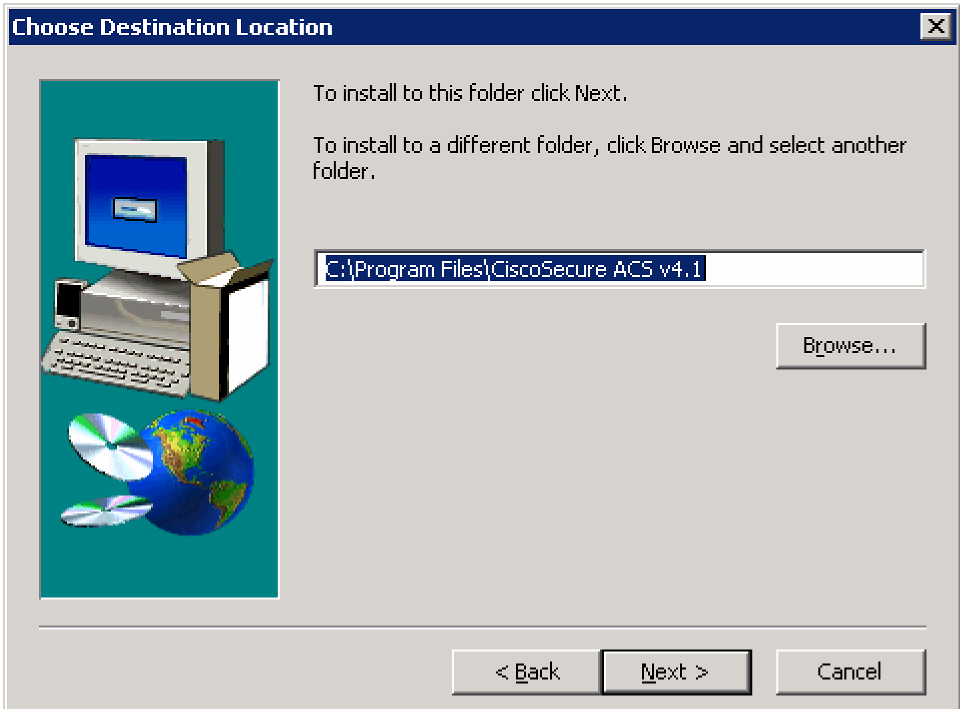


Figure 1-5: CiscoSecure ACS Installation Location

CiscoSecure has the ability to authenticate against the Windows User Database. However, for this lab, choose to only authenticate against the internal database. Click **Next**.

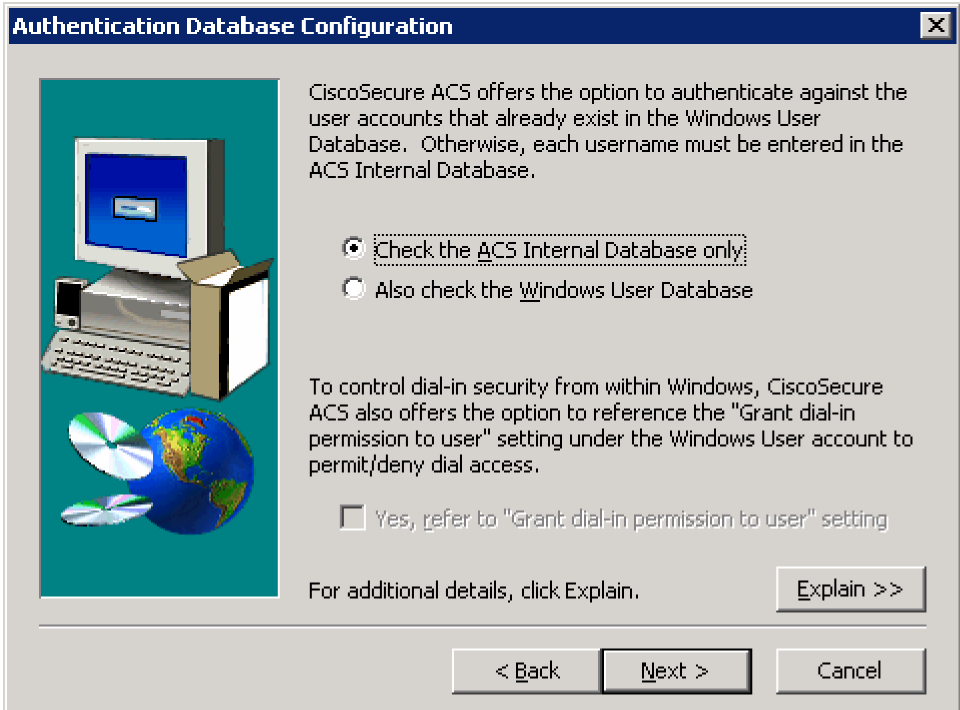


Figure 1-6: CiscoSecure ACS Authentication Database Options

The installer will then begin copying files and registry keys. This process may take a few minutes.

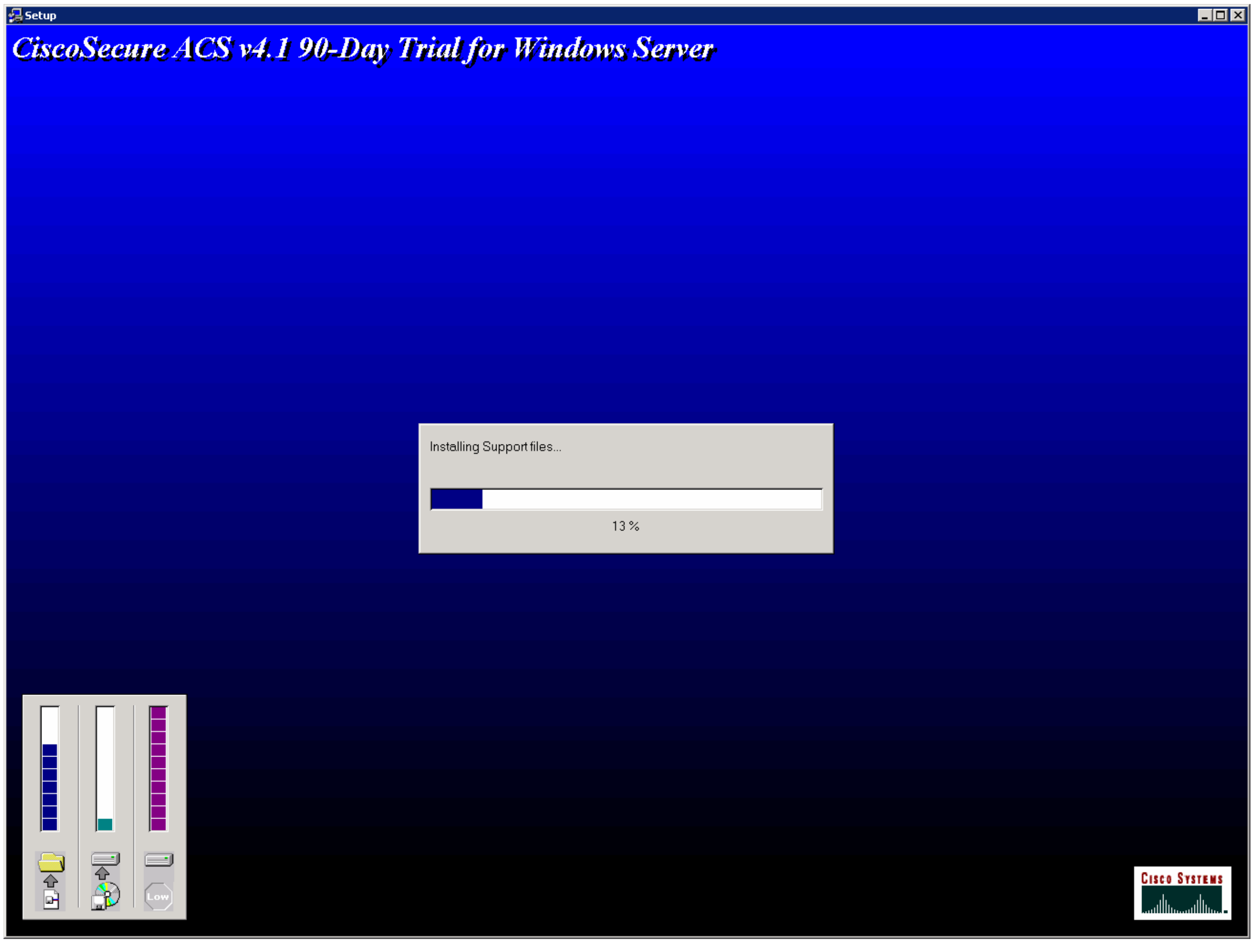


Figure 1-7: CiscoSecure ACS Installation Progress Indicator

At the end of the installation, you will be prompted to indicate if you want to see any advanced configuration options in the user interface. You do not need to check any of these. Click **Next** after reviewing the options.

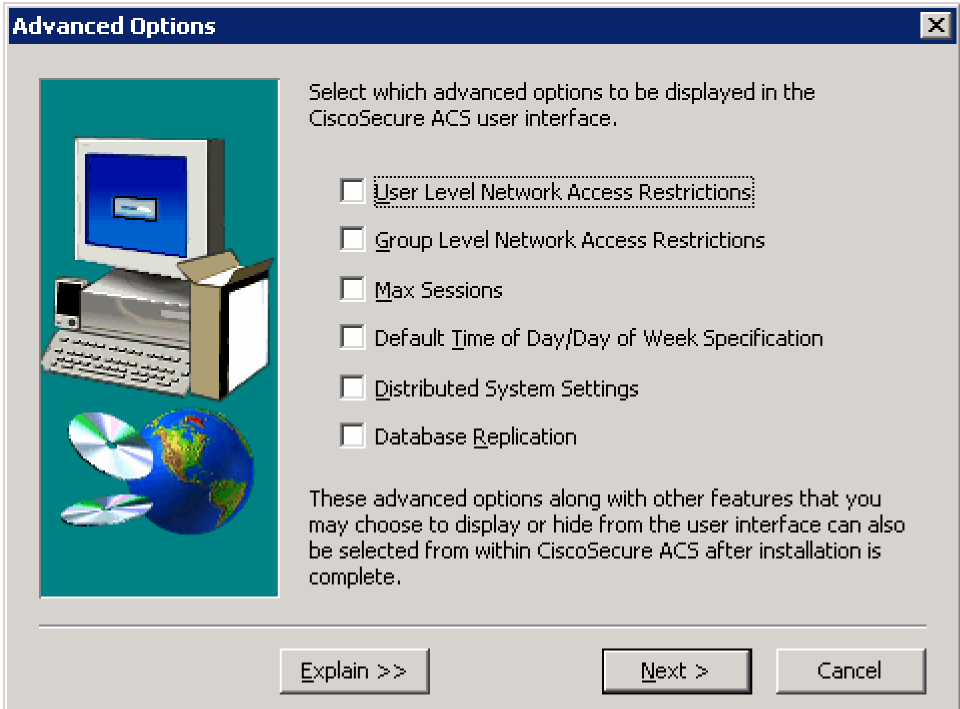


Figure 1-8: CiscoSecure ACS Advanced Configuration Options

Use the default settings in the next step of the installation wizard as well and click **Next**.

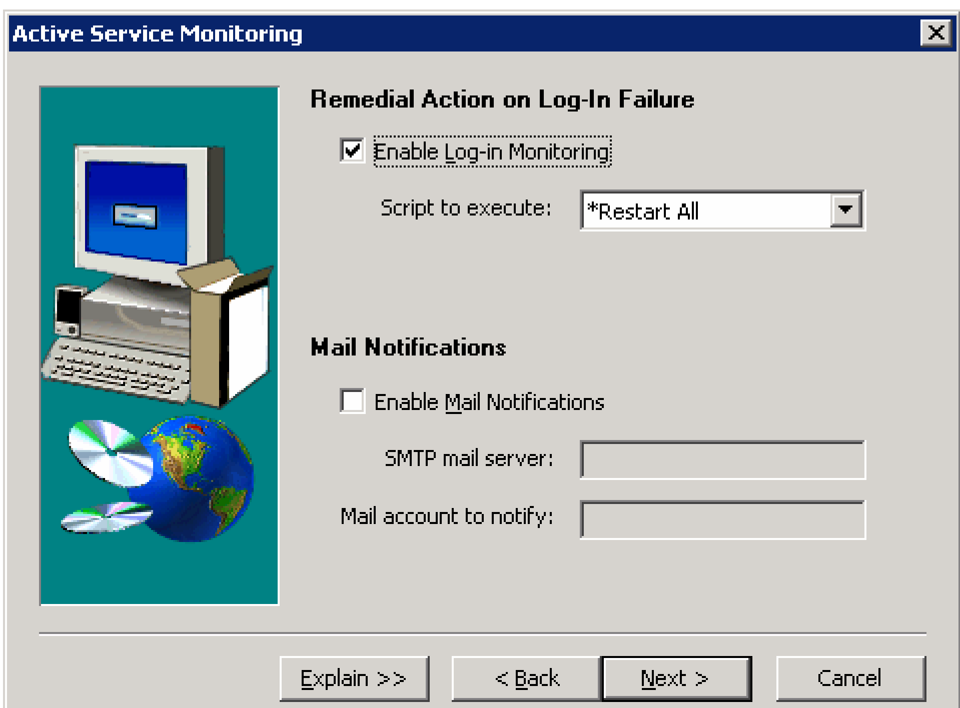


Figure 1-9: CiscoSecure ACS Log-In

You must create a password for ACS internal database encryption. It must be at least eight characters long and contain both letters and numbers. In the example below, “ciscoacs4” was used as a password. After configuring the password, click **Next**.

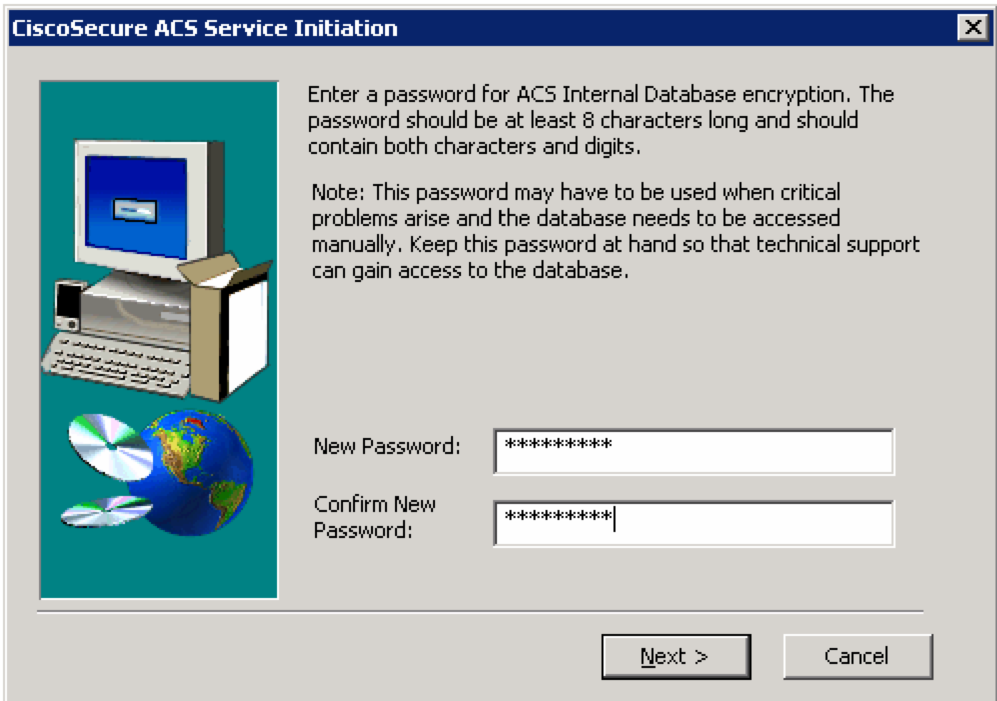


Figure 1-9: CiscoSecure ACS Password Configuration

Choose to start the ACS service on the host now. You should also select the option to start the administration window after the installer ends to verify the installation. Click **Next** after selecting the correct options.

