In this case, the destination router returns an ICMP Destination Unreachable message to the sending IP host with the code field indicating that fragmentation is needed to complete the delivery, but that the DF bit is set. The sending IP host must support the IP Path MTU Discovery mechanism in order to complete the delivery of the packet. If the sending IP host does not support the mechanism, the connection fails. The sending IP host will require that its largest packet size be reduced to a size manageable by the majority of the connected networks. Most network designers end up avoiding this problem by making the MTU of an interface, and consequently the IP MTU size, 1500 bytes, which is typically the largest packet size allowed on an Ethernet LAN.

The MTU packet size is the largest size available for any protocol on the interface. This varies depending on the interface type. The following command can be used to modify just the IP protocol MTU size when specified under the interface configuration mode,

ip mtu *bytes*

where *bytes* is the size of the largest IP MTU allowed on the network supported by the interface being defined. The minimum is 128 and the maximum is dependent on the media connected to the interface. If the DF bit is not set, the Cisco router will automatically fragment the packet to the supported MTU size of the next hop interface.

Filter IP Packets Using Access Lists

The most widely used method for controlling access to a router, and subsequently networks attached to a router, is the access list feature of Cisco IOS. The access list is a filter that either permits the packet to pass through the router or denies the packet and returns an ICMP Destination Unreachable message to the source IP host of the denied packet. There are four types of access lists: standard, extended, dynamic, and reflexive.

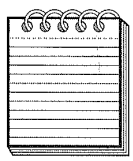
An access list is a sequential comparison of the packet to the filters defined by the access-list global configuration command. The filters created can be applied to the following:

- Control packet transmissions on an interface
- Control access via virtual terminal lines
- For restricting the contents of routing updates

The packet IP address is tested against the conditions defined for each access list. The Cisco IOS software performs the test sequentially down the list. The first match found is the action taken on the packet and further conditional testing on the packet stops.

A list number or a name identifies access lists. IP access list numbers 1 to 99 are for use by standard access lists, while list numbers 100 to 199 are reserved for extended access lists. Standard IP access lists apply the condition to the source IP address field only. Extended access lists apply the condition test to protocols and possibly port numbers along with the source and destination IP addresses.

The dynamic filter is a feature of the extended IP access list that is used to grant access on a per-user basis by applying a conditional test to the source or destination IP host using a user authentication process, thereby allowing dynamic access without compromising security. The reflexive access list is nested within an extended named IP access list, basing the filter on session information. Using a name to identify an IP access list enables the router administrator to use more than 199 filters against IP packets.



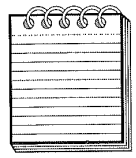
NOTE: *Cisco IOS Release 11.1 and higher have modified IP access lists. The IP access lists created by releases prior to IOS 11.1 are automatically converted to the new IP access list format.*

Do not use an access list created with Cisco IOS 11.1 or higher on pre-11.1 IOS releases. Pre-11.1 releases are not compatible with the new access list format. This may result in security violations.

The IP access lists are applied to an interface on which a filter is required. The application of the filter is specific to packets inbound to the interface or outbound to the interface. An IP access list is applied to an interface under interface configuration mode using

ip access-group *access-list-number* | *access-list-name* [**in** | **out**]

The access list by default is applied to the outbound packets when neither the **in** or **out** keywords are specified on the **ip access-group** command. The access list itself is identified by either specifying the appropriate *access-list-number* or *access-list-name* of a previously defined access list. If a value is used that is not defined by an access list statement, all packets are permitted. Either a number or a name can be used when the access list is applied to a router interface. Only a number is allowed when applying an access list to a virtual terminal line.



NOTE: *Enabling outbound access lists automatically disables autonomous switching for the interface. Inbound access lists on a cbus or cxbus interface board disable autonomous switching for all the interfaces on the board, with the exception of those using SSE switching and simple access lists for outbound packets, and they still perform SSE switching. All other packets are sent using process switching.*

Create Standard Access Lists Using Numbers and Names

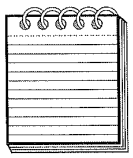
The format for creating a standard IP access list is

access-list *access-list-number* {**deny** | **permit**} **any** | *source* [*source-wildcard*] [**log**]

The *access-list-number* value is any available number in the range of 1 through 99, indicating a standard IP access list. More than one access-list command can be defined with the same access-list number. Be aware that all the conditions defined in the list group are applied to the packet being tested.

The **deny** | **permit** keywords indicate that if the condition being tested is true, the packet is dropped (deny) or forwarded (permit). The **any** keyword defines the use of the standard IP access list when the *source* is 0.0.0.0 and the *source-wildcard* is 255.255.255.255, essentially meaning all packets. The *source* variable is a four-part dotted decimal IP address and the *source-wildcard* is a dotted decimal value, indicating which bits of the *source* value are to be tested. The bits set to a one in the *source-wildcard* value are ignored.

The **log** keyword first appeared in IOS Release 11.3(3)T. Specifying this keyword causes the filtering process to write messages to the router message log. These messages include the *access-list-number*, whether the packet was permitted or denied, what the source address of the packet was, and the number of packets filtered. After the first successful match, a message is generated every five minutes with the number of packets permitted or denied within that interval.



NOTE: *An implicit deny filter for all packets can be found at the end of each access list.*

Suppose a resource with IP address 192.168.22.8, for example, requires access to the network, but all other hosts of the 192.168.0.0 network are restricted. The standard IP access list would be defined as

```
interface serial 0
ip address 10.10.100.100 25.255.255.0
ip access-group 1 in
access-list 1 permit 192.168.22.8 0.0.0.0
```

or

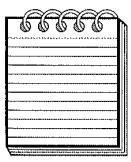
```
access-list 1 permit 192.168.22.8
```

The difference between the two statements is the exclusion of the *source-wildcard* value in the second access-list statement. Not coding the *source-wildcard* value assumes that the whole IP address entered as the *source* is tested. If a packet arrives with 192.168.22.10, it will fail the conditional testing, due to the implicit deny statement for the access list. All packets other than those with 192.168.22.8 will be discarded when applied against the defined access list.

In another example, suppose packets with 10.8.1.0 have restricted access with this router, but all other IP hosts in the 10.0.0.0 network are allowed. One other twist, address 10.8.1.23 needs access to the resources attached to this router. The standard access list is coded as

```
interface serial 0
ip address 10.10.100.100 25.255.255.0
ip access-group 1 in
access-list 1 permit 10.8.1.23
access-list 1 deny 10.8.1.0 0.0.0.255
access-list 1 permit 10.0.0.0 0.255.255.255
```

The statements' order of appearance in this access list is crucial for delivering the desired outcome. This example illustrates the best practice for defining access lists. Coding the most granular permits first, followed by the most granular denies, enables more global permits or denies to take place. Suppose a packet with IP address 10.8.2.200 is presented to the access list conditions. In this case, the packet is permitted because the IP address 10.8.2.200 falls into a different subnet than the deny statement.

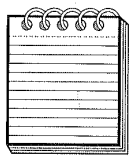


NOTE: *Coding an explicit deny all statement causes the Cisco IOS to add new access list statements after the explicit deny all statement, causing the new access-list statements to be ignored. Because of this, the implicit* ***deny all*** *statement is recommended.*

Using a name as the access list identifier alleviates the IP access list restriction to 199 conditions. Only packet and route filters are currently supported using named access lists. The use of a name was first introduced in Cisco IOS Release 11.2 software. Thus, Cisco IOS releases previous to Release 11.2 will not be compatible with the use of names for access lists. The global configuration commands required to assign a name to an IP standard access list are as follows:

ip access-list standard *name*
deny {*source* [*source-wildcard*] | **any**}[**log**]
permit {*source* [*source-wildcard*] | **any**}[**log**]

The variables and keywords of the deny and permit statements shown have the same meaning as that discussed for creating access lists with number identifiers. The difference in using names is that the **deny** and **permit** statements are subcommands of the **ip access-list standard** command. The *name* variable of the **ip access-list standard** command is a unique name assigned to this named list of deny and permit subcommands.



NOTE: *As with numbered lists, additional deny or permit statements are added to the bottom of the list. The no deny or no permit command, however, can be entered against a named access list to remove the condition taking immediate effect.*

Applying names to the previous numbered access lists in our examples results in the following configuration:

```
interface serial 0
ip address 10.10.100.100 255.255.255.0
ip access-group customer in
ip access-group sales out
!
ip access-list standard customer
permit 192.168.22.8 log
!
ip access-list standard sales
permit 10.8.1.23
deny 10.8.1.0 0.0.255.255
permit 10.0.0.0 0.255.255.255
```

The **log** keyword is added to the customer access list definition to log messages on the activity used by the permitted resource. The use of names is an added value since they now give an identity to the reason or group of users against which the filter is applied.

Extended Access Lists

The extended format of the IP access-list command is

access-list *access-list-number* {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [precedence *precedence*] [tos *tos*] [log]

The access-list-number is the extended access list identifier. The valid range for extended IP access lists is from decimal 100 to 199. The extended access list differs from a standard access list in its use of destination IP networks or IP host addresses, a precedence field, a Type of Service (TOS) field, and the keyword established. The variables are *protocol, source, source-wildcard, destination,* and *destination-wildcard*

The *protocol* variable can be any of the following IP protocol names or a valid integer assigned to the protocol, such as

eigrp, gre, icmp, igmp, igrp, ip, ipinip, nos, ospf, tcp, or **udp.**

A value of **ip** denotes that all IP protocols listed will provide a match.

The *source* and *destination* variable values identify the IP network or IP host address to which the filter is applied. The *source-* and *destination-wildcard* variable values define the interesting bits of the IP network or IP host address to which the filter applies. These can be specified in the following ways:

- Using dotted-decimal format, such as 10.10.200.20 or 192.168.63.0.
- Using the any keyword when the filter applies to a value IP network or IP host address of 0.0.0.0 and the accompanying wildcard variable is 255.255.255.255.
- Using the host keyword when the wildcard value being applied is 0.0.0.0, indicating all bits are interesting.

The optional *precedence* variable is optional and is used to further define the type of packet being filtered by referencing the importance of the packet type. The possible values for *precedence* are listed below:

critical matches packets with critical precedence (5).

flash matches packets with flash precedence (3).

flash-override matches packets with flash override precedence (4).

immediate matches packets with immediate precedence (2).

internet matches packets with internetwork control precedence (6).

network matches packets with network control precedence (7).

priority matches packets with priority precedence (1).

routine matches packets with routine precedence (0).

The optional *tos* variable allows the extended filter to be placed against a specific type of service that meets the source and destination packet criteria. The *tos* variable values that can be used are listed below:

<0-15> is the type of service value.

max-reliability matches packets with max reliable TOS (2).

max-throughput matches packets with max throughput TOS (4).

min-delay matches packets with min delay TOS (8).

min-monetary-cost matches packets with min monetary cost TOS (1).

normal matches packets with normal TOS (0).

The specification of the optional **log** keyword indicates that any packets meeting the filter criteria will be written to the router's log file.

Further detailed filtering of ICMP, IGMP, TCP, and UDP protocols is possible using the extended access list. Specifying any of these protocols for the protocol variable modifies the format of the command to accept parameters with specific criteria within the protocol.

The complete format for specifying the ICMP as the protocol variable value is

access-list *access-list-number* {**deny** | **permit**}
icmp *source source-wildcard destination destination-wildcard*
[*icmp-type* [*icmp-code*] | *icmp-message*] [**precedence** *precedence*] [**tos** *tos*]
[**log**]

In the **icmp** format of the extended access list are three optional variables for further granular filtering. The *icmp-type*, along with the optional *icmp-code* parameter, allows criteria for an ICMP packet against the numerical values associated with the ICMP protocol. The numeric value is 0–255 for both the *icmp-type* and *icmp-code* variables. The *icmp-message* variable can be used, instead of the paired *icmp-type* and *icmp-code* variables. Using the *icmp-message* variable, ICMP packets are filtered based on a valid name provided by Cisco IOS software that describes the ICMP message type or ICMP message type and code within the message type. The following is the list of possible icmp-message values available:

administratively-prohibited: Administratively prohibited

alternate-address: Alternate address

conversion-error: Datagram conversion

dod-host-prohibited: Host prohibited

dod-net-prohibited: Net prohibited

echo: Echo (ping)
echo-reply: Echo reply
general-parameter-problem: Parameter problem
host-isolated: Host isolated
host-precedence-unreachable: Host unreachable for precedence
host-redirect: Host redirect
host-tos-redirect: Host redirect for TOS
host-tos-unreachable: Host unreachable for TOS
host-unknown: Host unknown
host-unreachable: Host unreachable
information-reply: Information replies
information-request: Information requests
mask-reply: Mask replies
mask-request: Mask requests
mobile-redirect: Mobile host redirect
net-redirect: Network redirect
net-tos-redirect: Net redirect for TOS
net-tos-unreachable: Network unreachable for TOS
net-unreachable: Net unreachable
network-unknown: Network unknown
no-room-for-option: Parameter required but no room
option-missing: Parameter required but not present
packet-too-big: Fragmentation needed and DF set
parameter-problem: All parameter problems
port-unreachable: Port unreachable
precedence-unreachable: Precedence cutoff
protocol-unreachable: Protocol unreachable
reassembly-timeout: Reassembly timeout
redirect: All redirects
router-advertisement: Router discovery advertisements
router-solicitation: Router discovery solicitations
source-quench: Source quenches
source-route-failed: Source route failed
time-exceeded: All time exceeded

timestamp-reply: Timestamp replies

timestamp-request: Timestamp requests

traceroute: Traceroute

ttl-exceeded: TTL exceeded

unreachable: All unreachables

In the following example, the serial 0 interface connects the router to the Internet. The access list 100 is applied to the outbound packets of the serial 0 interface, allowing the internal network to ping devices on the Internet. The access-list 101 is applied to inbound packets of the serial0 interface, thereby prohibiting Internet IP hosts from being able to ping internal IP hosts. The filters also allow echo-reply messages to flow.

```
interface serial 0
ip address 209.196.9.34 255.255.255.0
 ip access-group 100 out
 ip access-group 101 in
access-list 100 permit icmp 10.0.0.0 0.255.255.255 any echo log
access-list 101 deny icmp any 10.0.0.0 0.255.255.255 echo log
```

When specifying the Internet Group Management Protocol (IGMP) on an extended access list, the format allows for the inclusion of specific IGMP message types. The format of the IGMP extended access list is

access-list *access-list-number* {**deny** | **permit**}
igmp *source source-wildcard destination destination-wildcard* [*igmp-type*]
[**precedence** *precedence*] [**tos** *tos*] [**log**]

The igmp-type variable can be a valid IGMP message type number ranging from 0 to 15, or it can be one of the IGMP message names found in the following list:

dvmrp: Distance Vector Multicast Routing Protocol

host-query: Host query

host-report: Host report

pim: Protocol Independent Multicast

trace: Multicast trace

In the following example, the serial0 interface connects the router to the Internet. The host-report IGMP message is prohibited from being transmitted between any internal hosts and any external hosts on the Internet.

```
interface serial 0
ip address 209.196.9.34 255.255.255.0
 ip access-group 102 out
access-list 102 deny igmp any any host-report log
```

Specifying the TCP protocol on an extended access list enables the Cisco IOS software to apply filter criteria to packets that indicate a TCP connection is already traversing the router along with applying a comparison to specific TCP applications using the applications port number. The format of the extended access list when applying the filter to TCP is

access-list *access-list-number* {**deny** | **permit**}
tcp *source source-wildcard* [*operator port* [*port*]] *destination destination-wildcard* [*operator port* [*port*]] [**established**] [**precedence** *precedence*] [**tos** *tos*] [**log**]

The optional **established** keyword is used to match the filter and the packet only if the ACK or RST bits of the TCP header are set. This means that the filter is not applied to the packets during the initial TCP handshake used to form the connection.

The optional *operator* variable can be applied to the source and/or destination IP network or IP host address. The variable can be any of the following values:

eq matches only packets on a given port number.

gt matches only packets with a greater port number.

lt matches only packets with a lower port number.

neq matches only packets not on a given port number.

range matches only packets in the range of port numbers.

The *operator* value specified indicates the scope of the filter. Using the range value for the operator variable requires a second port number, thereby restricting the match within a sequential range of TCP port numbers.

The port variable is a TCP port number used by the packets of interest. The Cisco IOS software has the following list of port numbers that can also be assigned using the associated TCP port number name. These are

bgp: Border Gateway Protocol (179)

chargen: Character generator (19)

cmd: Remote commands (rcmd, 514)

daytime: Daytime (13)

discard: Discard (9)

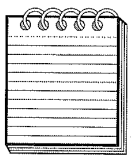
domain: Domain Name Service (53)

echo: Echo (7)

exec: Exec (rsh, 512)

finger: Finger (79)
ftp: File Transfer Protocol (21)
ftp-data: FTP data connections (used infrequently, 20)
gopher: Gopher (70)
hostname: NIC hostname server (101)
ident: Ident Protocol (113)
irc: Internet Relay Chat (194)
klogin: Kerberos login (543)
kshell: Kerberos shell (544)
login: Login (rlogin, 513)
lpd: Printer service (515)
nntp: Network News Transport Protocol (119)
pop2: Post Office Protocol v2 (109)
pop3: Post Office Protocol v3 (110)
smtp: Simple Mail Transport Protocol (25)
sunrpc: Sun Remote Procedure Call (111)
syslog: Syslog (514)
tacacs: TAC Access Control System (49)
talk: Talk (517)
telnet: Telnet (23)
time: Time (37)
uucp: Unix-to-Unix Copy Program (540)
whois: Nicname (43)
www: World Wide Web (HTTP, 80)

The TCP port numbers listed represent well-known port numbers reserved for standardized TCP protocols.



NOTE: *If applying a filter to a user-defined TCP application, the port number must be specified since the user-defined TCP application name is unknown to the Cisco IOS.*

In the following example, the TCP protocol is being filtered on the inbound side of the serial 0 interface, which connects the router to the Internet. In this filter, a remote IP host with the address 192.168.39.8 is allowed to access an internal IP host at IP address 10.1.1.200 with only the Telnet protocol. Placing the filter on the inbound side of the serial interface ensures that the remote IP host cannot try to access any other host on the internal network.

```
interface serial 0
ip address 10.200.20.1 255.255.255.0
 ip access-group 103 in
access-list 103 permit tcp host 192.168.39.8 eq telnet host
10.1.1.200 eq telnet log
```

Applying an extended access list to UDP, like TCP, allows for comparison to UDP port numbers on the source and/or destination IP network or IP host address. The format of the extended access list for use with filtering UDP messages is

access-list *access-list-number* {**deny** | **permit**}
udp *source source-wildcard* [*operator port* [*port*]] *destination destination-wildcard* [*operator port* [*port*]] [**precedence** *precedence*] [**tos** *tos*] [log]

The function of the operator and port variables is identical to the function and values of their use with TCP. However, UDP port numbers are unique to UDP applications. The UDP application names available for the port variable value, along with their associated port number, are as follows:

biff: Biff (mail notification, comsat, 512)

bootpc: Bootstrap Protocol (BOOTP) client (68)

bootps: Bootstrap Protocol (BOOTP) server (67)

discard: Discard (9)

dnsix: DNSIX security protocol auditing (195)

domain: Domain Name Service (DNS, 53)

echo: Echo (7)

mobile-ip: Mobile IP registration (434)

nameserver: IEN116 name service (obsolete, 42)

netbios-dgm: NetBios datagram service (138)

netbios-ns: NetBios name service (137)

ntp: Network Time Protocol (123)

rip: Routing Information Protocol (router, in.routed, 520)

snmp: Simple Network Management Protocol (161)

snmptrap: SNMP Traps (162)

sunrpc: Sun Remote Procedure Call (111)

syslog: System Logger (514)

tacacs: TAC Access Control System (49)

talk: Talk (517)

tftp Trivial File Transfer Protocol (69)

time: Time (37)

who: Who service (rwho, 513)

xdmcp: X Display Manager Control Protocol (177)

For both TCP and UDP, the valid range for the port value when using a number to identify the port is 0 through 65535.

In the following UDP example for extended access lists, the Ethernet interface 0 connects the router to the DMZ of the firewall for access to the Internet. To ensure that Internet RIP packets do not enter the router, an inbound filter denying RIP UDP packets is applied using access-list number 104.

```
interface ethernet 0
ip address 10.200.10.1 255.255.255.0
 ip access-group 104 in
access-list 104 permit udp any any neq rip log
access-list 104 deny udp any any eq rip log
```

The extended access lists can also be defined using names. The format of the commands are as follows:

ip access-list extended *name*
{**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log**]
or
{**deny** | **permit**} *protocol* **any any**
or
{**deny** | **permit**} *protocol* **host** *source* **host** *destination*

For ICMP filtering using names on extended access lists, you would use the following format:

{deny | permit} icmp *source source-wildcard destination destination-wildcard* [*icmp-type* [*icmp-code*] | *icmp-message*] [**precedence** *precedence*] [**tos** *tos*] [log]

For IGMP filtering using names on extended access lists:

{deny | permit} igmp *source source-wildcard destination destination-wildcard* [*igmp-type*] [**precedence** *precedence*] [**tos** *tos*] [log]

For TCP filtering using names on extended access lists:

{deny | permit} tcp *source source-wildcard* [*operator port* [*port*]] *destination destination-wildcard [operator port* [*port*]] [**established**] [**precedence** *precedence*] [**tos** *tos*] [log]

For UDP filtering using names on extended access lists:

{deny | permit} udp *source source-wildcard* [*operator port* [*port*]] *destination destination-wildcard* [*operator port* [*port*]] [**precedence** *precedence*] [**tos** *tos*] [log]

The values and placements for extended access lists using names are the same for specifying the denial or permission of a packet. The **ip access-list extended** command followed by a unique *name* indicates to the Cisco IOS software that the following statements are grouped under the specified *name*.

We can apply the numbered extended access list examples using a name by specifying them using the descriptions listed below.