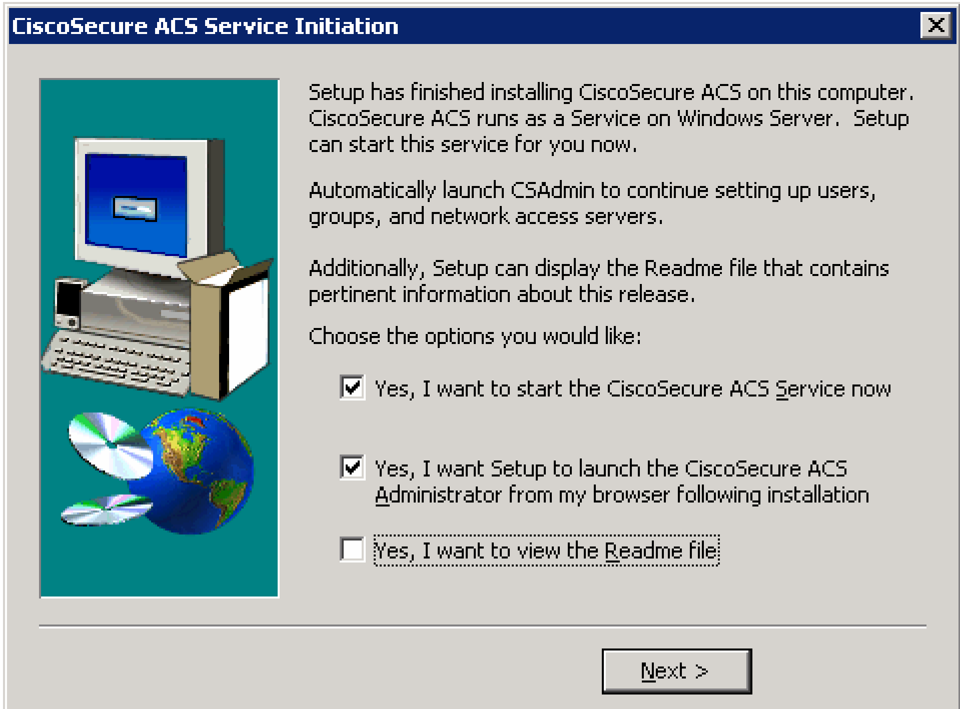


Figure 1-10: CiscoSecure ACS Service Configuration

Read the instructions and click **Finish**. You should also make sure your computer is compliant with all ACS access requirements, complying with the supported versions of Internet Explorer and the Java Runtime Environment.

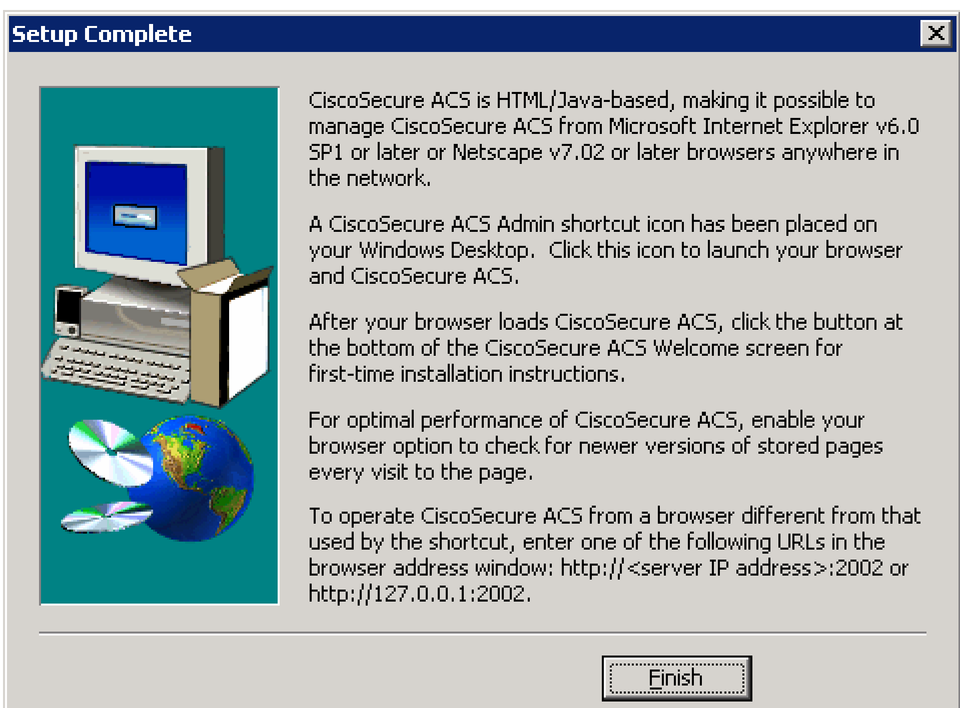


Figure 1-11: CiscoSecure ACS Installation Complete Window

If the Cisco Secure ACS administrative screen comes up when the installer ends, this signals that ACS was successfully installed.

Step 2: Set up ACS for LEAP

If you don't have the Cisco Secure ACS application open on Host A from the previous step, open it now by clicking the **Start** button and choosing **Programs > CiscoSecure ACS v4.1 Trial > ACS Admin**.

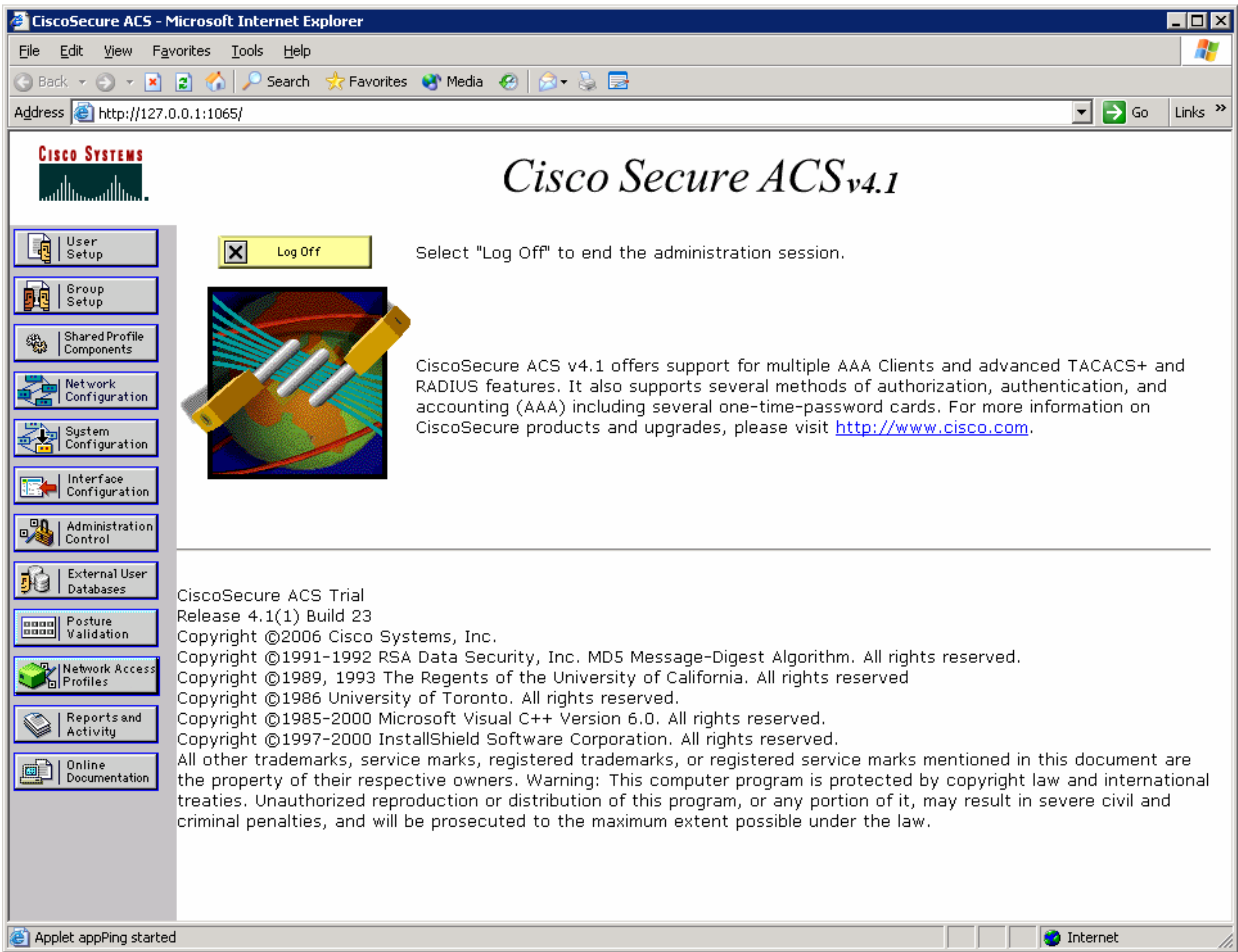


Figure 2-1: ACS Home Page

In the left pane, click **Network Configuration**. On the Network Configuration screen, you can configure authentication, authorization, accounting (AAA) clients directly. Click the **Add Entry** button under the heading **AAA Clients**.

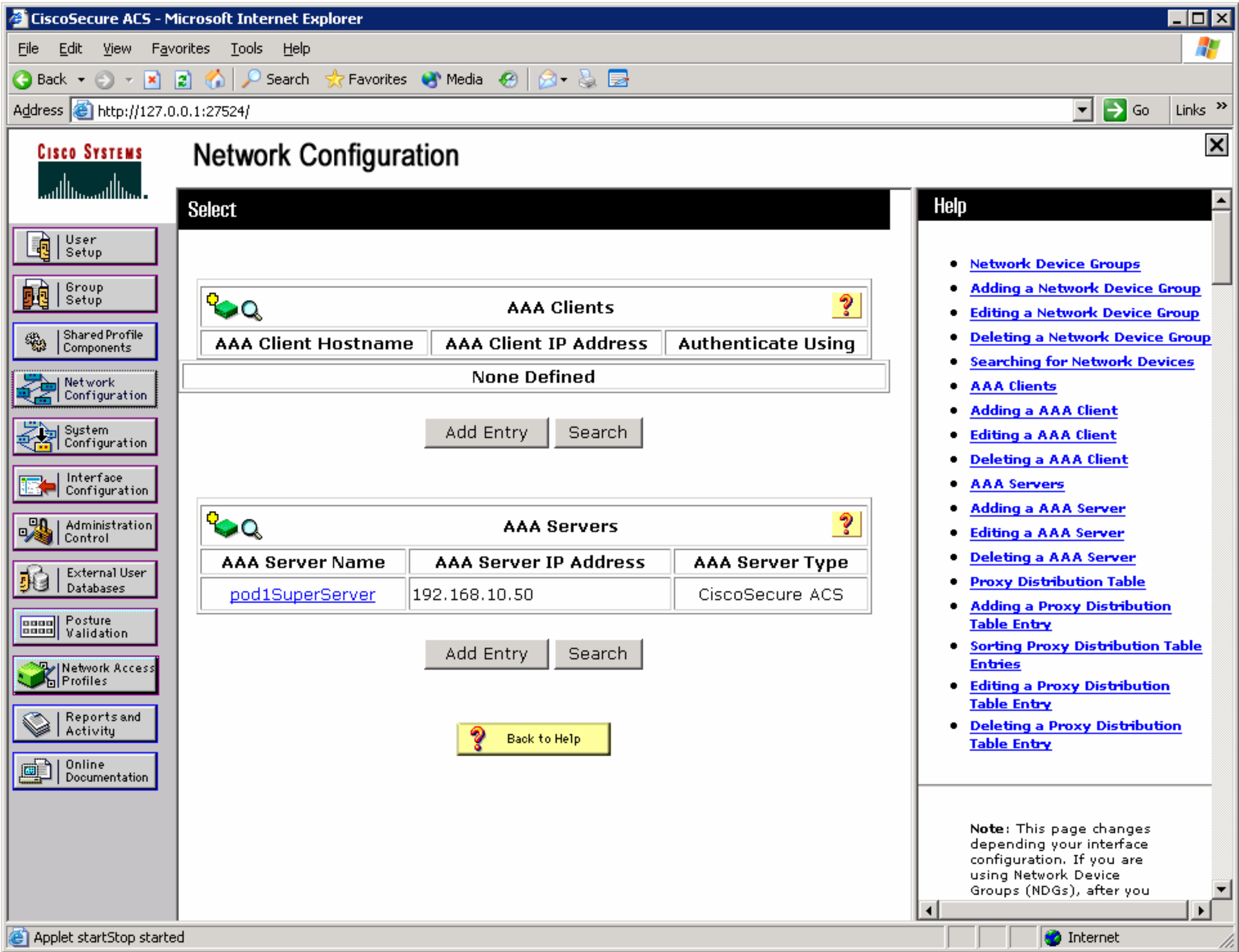


Figure 2-2: ACS Network Configuration Page

Enter the hostname of the WLC (you can get this from **show run-config** on the WLC command-line interface [CLI] or from its web interface), the management IP address of the WLC, and “cisco” as the shared secret. Change the value of the **Authenticate using:** field to **RADIUS (Cisco Airespace)**. After you have entered in everything, click **Submit + Apply**.

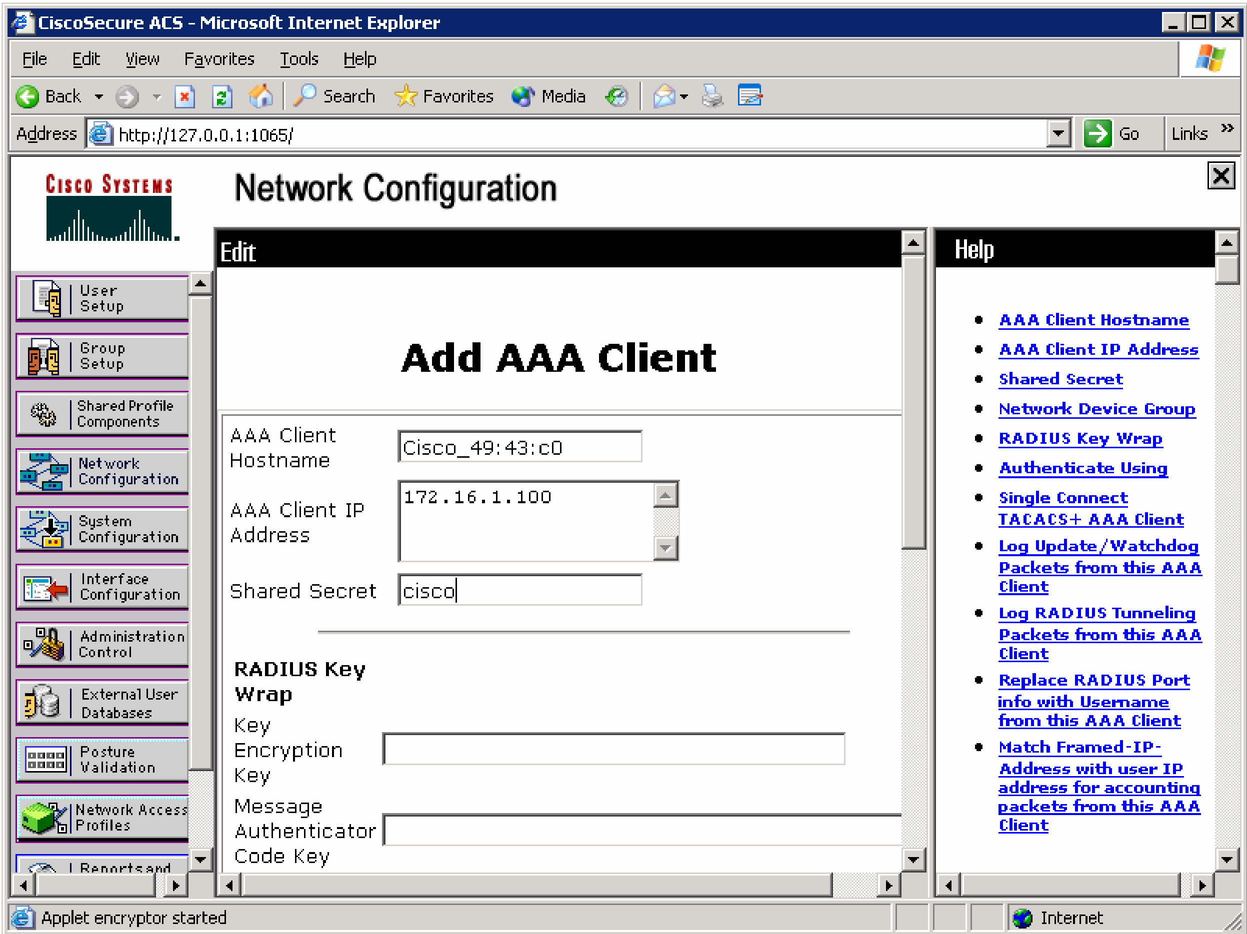


Figure 2-3: ACS AAA Client Configuration

You should now be able to see the WLC listed as an AAA client on the network configuration screen.