Using named extended access lists on ICMP echo messages to and from the Internet connection on serial 0 interface:

```
interface serial 0
ip address 209.196.9.34 255.255.255.0
 ip access-group pingnet out
 ip access-group pingnet in
ip access-list extended pingnet
 permit icmp 10.0.0.0 0.255.255.255 any echo log
 deny icmp any 10.0.0.0 0.255.255.255 echo log
```

Prohibiting the host-report IGMP message from leaving the router on interface serial0 to the Internet:

```
interface serial 0
ip address 209.196.9.34 255.255.255.0
 ip access-group igmpout out
ip access-list extended igmpout
 deny igmp any any host-report log
```

Permitting Telnet access only between an Internet IP host at 192.168.39.8 and an internal IP host at 10.1.1.200 through the serial 0 interface using named extended access lists:

```
interface serial 0
ip address 209.196.9.34 255.255.255.0
 ip access-group port25 in
ip access-list extended port25
 permit tcp host 192.168.39.8 eq telnet host 10.1.1.200 eq telnet
log
```

In the last example, an RIP protocol is filtered out on the inbound side of an Ethernet connection:

```
interface ethernet 0
ip address 10.200.10.1 255.255.255.0
 ip access-group norip in
ip access-list extended norip
 deny udp any any eq rip log
```

Fault-Tolerant Routing of IP Packets

The Cisco IOS software can provide backup and recovery of the IP default gateway address using the Hot Standby Router Protocol (HSRP). Let's use Figure 13-10 as the first example on understanding how HSRP works with Cisco routers.

The virtual IP address of 10.1.1.1 is the default IP gateway address for ELAN1 on the Cisco Catalyst 5000 switch in Figure 13-10. The actual IP addresses HSRP assigns a virtual MAC address for the LAN interface associating the virtual IP address. The ATM-specific definitions are discussed in detail in Chapter 20, "Defining ATM (LANE, Classical IP, and MPOA)."

The following is the configuration for PRI-Router:

hostname PRI-Router

```
!
interface ethernet 0
ip address 10.10.10.3 255.255.255.0
standby 1 ip 10.10.10.1
standby 1 preempt
standby 1 priority 110
standby 1 authentication group1
standby 1 timers 5 15
```

The following is the configuration for SEC-Router:

hostname SEC-Router

```
!
interface ethernet 0
ip address 10.10.10.2 255.255.255.0
standby 1 ip 10.10.10.1
standby 1 preempt
standby 1 authentication group1
standby 1 timers 5 15
```

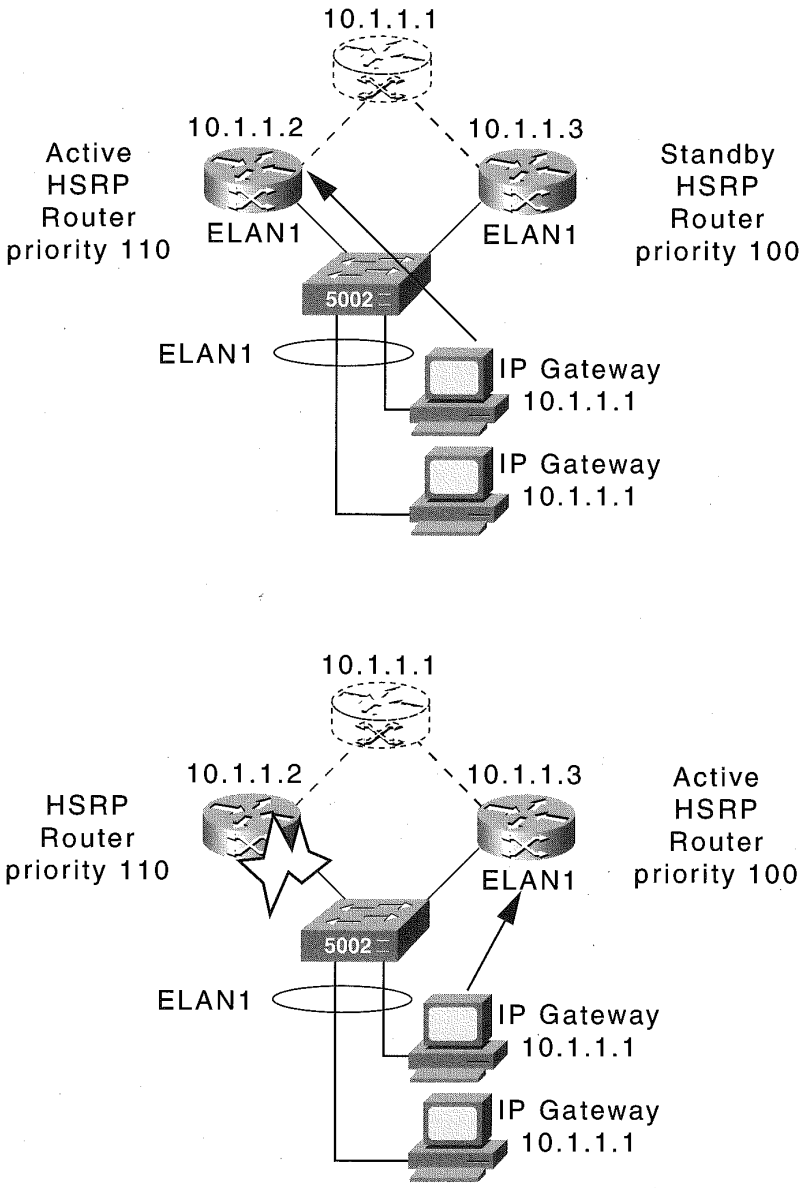
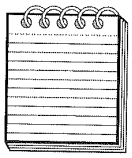


Figure 13-10
HSRP in support of providing IP gateway address backup and recovery.

The HSRP feature is enabled by specifying the interface configuration command, **standby ip**. The format of this command is

standby [*group-number*] **ip** [*ip-address* [**secondary**]]

The *group-number* is an optional numeric identifier used to designate the routers that belong to the HSRP group. The default is 0 with a valid range of 0 to 255. The *ip-address* variable is the virtual IP address being assigned to the HSRP group. The secondary keyword is used when HSRP is being applied as a **secondary** IP address. In our example, the HSRP group is group 1 and the 10.10.10.1 is the virtual IP address.



NOTE: *Ethernet, ATM LANE, and FDDI support up to 255 Hot Standby groups. Token Ring LANs can support only three Hot Standby groups numbered 0, 1, and 2.*

The **standby priority** HSRP interface command is used on a router interface with HSRP to ensure the election of this router interface as the primary router. The format of this command is

standby [*group-number*] **priority** *priority* [**preempt** [**delay** *delay*]]

The *group-number* associates this command with an HSRP group. The *priority* value is a defined numeric value in the range of 1 to 255, defaulting to 100 and denoting the preference of this interface to act as the primary for the virtual IP address. The **preempt** keyword, if specified, directs the Cisco IOS to assume control as the active virtual IP address if the router has a higher priority than the current primary virtual interface. The *delay* variable determines the amount of time in seconds that the router will postpone its attempt to become the active primary interface for the virtual IP address. The *delay* variable defaults to 0 and can range from 0 to 3600 seconds.

In the example, PRI-Router has the *priority* set to 110, while SEC-Router defaults to a priority of 100. During the UDP hello exchanges that occur with HSRP, PRI-Router is elected as the primary interface for servicing the virtual IP address. Should PRI-Router's HSRP interface go inactive, the Ethernet interface on SEC-Router will become the active interface. Once PRI-Router's Ethernet interface becomes active again, it immediately attempts to become the active interface since the **preempt** keyword is coded and the delay value defaults to a 0. If the **preempt** keyword is not coded, the PRI-Router becomes the active router again when SEC-Router Ethernet is inactivated.

An authentication string can be used that ensures the proper learning of the virtual IP address and timer values interpreted for the appropriate HSRP group. The format of the command is

standby [*group-number*] **authentication** *string*

The *group-number* identifies which HSRP group the authentication *string* applies. All routers participating in the same HSRP group must use the same authentication string. The string value defaults to the string "cisco" and is one to eight characters in length. In our example, the authentication string of group1 is used to verify that the parameters are for use with the HSRP standby group number 1.

HSRP uses UDP hello messages to discover and convey other HSRP routers' participation. The standby timer's interface command allows the router administrator to fine-tune the interval between Hello messages and the delay in determining that an active or other standby router is no longer available. The format for the command is

standby [*group-number*] **timers** *hellotime holdtime*

The *group-number* value identifies which standby group to apply to the timer values. The *hellotime* variable is the number of seconds from 1 to 255 the router will wait between sending Hello messages. Its default is three seconds. The *holdtime* value ranges from 1 to 255 and defaults to 10 seconds. This value denotes the amount of time that must pass without receiving a Hello message from a previously known HSRP router interface before it is determined as unavailable. The timers configured on the active router override all timer settings. If coding the standby timers interface command, be sure to code it the same on all interfaces participating in the same standby group.

The *holdtime* is typically greater than or equal to three times the *hellotime* value. In our example, we follow this rule by having the Hello messages sent every five seconds and the receipt of a Hello message every 15 seconds the sending router interface has become unavailable.

Using the concept of HSRP groups, multiple groups can be created that can enable a single router to be the backup for multiple routers or even for creating load sharing. Figure 13-11 illustrates the capability of providing load sharing on Cisco routers with HSRP.

The following is the HSRP configuration portion of router R1:

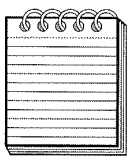
```
hostname R1
!
interface ethernet 0
ip address 10.1.1.1 255.255.255.0
standby 1 ip 10.1.1.253
standby 1 priority 110
standby 1 preempt
```

```
standby 2 ip 10.1.1.254
standby 2 preempt
```

The following is the HSRP configuration portion of router R2:

```
hostname R2
!
interface ethernet 0
ip address 10.1.1.2 255.255.255.0
standby 1 ip 10.1.1.253
standby 1 preempt
standby 2 ip 10.1.1.254
standby 2 priority 110
standby 2 preempt
```

The above configurations demonstrate the use of two Cisco routers supporting fault-tolerance for two IP gateway addresses on the same subnet. IP address 10.1.1.253 is the virtual IP address supporting LAN devices requiring 10.1.1.253 as the default IP gateway. Router R1 is the active router for LAN devices using 10.1.1.253 as the default gateway due to the higher priority defined on standby group 1, which defines 10.1.1.253 as the virtual IP address. Router R2 assumes the active router responsibilities, should the Ethernet interface on router R1 become unavailable. If this happens, the physical interface on router R2 supports three IP addresses and responds to all three: the interface IP address and the group1 and group 2 standby IP addresses.



NOTE: *In order to minimize the potential for connection or session outages for all HSRP configurations during the election of a new active router, it is best to use a fast-converging routing protocol like EIGRP.*

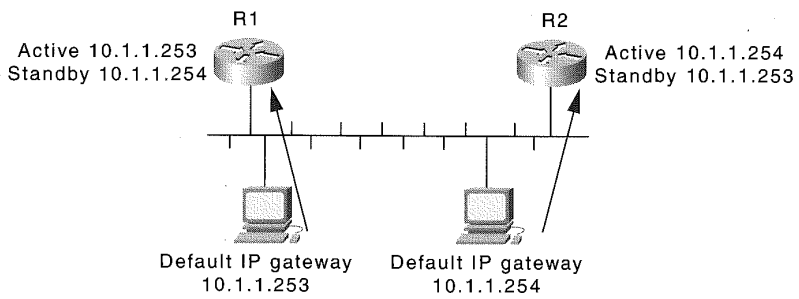


Figure 13-11 Support for fault-tolerant load sharing between two Cisco routers.

This same method can be used to plan load balancing over two routers that support the same multiple LANs. For instance, if both routers in Figure 13-11 support the same four Ethernet LAN segments, the configuration to load balance two segments on one router and two segments on the other router during a stable network would look like the following code.

The following section is the HSRP configuration portion of router R1:

```
hostname R1
!
interface ethernet 1/0
ip address 10.1.1.1 255.255.255.0
standby 1 ip 10.1.1.253
standby 1 priority 110
standby 1 preempt
standby 2 ip 10.1.1.254
standby 2 preempt
!
interface ethernet 1/1
ip address 10.1.2.1 255.255.255.0
standby 3 ip 10.1.2.253
standby 3 priority 110
standby 3 preempt
standby 4 ip 10.1.2.254
standby 4 preempt
!
interface ethernet 1/2
ip address 10.1.3.1 255.255.255.0
standby 5 ip 10.1.3.253
standby 5 priority 110
standby 5 preempt
standby 6 ip 10.1.3.254
standby 6 preempt
!
interface ethernet 1/3
ip address 10.1.4.1 255.255.255.0
standby 7 ip 10.1.4.253
standby 7 priority 110
standby 7 preempt
standby 8 ip 10.1.4.253
standby 8 preempt
```

The following is the HSRP configuration portion of router R2:

```
hostname R2
!
interface ethernet 0
ip address 10.1.1.2 255.255.255.0
standby 1 ip 10.1.1.253
standby 1 preempt
standby 2 ip 10.1.1.254
standby 2 priority 110
standby 2 preempt
```

IP Performance Tuning

The focus of IP performance-tuning in Cisco IOS is on TCP. The performance options available for tuning IP deal with the compression of TCP packet headers, MTU path discovery time, and selective acknowledgment to reduce retransmission of TCP packets, the TCP read size, the TCP window size, and outgoing queue size.

Compressing TCP Headers

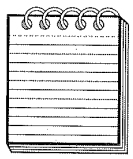
Compressing TCP headers of TCP segments reduces the size of the IP datagram and therefore reduces the bandwidth requirement on the WAN connections. TCP header compression is supported by the following:

- HDLC serial lines
- Frame Relay serial lines
- PPP encapsulation on serial lines

TCP packet header compression is best suited for networks in which many small TCP packets traverse the WAN connection. This type of traffic is typical of interactive transaction processing from display terminals. The format of the interface configuration command is as follows:

ip tcp header-compression [passive]

Specifying the command on the serial interface with the optional **passive** keyword indicates that outgoing TCP packets are compressed only if the incoming TCP packets of the same interface are compressed. Not coding the passive keyword indicates that all incoming and outgoing TCP headers are compressed.



NOTE: *Fast switching is disabled when compression is enabled for a serial interface. Because of this, using TCP header compression on T1 serial interfaces and higher bandwidths may result in router overload.*

An associated command used with **ip tcp header-compression** is the **ip tcp compression-connections** interface configuration command. The format of this command is

ip tcp compression-connections *number*

The number variable is the total number of header compression connections supported by the router on the serial interface. The range is 3 through 256, and it defaults to 16 cache entries if the ip tcp compression-connections interface command is not defined with the ip tcp header-compression interface command. A sample configuration is
Router 1

```
host router1
!
interface serial 1/1
 ip 10.20.30.40 255.255.255.0
 ip tcp header-compression
 ip tcp compression-connections 20
```

Router 2

```
host router2
!
interface serial 2/1
 ip 10.20.30.50 255.255.255.0
 ip tcp header-compression
 ip tcp compression-connections 20
```

When using TCP header compression, both sides of the serial line must have ip tcp header-compression specified for proper connectivity.

Enable TCP Path MTU Discovery

The discovery of the largest MTU size over a path between the source and destination router is performed on the initial setup of the TCP connection. Using a dynamic TCP Path MTU Discovery mechanism enables the MTU size to be updated prior to any new TCP connection request. This is done using the interface command:

ip tcp path-mtu-discovery [**age-timer** {*minutes* | **infinite**}]

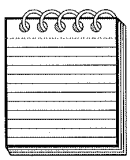
Applying this command on an interface definition enables the router to dynamically determine the largest MTU size between distinct subnets. This is useful to maximize bandwidth for connections that transmit bulk data between the two distinct subnets. The *minutes* variable is the number of minutes the discovery process will wait prior to discovering a new MTU size. The default is 10 minutes and the maximum is 30 minutes. Specifying the infinite value as the variable for the age-timer keyword turns off the age-timer and the MTU size discovery process uses the most current MTU size discovered, until modified by the router administrator.

Performance Enhancements Through Switching Features

Cisco IOS software and microcode together enable the router to move packets between interfaces based on the next-hop destination Layer 3 address. Fast switching is enabled on all interfaces by default. The types of switching supported with Cisco IOS software are

- Process
- Fast
- Optimum
- Silicon Switch Engine (SSE)
- Autonomous
- Distributed
- NetFlow

Access-control lists play a major role in the throughput of packets. Depending on the switching mechanism in use, along with the access filters being applied, packet performance is directly affected. Optimum switching is the most efficient with the highest throughput available on the routers when access filtering is not being used. NetFlow switching is the second-most efficient, even when applying access filters. Fast switching without access filters is the third-most efficient, followed by process switching.



NOTE: *Using optimum and fast switching is compromised to process switching when an access list is applied to an interface. NetFlow switching is the only mechanism that enables switching while access list processing is enabled.*

Enabling Process Switching

In process switching, each frame is sent to the route processor CPU and the CPU encapsulates or de-encapsulates the data. In addition, route selection and filtering is performed during process switching. Process switching is CPU-intensive and slows down packet throughput, using valuable CPU. If you use priority queuing, custom queuing, or filtering, frames become

process-switched. Since fast switching is enabled by default, process switching is administratively enabled by issuing the following interface command:
no ip route-cache

Entering this command on an interface disables all switching and forces process switching.

Enabling Fast Switching

Fast switching is the default method used on Cisco routers when a Cisco cbus interface processor is installed. Fast switching enables the passing of a frame that is destined to a port on another cbus interface processor directly over the router cbus backplane at the interrupt level without copying the frame to the system memory first. This bypasses the involvement of the router processor CPU and therefore provides better performance. Enabling fast-switch processing if it has been disabled is accomplished by entering the interface command:
ip route-cache

NOTE: *A bus is the common electrical connection used by all the interface processors on a router. The Cisco cbus architecture is a high-speed backplane for connecting the router's interface processors.*

Enabling Fast Switching on the Same Interface

Fast switching on the same interface allows the switching of packets back out the interface without passing over the bus. This is useful for NBMA networks like Frame Relay and ATM. To enable the same interface ith fast switching, enter the following interface command:
ip route-cache same-interface

This command is only viable for non-broadcast multi-access (NBMA) networks where subinterfaces are defined to create multipoint connections.

Enabling Autonomous Switching

Autonomous switching occurs without an interrupt or the route processor CPU intervening. Autonomous switching performs better than fast-switch

LANE interfaces also do not support the NetFlow switching feature at this time.

NetFlow switching creates a network flow cache based on the network layer source and destination IP address and the transport-layer port number. A flow is defined in the flow cache using the following fields found in IP datagram:

- Source/destination IP address
- Source/destination port number
- Protocol type
- Type of service
- Input/output interface
- Source/destination subnet mask
- BGP source/destination AS

NetFlow switching applies access list processing to the first packet of an identified flow. Subsequent packets are then handled on a connection-oriented basis, thereby bypassing the access-list processing of subsequent packets within a flow. This is in contrast to the standard access-list processing in which each packet is checked against the applied access list, thereby forcing process switching that causes degradation in the throughput.

Enabling NetFlow switching on an interface disables all other switching modes for the interface. The cache created is based on processing the first packet of flow through either fast or optimum switching processes. After the creation of the cached entry, the packet is flow switched based on the information found in the flow cache. NetFlow switching on a interface is enabled using the following interface command:

ip route-cache Flow

An advantage to using NetFlow switching is the capability to create call detail recording (CDR) records based on the flow entries found in the NetFlow cache. These data can be exported to the Cisco NetFlow Collector application residing on a workstation or server using the following global command:

ip flow-export destination {*hostname* | *ip-address*} *udp-port*

This command instructs the NetFlow feature to send the cached entry statistics of an expired entry to the NetFlow Collector application at *hostname* or *ip-address* using the *udp-port* number supported by the application. For example, if we entered

processing and is advantageous for traffic between cbus interfaces and functions for two-port or multiport bridging on the same router. Enabling autonomous switching is accomplished by entering the configuration interface command:

ip route-cache cbus

Entering the ip route-cache cbus interface command enables both autonomous and fast switching for the interface.

Enabling Silicon Switch Engine (SSE) Switching

The switching of packets using a software solution is now a viable alternative to the previous switching methods that utilized a hardware bus handler processor. The software solution speeds the switching of packets through the cbus since the SSE is a programmable cache. To enable SSE, the following configuration interface command must be entered:

ip route-cache sse

This interface command is specific to the Silicon Switch Processor (SSP) board of the Cisco 7000 series routers, which use a cxbus architecture.

Optimum Fast Switching

The highest switching throughput and most efficient mechanism for switching on the router is optimum switching. The feature is specified to the Cisco 7500 series routers where IP is enabled on Ethernet, FDDI, and serial interface. Optimum switching on serial interfaces is only supported when HDLC is the encapsulation method. To specify optimum switching, enter the following interface command:

ip route-cache optimum

NetFlow Switching

NetFlow switching is used on Cisco RSP7000/7000CI, 7200, and 7500 series routers. NetFlow switching is supported for IP and IP-encapsulated traffic on all interfaces. The exception to this is when more than one input access control list is applied to an ISL/VLAN, ATM, or Frame Relay interface. ATM

LANE interfaces also do not support the NetFlow switching feature at this time.

NetFlow switching creates a network flow cache based on the network layer source and destination IP address and the transport-layer port number. A flow is defined in the flow cache using the following fields found in IP datagram:

- Source/destination IP address
- Source/destination port number
- Protocol type
- Type of service
- Input/output interface
- Source/destination subnet mask
- BGP source/destination AS

NetFlow switching applies access list processing to the first packet of an identified flow. Subsequent packets are then handled on a connection-oriented basis, thereby bypassing the access-list processing of subsequent packets within a flow. This is in contrast to the standard access-list processing in which each packet is checked against the applied access list, thereby forcing process switching that causes degradation in the throughput.

Enabling NetFlow switching on an interface disables all other switching modes for the interface. The cache created is based on processing the first packet of flow through either fast or optimum switching processes. After the creation of the cached entry, the packet is flow switched based on the information found in the flow cache. NetFlow switching on a interface is enabled using the following interface command:

ip route-cache Flow

An advantage to using NetFlow switching is the capability to create call detail recording (CDR) records based on the flow entries found in the NetFlow cache. These data can be exported to the Cisco NetFlow Collector application residing on a workstation or server using the following global command:

ip flow-export destination {*hostname* | *ip-address*} *udp-port*

This command instructs the NetFlow feature to send the cached entry statistics of an expired entry to the NetFlow Collector application at *hostname* or *ip-address* using the *udp-port* number supported by the application. For example, if we entered

```
ip flow-export destination manager 125
```

the NetFlow process on the router sends the statistics to the workstation named manager using UDP port number 125. The data provided by the NetFlow entry can be used for network management and planning, enterprise accounting and departmental charge-backs, ISP billing, and data warehousing/mining for marketing purposes, as well as capacity planning.