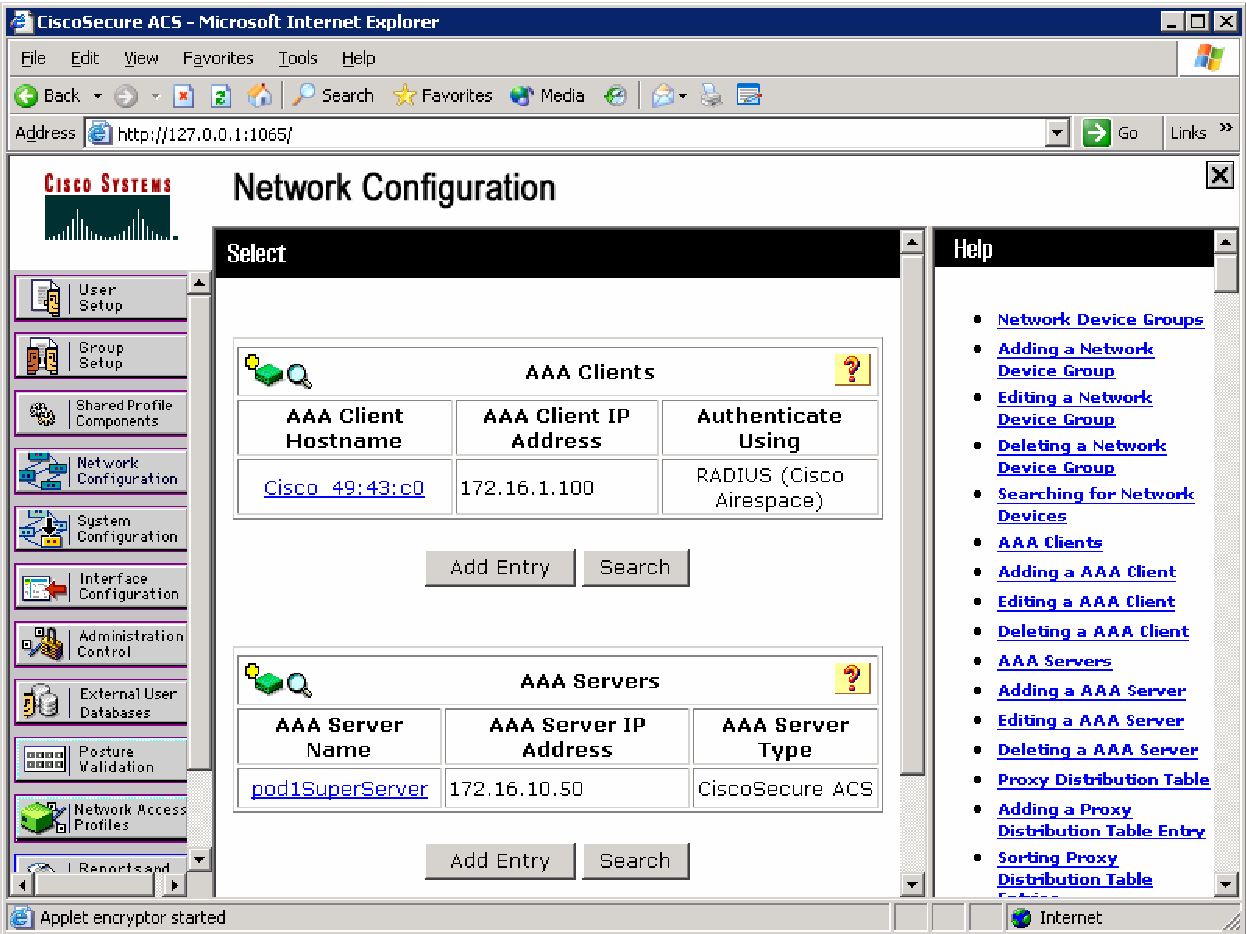


Figure 2-4: ACS Network Configuration Page, with Changes Applied

On the left pane, click **User Setup**. Add a user named “cisco,” and then click **Add/Edit**.

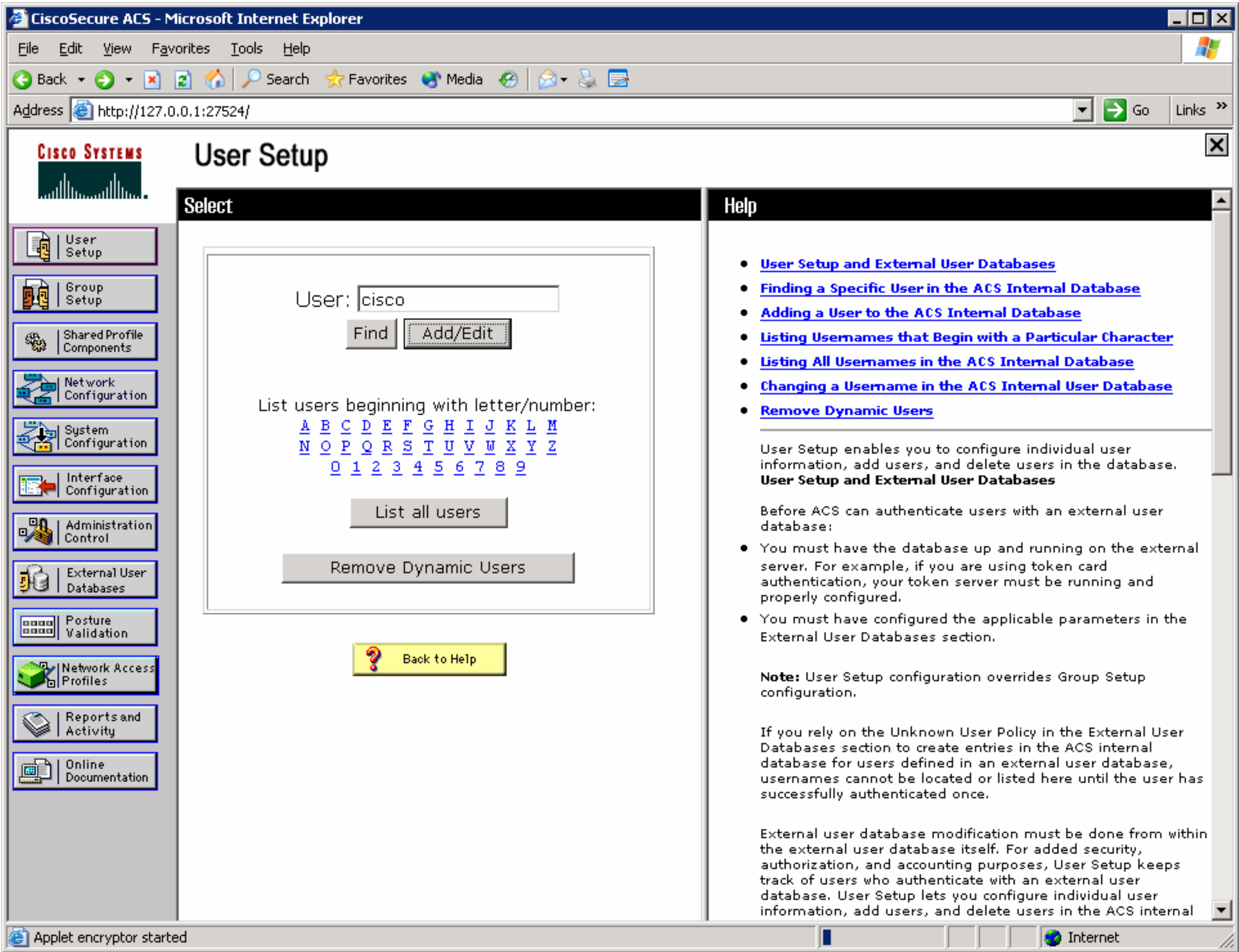


Figure 2-5: ACS User Configuration Page

Assign “cisco” as the user name, and set “cisco” as the password. Click **Submit**.

Why is the shared secret configured on a per-client basis?

You should see the WLC listed in the network configuration screen.

On the left pane, click **User Setup**. Type in “cisco” in the user field (this will be the name of the user we are creating), and then click **Add/Edit**.

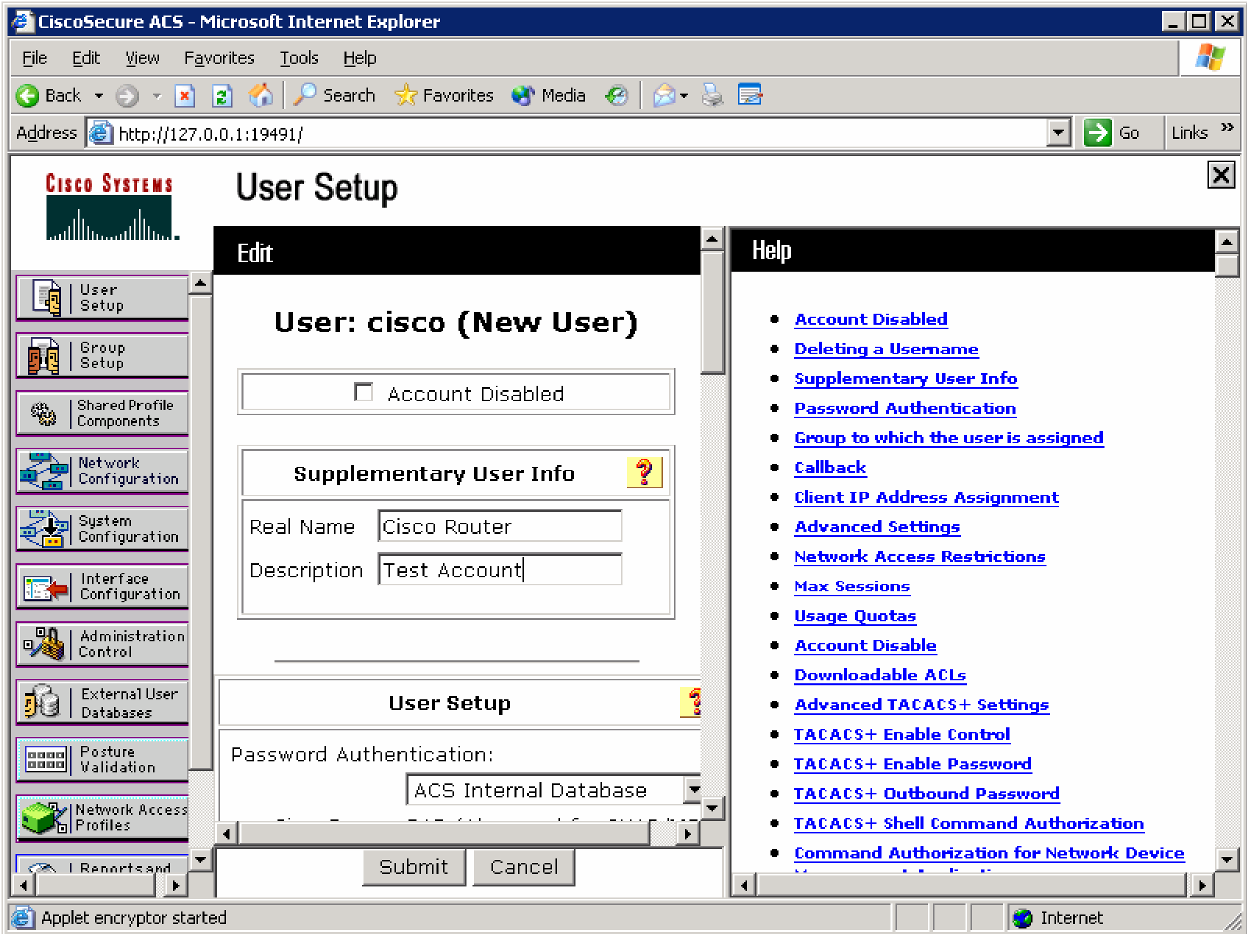


Figure 2-6: ACS User Configuration

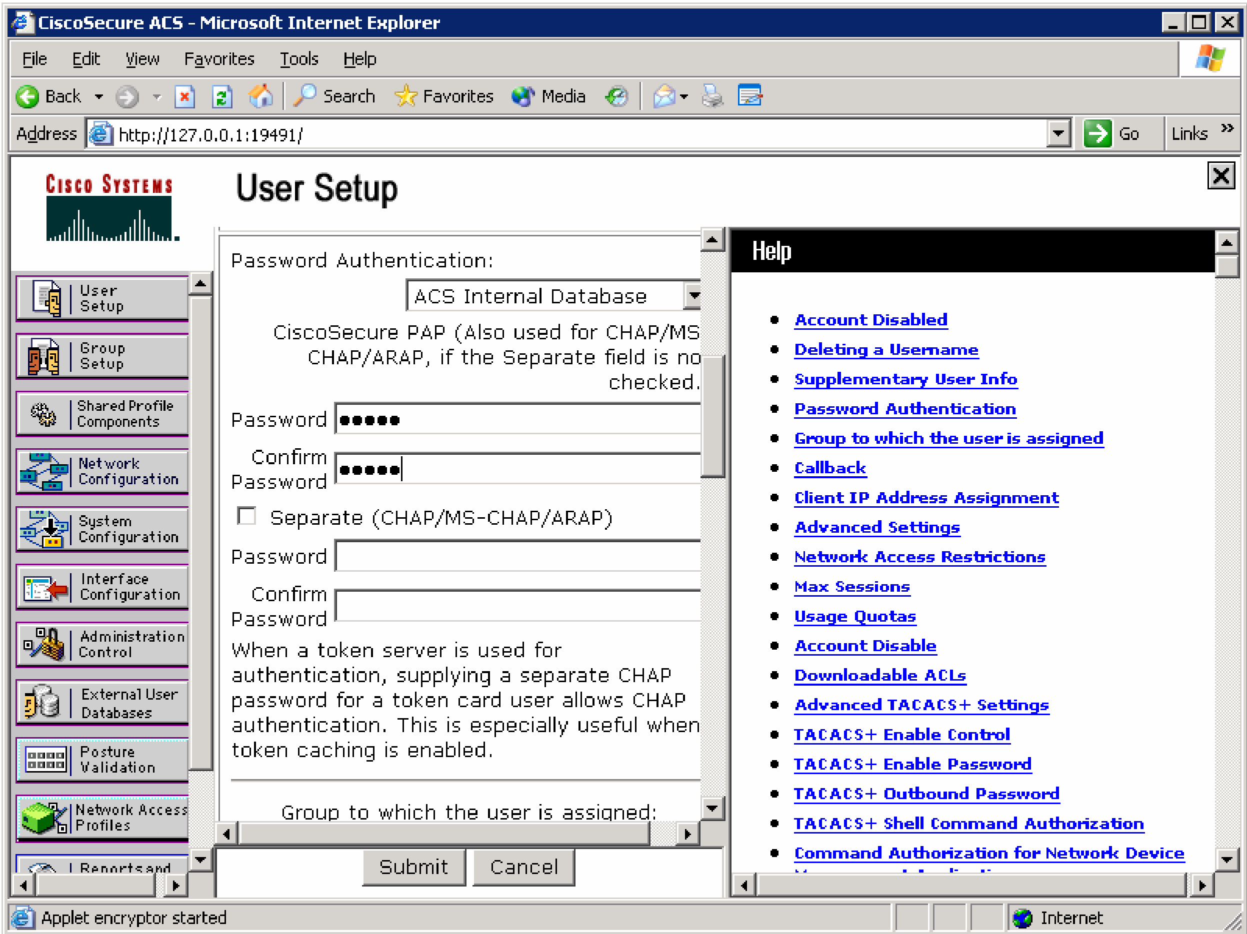


Figure 2-7: ACS User-level Password and Group Configuration

For what purpose will you use this user account?

Although it should be enabled by default, we will make sure that LEAP authentication is enabled in ACS.

Click **System Configuration** on the left pane.

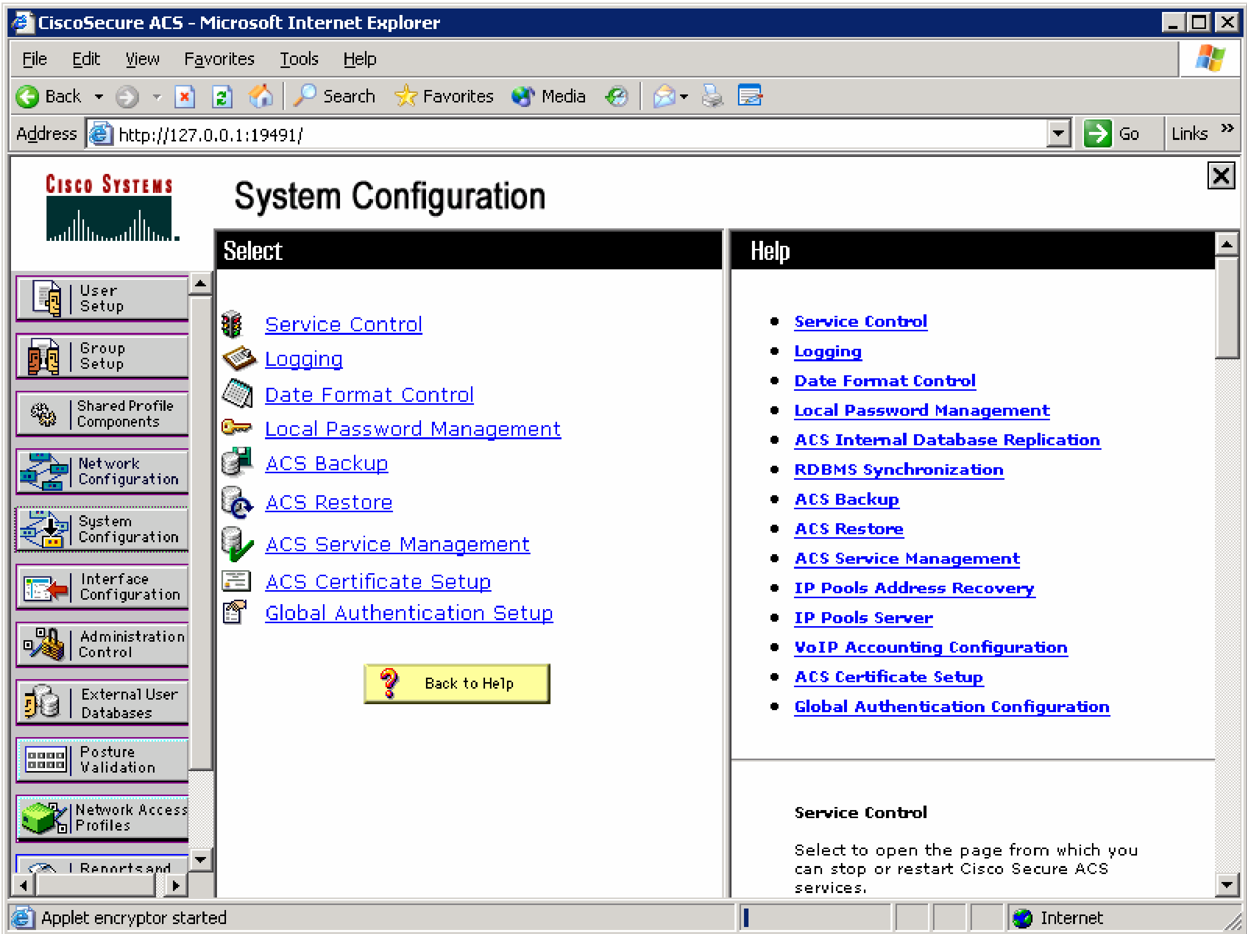


Figure 2-8: System Configuration Tab

Click **Global Authentication Setup** in the list of options. Scroll down and make sure that **Allow LEAP** is checked, as shown in Figure 2-9.

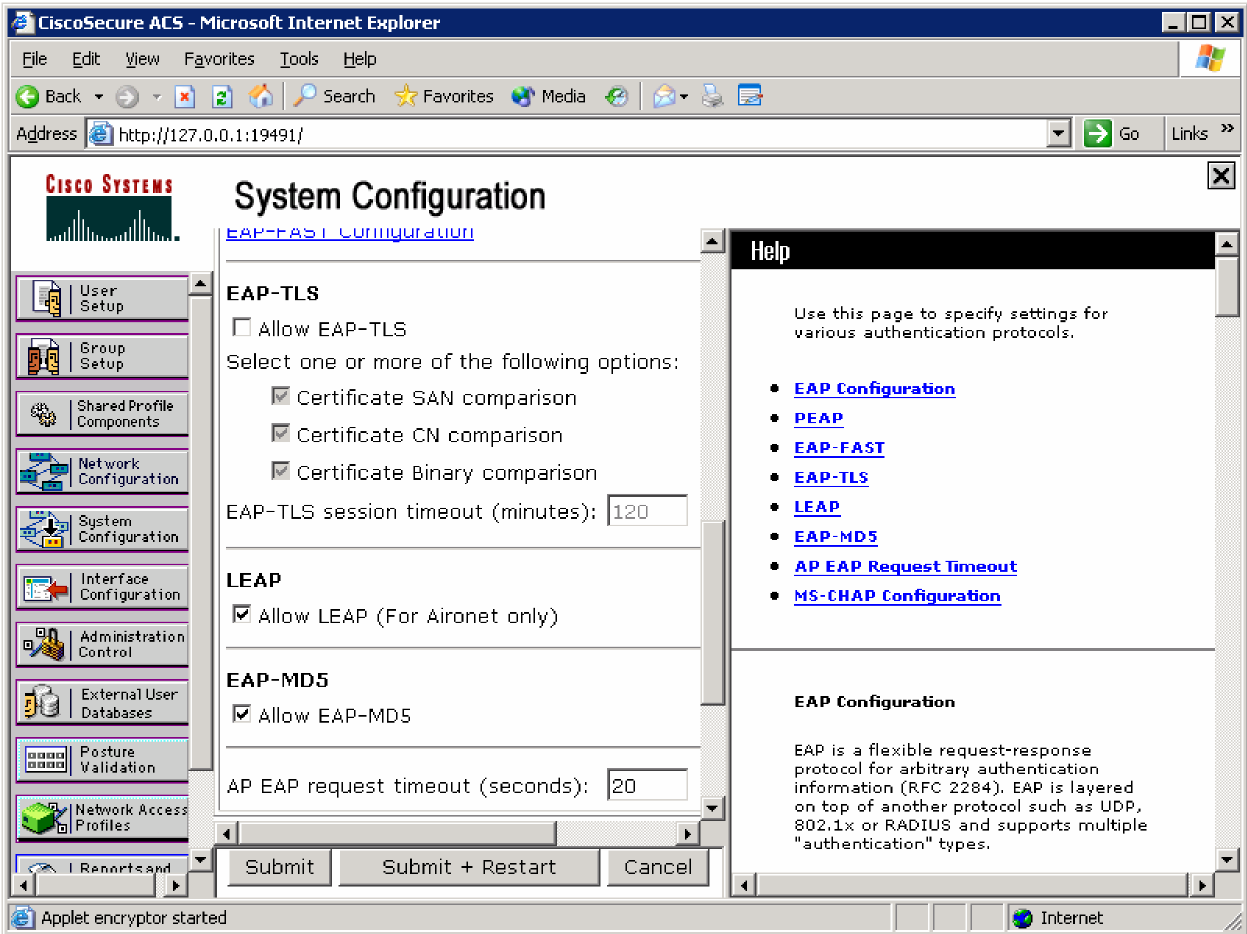


Figure 2-9: System Security Protocol Configuration

Step 3: Connect to the WLC from the Management Host

This lab will only go into the details of configuring WLAN security using 802.1X and RADIUS. For more information on using the web interface of the WLC, consult Lab 6.2: Configuring a WLAN Controller via the Web Interface.

On Host A, open up Internet Explorer, and go to the URL https://172.16.1.100. This is the secure method of connecting to the management interface of the WLAN controller. You can also use http://172.16.1.100 since we previously enabled regular insecure HTTP access in the CLI for Lab 6.1: Configuring a Wireless LAN Controller. If you connect to the secure address, you may be prompted with a security warning. Click **Yes** to accept it and you will be presented with the login screen for the WLAN controller. Click **Login** and an authentication dialog box will appear.

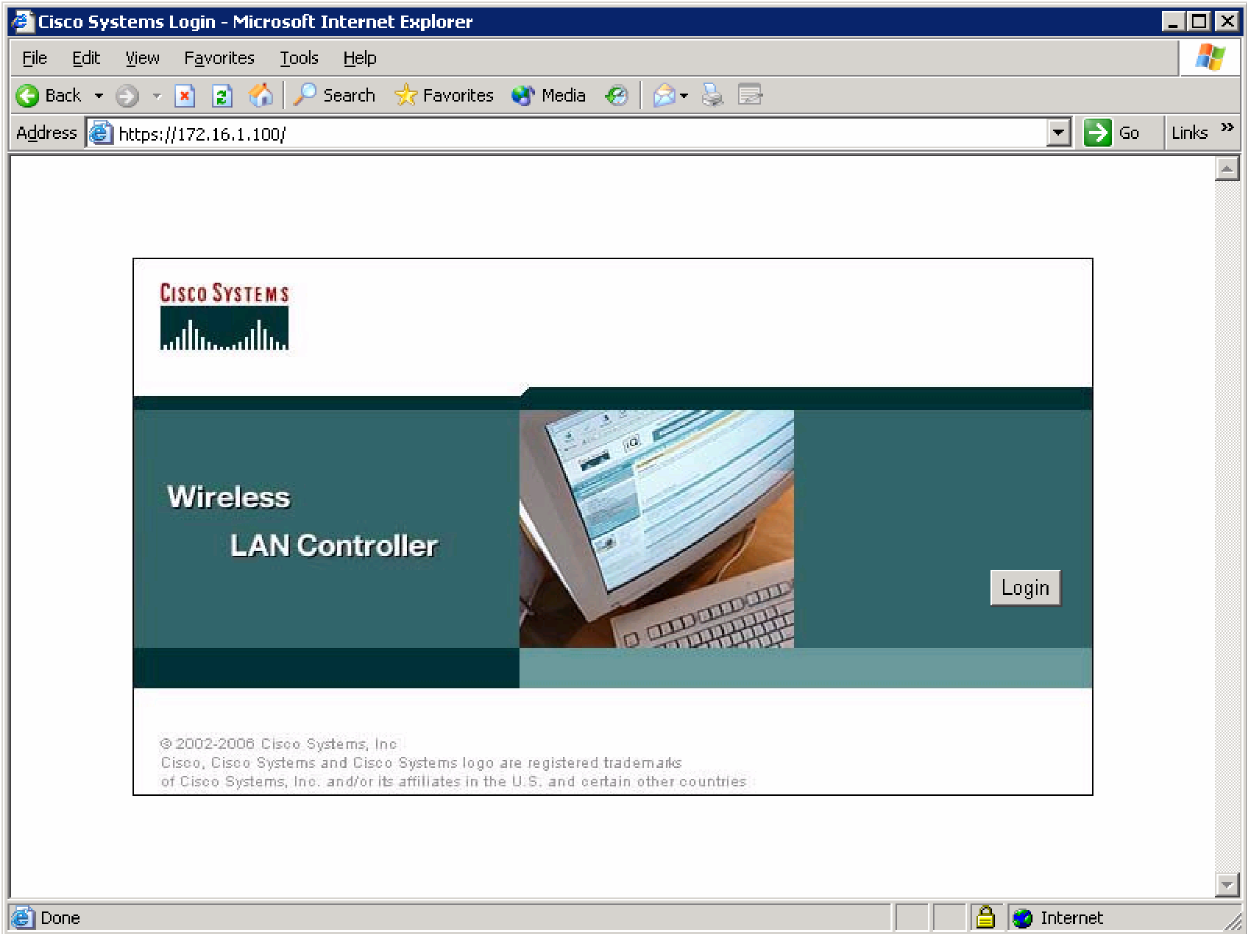


Figure 3-1: WLAN Controller Splash Screen

Use "cisco" as both the username and password. You configured these in the earlier lab. Click **OK** to get to the main page of the WLC web interface.

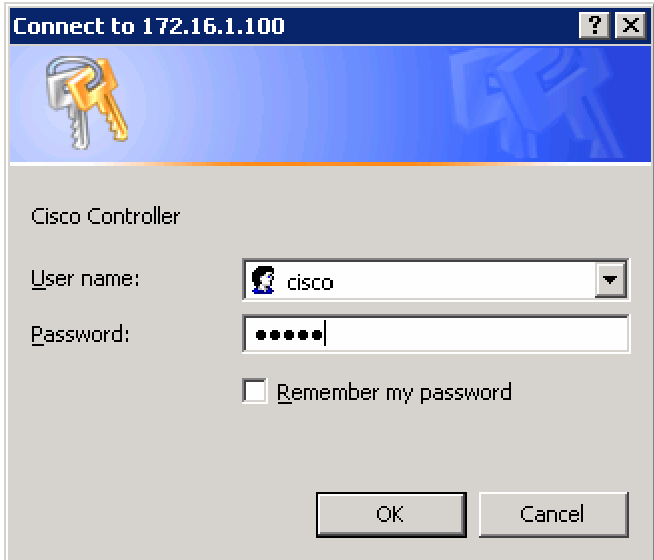


Figure 3-2: Authentication Dialog Box for HTTP Access to WLC

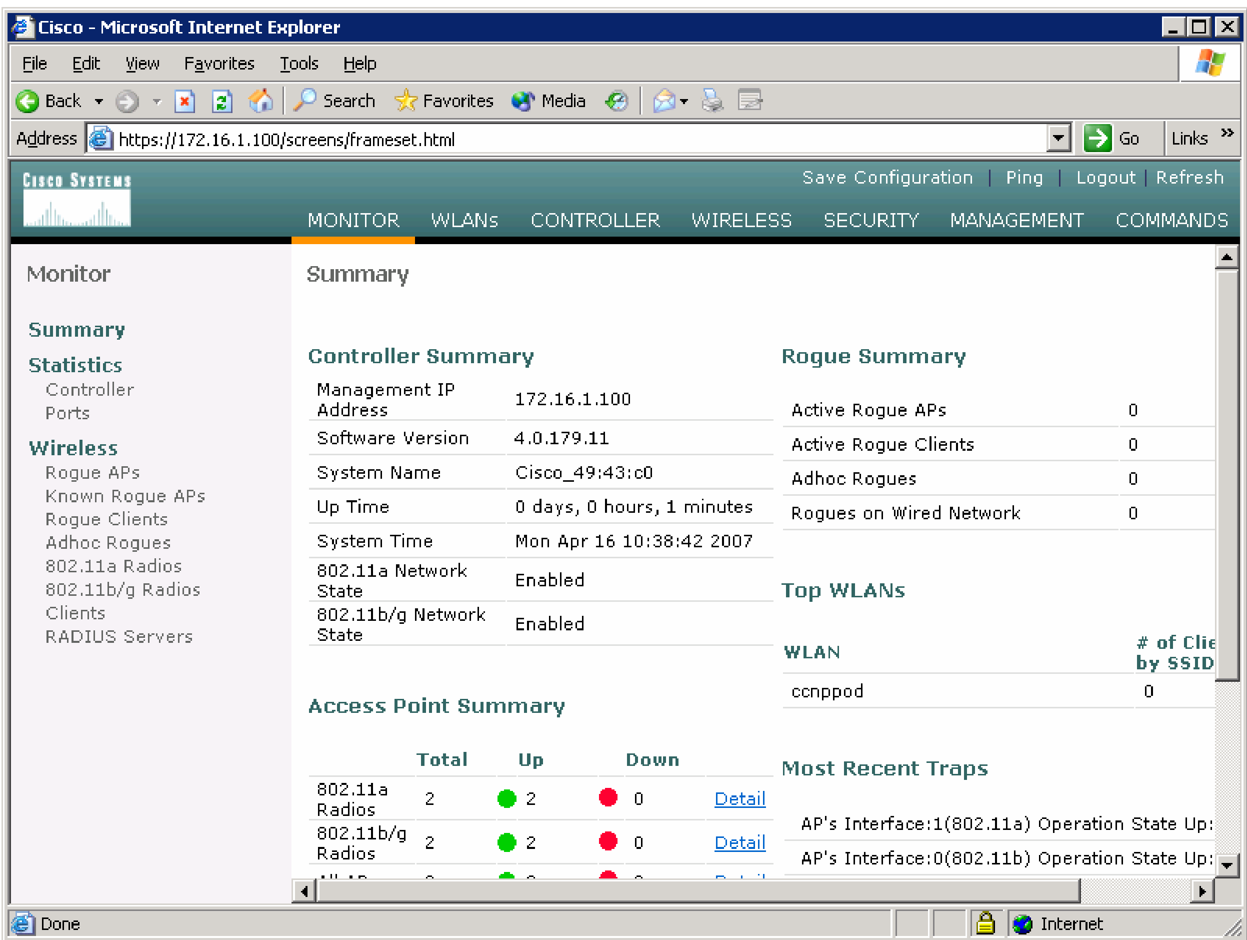


Figure 3-2: WLAN Controller Monitor Page

Make sure you see two access points under the "Access Point Summary" part of the page. If you don't, try reloading the LWAPs; otherwise, troubleshoot. You may also see it detecting rogue access points if your lab has other wireless networks around it; this behavior is normal. You can also see various port controller and port statistics by clicking their respective links on the left-hand menu on the screen.

Step 4: Set Up a RADIUS Server

In this step, we will set up a RADIUS server to be used for WLAN authentication. Click the **Security** link at the top of the WLC interface.