Because NetFlow tracks the identified flows along with traffic counts, the process uses more memory and CPU resources than the other types of switching mechanisms. The default size of the NetFlow cache is 64K of memory. Each cache entry requires 64 bytes of storage; thus, a default memory requirement is 4 MB of DRAM. The NetFlow process has a memory management mechanism that attempts to age 30 flow entries using an accelerated timeout when approximately 10 remaining free flow entries are available in the cache. If the number of free flow entries is one, NetFlow automatically ages the oldest 30 flow entries. This mechanism is used each time a free flow entry in the cache is used for a new flow entry. The following global command can be used to change the NetFlow DRAM memory allocation:

ip flow-cache *entries*

The *entries* variable is the number of entries to be supported on the router. The range is 1,024 to 524,288 with the default number of entries being 65,536. Cisco recommends that the default be used as much as possible since increasing the entries value may cause poor network performance and problems.

The record type exported to the NetFlow Collector on a workstation is defined by using the global command:

ip flow-export version {1 | 5 [**origin-as** | **peer-as**] }

If this command is not specified, the default record type is version 1. The version type is dependent on the version supported by the NetFlow Collector application on the workstation. If version 5 is supported, you may also specify either the origin (origin-as) or peer (peer-as) BGP autonomous-system number. Specifying version 5 defaults to neither origin nor peer AS numbers for the record and provides for better performance. The version 5 format also includes the source and destination AS addresses, source and destination prefix masks, and a sequence number to determine lost UDP packets.

The NetFlow collector application on a workstation can request NetFlow data and identify an IP address as the source of the NetFlow record through the use of the following global configuration command:

ip flow-export source ***interface***

The interface variable provides the name of the interface that identifies the router IP address to use as the source of the NetFlow data and to allow the NetFlow Collector application to perform SNMP queries to the router using the IP address of the specified interface. In this instance, it is good practice to use the loopback interface IP address since it is not tied to a physical interface and is always considered active. An example of specifying the NetFlow source address is

```
interface loopback 0
 ip address 10.1.1.1 255.255.255.0
interface etherent 0/1
 ip address 10.2.2.1 255.255.255.0
 ip route-cache flow
ip flow-export destination manager 125
ip flow-export source loopback0
```

Using NetFlow switching provides a powerful tool for network-analysis gathering in order to determine traffic characteristics, which can then lead to network topology changes and performance tuning.

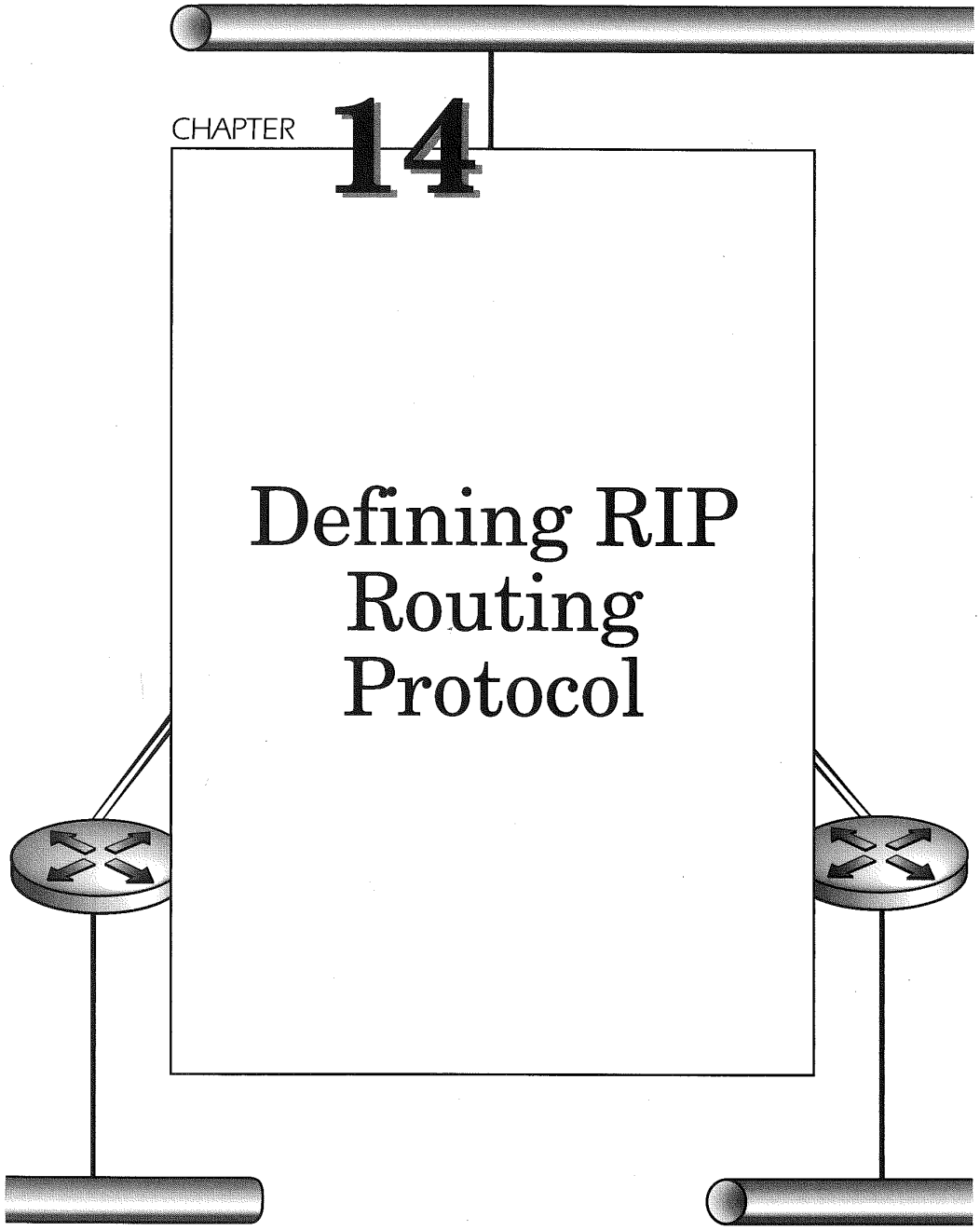
Distributed Switching

When using distributed switching, a cache entry is built and propagated to all switching caches of the VIPs that are enabled for distributed switching. Thus, switching caches are synchronized for all VIPs that are enabled for distributed switching. This allows VIP-to-VIP switching decisions to be made by the inbound VIP, even if the destination is to a different VIP card. The format of the interface command for implementing distributed switching is

```
ip route-cache distributed
```

CHAPTER 14

Defining RIP Routing Protocol



Routing Information Protocol (RIP) is the first open standard used between router vendors and, as such, is a simple protocol. RIP comes in two versions. The initial introduction of RIP is based on IETF RFC 1058 and was developed to support small networks. It is now referred to as RIP version 1. The newest version of RIP, named RIP version 2 (RIP-2), like RIP version 1, is still based on classical distance-vector routing algorithms. RIP-2, however, incorporates some of the advanced features required in today's larger networks such as authentication, route summarization, classless inter-domain routing (CIDR), and variable-length subnet masks (VLSM). None of these advanced features are not supported by RIP version 1.

The Basics of RIP

RIP uses broadcast User Datagram Protocol (UDP) data packets for sending routing table entries to neighboring routers. Since RIP uses UDP as its delivery mechanism, the routing table updates sent to the neighboring routers are not guaranteed. The sending of the RIP table entries between routers defaults to 30 seconds after the initial startup of the router. This "advertising" of routes occurs also between two routers when a router becomes active on a connection to an already active router. Figure 14-1 illustrates this advertising of routing tables.

Routers using RIP expect an update from a neighboring router within 180 seconds. If a routing table update is not received from the neighboring router within this time, the routes to networks through the non-updating router are marked as unusable, forcing returned ICMP network-unreachable messages to originating requesters for resources connected through the non-updating router. Once the received update timer has reached 240 seconds, the non-updating router route entries are removed from the routing table. Packets now received by the router for networks connected through the non-updating router can now be directed to the default network path for this router. The "default route" is learned by RIP or a gateway of last resort is defined with a default RIP metric. Packets with destination networks not found in the routing table are directed out the interface on which the default route is defined.

In Figure 14-2, the Cisco IOS command, show ip route, is entered on a Cisco router. The displayed output identifies the default route for this router to be found at address 10.163.17.5 to network 0.0.0.0, the pseudo network used by Cisco IOS for implementing the default-routing feature. The result of this feature is that any destination IP addresses within the networks found in the routing table that are not found explicitly in the routing

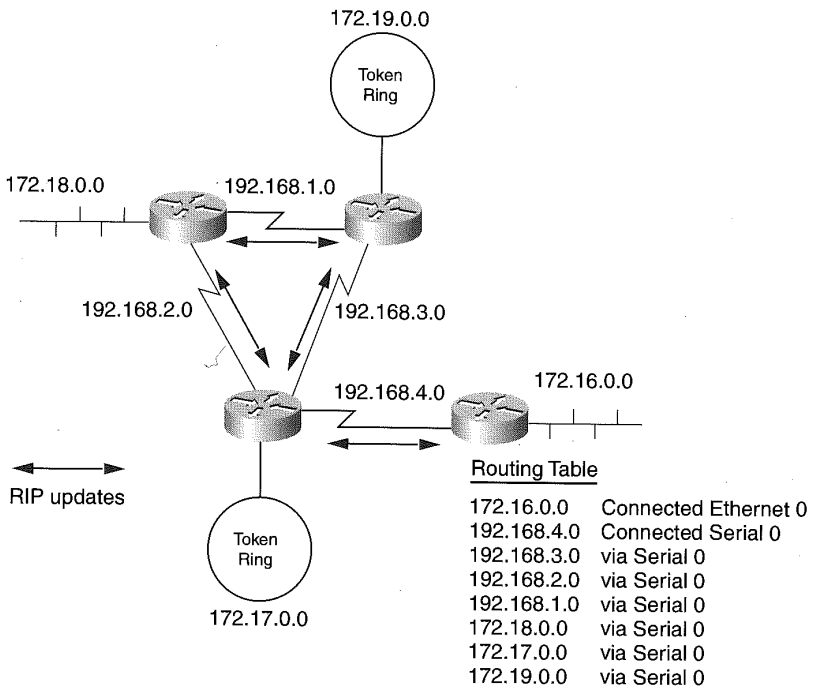


Figure 14-1
Basic RIP route table advertising.

table are routed out of the interface defined by IP address 10.163.17.5. In this case, the ATM interface on slot 3 subinterface 0.2 is used as the default route to subnets within the known and unknown networks.

Why Use RIP as the Routing Protocol?

RIP is the most commonly found routing protocol and, as such, is available on all IP-routing platforms. For example, as depicted in Figure 14-3, when using RIP, a Cisco router can use another vendor's router, such as a Bay Network router, for connecting network resources.

Also found in many networks is the use of Sun workstations as a router supporting RIP as the routing protocol. Because RIP is the most common denominator for connecting different vendor equipment, it is often used in vendor equipment migrations and to grandfather entrenched, antiquated network topologies, along with its ease of implementation.

```
r1 >SH IP ROUTE
Codes: C - connected, S - static, I - IGRP, R - RIP,
M - mobile, B - BGP
 D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF
inter area
 N1 - OSPF NSSA external type 1, N2 - OSPF NSSA
external type 2
 E1 - OSPF external type 1, E2 - OSPF external type
2, E - EGP
 i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2,
* - candidate default
 U - per-user static route, o - ODR

Gateway of last resort is 10.163.17.5 to network
0.0.0.0

R*  0.0.0.0/0  [120/1]  via  10.163.17.5,  00:00:18,
ATM3/0.2
 [120/1] via 10.163.17.2, 00:00:27, ATM3/0.2
 [120/1] via 10.163.17.4, 00:00:16, ATM3/0.2
```

Figure 14-2
The Cisco IOS show ip route command for identifying the default route.

Defining RIP as a Routing Protocol on the Router

Cisco routers enable routing protocols through the use of the following global configuration command:

router

The IOS enables the RIP-routing protocol by entering the following global configuration command:

router rip

After entering this command, the router is running the RIP process, but no RIP routing table is created. Creating the RIP-routing table is accomplished by following the **router rip** global configuration command with the network command. The format of the network command is

network *network-number*

The variable network number is the network number of a network directly connected to the router. For example, in Figure 14.4, three routers are connected to each other using the RIP routing protocol. Each router must specify the network addresses directly connected to it in order for RIP

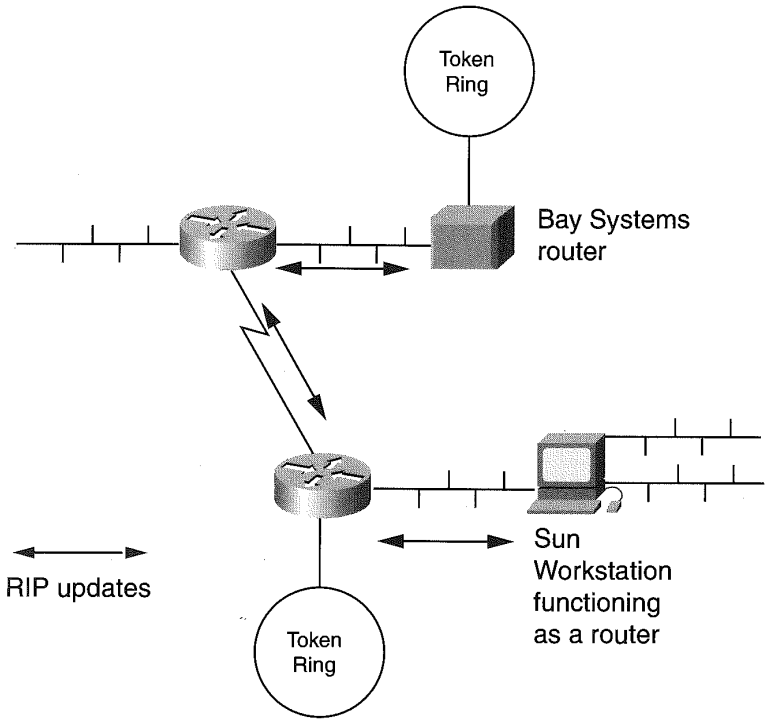


Figure 14-3 RIP used as the common routing protocol in a multi-vendor network topology.

to advertise the IP network to its neighbors. The network number is the Class A, B, or C IP network assigned to the interfaces on the router. The use of subnet masks for identifying the network is not needed on the network command. In Figure 14-4, network 10.0.0.0 is defined to all the routers, while network 192.168.5.0 is defined to only router 1, network 192.168.6.0 is defined to router 2, and network 192.168.4.0 is defined to router 3 only.

The RIP specific commands for each router are as follows:

Router 1:

```
router rip
network 10.0.0.0
network 192.168.5.0
```

Router 2:

```
router rip
network 10.0.0.0
network 192.168.6.0
```

Router 3:

```
router rip
network 10.0.0.0
network 192.168.4.0
```

RIP sends and receives routing updates only on the interfaces matching the networks defined by the network command. If a network is defined on an interface but not defined to the RIP process using a network statement, that network will not be advertised to the rest of the network and is essentially isolated from that part of the network.

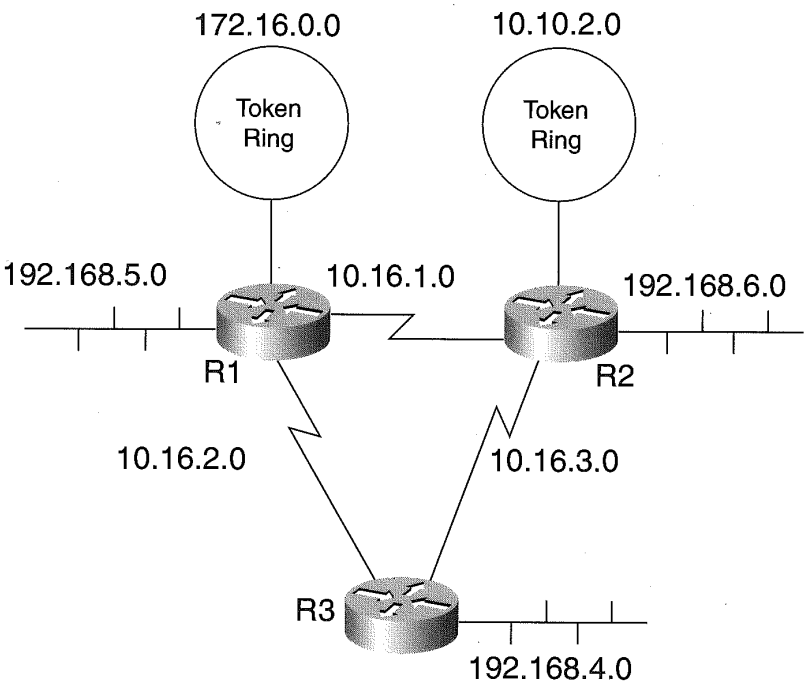


Figure 14-4 *RIP-routing protocol used between Cisco routers.*

As shown in Figure 14-4, the serial interfaces connecting the routers along with the Ethernets on all the routers have their networks in the routing tables. The Token Ring network on router 1, however, is not included in the network statements under the router rip definition and therefore is not in the RIP-routing table. The devices on the Token Ring thus do not receive RIP updates from router 1. The Token Ring segment on router 2, however, is included in the RIP-routing tables because the interface has the IP address, 10.10.2.1, and a mask of 255.255.255.0, falling within network 10.0.0.0, which is defined by the network command on router 2.

Allow Point-to-Point (NBMA) Updates for RIP

X.25 and Frame Relay Non-Broadcast Multi-Access (NBMA) networks provide multiple connections over a single interface. Shown in Figure 14-5, the NBMA topology provides a single physical interface on the router to a public network. The public network provider then delivers the packets or frames to the destination NBMA address or identifier. The Cisco IOS software must be defined in such a scenario to specifically identify the RIP routers over the single interface by specifying the neighboring routers' IP addresses. For example, in Figure 14-5, router R1 must define the IP addresses of router R2 and R3 under the RIP-routing protocol definition to send RIP updates to the adjacent routers R2 and R3 through the single Frame Relay interface.

The router R1 specifies the IP address of the neighboring routers R2 and R3, connecting over the single frame relay interface using the global configuration command neighbor. The format of the command is

neighbor *ip-address*

The *ip-address* value is the IP address assigned to the routers on the far end of the public NBMA network connection. In Figure 14-5, the configuration in router R1 is specified as the following:

Router R1 Configuration:

```
interface serial 0
encapsulation frame-relay
ip address 10.163.7.1 255.255.255.0
!
interface ethernet 0
```

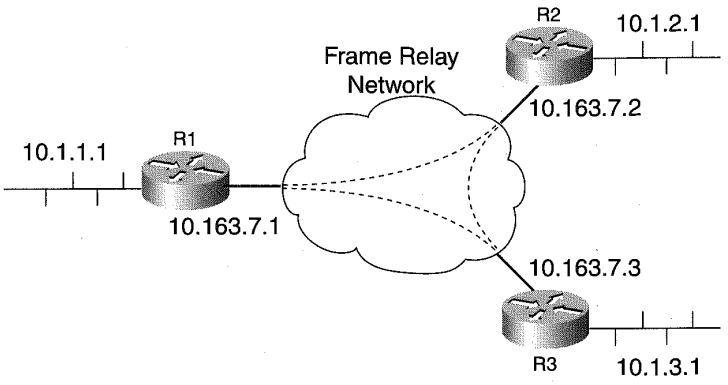


Figure 14-5
RIP-routing updates over a single Frame Relay interface using the neighbor command.

```
ip address 10.1.1.1 255.255.255.0
!
router rip
network 10.0.0.0
neighbor 10.163.7.2
neighbor 10.163.7.3
```

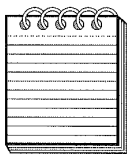
Router R1 sends RIP updates to the IP addresses 10.163.7.2 and 10.163.7.3 and has learned of the DLCI for each of these on the Frame Relay multipoint connection using Frame Relay Inverse ARP. The Frame Relay Inverse ARP builds a table, mapping the DLCI numbers to the IP addresses dynamically. The configuration for router R2 and router R3 are as follows:

Router R2 Configuration:

```
interface serial 0
encapsulation frame-relay
ip address 10.163.7.2 255.255.255.0
!
interface ethernet 0
ip address 10.1.2.1 255.255.255.0
!
router rip
network 10.0.0.0
neighbor 10.163.7.1
neighbor 10.163.7.3
```

Router R3 Configuration:

```
interface serial 0
encapsulation frame-relay
ip address 10.163.7.3 255.255.255.0
!
interface ethernet 0
ip address 10.1.3.1 255.255.255.0
!
router rip
network 10.0.0.0
neighbor 10.163.7.1
neighbor 10.163.7.2
```



NOTE: *It is recommended that dynamic routing be enabled to avoid configuration changes and to allow the network to change dynamically. This is done in frame relay networks by defining subinterfaces as point-to-point. See Chapter 5, "Frame Relay Network Design," for more information.*

Another use of the neighbor command is to selectively define which routers on a given interface will participate in routing updates along with the passive-interface command. In this sample configuration, the workstations connected to R1 LAN segment Ethernet 1 need only to communicate with the R2 router to connect to the WAN and the stations on segment Ethernet 1 router R2. The following configuration demonstrates this concept:

Router R1 Configuration:

```
interface Ethernet 0
ip address 10.17.2.1 255.255.255.0
!
interface Ethernet 1
ip address 10.1.2.1 255.255.255.0
!
router rip
network 10.0.0.0
neighbor 10.17.2.2
passive-interface Ethernet 0
passive-interface Ethernet 1
```

Router R2 Configuration:

```
!
interface Serial 0
ip address 172.16.1.1 255.255.255.0

interface Ethernet 0
ip address 10.17.2.2 255.255.255.0
!
```

```
interface Ethernet 1
ip address 10.2.2.1 255.255.255.0
!
router rip
network 10.0.0.0
network 172.16.0.0
passive-interface Ethernet 1
```

Router R3 Configuration:

```
interface Ethernet 0
ip address 10.17.2.3 255.255.255.0
!
interface Ethernet 1
ip address 10.3.2.1 255.255.255.0
!
router rip
network 10.0.0.0
neighbor 10.17.2.2
passive-interface Ethernet 0
passive-interface Ethernet 1
```

In this configuration example, router R3 does not receive updates from router R1 because the passive-interface command is specified for the Ethernet interface, connecting router R1 on IP subnet 10.17.2.0. The same holds true for R1 for receiving updates from R3. The passive-interface command indicates to the Cisco IOS that the interface specified on the command will not be used for sending routing updates, but the specified interface can receive updates. Router R2 receives and exchanges routing updates with R1 and R3 because the IP address of R2 is defined on the neighbor command of router R1 and R3. Figure 14-6 illustrates this network topology. Further RIP-update reduction is possible by implementing the passive-interface command for LAN segments connecting end-user stations only. This eliminates the unnecessary RIP-update traffic entering the network interface cards of the workstations.