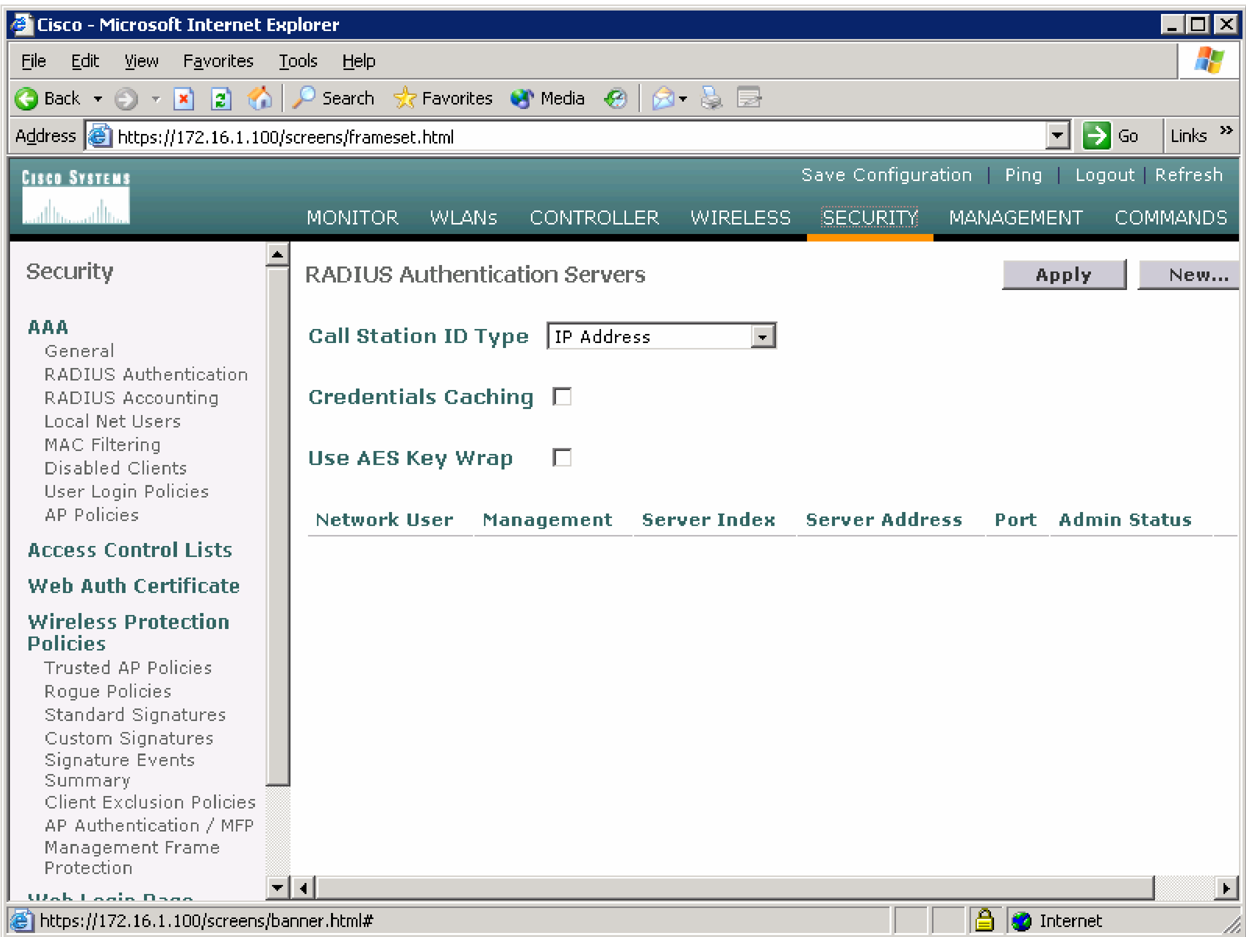


Figure 4-1: WLC RADIUS Server Configuration

Click **New** to add a new server. Set the IP address to the IP address of the server running ACS, and set the shared secret to "cisco" as configured on the ACS server for this device. Click **Apply** when done.

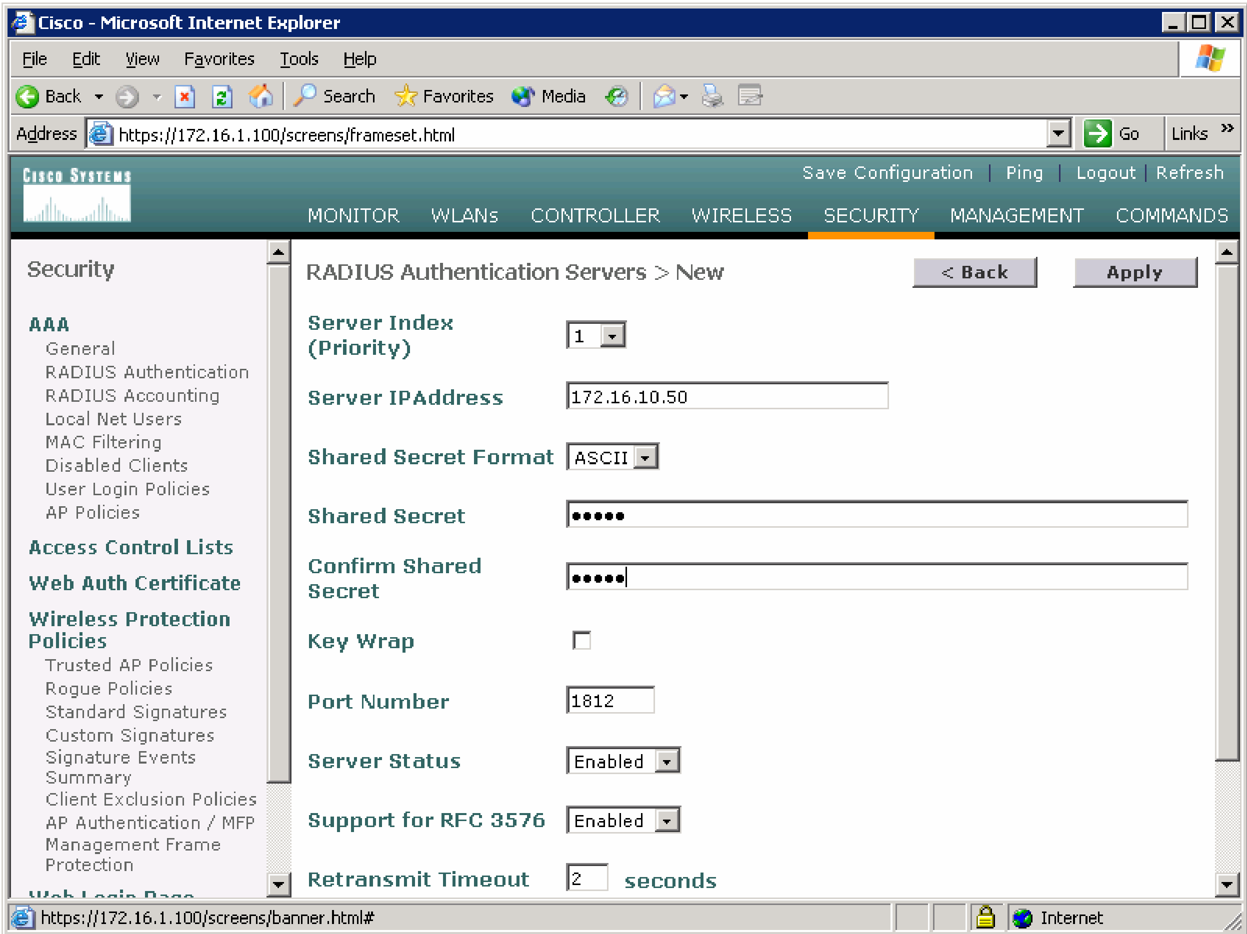


Figure 4-2: New RADIUS Server Configuration

You should see the new server added to the list.

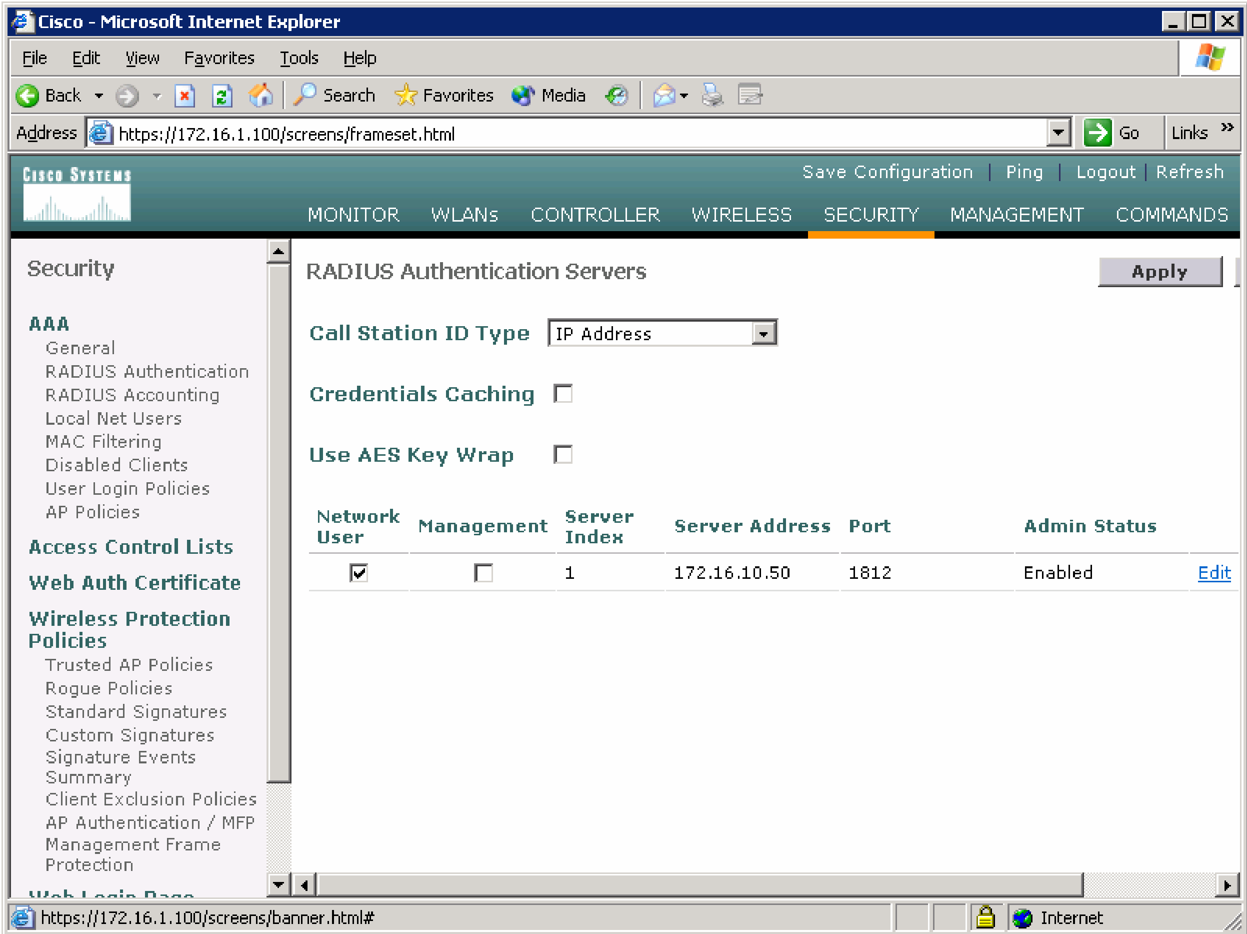


Figure 4-3: WLC RADIUS Server Configuration with Changes Applied

Step 5: Assign a WLAN to a VLAN

Click the **Controller** button at the top of the WLC interface. On the left pane, click **Interfaces** to see the current configured IP interfaces on the WLC. Click **New** to create a new interface.

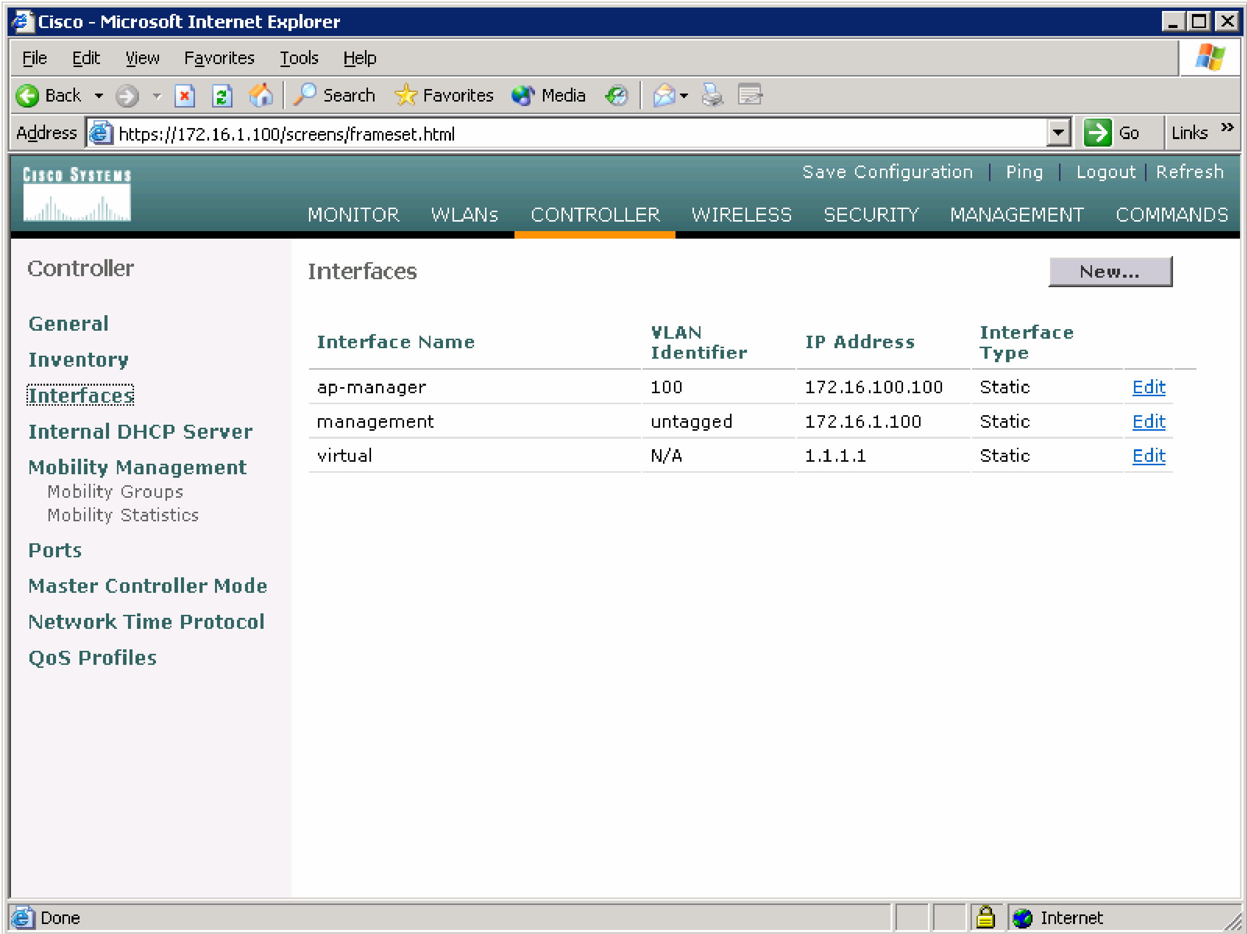


Figure 5-1: Interface Configuration Page

Name the interface “VLAN2” and assign it to 802.1Q tag 2, just like in Lab 6.2. Click **Apply** when you have completed this.

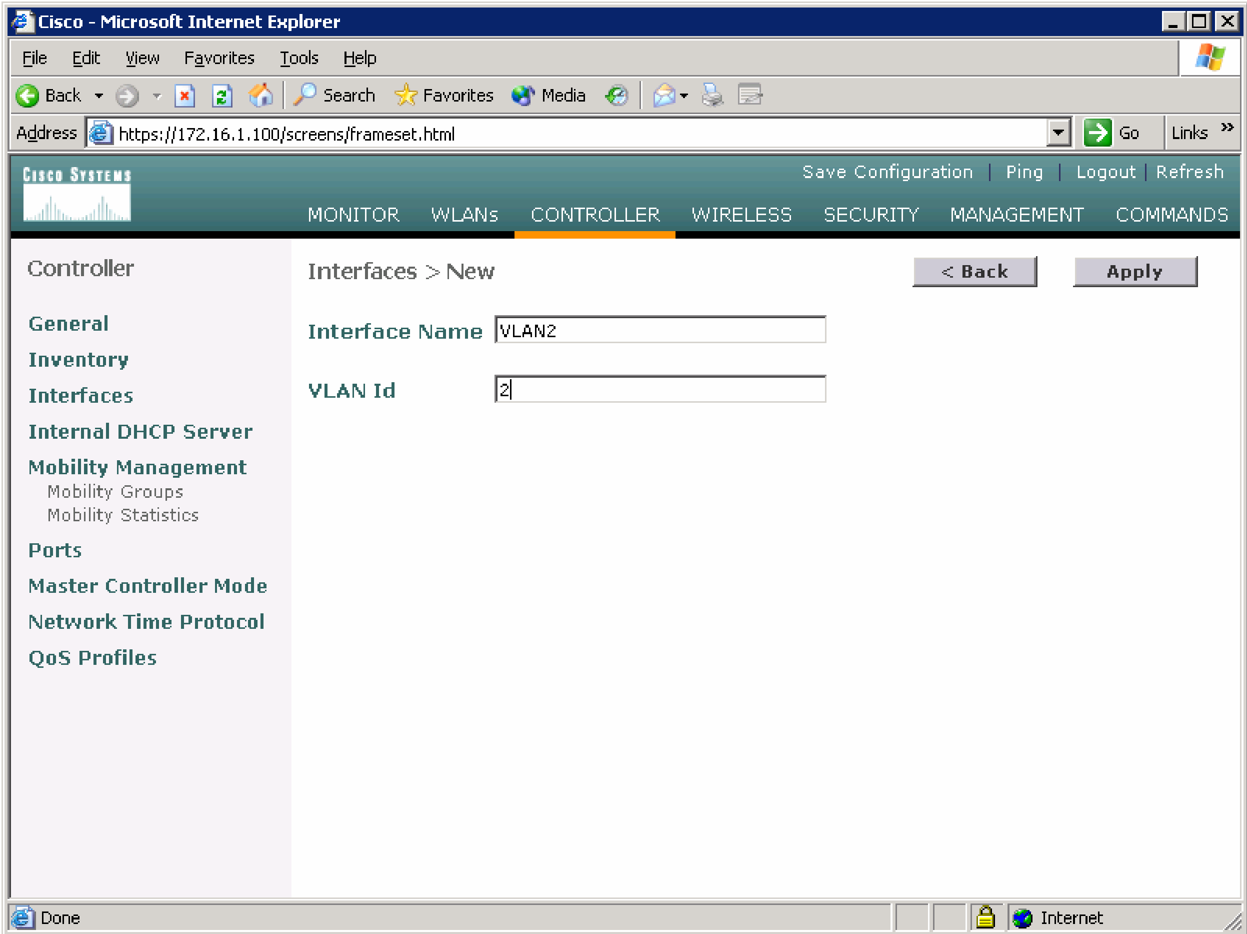


Figure 5-2: Creating a New VLAN Interface

Configure the IP address, default gateway, port number, and Dynamic Host Configuration Protocol (DHCP) server for this interface as shown in the following figure, and then click **Apply**. Accept the warning that comes up by clicking **OK**.

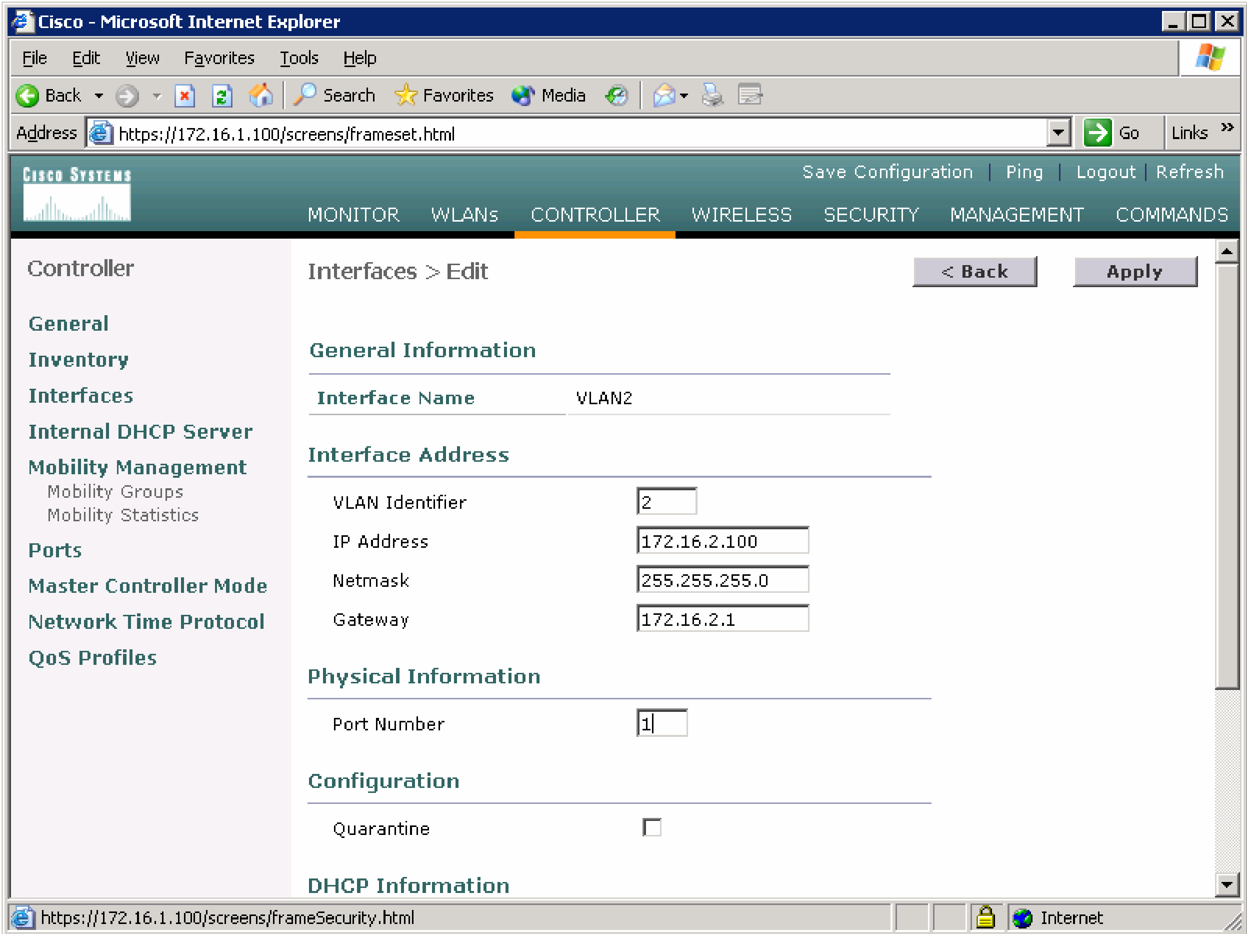


Figure 5-3: Configuring VLAN Interface Properties

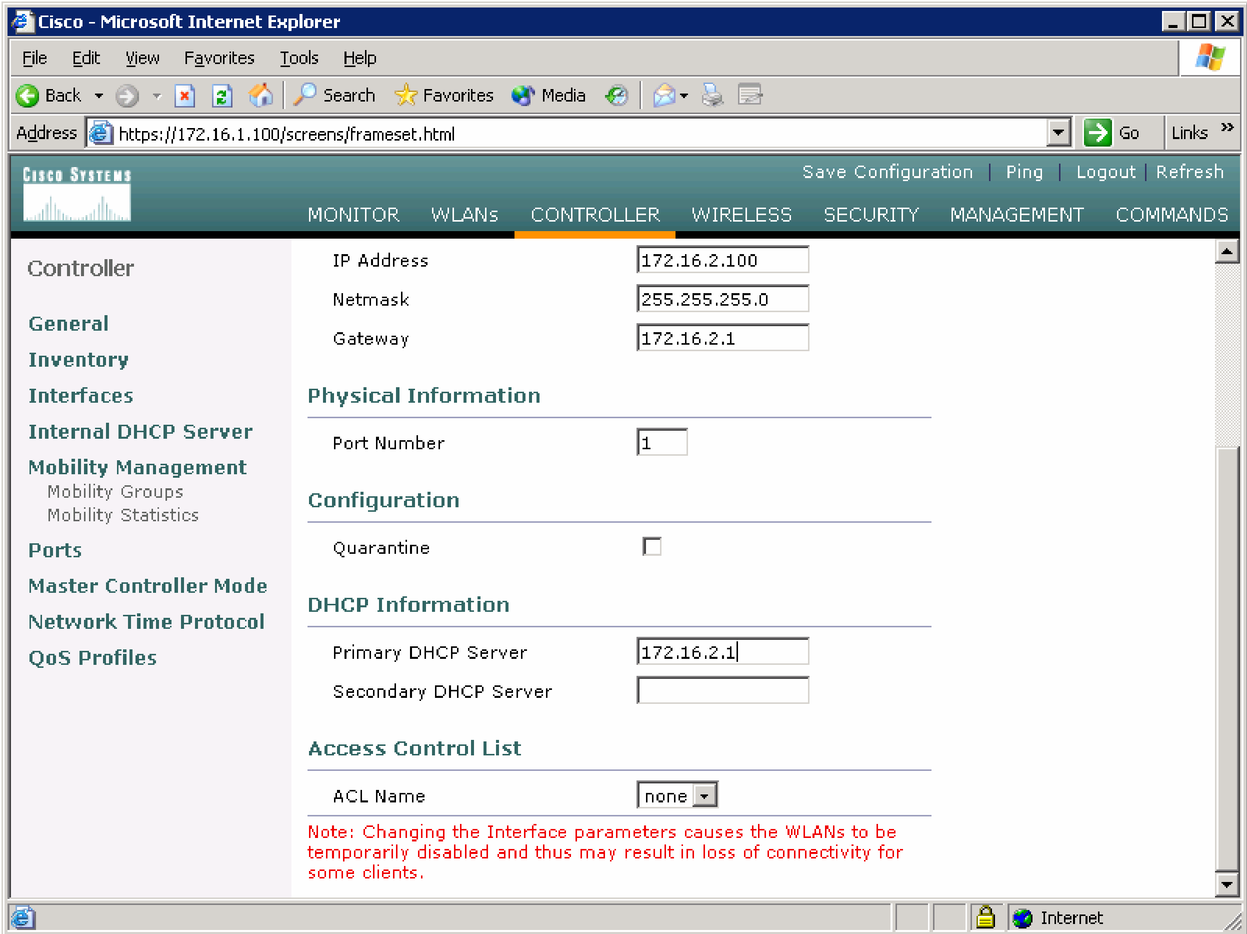


Figure 5-4: Configuring VLAN Interface Properties, DHCP Options

The new interface should appear in the interfaces list.

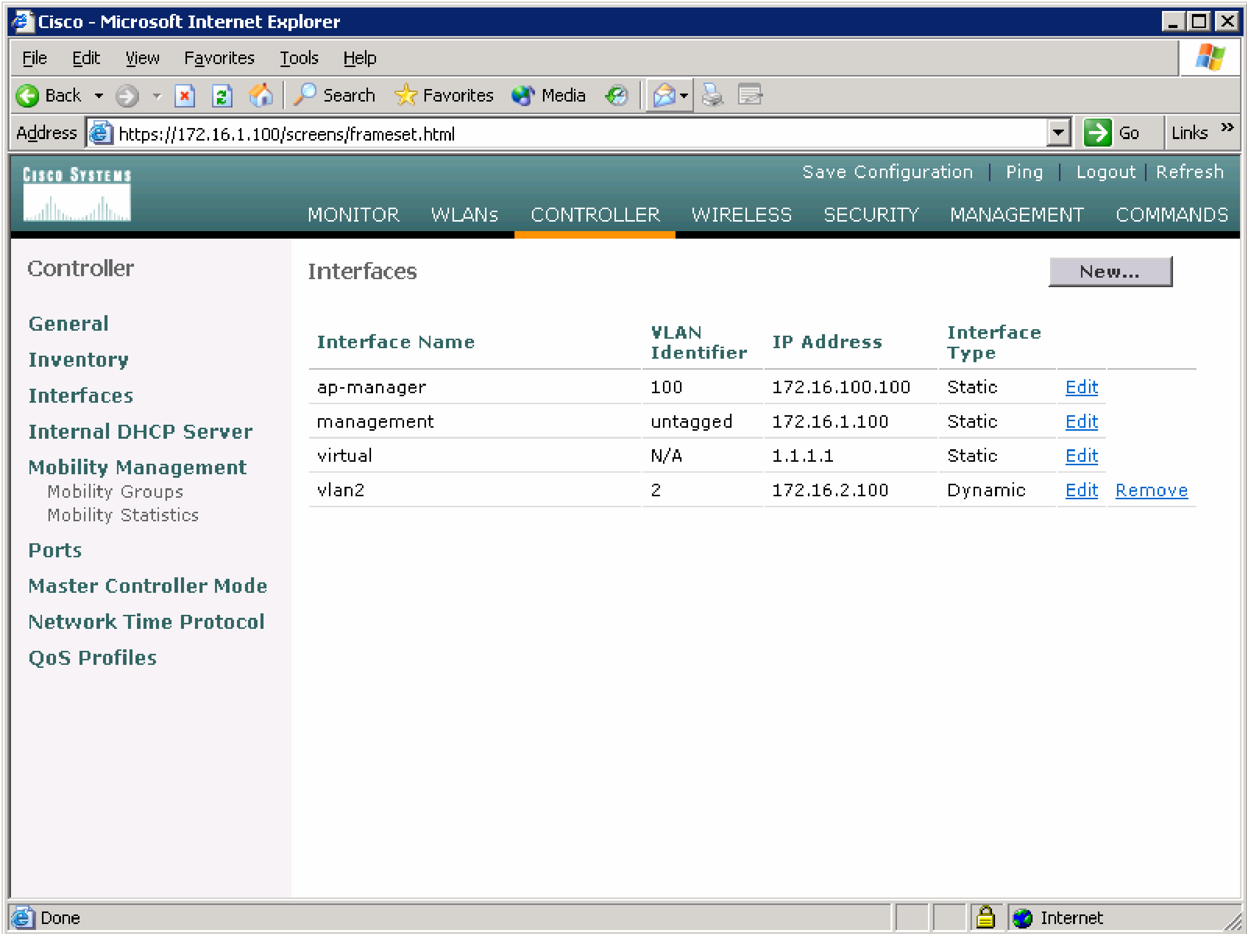


Figure 5-5: Verify Existing VLAN Interfaces

Click the **WLANs** button at the top of the web interface. This shows you all configured WLANs on the WLC. Currently the only one listed is the one created during the setup wizard.

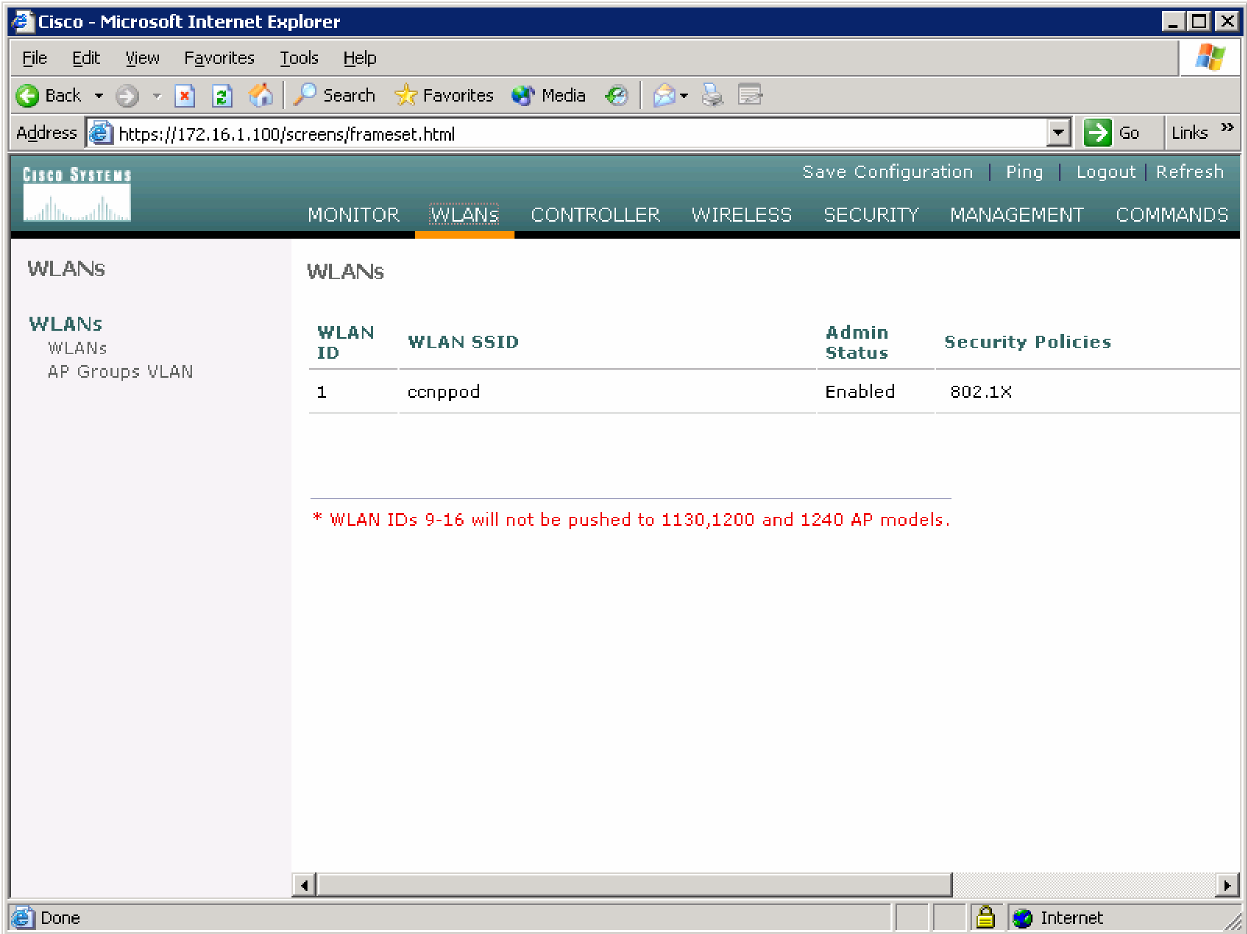


Figure 5-6: Viewing Existing WLANs with Security Policies

Click **Edit** for the WLAN listed. The default security policy is 802.1X, which is the security policy we want. Make sure that the administrative status of the WLAN is enabled. Change the IP interface of the WLAN to VLAN2, and assign the RADIUS server created earlier. Click **Apply** when all changes are configured. Click **OK** if a warning appears.