The result of this configuration is that update packets are exchanged only between R1 and R2 and between R2 and R3. Using this method, the interface cards on routers not requiring RIP-routing table updates will not have to process unneeded packets. In another example, if the same three routers are attached through a switch, then the workstations attached to the same switch no longer receive unnecessary RIP-routing updates. This is especially true in switched networks employing ATM LANE topologies

where unicast-directed RIP-routing updates are only forwarded to the destination MAC address within the packet, instead of an all-broadcast destination MAC address, as is normally found with RIP update packets.

Also required here is the no ip slit-horizon command on the interface E0 of R2. Figure 14-7 illustrates this configuration. Using this concept on switched connections saves switch processor resources and eliminates the RIP updates from being propagated to all the workstations also connected on the same switch.

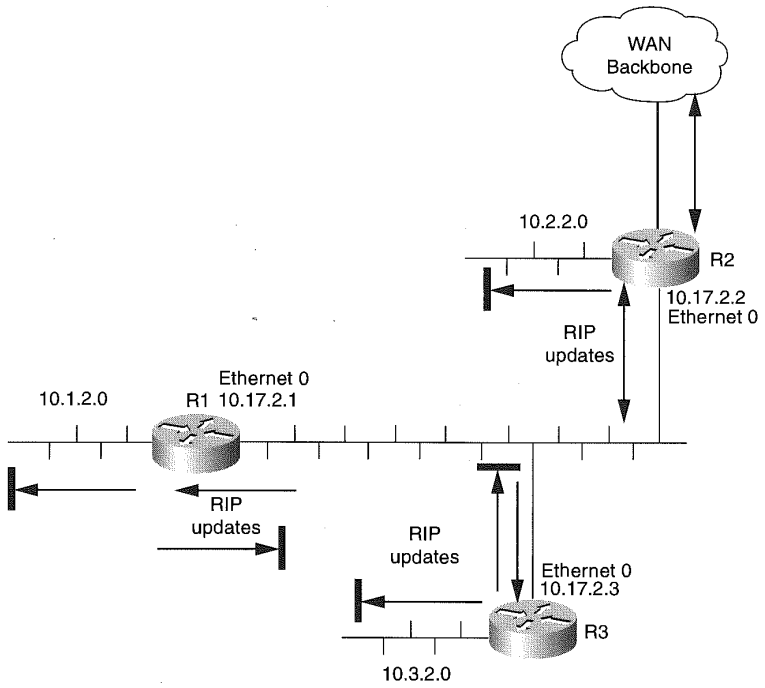


Figure 14-6
Reducing RIP updates on an Ethernet backbone.

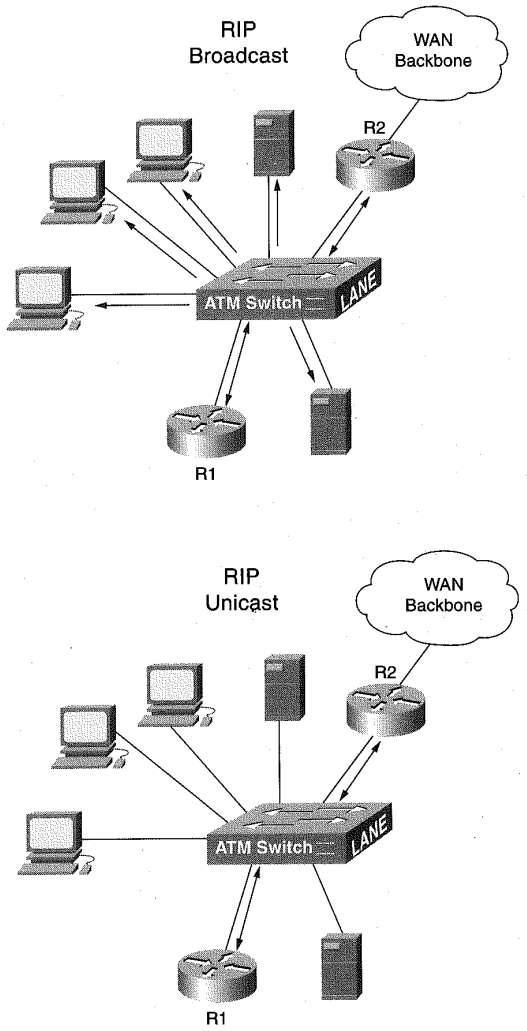


Figure 14-7
Reducing RIP updates in an ATM LANE network.

Specifying the Version of RIP

Two versions of RIP are available using Cisco IOS software. Specifying the following command in the configuration

```
router rip
```

enables the Cisco IOS to receive RIP version 1 and version 2 packets. However, without specifically specifying the following commands

```
router rip
version 2
```

only RIP version 1 is sent. Specifying the command version 2 as above indicates to the Cisco IOS that only RIP version 2 packets are sent and received. The Cisco router is instructed to send and receive RIP version 1 packets only by the following commands:

```
router rip
version 1
```

Table 14-1 lists the various combinations of RIP packets being sent and received as the default behavior of RIP on a Cisco router.

Because there is the potential for two versions of RIP, the Cisco IOS software allows the router administrator to specify the version used on any specific interface to send and receive packets. This is useful in networks where legacy configurations are using UNIX workstations as routers. These older UNIX systems use RIP version 1 packet formats. For example, in Figure 14-8, a Sun workstation has multiple network interface cards and acts as a router to the various Ethernet segments. Subnet 10.1.1.0 on the Sun workstation is connected to the department backbone that connects the Cisco router.

The other two subnets on the Sun workstation, 10.1.2.0 and 10.1.3.0, require connectivity to the network 10.1.4.0, which is connected to the Cisco router R2. The RIP update between the routers is version 2. The Sun workstation acting as a router needs to learn of the 10.1.4.0 network through Cisco router R1.

The configuration on router R1 to accomplish RIP version 2 updates for router R2 and RIP version 1 updates for the Sun workstation is as follows:

Table 14-1

RIP Packet Default Support Based on RIP Definitions

Command Sequence	RIP Packet Support
router rip	Receives version 1 and version 2 / Sends version 1
router rip version 1	Receives and sends version 1 packets only
router rip version 2	Receives and sends version 2 packets only

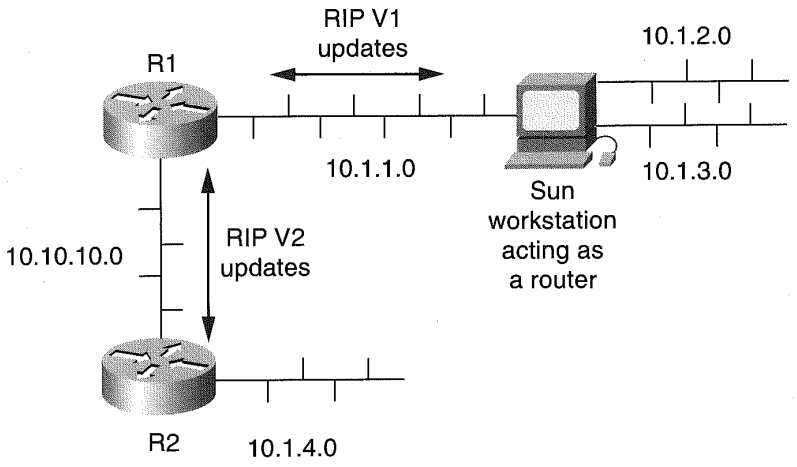


Figure 14-8
RIP version packet control on router interfaces.

Router R1 Configuration:

```
interface Ethernet 0
description To Sun workstation
ip address 10.1.1.2 255.255.255.0
ip rip send version 1
ip rip receive version 1
!
interface Ethernet 1
description To router R2
ip address 10.10.10.1 255.255.255.0
!
router rip
 version 2
 network 10.0.0.0
```

There is no need to specifically define the RIP version type on the interface Ethernet 1 since it defaults to the packet specified under the router rip command. In this case, interface Ethernet 1 sends and receives version 2 packets only. Table 14-2 lists the interface commands and their resulting functions.

If a Cisco router is placed on the departmental backbone, connecting router R1, and the Sun workstation acts as a router while RIP version 2 is used between the routers to support variable length subnet masks (VLSM) for networks connected to router R3, as shown in Figure 14.9, the configuration on router R1 is modified as follows:

Table 14-2 RIP Packet Version Interface Specific Commands

RIP Version Interface Command	Resulting Function
ip rip send version 1	Sends only RIP version 1 packets.
ip rip receive version 1	Accepts only RIP version 1 packets.
ip rip send version 2	Sends only RIP version 2 packets.
ip rip receive version 2	Accepts only RIP version 2 packets.
ip rip send version 1 2	Sends RIP version 1 and version 2 packets.
ip rip receive version 1 2	Accepts both RIP version 1 or 2 packets.

Router R1 Configuration:

```
interface Ethernet 0
description To Sun workstation and router R3
ip address 10.1.1.2 255.255.255.0
ip rip send version 1 2
ip rip receive version 1 2
!
interface Ethernet 1
description To router R2
ip address 10.10.10.1 255.255.255.0
!
router rip
 version 2
 network 10.0.0.0
```

Router R1 requires the specification of both packet versions on interface Ethernet 0. This is because the packet version specified on the interface overrides the global default packet version defined under the router rip definition.

Enabling RIP Version 2 Authentication

RIP Version 2 supports router authentication as a means of providing security for routing table updates. RIP Version 1, however, does not support authentication. When using authentication, Cisco routers are protected from both advertising routes to and receiving routes from promiscuous

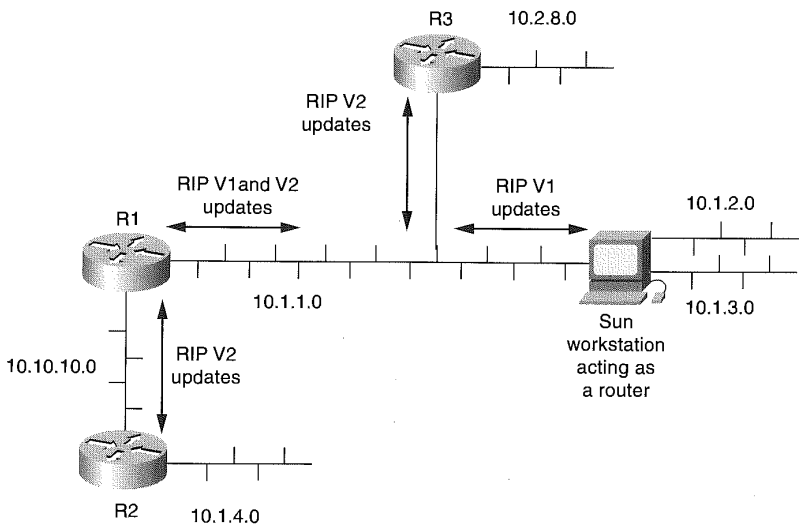


Figure 14-9
Use of RIP version 1 and 2 for sending and receiving on the same interface.

unauthorized routers. Authentication is defined on the interface level and hence provides for discreet selection of the interfaces that must use authentication.

Authentication is determined by the configuration of a key chain specific to the interface. The key chain is identified by using the following interface command:

ip rip authentication key-chain *name-of-chain*

The *name-of-chain* operand is the name given to the valid group of keys. Specifying the **ip rip authentication key-chain** command enables RIP version 2 authentication. If this command is not specified on an interface using RIP Version 2, then authentication is not performed.

Along with enabling authentication, the mode of authentication can be specified as well. The mode of authentication defaults to plain clear text, which becomes a security risk on connections that are deemed unsecured. Encrypted authentication must be specifically specified. The following interface command is used to define the type of mode used for authentication on an interface:

ip rip authentication mode {text | md5}

The **text** value is the default and should only be used on connections where security is not an issue. The **md5** value implements a keyed authentication based on the MD5 standard.

Authentication is enabled using the ip rip authentication key-chain interface command, but the authentication keys must be defined in order for authentication to occur. Authentication keys themselves are defined using the following global configuration commands:

key chain *name-of-chain*
key *number*
key-string *text*
accept-lifetime *start-time* {**infinite** | *end-time* | **duration** *seconds*}
send-lifetime *start-time* {**infinite** | *end-time* | **duration** *seconds*}

The value of the *name-of-chain* variable on the key-chain command must match a defined **ip rip authentication key-chain** command on an interface to exercise the authentication key chain. The *number* variable of the **key** command identifies the number of the key being defined. The *text* variable on the *key-string* command is the authentication text identified by the **key** *number* command used for authentication. Multiple keys can be configured. The key numbers supplied in the key chain definition are searched from lowest to highest, and the first valid key found is then authenticated. The router sends only one authentication packet despite multiple key definitions. The **key** *number* value in combination with the interface associated with the packet is used by the authentication algorithm and MD5 authentication to uniquely identify the authentication key.

The keys can be set to be active during specific times of a given day. This is done with the **accept-lifetime** and **send-lifetime** commands. The *start-time* variable of the accept-lifetime and send-lifetime commands defines the beginning time when the key will be in use. The *end-time* of the accept-lifetime and send-lifetime commands defines the end on a given day when the key will no longer be in use. The format of the *start-time* and *end-time* variable is

hh:mm:ss month day year

The *hh:mm:ss* variable is the two-digit 24-hour clock representation of the time of day. The month variable is the first three letters of the month. The day is the number of the day of the month, and the year variable is the four-digit number for the year. The *start-time* and *end-time* variables

default to an infinite time period starting from January 1, 1993. The **infinite** keyword can be used as the end-time value, indicating that the key never times out after the value of the *start-time* variable has been reached. The **duration** keyword, followed by a valid value for the *seconds* variable, can also be used instead of the end-time variable value to define the usage length for the key. The following is an example of a configuration using RIP Version 2 and authentication between two Cisco routers.

Router R1 Configuration:

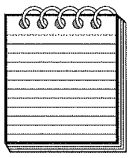
```
interface Ethernet 0
 ip address 10.163.8.1 255.255.255.0
 ip rip authentication key-chain colors
 ip rip authentication mode md5
!
router rip
 network 10.0.0.0
 version 2
!
key chain colors
 key 1
 key-string blue
 accept-lifetime 07:00:00 Feb 28 1999 duration 7200
 send-lifetime 08:00:00 Feb 28 1999 duration 3600
 key 2
 key-string green
 accept-lifetime 00:00:00 Mar 1 1999 infinite
 send-lifetime 00:30:00 Mar 1 1999 infinite

Router R2 Configuration:

interface ethernet 0
 ip address 10.163.8.2 255.255.255.0
 ip rip authentication key-chain colors
 ip rip authentication mode md5
!
router rip
 network 10.0.0.0
 version 2
!
key chain colors
 key 1
 key-string blue
 accept-lifetime 07:30:00 Feb 28 1999 duration 7200
 send-lifetime 06:30:00 Feb 28 1999 duration 3600
 key 2
 key-string green
 accept-lifetime 23:30:00 Feb 28 1999 infinite
 send-lifetime 00:00:00 Mar 1 1999 infinite
```

In these configurations, the two routers first use the keys identified by number 1 because the search order is from lowest to highest. The routers then use the number 2 key to authenticate routing updates between them.

The start time on router R1 is defined to ensure the acceptance and delivery of RIP version 2 packet updates to router R2, given a 30-minute cushion on the time differences between the routers. The second key (number 2) on each router takes effect once the clocks on each router have met the start-time criteria and last indefinitely.



NOTE: *Both routers must use RIP Version 2 and have authentication enabled, in addition to the accept and send lifetime value being coordinated. Both routers must also specify the same key-string that will then force the router to examine the accept and send lifetime values.*

Disable RIP Version 2 Route Summarization

Route summarization is enabled by default when using the RIP Version 2 routing protocol. The Cisco IOS software summarizes the RIP Version 2 routes across classful network boundaries. This is in contrast to RIP Version 1, which is also a classful routing protocol but does not communicate the subnets across different major networks.

Route summarization is the process of summarizing routes with long masks to routes with shorter masks that still meet the initial routing requirement. Classful routing protocols only use the full prefixes of each IP network classification if a subnetted network is directly connected to the router, meaning 8 bits for a Class A, 16 bits for a Class B, and 24 bits for a Class C address. Summarization is useful in RIP Version 2 due to its support of VLSM. In Figure 14.10, for example, router R1 connects to router R2 through serial interface 0 and router R3 through serial interface 1.

The networks connected to router R2 and router R3 are summarized in the RIP Version 2 update packets to router R1 from router R2 and R3. RIP Version 2 summarizes the following network addresses into two routing entries using RIP Version 2:

Using the router sh ip route command on router R1, the summarized routing entries will be displayed as

```
R 172.16.0.0/16 [120/1] via 172.30.1.1, 00:00:01, Serial0/0
R 172.20.0.0/16 [120/1] via 172.31.1.1, 00:00:01, Serial0/1
```

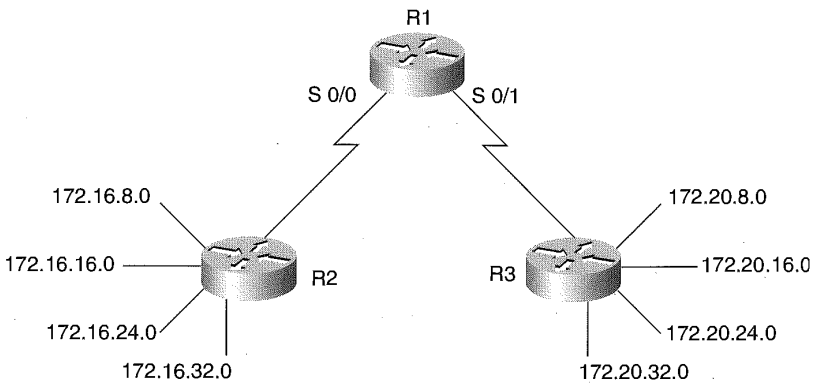


Figure 14-10 RIP Version 2 network configuration used for route summarization.

Table 14-3 Network addresses to be summarized in Figure 14-10.

IP Subnet	Subnet Mask
172.16.8.0	255.255.255.0
172.16.16.0	255.255.255.0
172.16.24.0	255.255.255.0
172.16.32.0	255.255.255.0
172.20.8.0	255.255.255.0
172.20.16.0	255.255.255.0
172.20.24.0	255.255.255.0
172.20.32.0	255.255.255.0

Using route summarization with RIP Version 2 reduces router overhead and bandwidth required to send routing updates.

Route summarization is the desired approach when using RIP Version 2. If a subnet is disconnected, however, the routers will not be able to correctly advertise the subnets. Figure 14-11 illustrates this issue. Router R2 has subnet 172.20.8.0 and router R3 has subnet 172.20.16.0, causing a disconnect of the subnets. Router R1, after receiving summarized entries from router R2 and R3, indicates two ways to the 172.20.0.0 network, which is not really the case.

The sh ip route command displays the following on router R1, based on router summarization updates from R2 and R3:

```
R 172.20.0.0/16 [120/1] via 172.31.1.1, 00:00:01, Serial0/0
R 172.20.0.0/16 [120/1] via 172.30.1.1, 00:00:01, Serial0/1
```

The resulting routing table will cause connectivity problems for two reasons. First, due to the fact that the hop count is the same, the router will load balance packets between serial interface 0/0 and serial interface 0/1. Secondly, this will cause only 50 percent of the packets to successfully reach their true destination. This type of topology can be corrected by disabling route summarization using the following router configuration mode command:

no auto-summary

Entering this in the configuration for routers R2 and R3 will disable summarization and cause the full subnet and host-routing information to be transmitted to router R1. The sh ip route command issued on router R1 will now display the following entries:

```
R 172.20.8.0/24 [120/1] via 172.30.1.1, 00:00:01, Serial0/0
R 172.20.16.0/24 [120/1] via 172.31.1.1, 00:00:01, Serial0/1
```

The no–auto-summary command on router R2 and R3 forces the RIP Version 2 updates to advertise the full mask for each of the subnets. Now packets through router R1 for resources on subnet 172.20.8.0 will only be directed out serial interface 0/0. Likewise, packets through router R1 for resources on subnet 172.20.16.0 will only be directed out serial interface 0/1.

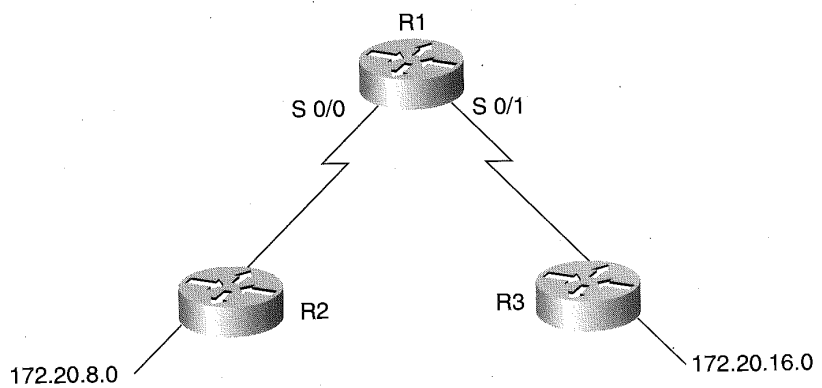


Figure 14-11 Disconnected networks and RIP V2 route summarization.