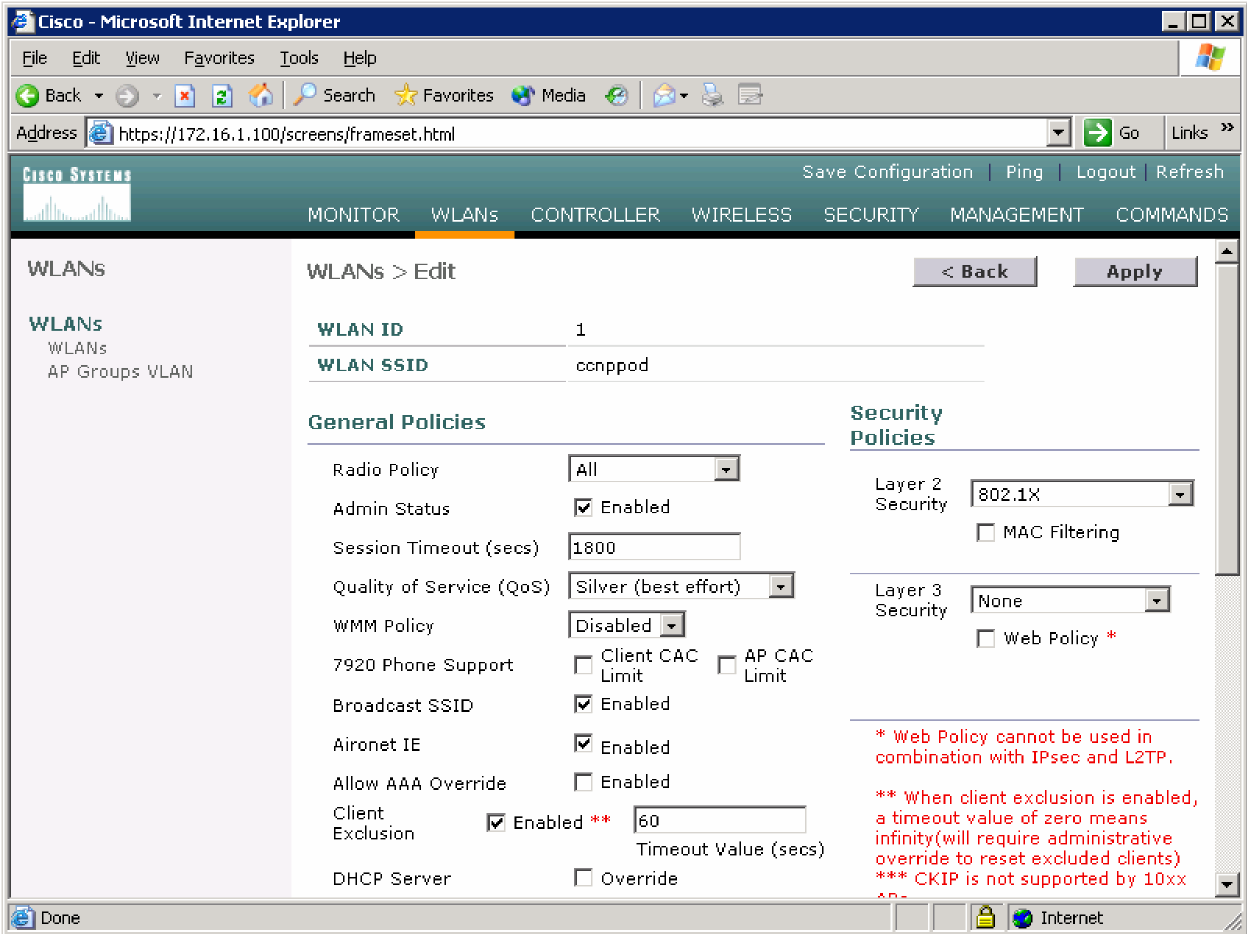


Figure 5-7: Editing the Configuration for WLAN 1

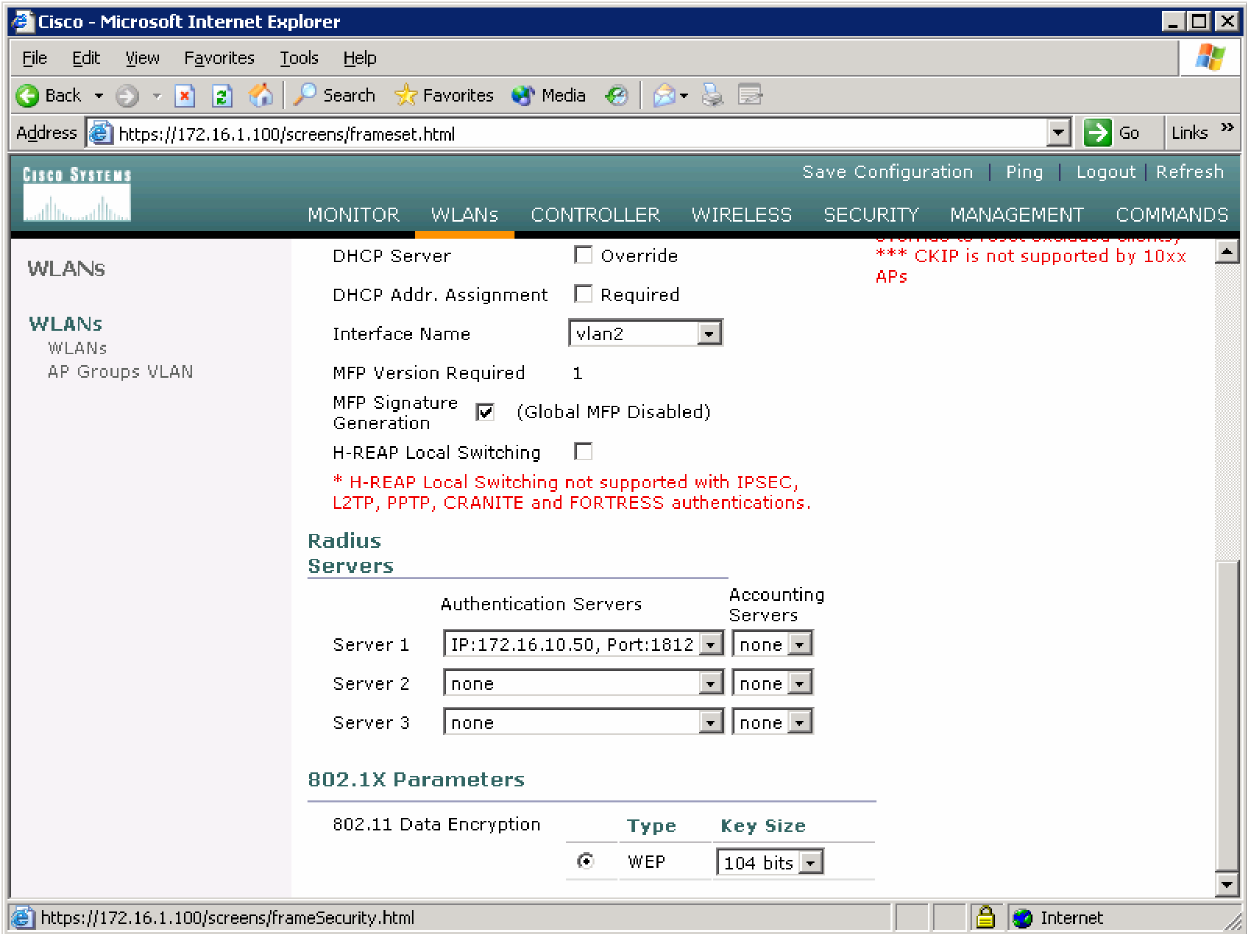


Figure 5-8: Editing the Configuration for WLAN 1, Security Options

Step 6: Configure the Wireless Client

On Host B, open up the Cisco Aironet Desktop Utility (ADU) either using the icon on the desktop or the program shortcut in the start menu. If you do not have the Cisco Aironet Desktop Utility installed, consult Lab 6.3: Configuring a Wireless Client. Once in the ADU, click the **Profile Management** tab. Next, click **New** to make a new profile.

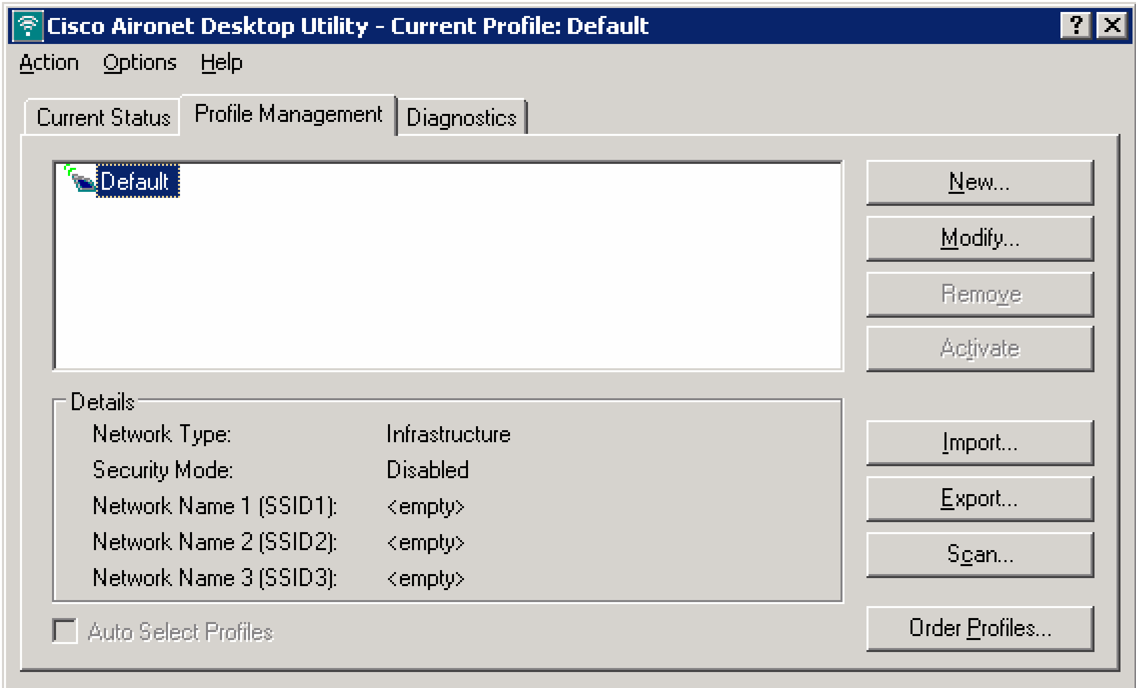


Figure 6-1: Cisco ADU Profile Management Tab

Use a profile name and service set identifier (SSID) of “ccnppod” since this was the SSID configured earlier. Use any client name desired. Here, “cisco” is the name used.

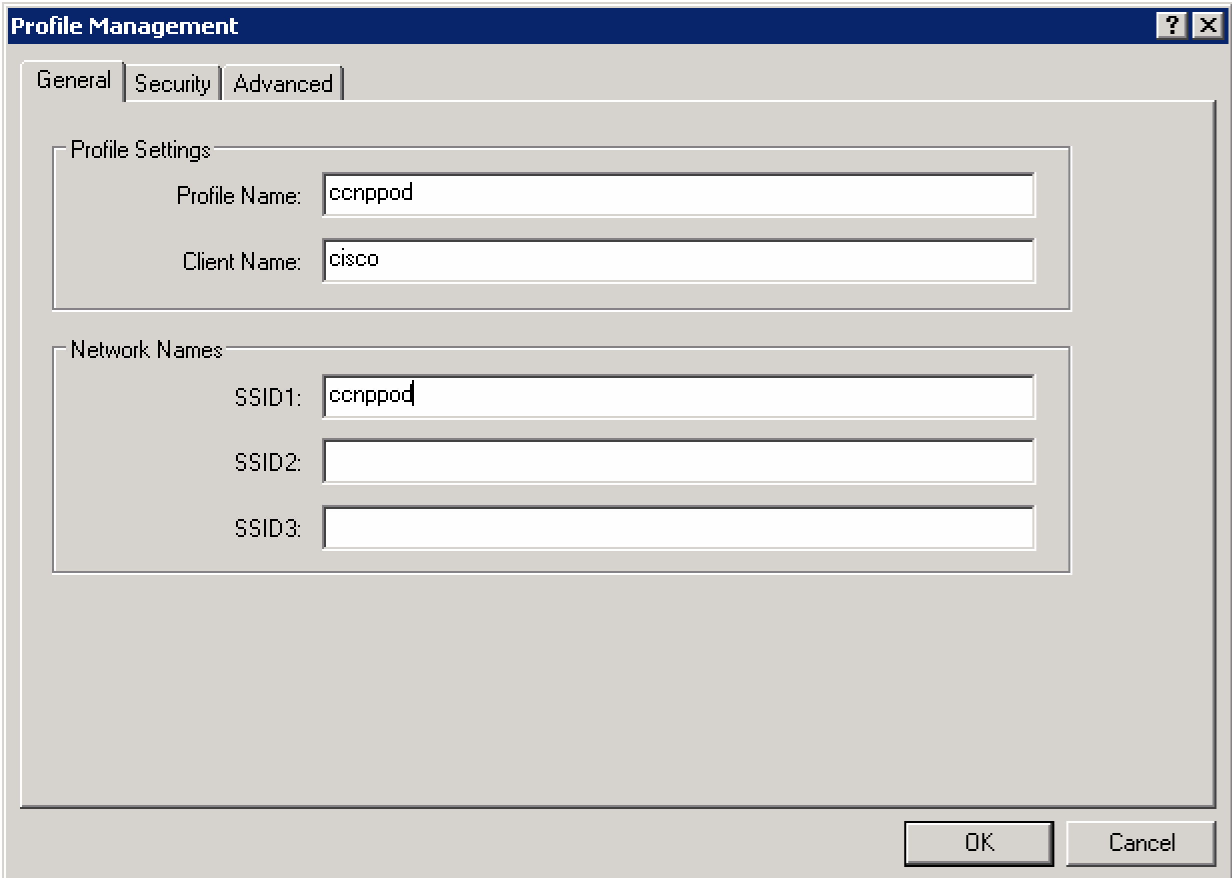


Figure 6-2: Configuring Profile Options and SSID

Under the **Security** tab, set the security type as **802.1x**. After selecting the security method, click **Configure**.

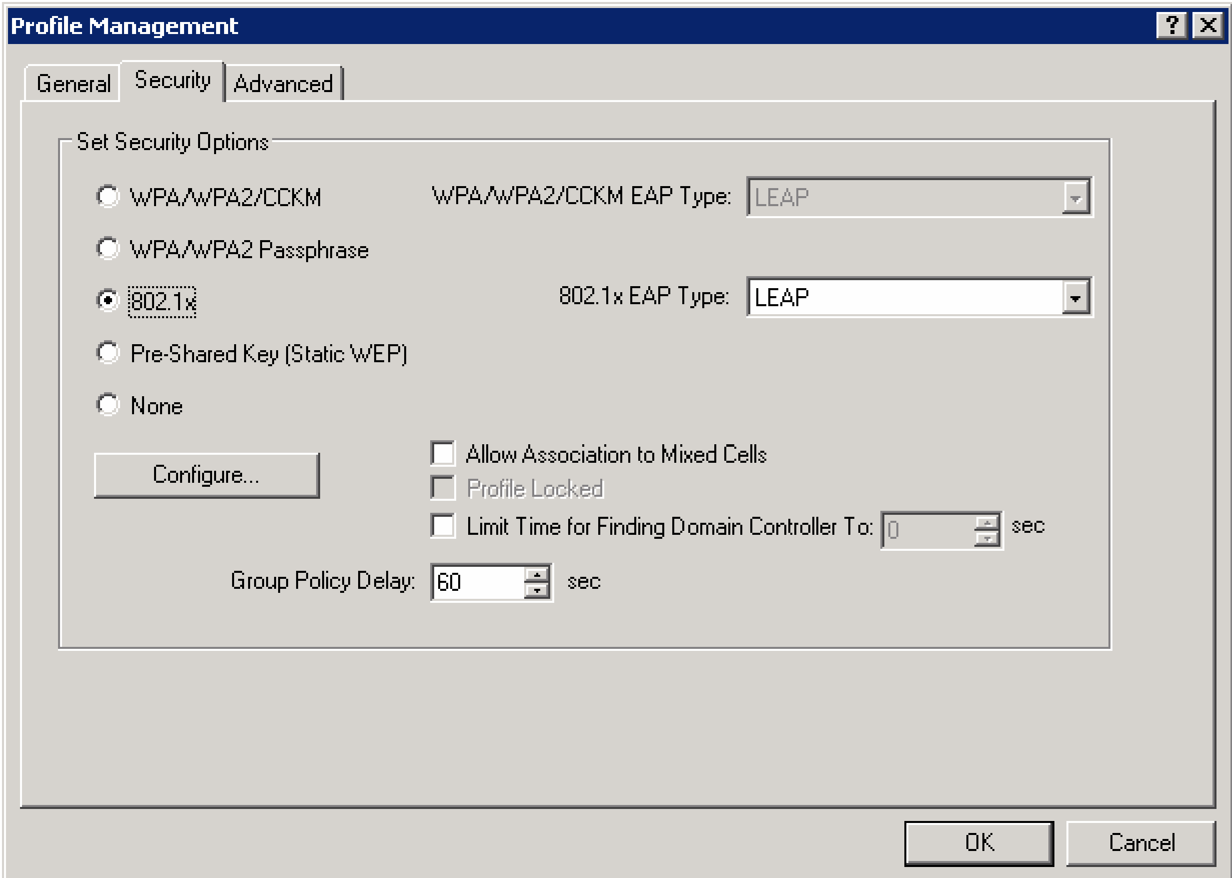


Figure 6-3: Wireless Security Options

Choose **Automatically Prompt for User Name and Password** as the authentication setting. Click **OK** when done, and then click **OK** again to close the new profile window.

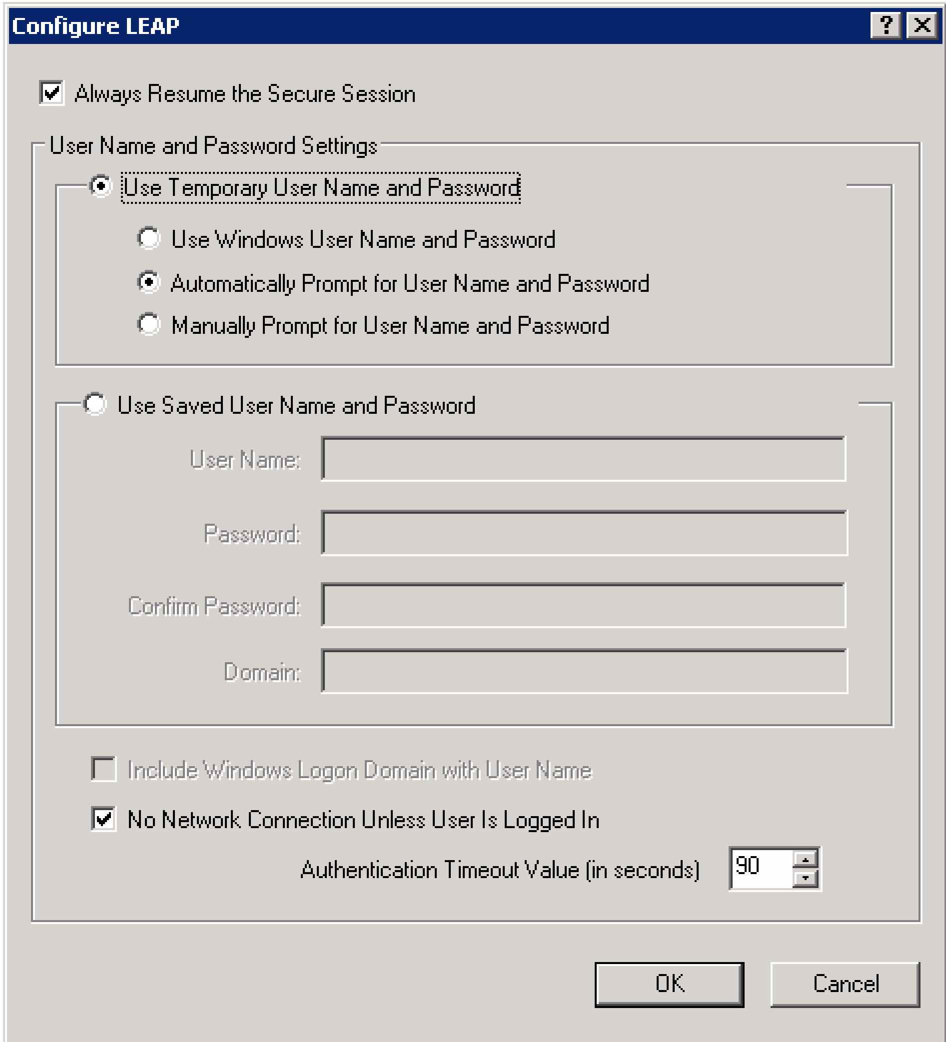


Figure 6-4: LEAP Configuration Options

On the profile list, select the new profile and click **Activate**.

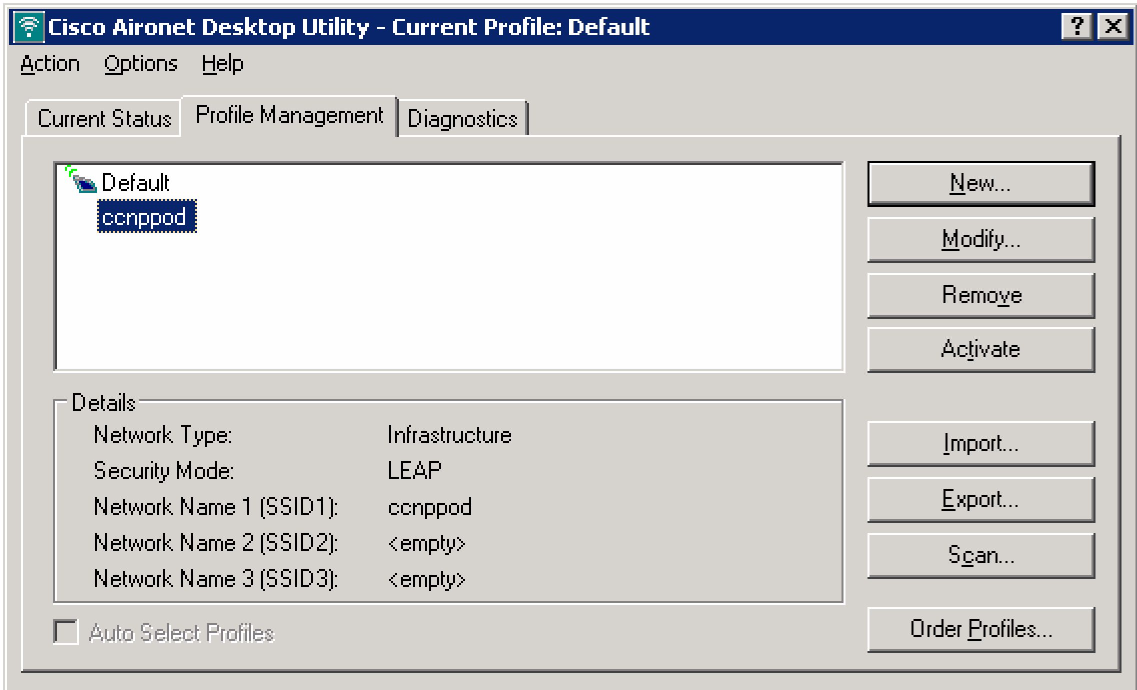


Figure 6-5: Selecting a Wireless Profile

When prompted to enter a username and password, enter in the credentials created earlier on the ACS server, and then click **OK**. (username and password of “cisco”).

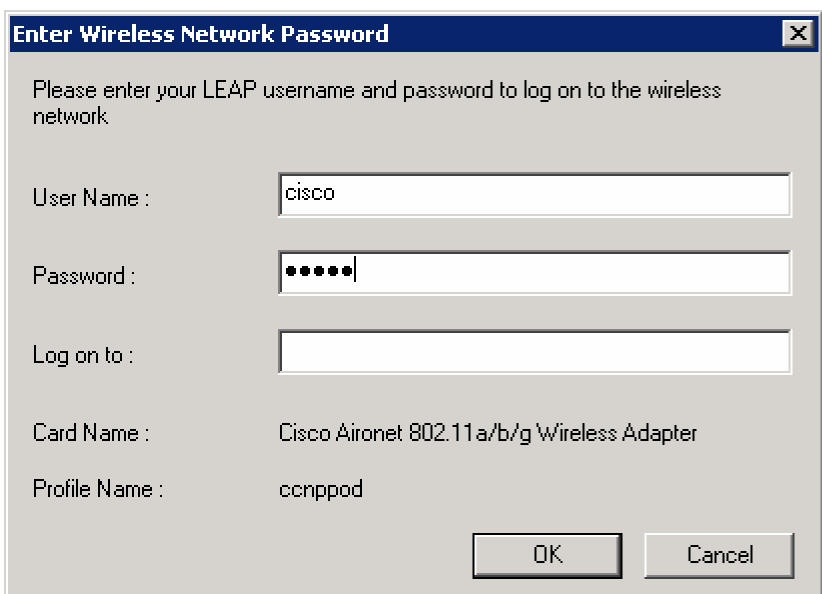


Figure 6-6: ADU LEAP Authentication Dialog

You should see all authentication steps be successful. If not, troubleshoot.

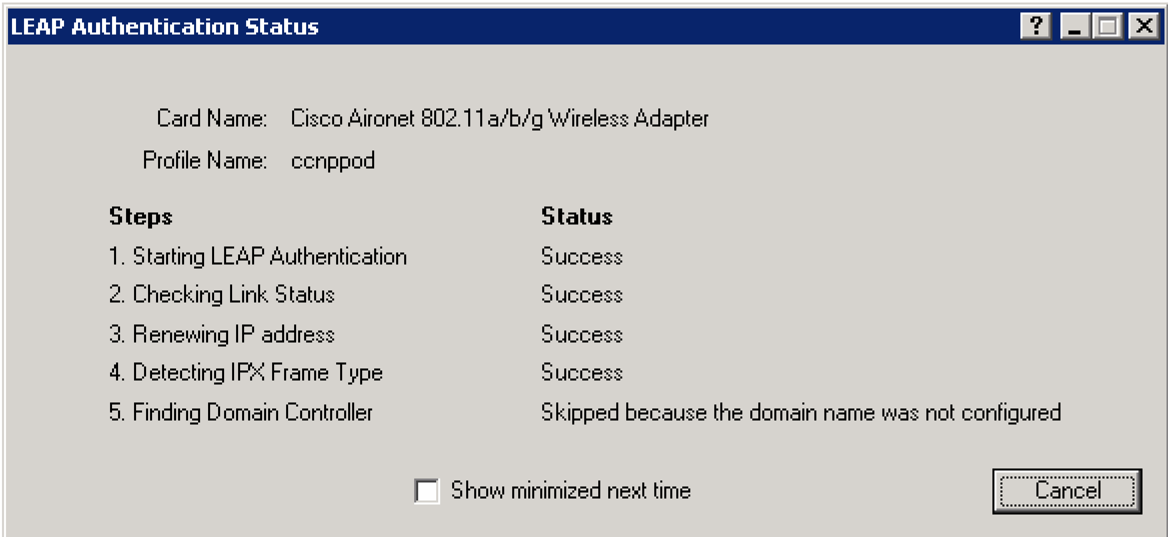


Figure 6-7: ADU LEAP Authentication Checklist

Under the **Current Status** tab, make sure you have received a correct IP address for the VLAN and the link is authenticated.

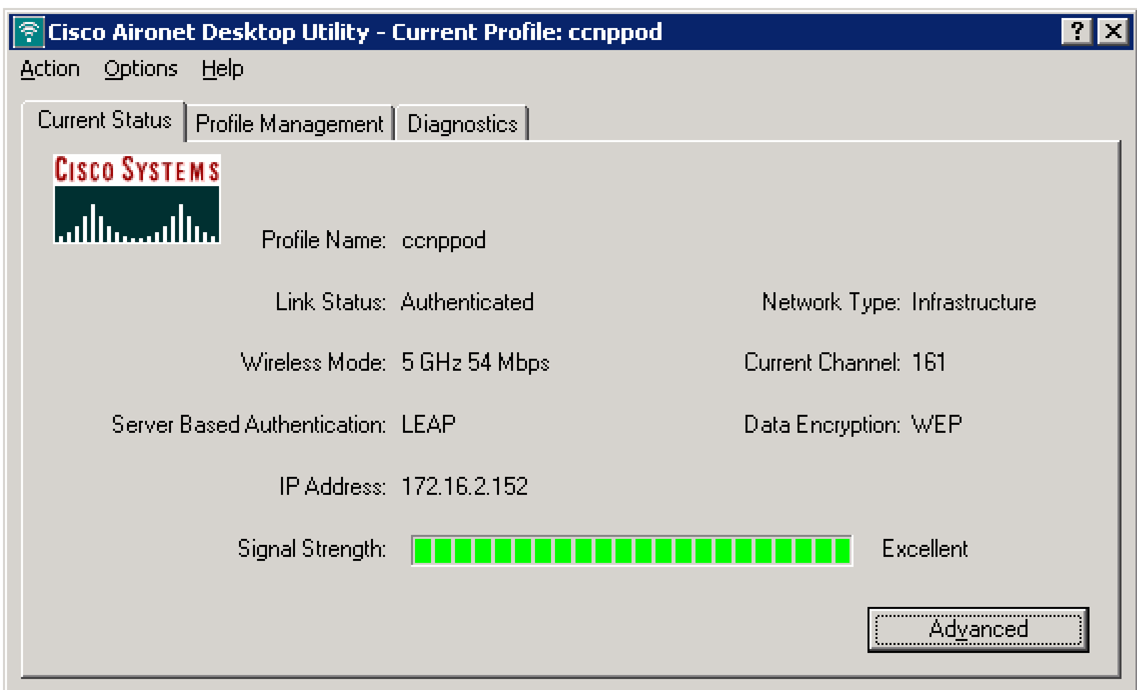


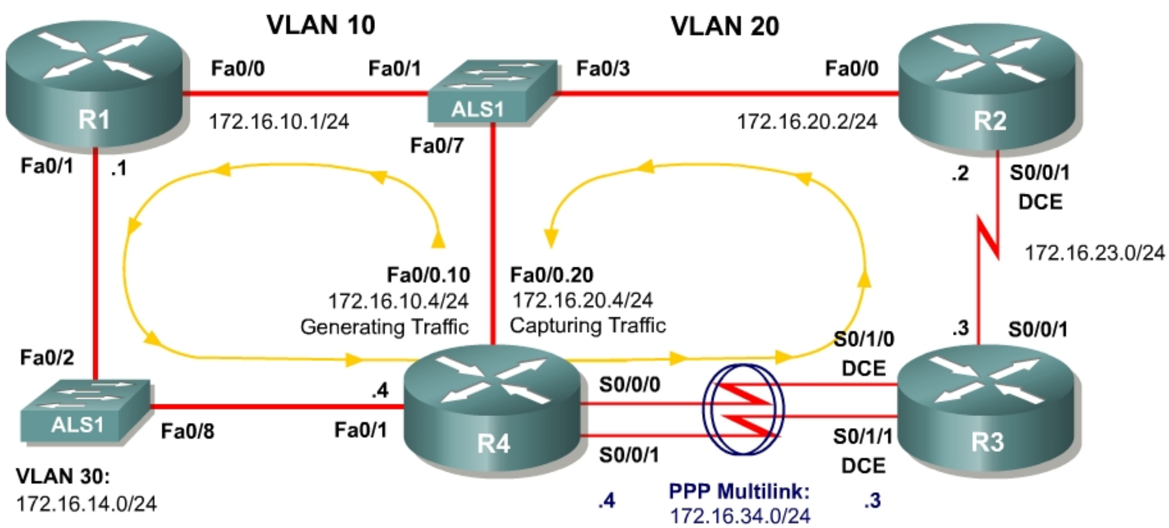
Figure 6-8: Current Wireless Profile Status

Case Study: QoS and MLPPP

Instructions

Implement the International Travel Agency network shown in the topology diagram and using the information and the instructions in the scenario. Implement the design on the lab set of routers. Verify that all configurations are operational and functioning according to the guidelines. This lab requires you to have the advanced Pagent configuration set up as shown in Lab 3.1: Preparing for QoS.

Topology Diagram



Scenario

The International Travel Agency is evaluating Quality of Service (QoS) strategies in their test environment using a traffic generator. This lab should be completed using the IOS command-line interface (CLI), without using Cisco Security Device Manager (SDM).

- Set up R4 using the advanced Pagent configuration and start traffic generation. (Certain configuration changes may change the traffic generation status so traffic generation may need to be restarted later in the lab.)
- Configure all interfaces using the subnetting scheme shown in the diagram, with the exception of the serial links between R3 and R4.
- Use a clock rate of 800000 on the serial link between R2 and R3.
- Configure the serial links between R3 and R4 to run at 2 mbps.
- Bind the serial links between R3 and R4 using PPP multilink and address it as shown in the diagram.

- Use weighted fair queuing (WFQ) on the PPP multilink.
- Enable the PPP multilink interleaving with a maximum interleaving delay of 20 ms.
- The International Travel Agency network should be running Open Shortest Path First (OSPF) in AS 1.
- Use Network-based Application Recognition (NBAR) on R1 to discover which traffic types are being generated from the traffic generator.
- Determine three different traffic classes and mark them with varying IP precedence for each class (this is subjective).
- Use NBAR to classify packets.
- Perform this marking outbound on R1 towards R4.
- Make sure the various classes do not exceed 3 megabits/second for each class.
- Do not configure queuing strategies to accomplish this task.
- Configure low latency queuing (LLQ) on R3 for the link between R2 and R3.
- Allocate bandwidth for each IP precedence you configured earlier.
- Also allocate some bandwidth for OSPF packets, and place this traffic in the priority queue.
- Bandwidth amounts are subjective, but do not exceed the capacity of the link.