Disabling the Validation of Source IP Addresses

Many network configurations require multiple IP network definitions. RIP validates the source IP address of the routing update being received. If the source IP address of the routing update packet being received is on the same network as the IP address of the interface receiving the update, the routing update packet is processed. If the received routing update packet source IP address is not on the same network as the interface receiving the routing update packet, then the packet is discarded. A scenario where this is possible is the use of secondary IP addressing on interfaces. Figure 14-12 illustrates this configuration.

Router R1 in Figure 14-12 is connected to the same Ethernet segment as router R2 via a hub. Router R1 has its Ethernet interface defined as IP address 172.16.1.1, using 255.255.0.0 as the subnet mask. This same interface on router R1 also has a secondary IP address defined as 172.17.1.1, using 255.255.255.0 as the subnet mask. Router R2 connected to the same hub defines its Ethernet connection as 172.17.1.2, using 255.255.255.0 as the subnet mask. When the routers begin sending RIP-routing updates, the following message can be observed in the log of router R2, using the Debug IP Routing Information Protocol privileged command:

Ignored v1 update from bad source 172.16.1.1 on Ethernet0

This message indicates that the RIP update packets being received on Ethernet 0 of router R2 are not on a connected network. The source IP address of the routing packet received from router R1 is not a network connected to interface Ethernet 0 on router R2 and is hence discarded by router R2. To overcome this in the network configuration presented in Figure 14-12, the validation of the source IP address can be disabled using the following command:

no validate-update-source

Using this command, the validation process of incoming RIP update packets is disabled and the router, which in this case is router R2, receives the routing updates. The configuration of the two routers in Figure 14.12 that utilize this command is as follows:

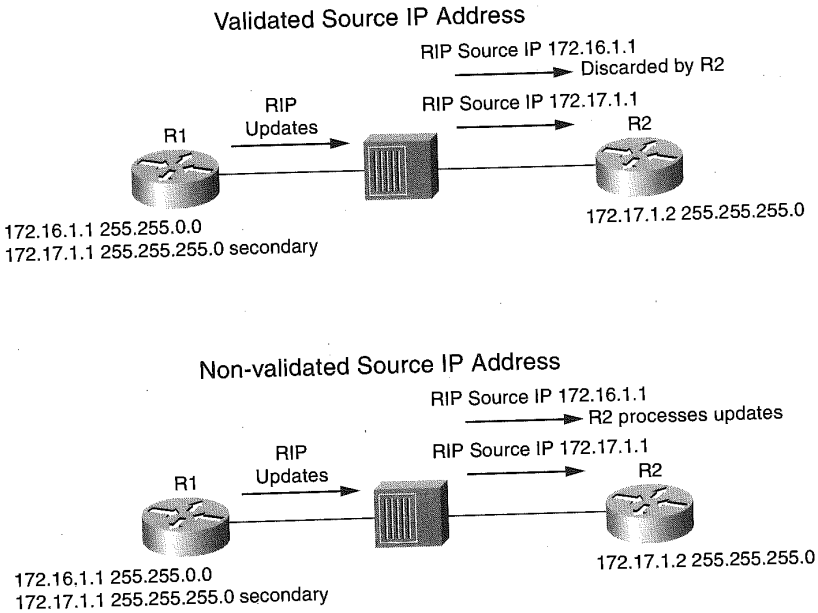


Figure 14-12
The off-network routing configuration and the effect of disabling source IP address validation.

Configuration router R1:

```
interface Ethernet 0
ip address 172.17.1.1 255.255.255.0 secondary
ip address 172.16.1.1 255.255.0.0
!
router rip
 network 172.16.0.0
 network 172.17.0.0
```

Configuration router R2:

```
interface Ethernet 0
ip address 172.17.1.2 255.255.255.0
!
router rip
 network 172.17.0.0
 no validate-update-source
```

This enables router R2 to place updates provided by router R1 source IP address 172.16.1.1 into its routing table. It is recommended that this command not be used because it can lead to confusion in the routing tables and possibly cause routing loops.

Reducing Routing Loops with Split-horizon

A router advertising routes out of an interface that the routing information was received on frequently causes routing loops. The split-horizon feature employed for distance-vector routing protocols like RIP reduces the possibility of routing loops by blocking routing updates to connected routers from which the routing update was originally received. Figure 14-13 diagrams this scenario.

This type of scenario is most prevalent on broadcast type IP networks such as LANs. As shown in Figure 14-13, the routing advertisements received by router R1 for networks connected to router R2 are not sent to router R2 from router R1 in its routing update packets. Likewise, router R2 does not send routing advertisements to router R1 for networks attached to router R1.

Non-broadcast networks like Frame Relay, however, require split-horizon to be disabled. The Cisco IOS automatically disables split-horizon for Frame Relay, ATM, and SMDS networks when subinterfaces are defined on the physical interface, but split-horizon is enabled for all other network connections. If split-horizon is disabled on an interface connecting a packet-switched network like X.25, it must be disabled on all other routers connecting to the same packet-switched network.

Spilt-horizon is disabled using the following interface configuration command:

no ip split-horizon

Enabling split-horizon for a specific interface is accomplished by entering the following interface command:

ip split-horizon

Tuning RIP Update Packet Delays

In many situations, the network topology consists of a high-end Cisco router acting as a hub router that connects multiple remote routers. The remote routers are usually lower-end routers servicing small remote offices. The high-end Cisco router at the hub is a higher-performance router and hence can cause the lower-end remote office router CPU to become overutilized

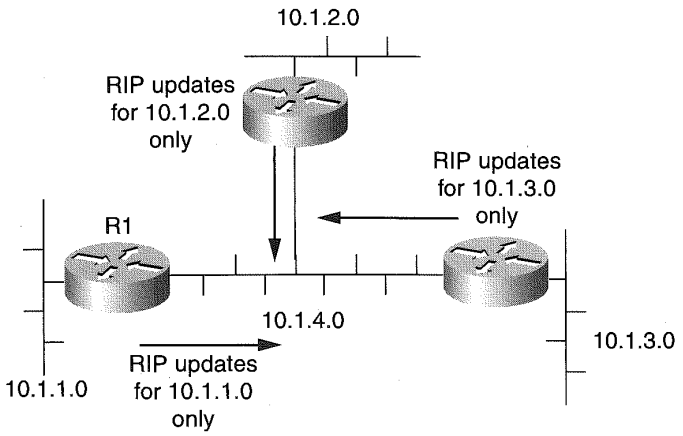


Figure 14-13
The effect of split-horizon on broadcast networks.

when processing RIP updates. The Cisco IOS allows for the router administrator to delay the packets for delivery through the use of the following router configuration command:

output-delay *delay*

The router will delay the sending of RIP update packets to all RIP interfaces by the value defined for the delay variable in the output-delay router command. The delay variable can be specified in the range of 8 to 50 expressed in milliseconds.

RIP Updates and the Effect on Bandwidth

RIP-routing updates are sent every 30 seconds to all RIP routers on interfaces supporting the RIP-routing protocol. Remember that the RIP process on the router can determine which interface to send RIP updates on by using the networks specified under the router-rip configuration. Because RIP automatically sends updates every 30 seconds, there can be a sudden increase in temporary bandwidth required.

An RIP update packet can have up to 25 route entries. The RIP update header is 36 bytes and each route entry is 20 bytes. A single routing update packet can then be as large as 536 bytes.

To determine the amount of bandwidth used by RIP in your network, use the following formula:

# of advertised routes / 25 entries = # of packets
# of packets x 36 bytes of header = # of bytes for headers
# of advertised routes x 20 bytes for each entry = # of bytes for entries
# of bytes for headers + # of bytes for entries = # of bytes (every 30 seconds)
# of bytes / 30 second updates x 8 bits per second = bandwidth required for sending RIP updates

Suppose the RIP network has 200 advertised routes, for example. Using the formula, the bandwidth used every 20 seconds on RIP interfaces is as follows:

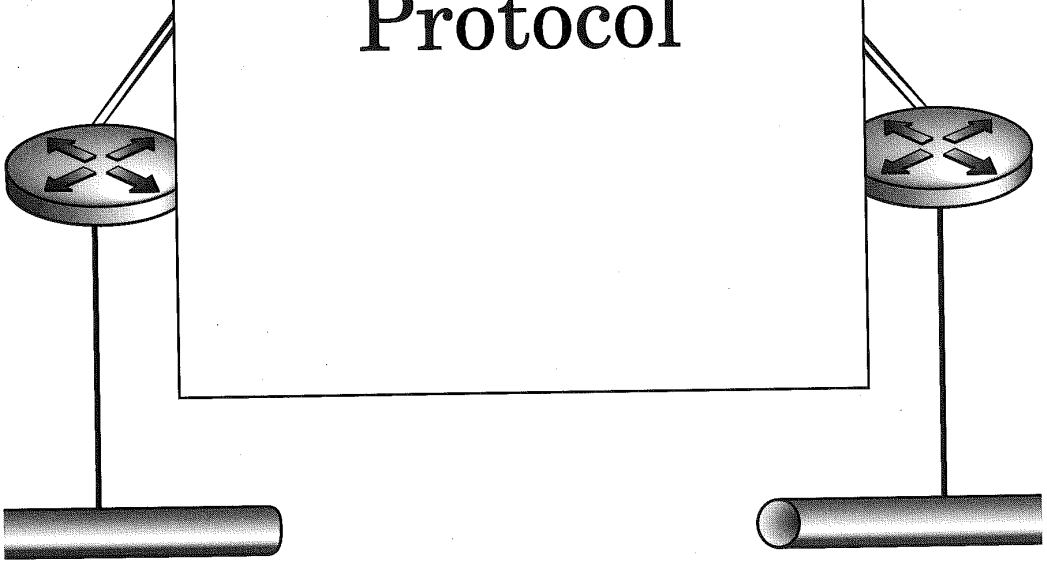
200/25 = 8 packets
8 x 36 = 288 bytes for headers
200 x 20 = 4,000 bytes for entries
288 + 4000 = 4,288 bytes
(4288/30) x 8 = 1,143 bps or 1.1 Kbps

This does not seem like much for many serial-line connections. But suppose the serial line is connecting to a Frame Relay network with 50 DLCIs connecting to the interface. The bandwidth requirement is now 50 x 1.1 Kbps, or 50.5 Kbps on the serial interface. This can lead to severe bandwidth constraints on the Frame Relay connection for remote locations where 56 Kbps is often used as the Frame Relay bandwidth.

Connecting 50 remote locations over the 56-Kbps Frame Relay connection in this scenario will require a full-committed information rate (CIR) of 56 Kbps to guarantee that the RIP update packets will be sent through the network. If the CIR is low, then there is the potential that the RIP updates will be dropped by the public Frame Relay network switches and hence the routers will not be updated appropriately.

CHAPTER 15

Configuring IGRP Routing Protocol



Interior Gateway Routing Protocol (IGRP) is a Cisco proprietary distance-vector-based routing protocol. IGRP uses metrics (termed vectors) to determine the best route to a network. IGRP uses delay, bandwidth, reliability, and load for determining the calculated optimal route. Cisco IOS allows router administrators to set weights to these metrics in order to influence the metric calculation.

Cisco IGRP enhances routing through the use of multiple paths between two networks. IGRP can use these equal-cost paths to perform load balancing on a packet-by-packet or destination-by-destination round-robin fashion. If a connection becomes inoperative in a dual equal-bandwidth configuration, IGRP can perform an automatic switchover to the remaining connection. In addition, IGRP provides the mechanism with multiple paths of unequal cost between routers in order to improve performance. In such a situation, the lower-cost path will be used more often than the higher-cost one.

IGRP has three flavors of route entries in its routing table: interior, system, and exterior routes. As shown in Figure 15-1, interior routes are advertisements describing paths between subnetworks attached to a router interface. Interior route entries include the subnet mask for the interface. If the network defined to a router interface is not subnetted, IGRP does not advertise this route as an interior route. Instead, non-subnetted interfaces are advertised using system routes.

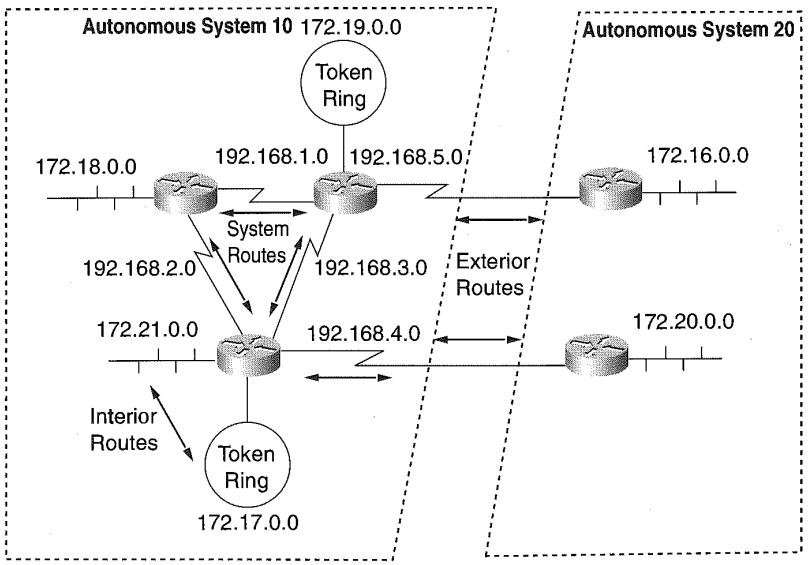


Figure 15-1 IGRP route types within a network topology.

System route advertisements describe routes between networks within the same autonomous system. IGRP uses the concept of autonomous systems (AS) to isolate routing domains. The system routes are determined by the network definition of an interface and system route information gathered from the receipt of other IGRP routers within the autonomous system. The system routes describe the path on the network boundary and do not include subnet mask information.

The final route advertisement describes paths to exterior routes. Exterior routes describe to other autonomous systems the path that is used by the routing in determining the gateway of last resort. The gateway of last resort is chosen from the exterior routes and is chosen for packets that are destined for networks that are not known within the autonomous system routing tables. In networks with multiple routers, connecting the same two autonomous systems can have different gateway-of-last-resort routes.

Managing IGRP Updates and Route Advertisements

IGRP, like RIP, has a set period for delivering the routing table entries between adjacent routers. IGRP defaults the period to 90 seconds. This is a modifiable time period, but most networks operate smoothly using this default value.

The IGRP process must first recognize that an interface has become active and the IP network address assigned to the interface is defined for IGRP routing. After this recognition, the router broadcasts the IGRP routing table out of the interface. At this point, the IGRP process begins the 90-second timer.

Like RIP, IGRP uses UDP to send the route entries and hence the router has no knowledge as to whether the updates were successfully delivered to an adjacent router. Because of this, IGRP will declare a route as inaccessible if an update is not received from a router on the interface defined as the outbound port within three routing updates. That is, 270 seconds must pass by for the route to be labeled as inaccessible. If seven route-update periods have elapsed (630 seconds), the IGRP process removes the route from the routing table.

One of the advantages of using IGRP over RIP version 1 is the reduced time for route convergence. IGRP implements three different algorithms to minimize convergence time: holddowns, split-horizons, and poison-reverse updates.

The Holddown Algorithm for IGRP

The holddown algorithm instructs the IGRP process in updating and then propagating a new calculated route due to a connection that has become inoperable for a period of time. The time period calculated by IGRP is just a little longer than the time period calculated to update all the routers within the network of the new calculated route(s). Using this holddown algorithm, a Cisco router using IGRP avoids erroneous routing updates.

For example, in Figure 15-2, a router (R1) looses its connection to router R2, which detects the outage due to the lack of routing updates. Router R2 recalculates new routes for networks once reachable through router R1 and sends the update routes to its neighbors. This causes router R3 in Figure 15-2 to then recalculate its routes to the networks connected to router R1 based on information from router R2. This triggered update mechanism is propagated throughout the network.

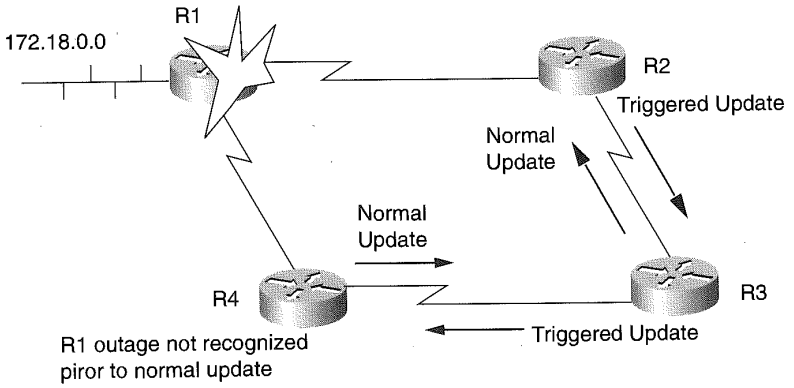
Herein lies the problem. Some routers in the network, not yet being updated with the outage, will send the previously known viable calculated route, as is the case with router R4. Router R4 sends its update prior to recognizing the outage of router R1, as if it is still available. Router R3 now contains an update that router R4 can get to the networks on router R1. This update is sent to router R2, which then calculates a new route for networks off of router R1 based on the originating update from router R4. Router R1 now believes that it can route packets destined for networks connected to router R1 through router R3, which then in turn will route the packets to router R4.

Now router R4 has recognized that connectivity to router R1 is down and sends an update to R3, which then forwards an update to R2. The holddown mechanism prevents router R2 from updating its routing table with the outage and then sends an updated routing table after the above scenario has taken place, thus avoiding an erroneous routing table information and long convergence time.

The Split-horizon Algorithm for IGRP

Split-horizons work for IGRP, just as with RIP. Route information received on an interface is never sent back out the same interface. Without split-horizon in effect, a routing loop, as shown in Figure 15-3, is possible.

In Figure 15-3, a routing loop is created when the Ethernet to network 192.168.7.0 becomes inactive if split-horizon is not implemented. Router R2, having learned of the existence of network 192.168.7.0 from router R1,



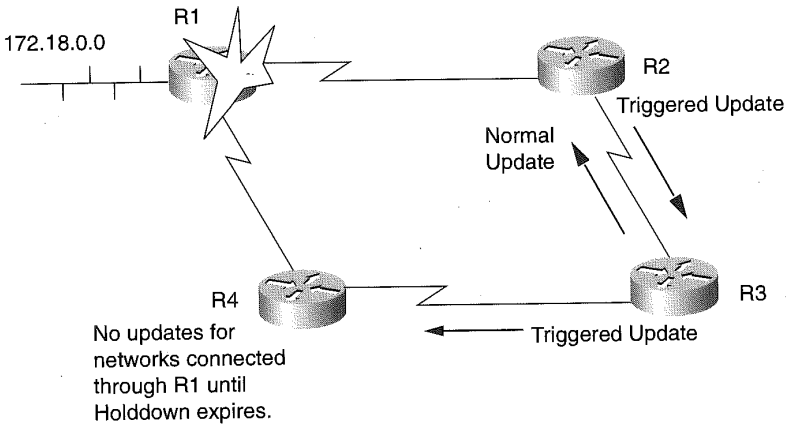


Figure 15-2 IGRP-routing updates of a lost connection with and without the use of holddown.

sends an update to router R1 that network 192.168.7.0 can be reached in two hops. Router R1 now sends an update back to router R2, increasing the hop count and updates the routing table in R2. Now router R2 does it again. This iteration continues until delay has reached 167 seconds and the network connection is marked as unreachable. Split-horizon prevents such a loop from happening. Holddowns, in concert with split-horizon for IGRP-based routing environments, add a greater degree of stability to the network than with RIP V1.

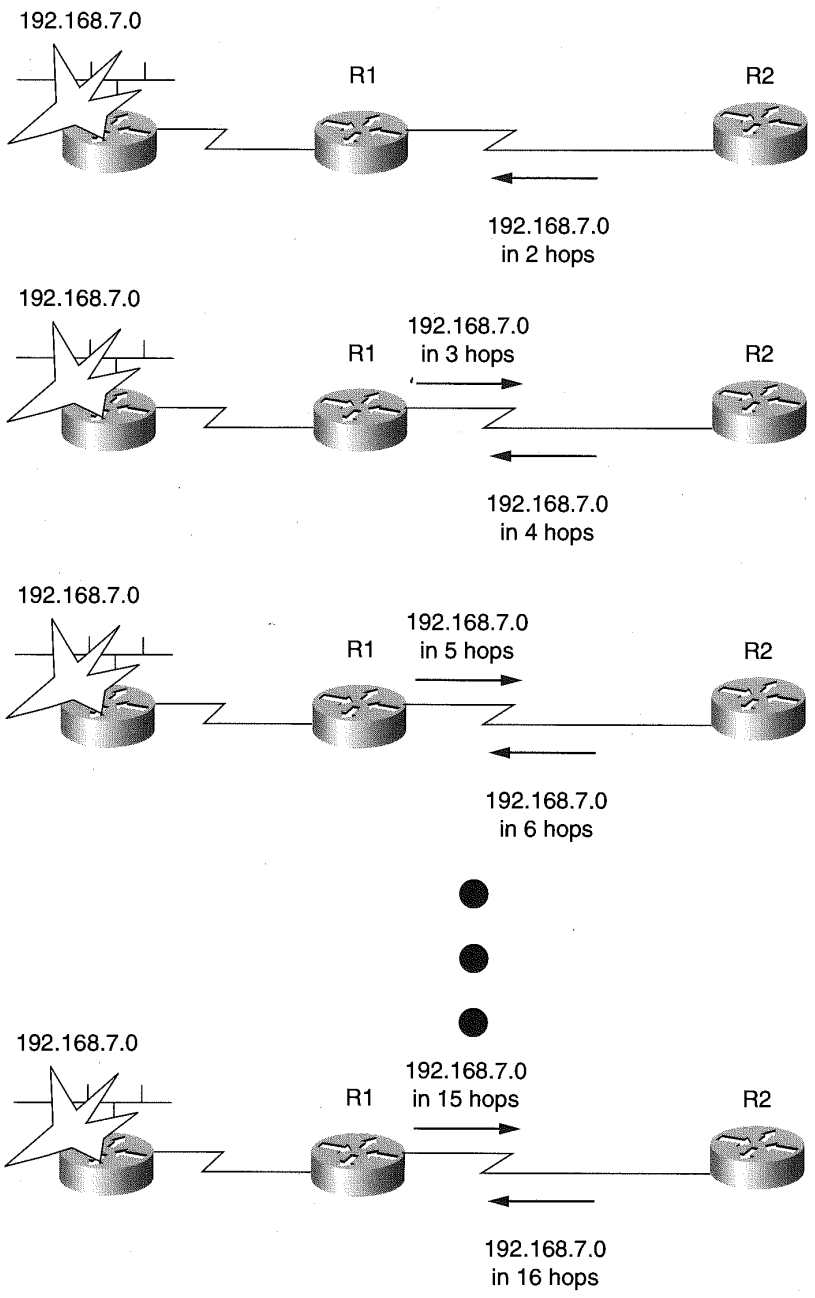


Figure 15-3
A routing loop created from split-horizon being disabled.

The Poison-reverse Updates Algorithm for IGRP

As a router receives an update from a neighbor with an increased metric, it marks the route as unusable. The router will update the path as usable after it has received a follow-up update confirming the metric change. The confirmation happens within the periodic routing-table update time of 90 seconds. This enables IGRP routers to defeat larger routing loops that may occur within the network.

As the increased metric indicates a loop, it does not necessarily mean that there is a loop. IGRP sends a poison-reverse update to adjacent routers that instructs the IGRP process to place the route in question in holddown. Cisco's IGRP sends poison-reverse updates when a metric has increased by a factor of 1.1 or greater.

Defining IGRP as a Routing Process

The following global configuration command is used to identify to the Cisco IOS that a routing protocol is being defined:

router

Cisco IOS starts the IGRP routing protocol process by entering the following global configuration command:

router igrp *autonomous-system*

The ***autonomous-system*** variable is either a registered or an internal-use-only autonomous-system number that identifies the routes of the IGRP network. This value is also used to identify the IGRP process executing in the router. It can also be referred to as the process-id for the IGRP route process being defined. Each router using IGRP must specify the same autonomous-system number in order for the routers to exchange and update IGRP routing tables.

The routing tables are defined by entering the network command under the router igrp command. The format of the network command is

network *network-number*

The variable *network-number* is the network number of a network directly connected to the router. For example, in Figure 15-4, the three routers are connected using IGRP as the routing protocol. Each router specifies the network address of a directly connected network under the router IGRP definition. This enables each router to advertise the IP network to its neighboring IGRP routers. The *network-number* is the Class A, B, or C IP network assigned to the interfaces on the router. The use of subnet masks for identifying the network is not used on the network command.

Also in the figure, network 10.0.0.0 is defined to all the routers, while network 192.168.15.0 is defined to only router 1, network 192.168.16.0 is defined to router 2, and network 192.168.14.0 is defined to router 3 only.

The IGRP configuration commands for each router are as follows:

Router 1:

```
interface serial 0
 ip address 10.16.1.1 255.255.255.0
!
interface serial 1
 ip address 10.16.2.1 255.255.255.0
!
interface Ethernet 0
 ip address 192.168.15.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
network 192.168.15.0
```

Router 2:

```
interface serial 0
 ip address 10.16.1.2 255.255.255.0
!
interface serial 1
 ip address 10.16.3.2 255.255.255.0
!
interface Ethernet 0
 ip address 192.168.16.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
network 192.168.16.0
```

Router 3:

```
interface serial 0
 ip address 10.16.2.3 255.255.255.0
!
interface serial 1
 ip address 10.16.3.3 255.255.255.0
!
```

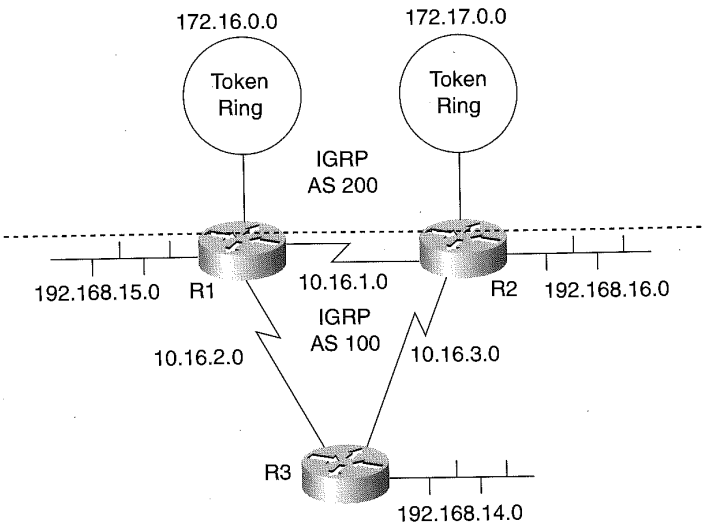


Figure 15-4
IGRP-routing protocol used between Cisco routers.

```
interface Ethernet 0
 ip address 192.168.14.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
network 192.168.14.0
```

IGRP sends and receives updates for networks only on interfaces where the network is connected and only to the IGRP autonomous system neighbors being advertised with autonomous system number 100. If a second IGRP process were defined with different networks, they would not communicate with the networks of the other autonomous system.

As seen in Figure 15-4, the serial interfaces connecting the routers, along with the Ethernets on all the routers that have their networks in the routing tables, are in IGRP 100. The Token Ring networks on the routers, as well as the serial links, are defined in IGRP network 200. The devices on the Token Ring cannot communicate with the devices on the Ethernet in this configuration. This is because the IGRP autonomous systems are viewed as two independent networks. Routers R1 and R2 have two separate routing tables, one for each AS defined. The following configuration illustrates the new definitions to support the two IGRP autonomous system networks:

Router 1:

```
interface serial 0
 ip address 10.16.1.1 255.255.255.0
!
interface serial 1
 ip address 10.16.2.1 255.255.255.0
!
interface Ethernet 0
 ip address 192.168.15.1 255.255.255.0

interface Tokenring 0
 ip address 172.16.1.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
network 192.168.15.0
!
router igrp 200
network 10.0.0.0
network 172.16.0.0
```

Router 2:

```
interface serial 0
 ip address 10.16.1.2 255.255.255.0
!
interface serial 1
 ip address 10.16.2.2 255.255.255.0
!
interface Ethernet 0
 ip address 192.168.16.1 255.255.255.0

interface Tokenring 0
 ip address 172.17.1.1 255.255.255.0

router igrp 100
network 10.0.0.0
network 192.168.16.0
!
router igrp 200
network 10.0.0.0
network 172.17.0.0
```

Router 3:

```
interface serial 0
 ip address 10.16.2.3 255.255.255.0
!
interface serial 1
 ip address 10.16.3.3 255.255.255.0
!
interface Ethernet 0
 ip address 192.168.14.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
network 192.168.14.0
```

In order for devices on these two IGRP networks to communicate, the router must redistribute the routes between the two IGRP networks. Router redistribution among routing protocols is discussed in Chapter 19, "Route Redistribution."

Using Unicast IGRP Routing Updates

Non-Broadcast Multi-Access (NBMA) networks, like X.25 and Frame Relay, provide multiple connections over a single interface. As in Figure 15-5, the NBMA topology provides for a single physical interface on the router to a public network. The public network provider delivers the packets or frames to the destination NBMA address or identifier.

The Cisco IOS software must be defined in such a scenario to specifically identify the IGRP routers over the single interface by specifying the neighboring routers' IP addresses. For example, in Figure 15-5, router R1 must define the IP addresses of router R2 and R3 under the IGRP-routing protocol definition to send IGRP updates to the adjacent routers R2 and R3 through the single Frame Relay interface.

Router R1 specifies the IP addresses of the neighboring routers R2 and R3 connecting over the single Frame Relay interface by using the global configuration command neighbor. The format of the command is

neighbor *ip-address*

The *ip-address* value is the IP address assigned to the routers on the far end of the public NBMA network connection. In Figure 15-5, the configuration in router R1 is specified as the following:

Router R1 Configuration:

```
interface serial 0
encapsulation frame-relay
ip address 10.16.17.1 255.255.255.0
!
interface ethernet 0
ip address 10.1.10.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
neighbor 10.16.17.2
neighbor 10.16.17.3
```

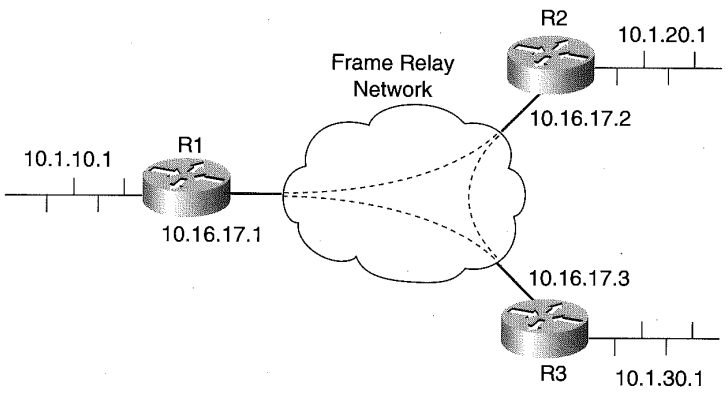


Figure 15-5
IGRP unicast updates using the neighbor command.

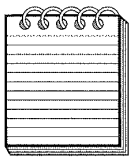
Router R1 sends IGRP updates to the IP addresses 10.16.17.2 and 10.16.17.3 and has learned of the DLCI for each of these on the Frame Relay multipoint connection using Frame Relay Inverse ARP. The Frame Relay Inverse ARP builds a table mapping the DLCI numbers to the IP addresses dynamically. The configuration for router R2 and router R3 are as follows:

Router R2 Configuration:

```
interface serial 0
encapsulation frame-relay
ip address 10.16.17.2 255.255.255.0
!
interface ethernet 0
ip address 10.1.20.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
neighbor 10.16.17.1
neighbor 10.16.17.3
```

Router R3 Configuration:

```
interface serial 0
encapsulation frame-relay
ip address 10.16.17.3 255.255.255.0
!
interface ethernet 0
ip address 10.1.30.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
neighbor 10.16.17.1
neighbor 10.16.17.2
```



NOTE: *It is highly recommended to use subinterfaces to enable broadcast-routing updates. See Chapter 21, "Defining Frame Relay" for more information.*

As was discussed with RIP unicast topologies, the neighbor command can be used to selectively define which routers on a given interface will participate in routing updates along with the passive-interface command. In this sample configuration, the workstations connected to R1 LAN segment Ethernet 1 need only communicate with the R2 router to connect to the WAN and the stations on segment Ethernet 1 router R2. The network topology is diagrammed in Figure 15-6 and the following IGRP configuration demonstrates this concept:

Router R1 Configuration:

```
interface Ethernet 0
ip address 10.17.2.1 255.255.255.0
!
interface Ethernet 1
ip address 10.12.2.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
neighbor 10.17.2.2
passive-interface Ethernet 0
passive-interface Ethernet 1
```

Router R2 Configuration:

```
!
interface Serial 0
ip address 172.16.10.1 255.255.255.0

interface Ethernet 0
ip address 10.17.20.2 255.255.255.0
!
interface Ethernet 1
ip address 10.22.2.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
network 172.16.0.0
passive-interface Ethernet 1
```

Router R3 Configuration:

```
interface Ethernet 0
ip address 10.17.2.3 255.255.255.0
!
interface Ethernet 1
```

```
ip address 10.32.2.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
neighbor 10.17.2.2
passive-interface Ethernet 0
passive-interface Ethernet 1
```

In this configuration example, router R3 does not receive updates from router R1 because the passive-interface command is specified for the Ethernet interface connecting router R1 on IP subnet 10.17.2.0. The same holds true for R1 for receiving updates from R3. The passive-interface command indicates to the Cisco IOS that the interface specified on the command will not be used for sending routing updates, but the specified interface can receive updates. Router R2 receives and exchanges routing updates with R1 and R3 because the IP address of R2 is defined on the neighbor command of router R1 and R3. Figure 15-6 illustrates this network topology.

Further IGRP update reduction is possible by implementing the passive-interface command for LAN segments connecting end-user stations only.

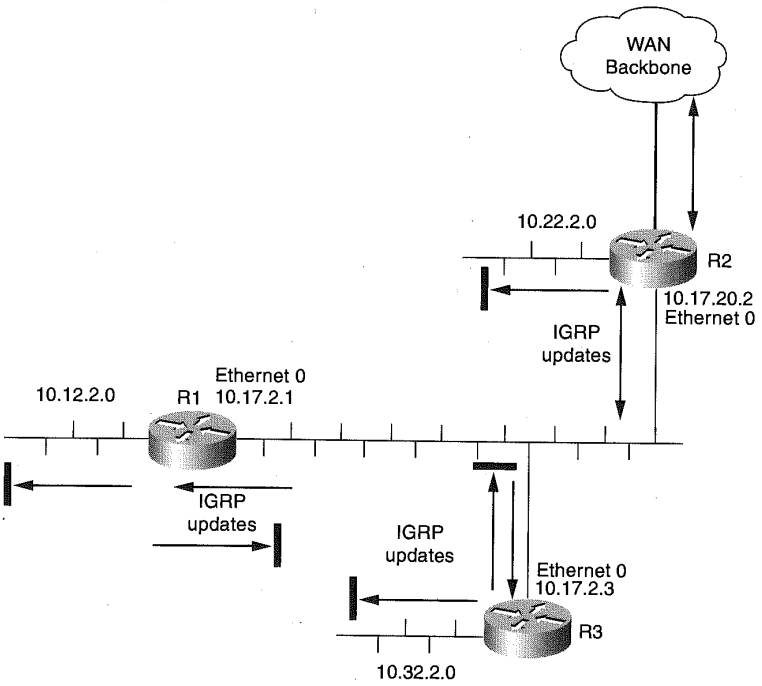


Figure 15-6 Reducing IGRP-update reduction on an Ethernet backbone using unicast.

This eliminates the unnecessary IGRP update traffic entering the network interface cards of the workstations.

The result of this configuration is that update packets are exchanged only between R1 and R2 and between R2 and R3. Using this method, the interface cards on routers not requiring IGRP routing table updates will not have to process unneeded packets. In another example, if the same three routers are attached through a switch, then the workstations attached to the same switch no longer receive unnecessary IGRP-routing updates. This is especially useful in switched networks employing ATM LANE topologies where unicast-directed IGRP-routing updates are only forwarded to the destination MAC address within the packet, instead of an all-broadcast destination MAC address, as is normally found with IGRP update packets. Figure 15.7 illustrates this configuration.

Using this concept on switched connections saves switch processor resources and eliminates the IGRP updates from being propagated to all the workstations also connected on the same switch.

Increasing Throughput and Reliability Using Unequal-cost Paths

Both RIP and IGRP distribute IP packets over equal cost paths on a packet-by-packet basis or on a destination-by-destination basis. IGRP, however, can provide for load balancing over unequal cost paths, enabling greater throughput and reliability with minimal convergence. The unequal-cost load balancing is provided over a maximum of four paths to a given destination network. The paths considered by IGRP must be calculated as feasible paths. A feasible path is determined by the route update from an adjacent router. If the adjacent router's routing update for a destination network has a lower value than the calculated metric of this router and the final calculated metric for the alternate route is equal to or greater than the best metric for the destination network on this router.

The following configuration illustrates the use of the variance definition for determining the feasibility of alternate routes for use in unequal-cost load balancing.

Router R1 in Figure 15-8:

```
router igrp 100
 variance 10
```

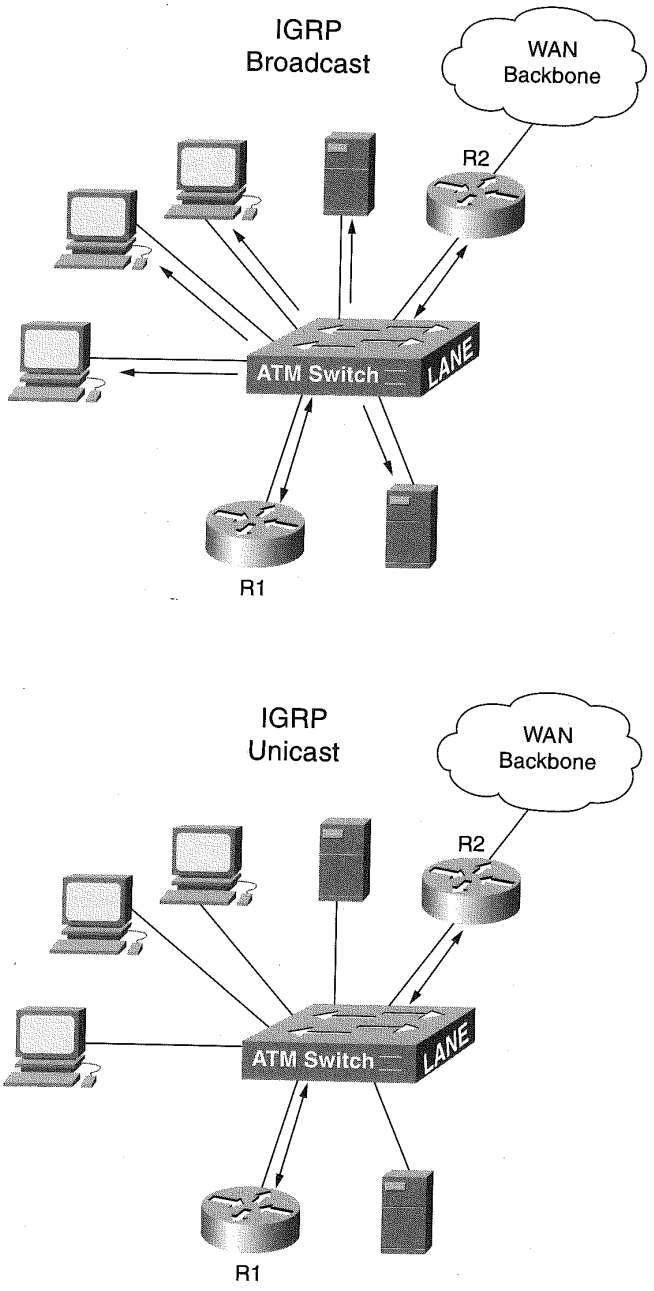


Figure 15-7
IGRP updates reduction in an ATM LANE network using unicast.

The variance command value of 10 in the example is the multiplier used for determining the feasibility of the alternate routes. It represents the metric multiplier. If you multiply the metric of the route with the best (lowest) metric by the variance value, all routes with a metric less than or equal to that result will be considered equal cost paths, provided that the loop prevention conditions (metric through the next router) are met. Metrics with resulting values equal to or greater than that will be considered unequal-cost paths. The metric value for the variance command can range from 0 to 128. The default multiplier is 1 for the variance command.

Figure 15-8 illustrates the unequal-cost path feasibility calculation. Router R1 connects to network 172.16.0.0 and has a metric of 50. Router R2 updates router R1 for network 172.16.0.0 with a metric of 40. Using the variance value of 10 for the IGRP autonomous system router R1 will add and use the route through router R2 for connection to network 172.16.0.0. This occurs because:

1. The metric for network 172.16.0.0 is greater than the received metric from router R2.
2. The resulting product of router R1's metric (50) multiplied by the variance value (10) is greater than or equal to the calculated metric of router R1 using router R2 to route traffic to the 172.16.0.0 network.

In concert with the variance command, the traffic-share command can be specified to manage just how packets are balanced across unequal-cost paths. The command format is

traffic-share {balanced | min}

The **traffic-share balanced** command shares the traffic proportionately to the ratio of metrics for the routes. The **traffic-share min** command uses the route with the minimum cost. The **variance** command allows you to specify the multiplier for the load balancing to skew the load. Both commands are for IGRP and EIGRP only.

Altering IGRP Routing and Metric Computations

IGRP router metrics are altered using five constants and the type of service for the route. A router administrator can alter these metric values

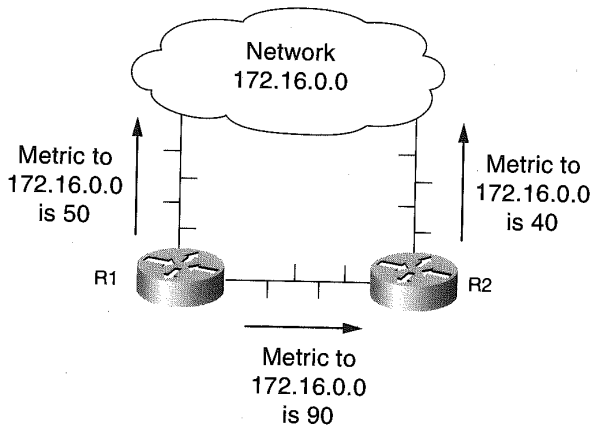


Figure 15-8
IGRP unequal-cost path feasibility topology.

using the metric weights router configuration command. The format of the command is

metric weights *tos k1 k2 k3 k4 k5*

The *tos* variable currently defaults to the value of 0 and must always be specified as 0. The *k1* through *k5* values default to the following:

k1: 1

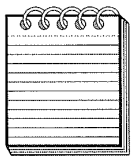
k2: 0

k3: 1

k4: 0

k5: 0

The values placed in the positional parameters of the metric weights command provide the values used as IGRP metric coefficients when determining the calculated metric for an IGRP route. Changing the defaults for these values changes the default behavior for all IGRP routing and metric calculations. Thus, changing the metrics can greatly affect network performance. The valid range for the numeric value of *k1-k5* is 0 to 4,294,967,295. Using 4,294,967,295 as a value for any of the coefficients results in the route being marked as unusable. The k1-k5 values are used in the IGRP metric calculation against the four major metric components of IGRP route calculation. These components are bandwidth, load, delay, and reliability.



NOTE: *Great care should be taken when modifying the default metric values for IGRP. It is strongly advised that the metric weight values should not be modified unless consulting with the Cisco TAC prior to and during the modification.*

The IGRP process calculates the metric by adding weighted values of different characteristics of the route entry. The standard formula for determining the metric is as follows:

metric = k1 x bandwidth + (k2 x bandwidth / (256 - load) + k3 x delay

The default metric values for k1-k5 affect this overall calculation by realizing that the k1 and k3 values both are 1 and the k2, k4, and k5 values equal 0. Applying the default values for the **metric weights** router command results in the following calculation:

metric = bandwidth + delay

Understanding this results in the need to understand the interface command `bandwidth`. The interface configuration command bandwidth allows for a router administrator to modify the bandwidth of the physical medium. For example, entering the following command

```
interface etherent 0
 bandwidth 5000
```

will end up causing interface Ethernet 0 on this router to have a less favorable metric than a router that uses the default bandwidth for a 10-Mbps Ethernet, which is 10,000 Kbps. The bandwidth command is expressed as

bandwidth *kilobits*

The *kilobits* value is in the valid range of 1 to 10,000,000, expressed in Kbps. This allows the router administrator to place a value as high as 10 terabits per second (Tbps) on an interface. With 1 Tbps technology proven and the capability to continually increase the bandwidth on fiber optic networks, this value may have a shorter life span than anticipated. The metric bandwidth value used by Cisco IOS running the IGRP process for serial interfaces defaults to 1.544 Mbps (T1 speed). This warrants that if the serial lines connecting to the router are less than a T1, then the bandwidth interface command must be specified to allow IGRP to calculate the true metric. For example, a 56-Kbps serial connection should be defined as

```
interface serial 0
 bandwidth 56
```

Since bandwidth is such a major factor in determining the metric for a route, great care must be used when using the bandwidth interface command.

Suppose the metric weights command were entered where the k2 value was changed from 0 to a value of 1. The command sequence is as follows:

```
router igrp 100
 metric weights 0 1 1 1 0 0
```

In this instance, the load metric becomes the deciding variable. The load value for an interface is calculated every five seconds, based on a five-minute weighted average. The load is presented as a fraction of 255, where 255 is a saturated link experiencing 100 percent utilization.

Setting the k2 coefficient to 1 can result in a route going into holddown when there is an increase in load, due to a large FTP or some other type of action that can consume bandwidth over a period of time. The net effect is that the metric ends up doubling the bandwidth, which can possibly make a slow link artificially inflated with bandwidth. In such a scenario, not only can a load rise fast enough to make a route unstable, it can also fall just as fast. A falling load will change the metric to become favorable and cause the Cisco router to use the Cisco IOS feature called flash update.

Flash update is a mechanism whereby the routing table is broadcast to the router's neighbors, prior to the update timer being expired. In the scenario described, multiple flash updates may occur due to the variant load, thus causing not only an unstable route for the specific path but causing the network to continuously perform convergence, which ends up in a convergence storm and an unstable network. It is because of the undeterministic characteristics found with IP networks that the metric defaults remain unaltered.

In addition to the formula noted above involving `k` values k1 through k3, k5 and k4 come into play if the k5 value specified on the metric weights IGRP router command is not zero. When this occurs, the following formula is also calculated to create the final metric for the route:

metric = metric x [k5 / (reliability + k4)]

A long-hand method for determining the metric can be performed for a specific router by performing the following:

1. Determine the smallest of all the bandwidths from outgoing interfaces and divide 10,000,000 by that number. (The bandwidth is scaled by 10,000,000 in kilobits per second.)

2. Add all the delays from the outgoing interfaces and divide this number by 10. (The delay is in 10s of microsecs.)

The path with the smallest metric is the best path. The router configuration for determining metric values is shown in Figure 15-9 and the default delay and bandwidth values used by IGRP for specific media types are shown in Table 15-1. These values can be found by issuing the router operation command show interfaces. An example for this is as follows:

```
R1#sh int e 0
Ethernet 0 is up, line protocol is up
Internet address is 192.168.9.22, subnet mask is 255.255.255.0
MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec, rely 255/255, load
  1/255
Serial 0 is up, line protocol is up
Internet address is 192.168.19.9, subnet mask is 255.255.255.224
MTU 1500 bytes, BW 784 Kbit, DLY 20000 usec, rely 255/255, load
  20/255

R2#sh int s 0
Serial 0 is up, line protocol is up
Internet address is 192.168.42.1, subnet mask is 255.255.255.224
MTU 1500 bytes, BW 448 Kbit, DLY 20000 usec, rely 255/255, load
  1/255
Serial 1 is up, line protocol is up
Internet address is 192.168.19.8, subnet mask is 255.255.255.224
MTU 1500 bytes, BW 224 Kbit, DLY 20000 usec, rely 255/255, load
  3/255

R1# sh ip route `R2-s0`
Total delay is 40000 microseconds, minimum bandwidth is 448 Kbit
  Known via "igrp 2022", distance 100, metric 26321
```

The calculation:

```
Metric = bandwidth + delay
       = 10,000,000/448 + (20,000+20,000)/10
       = 26321

R2# sh ip route `R1-e0`
Total delay is 21000 microseconds, minimum bandwidth is 224
Kbit Known via "igrp 2022", distance 100, metric 46742

Metric = bandwidth + delay
       = 10,000,000/224 + (20.000 + 1000)/10
       = 46742
```

Decreasing IGRP Route Convergence

The holddown mechanism is used by IGRP to assist in avoiding routing loops due to changes in route metrics. The Cisco IOS software places a route in a holddown state after receiving an update from a neighboring router

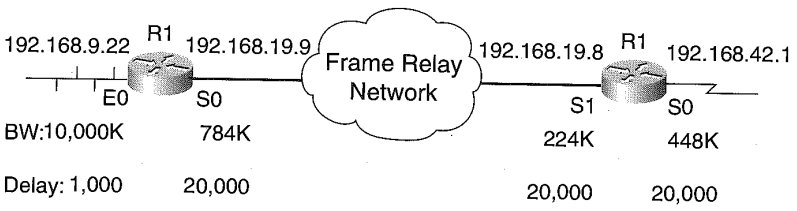


Figure 15-9 The router configuration for determining metric values.

Table 15-1 Default Delay and Bandwidth Values Used by IGRP for Specific Media Types

Media Type	Delay	Bandwidth
Satellite	5120 (2 seconds)	5120 (500 Mbits)
Ethernet	25600 (1 ms)	256000 (10 Mbits)
1.544 Mbps	512000 (20,000 ms)	1,657,856 bits
64 kbps	512000 (20,000 ms)	40,000,000 bits
56 kbps	512000 (20,000 ms)	45,714,176 bits
10 kbps	512000 (20,000 ms)	256,000,000 bits
1 kbps	512000 (20,000 ms)	2,560,000,000 bits

that the network found on the route in question is now a greater distance from the router or the network is no longer available. During the holddown period, the known route metrics for the network in question is still advertised to the neighboring routers. Advertisements received from other routers concerning the network in question are ignored. The ignoring of updates to the network in question after receiving a metric update change helps avoid the route loop. The time it takes for the routers to determine the correct route and metric, however, can them increase the convergence time for IGRP. In some instances, it is favorable to eliminate the holddown mechanism, which reduces the time for convergence. Disabling the holddown mechanism is accomplished by entering the following configuration IGRP router command:

```
no metric holddown
```

Entering this command disables the holddown mechanism. This configuration, however, must be the same for all routers in the IGRP autonomous system. Not disabling the holddown metric on all the IGRP autonomous system routers may result in routing loops.

To reenable the holddown mechanism after it has been disabled, the following IGRP router command must be entered on all the IGRP autonomous system routers:

```
metric holddown
```

Tuning IGRP Route Convergence

IGRP uses five timers for determining how routing tables are managed. These timers include an update timer, an invalid timer, a holddown timer, a flush timer, and a sleeptime timer.

The `update timer` determines the amount of time measured in seconds for sending routing updates. The default value is 90 seconds.

The `invalid timer` determines the amount of time measured in seconds that must expire without the router receiving route updates from neighboring routers for a known route. The default value is 270 seconds. If the invalid timer stops, the route is marked inaccessible and the route is advertised by the router to neighboring routers as unreachable. The route at this point has entered holddown state. Packets destined for the network in question, however, are still forwarded on the route.

The `holddown timer` is measured in seconds and defaults to 280 seconds. This is the amount of time that updates for the route in holddown state are ignored. After the holddown period, updates for the route are now applied and the route is now reachable.

The `flush timer` is the number of seconds that must pass prior to a route being removed from the routing table. This timer must be at minimum the sum of the interval and holddown timer values. The default is 630 seconds. Specifying a value that is less than the sum of the interval and holddown timer values results in the holddown timer never expiring. This allows an improper route to be added to the routing table on the update received.

The `sleeptime timer` is measured in milliseconds and delays routing updates due to a flash update. The sleeptime timer value should be less than the update timer to avoid unsynchronized routing tables. The default for this timer is 0 milliseconds.

The timers start after the routing table has been calculated. Modifying the timer values is accomplished using the IGRP router command:

timers basic *update invalid holddown flush* [*sleeptime*]

The *update* and *holddown* variables' range is from 0 to 429,496,729. The *invalid, flush,* and *sleeptime* variables' range from 1 to 4294967295.

In the following example, the IGRP process sends routing updates every 20 seconds.

```
router igrp 100
 timers basic 20 60 70 130
```

In 60 seconds, a route to a neighboring router is determined to be inaccessible if an update has not been received from the neighbor. The route then enters holddown. The holddown time of 70 seconds indicates that the updating of the route is ignored for an additional 70 seconds. At the end of the holddown period, the route is flushed from the routing table. The router then adds a new route based on the next received routing update for the original network in question. Making the timers to small may result in increased router utilization in large networks.

Controlling the Logical Size of an IGRP Network

IGRP being a distance-vector routing protocol measures distance between routers based on the hop count. Although RIP reaches its maximum at 16 (meaning infinity), IGRP can reach as high as 255 hop counts before declaring a route unreachable. Using the following command with IGRP, an autonomous system can be protected from a potential count-to-infinity problem. The format of this command is

metric maximum-hops *hops*

The *hops* value is a decimal value ranging from 1 to 255. The default is 100. For large networks, it may be necessary to increase the default value to enable routes over a complex routed WAN environment. The default of 100 indicates that any route with a hop count greater than 100 is advertised to its neighbors as unreachable.

Validate Source IP Addresses

Many network configurations require multiple IP network definitions. IGRP validates the source IP address of the routing update being received. If the source IP address of the routing update packet being received is on the same network as the IP address of the interface receiving the update, the routing update packet is processed. If the received routing update packet source IP address is not on the same network as the interface receiving the routing update packet, the packet is discarded. A scenario where this is possible is the use of secondary IP addressing on interfaces. Figure 15-10 illustrates this configuration.

Router R1 in Figure 15-10 is connected to the same Ethernet segment as router R2 via a hub. Router R1 has its Ethernet interface defined as IP address 192.168.16.1, using 255.255.0.0 as the subnet mask. This same interface on router R1 also has a secondary IP address defined as 192.168.17.1, using 255.255.255.0 as the subnet mask. Router R2 connected to the same hub defines its Ethernet connection as 192.168.17.2, using 255.255.255.0 as the subnet mask. When the routers begin sending RIP routing updates, the following message can be observed in the log of router R2, using the Debug IP Routing Information Protocol privileged command:

```
Ignored v1 update from bad source 192.168.16.1 on Ethernet0
```

This message is an indicator that the IGRP update packets being received on Ethernet 0 of router R2 are not on a connected network. The source IP address of the routing packet received from router R1 is not a network connected to interface Ethernet 0 on router R2 and is hence discarded by router R2. To overcome this in the network configuration presented in Figure 15-10, the validation of the source IP address can be disabled using the following command:

```
no validate-update-source
```

Using this command, the validation process of incoming IGRP update packets is disabled and the router, in this case router R2, receives the routing updates. The configuration of the two routers in Figure 15-10 using this command is as follows:

Configuration router R1:

```
interface Ethernet 0
ip address 192.168.17.1 255.255.255.0 secondary
ip address 192.168.16.1 255.255.0.0
!
router igrp 100
 network 192.168.16.0
 network 192.168.17.0
```

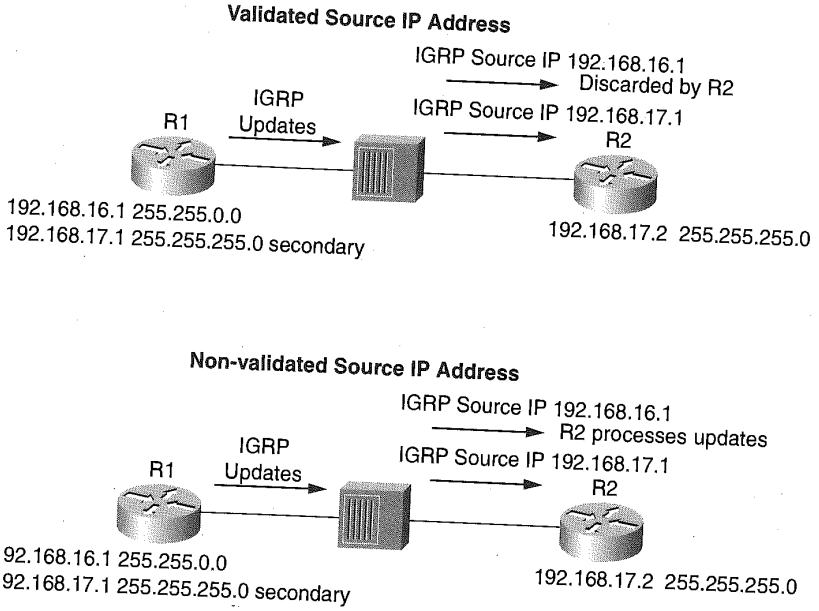


Figure 15-10
The off-network routing configuration and the effect of disabling the source IP address validation.

```
Configuration router R2:

interface Ethernet 0
ip address 192.168.17.2 255.255.255.0
!
router igrp100
 network 192.168.17.0
 no validate-update-source
```

This enables router R2 to place updates provided by router R1 source IP address 192.168.16.1 into its routing table. It is recommended that this command not be used, however, since it can lead to confusion in the routing tables and potentially cause routing loops.

IGRP Updates and the Effect on Bandwidth

Generally, IGRP updates flow every 90 seconds. Each IGRP packet can contain 104 route entries, for a total of 1,492 bytes, 38 of which are header information, and each route entry is 14 bytes. Advertising 1,000 routes over

a Frame Relay link configured with 50 DLCIs, the request is approximately 720 KB of routing update data every 90 seconds, or 64 Kbps of bandwidth consumed. On a T1 link, this bandwidth would represent 4.2 percent of the bandwidth, with each update duration being 3.7 seconds. This overhead is an acceptable amount:

1000 / 104 = 9 packets X 38 = 342 header bytes

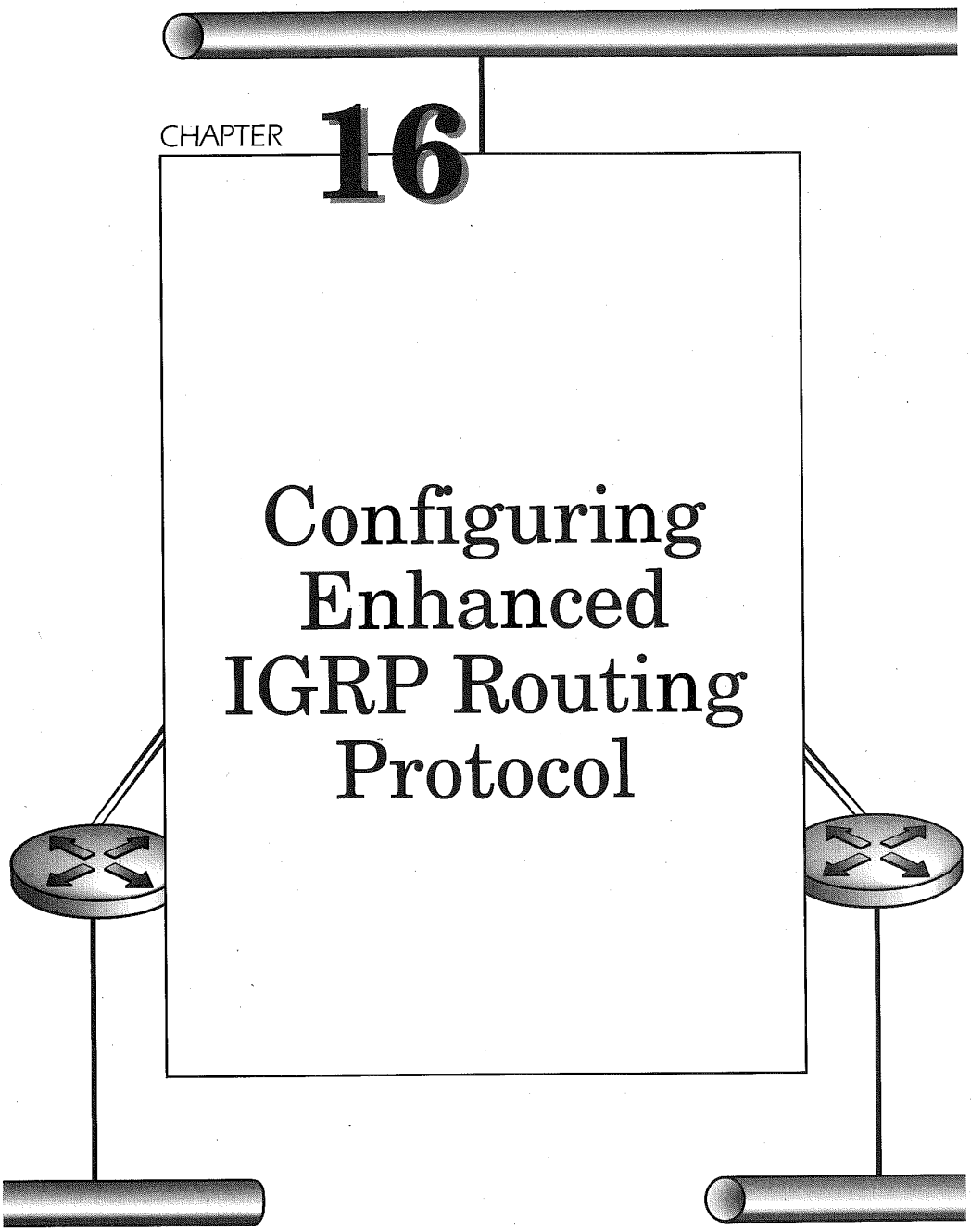
1000 X 14 = 14,000 bytes of route entries

Total = 14,342 bytes X 50 DLCIs
= 717 KB of IGRP updates every 90 seconds

717,000 bytes / 90 X 8 bits = 63.7 kbps

CHAPTER 16

Configuring Enhanced IGRP Routing Protocol



Enhanced IGRP (EIGRP) is the Cisco proprietary routing protocol that fuses the best attributes of distance-vector routing protocols with those of link-state routing protocols. EIGRP is configured just like IGRP and uses the same metrics as IGRP. The enhancement is provided by the inclusion of using the Diffusing Update Algorithm (DUAL). DUAL was developed by SRI International under the direction of J.J. Garcia to achieve a rapid convergence routing protocol that virtually guarantees a loop-free network. The marriage of distance-vector, link-state, and DUAL results in the following characteristics of EIGRP:

- Rapid convergence
- Reduced bandwidth
- Increased network size
- Reduced router CPU utilization

Rapid convergence is due to the use of DUAL. This rapid convergence is achieved by a route using EIGRP to keep backup routes within the routing table. In other words, the least-cost (successor) and second-to-least-cost (feasible successors) routes for a destination network are kept within the routing table. This enables a router to quickly adapt to link outages without causing major network disruption within the network. The favored and backup routes are recalculated based on updates from neighboring routers. EIGRP updates neighboring routers after the initial convergence only when there is a change in a route and only for the changed route.

Because EIGRP sends updates on routes to neighbor EIGRP routes only when the state of a route to a destination network changes or when the metric for a route changes, these partial updates require considerably less bandwidth. In addition, the route update is sent only to the neighboring routes that require knowledge of the state change. Due to the incremental updates, EIGRP uses less CPU than IGRP.

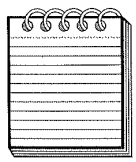
Large networks have difficulty using RIP as the IP routing protocol due to the 15 hop limit. EIGRP enables much larger networks to be architected, increasing the hop limit to 255. This means that the EIGRP-calculated metric supports hop counts in the thousands, enabling very large network configurations. Using EIGRP also moves the network size restrictions up the protocol stack to the transport layer. EIGRP facilitates the transport layer deficiency of 15 hops by incrementing the transport control field by one after the packet has passed through 15 EIGRP routers and the next hop is an EIGRP router. If non-EIGRP routers on the packets utilize next hop, the transport control field is incremented.

Because EIGRP is a Cisco proprietary routing protocol, it has an additional benefit not found with open-standard routing protocols. EIGRP can

also be used for delivering the routing and services information of Novell IPX RIP/SAP updates and AppleTalk Routing Table Maintenance Protocol (RTMP). The greater advantage in using EIGRP on these other routing protocols is that with Novell IPX networks, EIGRP sends incremental RIP/SAP updates between EIGRP routers. These updates are sent only when there is a change in the IPX entry. Furthermore, when using EIGRP for Novell networks, the hop count is now 255 versus the IPX RIP hop count of 15. EIGRP used for Novell IPX selects the best route to a destination based on the EIGRP metrics of bandwidth and delay, instead of the IPX metrics of tick and hop counts.

EIGRP uses three types of tables to determine routes. All these tables are used for the three networking protocols supported by EIGRP. These tables are named Neighbor, Topology and Routing. Each EIGRP router lists the address of the next hop router (the neighbor network layer address) and the interface on the router to which the neighbor is connected. The EIGRP process to verifies bidirectional communication using this table. The topology table contains the destination networks and up to six learned routes to each destination. These include the successor (best route) and feasible successors (backup routes). The routing table is the list of the best route (successor) to a destination network. The routing table is populated from the best route entries in the routing table for each destination network.

EIGRP maintains a set of tables for all enabled, supported network layer protocols. The IP network layer protocol is further supported by EIGRP using Variable Length subnet mask (VLSM) IP addressing and route summarization.



NOTE: *This chapter concerns the configuration of IP network layer protocol with EIGRP. Configuring EIGRP for Novell networks is discussed later in the book.*

Enable EIGRP as a Routing Protocol

Cisco IOS starts the EIGRP routing protocol process by entering the following global configuration command:

router eigrp *autonomous-system*

The *autonomous-system* variable is either a registered or an internal-use-only autonomous system number that identifies the routes of the EIGRP network. This value ranges from 1 to 65535 and is also used to identify the EIGRP process executing in the router. It can also be referred to as the process-id for the EIGRP route process being defined. Each router using EIGRP must specify the same autonomous system number in order for the routers to exchange and update EIGRP routing tables.

The topology tables are defined by entering the network command under the router eigrp command. The format of the network command is

network *network-number*

The variable *network-number* is the classful network number of a network directly connected to the router. For example, in Figure 16-1, the three routers are connected using EGRP as the routing protocol. Each router specifies the network address of a directly connected network under the router EIGRP definition. This enables each router to advertise the IP network to its neighboring IGRP routers. The network number is the Class A, B, or C IP network assigned to the interfaces on the router. The use of subnet masks for identifying the network is not used on the network command. In Figure 16-1, network 10.0.0.0 is defined to all the routers, while network 172.16.0.0 is defined to only router 1, network 172.17.0.0 is defined to router 2, and network 172.16.0.0 is defined to router 3 only.

The EIGRP configuration commands for each router are as follows:

Router 1:

```
interface serial 0
 ip address 10.16.1.1 255.255.255.0
!
interface serial 1
 ip address 10.16.2.1 255.255.255.0
!
router eigrp 200
network 10.0.0.0
network 172.16.0.0
```

Router 2:

```
interface serial 0
 ip address 10.16.1.2 255.255.255.0
!
interface serial 1
 ip address 10.16.3.2 255.255.255.0
!
router eigrp 200
network 10.0.0.0
network 172.17.0.0
```

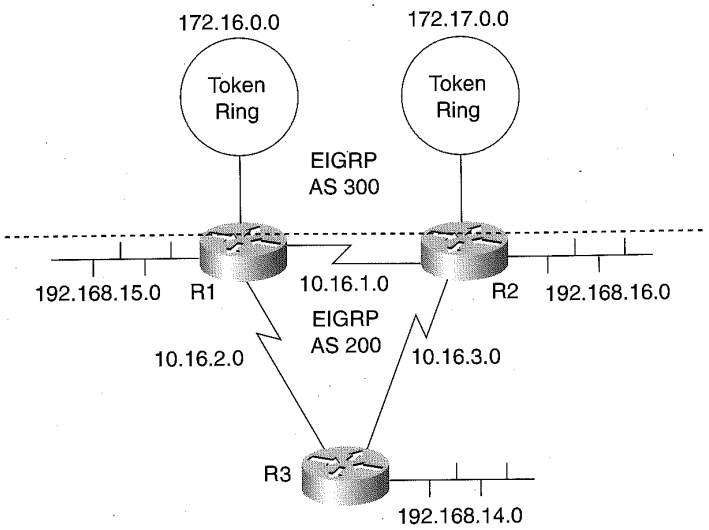


Figure 16-1
EIGRP routing protocol used between Cisco routers.

Router 3:

```
interface serial 0
 ip address 10.16.2.3 255.255.255.0
!
interface serial 1
 ip address 10.16.3.3 255.255.255.0
!
router eigrp 200
network 10.0.0.0
network 172.18.0.0
```

EIGRP sends and receives updates for networks only on interfaces where the network is connected and sends updates only to the EIGRP autonomous system neighbors being advertised with autonomous system number 200. If a second EIGRP process is defined with different networks, they would not communicate with the networks of the other autonomous system. As seen in Figure 16-1, the serial interfaces connecting the routers, along with the Ethernets on all the routers that have their networks in the routing tables, are in EIGRP 200. The Token Ring networks on the routers as well as the serial links are defined in EIGRP network 300. The devices on the Token Ring cannot communicate with the devices on the Ethernet in this configuration. This is because the EIGRP autonomous systems are viewed as two independent networks. Each router will have two separate routing tables. The following configuration illustrates the new definitions to

support the two EIGRP autonomous system networks. Expand these configurations using the serial, Ethernet, and Token Ring definitions.

Router 1:

```
interface serial 0
 ip address 10.16.1.1 255.255.255.0
!
interface serial 1
 ip address 10.16.2.1 255.255.255.0
!
router eigrp 200
network 10.0.0.0
network 172.16.0.0
!
router eigrp 300
network 10.0.0.0
network 192.168.100.0
```

Router 2:

```
interface serial 0
 ip address 10.16.1.2 255.255.255.0
!
interface serial 1
 ip address 10.16.2.2 255.255.255.0
!
interface Ethernet 0
 ip address 192.168.16.1 255.255.255.0

interface Tokenring 0
 ip address 172.17.1.1 255.255.255.0
!
router eigrp 200
network 10.0.0.0
network 172.17.0.0
!
router eigrp 300
network 10.0.0.0
network 192.168.200.0
```

Router 3:

```
interface serial 0
 ip address 10.16.2.3 255.255.255.0
!
interface serial 1
 ip address 10.16.3.3 255.255.255.0
!
interface Ethernet 0
 ip address 192.168.14.1 255.255.255.0
!
router eigrp 200
network 10.0.0.0
network 192.168.14.0
```

In order for devices on these two EIGRP networks to communicate, the router must redistribute the routes between the two EIGRP networks. Router redistribution among routing protocols is discussed in a later chapter.

Migrating to EIGRP from IGRP

Because EIGRP has its foundation in IGRP and both these protocols are proprietary to Cisco IOS, the migration from IGRP to EIGRP is extremely simple. The topology in Figure 16-2 illustrates the use of having IGRP and EIGRP in the same network. When both IGRP and EIGRP are configured using the same autonomous system number, the routes learned by both protocols are redistributed automatically. This makes for an ease of migration between IGRP and EIGRP.

In Figure 16-3, router R2 acts as the autonomous system boundary router (ASBR) between the IGRP network and the EIGRP network. Selecting an ASBR is paramount to the success of the migration effort. The configuration for the routers is listed below. Expand these configurations using the serial and Ethernet and Token Ring definitions.

Router 1:

```
interface serial 1
 ip address 10.1.2.1 255.255.255.0
!
interface Ethernet 0
 ip address 172.16.1.1 255.255.255.0
!
router igrp 100
network 10.0.0.0
network 172.16.0.0
```

Router 2:

```
interface serial 0
 ip address 10.2.3.2 255.255.255.0
!
interface serial 1
 ip address 10.1.2.2 255.255.255.0
!
router eigrp 100
network 10.0.0.0
passive-interface serial 1
!
router igrp 100
network 10.0.0.0
passive-interface serial 0
```

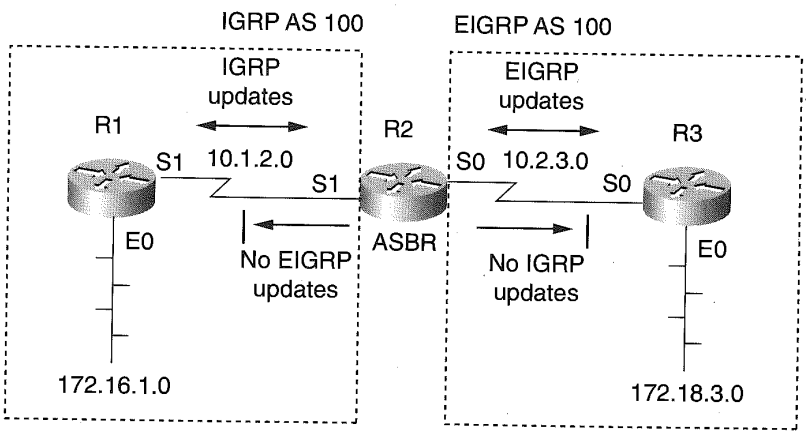


Figure 16-2
Migrating from IGRP to EIGRP.

Router 3:

```
interface serial 0
 ip address 10.2.3.3 255.255.255.0
!
interface Ethernet 0
 ip address 172.18.3.3 255.255.255.0
!
router eigrp 100
network 10.0.0.0
network 172.18.0.0
```

The routes in router R2 are automatically redistributed between the EIGRP and IGRP processes, enabling router R1 and its attached network to use IGRP as its routing protocol while learning the routes to networks connected to router R3. Likewise, router R3 EIGRP routing tables are populated with routes for networks attached to router R1 while using EIGRP as the routing protocol. If the autonomous system numbers used by IGRP and EIGRP are different, then router R2 would have to redistribute the IGRP-learned routes into EIGRP routes and EIGRP-learned routes into IGRP routes. Using the same AS number removes the necessity of route redistribution, which reduces CPU utilization.

In the configuration example, the passive-interface command is used on router R2 to eliminate unnecessary routing update traffic not used by the adjacent router. The command passive-interface serial 2 under the router eigrp 100 command blocks EIGRP routing updates and hello packets from being sent to the adjacent IGRP router that would just discard these packets anyway. Likewise, the passive-interface serial 3 statement under the router igrp 100 command blocks IGRP routing protocol packets from being

sent to the adjacent EIGRP-only router. Running EIGRP with RIP and OSPF is discussed in a later chapter.

Monitoring Neighbor Adjacency Changes

The establishment of EIGRP neighbors and the building of neighbor tables is a key element in the rapid convergence of EIGRP. Figure 16-3 provides a sample output of an EIGRP neighbor table for IP network layer protocol. EIGRP maintains a similar table for both IPX and AppleTalk.

The neighbor table is built on hello packet responses from adjacent EIGRP routers, and the hello packets are multicast packets. EIGRP uses the IP multicast group address 224.0.0.10 for the hello packets, which are sent by default every five seconds on a multi-access or point-to-point network. On NBMA networks like frame relay or X.25, the sending of hello packets defaults to 60 seconds. The hello packets are not acknowledged, while the update packets are unicast to neighbors, guaranteed delivery, and are acknowledged.

The neighbor table display resulting from the IOS EXEC command show ip eigrp neighbors lists the contents of the neighbor table. The Address column identifies the IP address of an adjacent neighbor. The Interface column identifies the interface on which the adjacent neighbor is attached. The Hold column specifies the number of seconds the router will wait without receiving any packets from the neighbor before marking the connection unavailable.

Figure 16-3 Router output of EIGRP neighbor table.

```
r1-njdc#SH IP EIGRP NEI
IP-EIGRP neighbors for process 10
H   Address         Interface   Hold Uptime   SRTT   RTO  Q   Seq
                                (sec)         (ms)       Cnt Num
6   10.16.1.4       AT3/0.2     13 2d19h      132    792  0   185923
11  10.16.1.6       AT3/0.2     12 2d20h       30    200  0   15956409
5   10.16.1.25      AT3/0.2     12 2d20h       32    200  0   2024
8   10.16.1.3       AT3/0.2     11 2d20h        9    200  0   8657
4   10.16.1.2       AT3/0.2     12 2d21h       26    200  0   7307774
0   10.16.1.5       AT3/0.2     14 2d21h       40    240  0   143003
9   10.16.1.7       AT3/0.2     12 2d21h       36    216  0   16160182
7   10.16.1.8       AT3/0.2     14 2d21h       44    264  0   312020
3   10.16.1.10      AT3/0.2     12 2d21h       28    200  0   7045242
2   10.16.1.1       AT3/0.2     11 2d21h       20    200  0   8660
1   10.16.1.9       AT3/0.2     10 2d21h       37    222  0   7080567
```

In the early releases of EIGRP, only the receipt of hello packets was measured. The current releases now measure the hold time for any EIGRP packet received after the first hello packet is received from the neighbor. The Uptime column indicates the amount of time this neighbor has been connected. The SRTT column represents the Smooth Round Trip Time (SRTT), which is a measurement in milliseconds of the average round-trip time for packets to travel between the routers. This time is used to determine the retransmit interval (RTO) values. The Q CNT (Queue Count) column is the number of packets waiting in the send queue. A value that is consistently higher than zero indicates link congestion.

The status of neighbor routers can assist in determining routing problems within the network. Changes to adjacent neighbors are normally not logged by Cisco IOS software. Logging is enabled using the following router eigrp configuration command:

eigrp log-neighbor-changes

Figure 16-4 provides a listing of a sample output from the Cisco IOS log when neighbor logging is enabled.

The `%DUAL-5-NBRCHANGE` message identifier indicates that EIGRP process ID 10 for IP routing has recognized a change in neighbor status. Neighbor 10.16.3.3, upon connecting to the network, has contacted this router and exchanged identification to form an adjacency. The neighbor connected using IP address 10.16.1.2 has not responded to EIGRP messages past the hold time and, as such, the router has determined that the neighbor is down. Once this neighbor rejoins the network, the logging process records it. Note that if a connection on this router to a neighbor goes down administratively, as shown in Figure 16-4, the logging process for ELAN test1 does not indicate a neighbor being disconnected. However, once the interface is brought back to an active state, the logging process indicates connection to the neighbor and forms an adjacency.

Figure 16-4 EIGRP neighbor changes logging output.

```
Feb 8 16:23:09: %DUAL-5-NBRCHANGE: IP-EIGRP 10: Neighbor 10.16.3.3
    (ATM3/0.9) is up: new adjacency
Feb 8 16:24:48: %DUAL-5-NBRCHANGE: IP-EIGRP 10: Neighbor 10.16.1.2
    (ATM3/0.2) is down: holding time expired
Feb 8 16:25:47: %DUAL-5-NBRCHANGE: IP-EIGRP 10: Neighbor 10.16.1.2
    (ATM3/0.2) is up: new adjacency
Feb 8 16:26:38: %LANE-5-UPDOWN: ATM3/0.9 elan test1: LE Client changed
    state to down
Feb 8 16:27:12: %LANE-5-UPDOWN: ATM3/0.9 elan test1: LE Client changed
    state to up
Feb 8 16:27:16: %DUAL-5-NBRCHANGE: IP-EIGRP 10: Neighbor 10.16.3.3
    (ATM3/0.9) is up: new adjacency
```

Managing EIGRP Bandwidth Utilization

Another advantage EIGRP has over other routing protocols is its inherent limit on the amount of bandwidth that routing updates will consume. Other routing protocols can quickly consume bandwidth during initial router connectivity and during normal periodic routing updates for distance-vector protocols and convergence in unstable networks. EIGRP guards against overutilizing the link for its own requirements by having an inherent limit of 50 percent link utilization imposed. The link utilization is based on the media default or value specified on the bandwidth interface command. Modifying the EIGRP utilization on a link is performed by entering the following interface command under the interface of the affected link:

ip bandwidth-percent eigrp *as-number percent*

The *as-number* value identifies the EIGRP process defined on this router that is being modified for link utilization. The percent value is the maximum amount of link utilization that EIGRP can consume on the link connected to the interface in question. The value 1 through 999,999 is valid for the percent variable.

Because the bandwidth interface command is used in some instances to influence the metric calculation for routes, EIGRP link utilization may end up becoming a factor that inadvertently causes performance problems. Or, for that matter, may require a link utilization that is greater than 100 percent due to bandwidth values that are specified as being much lower than the actual bandwidth of the connected link.

In Figure 16-5, we see a typical configuration where EIGRP would select the route R1-S1-R2-S0-R3-S0-R4-E0 to get from network 172.16.0.0 on router R1 to 172.24.0.0 on router R4. This route is simply based on the higher bandwidth and minimal delay. R1-S0 connects to R4-S0, however, using a direct 56-Kbps connection.

Because 56 Kbps is perceived by EIGRP to be a less favorable route, it is not the preferred path. The router administrator knows, however, that the connection between R1 and R4 over the 56-Kbps link have been traditionally more stable than the link between R2 and R3. Because of this, the bandwidth value for the serial interfaces connecting router R1 and router R4 are artificially increased to that of an T1 (1.544 Mbps) to establish this as the preferred route.

EIGRP, however, will now use this new value to determine its link utilization. At a default of 50 percent link utilization, the link would require

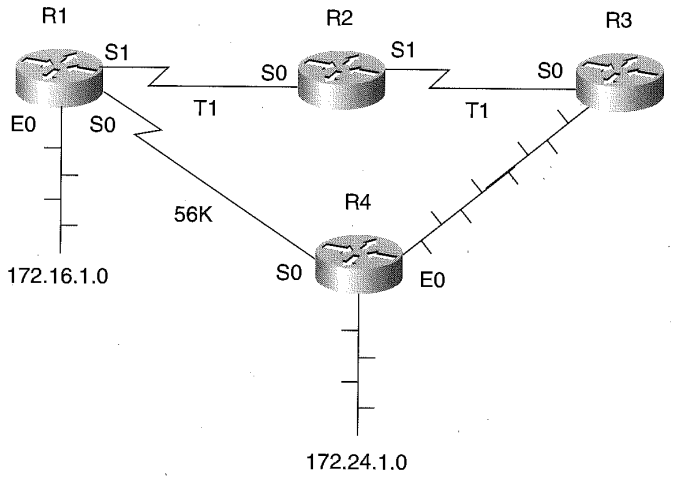


Figure 16-5 Using EIGRP link percentage on modified bandwidth values.

722 Kbps of bandwidth to support the EIGRP link utilization default. Obviously, this will saturate a 56-Kbps line. To protect the line from EIGRP overutilization, the following configuration for the serial interfaces on router R1 and router R4 is defined:

Router R1:

```
interface serial 0
 description Actual link bandwidth is 56Kbps
 bandwidth 1544
 ip bandwidth-percent eigrp 200 2
```

Router R4:

```
interface serial 0
 description Actual link bandwidth is 56Kbps
 bandwidth 1544
 ip bandwidth-percent eigrp 200 2
```

Utilizing the value of two percent on the artificially raised bandwidth, EIGRP uses a maximum of 30 Kbps, which is similar to the normal default of 28 Kbps for a 56-Kbps link.

In Figure 16-6, we see a configuration in which the serial link between R1 and R4 is actually a T1, but the router administrator wants to force EIGRP to use a public Frame Relay network. The frame relay multipoint connection has a total CIR of 768 Kbps. The router administrator modifying the bandwidth values for router R1 and router R4 on the T1 connection

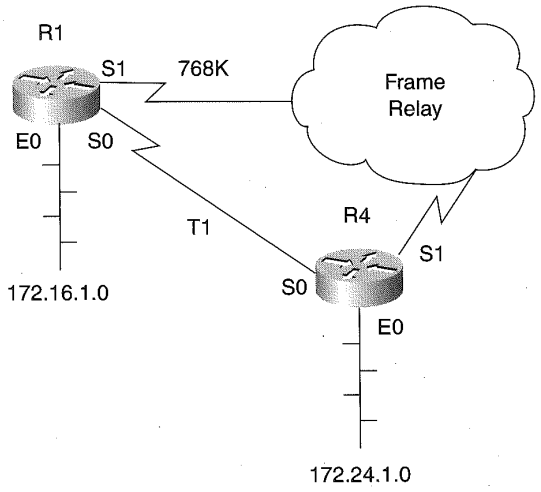


Figure 16-6
EIGRP link utilization configuration for specifying more than 100 percent of a link.

accomplishes this. The frame relay connection link between R1 and R4 is now favored.

Router R1:

```
interface serial 0
 description Actual link bandwidth is T1
 bandwidth 56
 ip bandwidth-percent eigrp 200 1000

interface serial 1
 description Frame Relay CIR bandwidth is 768K
 bandwidth 768
```

Router R4:

```
interface serial 0
 description Actual link bandwidth is T1
 bandwidth 56
 ip bandwidth-percent eigrp 200 1000

interface serial 1
 description Frame Relay CIR bandwidth is 768K
 bandwidth 768
```

The router R1 has its T1 serial interface bandwidth command specified as 56 Kbps. Likewise, router R4 has the T1 serial bandwidth specified as 56 Kbps. The requirement is met here, yet EIGRP router convergence will be affected, due to the default of 50 percent link utilization based on the bandwidth value. In order to compensate for this, the percent variable on the ip

bandwidth-percent eigrp interface command is set above 100 percent to 1000 percent, allowing EIGRP to use the actual available bandwidth on the link. The Frame Relay links must have the bandwidth interface command specified, since EIGRP will default the route calculation for the serial interface to that of a T1 (1.544 Mbps).

Modifying EIGRP Metric Weights

EIGRP router metrics are altered using five constants and the type of service for the route. A router administrator may alter these metric values using the metric weights router configuration command. The format of the command is

metric weights tos k1 k2 k3 k4 k5

The *tos* variable currently defaults to the value of 0 and must always be specified as 0. The *k1* through *k5* values default to the following:

k1: 1

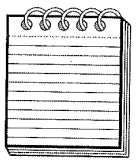
k2: 0

k3: 1

k4: 0

k5: 0

The values placed in the positional parameters of the metric weights command provide the values used as EIGRP metric coefficients when determining the calculated metric for an EIGRP route. Changing the defaults for these values thus changes the default behavior for all EIGRP routing and metric calculations. Therefore, changing the metrics can greatly affect network performance. The valid range for the numeric value of *k1-k5* is 0 through 4294967295. Using 4294967295 as a value for any of the coefficients results in the route being marked unusable. The k1-k5 values are used in the EIGRP metric calculation against the four major metric components of IGRP route calculation. These components are bandwidth, load, delay, and reliability.



NOTE: *It is strongly advisable to not modify the metric weight values unless consulting with the Cisco TAC prior to and during the modification.*

The EIGRP process calculates the metric by adding together weighted values of different characteristics of the route entry. The standard formula for determining the metric is as follows:

metric = k1 x bandwidth + (k2 x bandwidth / (256 - load) + k3 x delay

The default metric values for k1-k5 affect this overall calculation by realizing that the k1 and k3 values both are 1 and the k2, k4, and k5 values equal 0. Applying the default values for the **metric weights** router command results in the following calculation:

metric = bandwidth + delay

Understanding this results in the need to understand the interface command **bandwidth.** It allows for a router administrator to modify the bandwidth of the physical medium. For example, entering the following command:

```
interface ethernet 0
 bandwidth 5000
```

will end up causing interface Ethernet 0 on this router to have a less favorable metric than a router that uses the default bandwidth for a 10-Mbps Ethernet, which is 10,000 Kbps. The bandwidth command is expressed as

bandwidth *kilobits*

The *kilobits* value is in the valid range of 1 to 10,000,000, expressed in Kbps. This allows the router administrator to place a value as high as 10 terabits per second (Tbps) on an interface. With one-Tbps technology proven and the capability to continually increase the bandwidth on fiber optic networks, this value may have a shorter life span than anticipated. The metric bandwidth value used by Cisco IOS running the EIGRP process for serial interfaces defaults to 1.544 Mbps (T1 speed). This warrants that if the serial lines connecting to the router are less than a T1, then the bandwidth interface command can be specified to allow EIGRP to calculate the true metric. For example, a 56-Kbps serial connection should be defined as

```
interface serial 0
 bandwidth 56
```

Since bandwidth is such a major factor in determining the metric for a route, great care must be used when using the bandwidth interface command.

Suppose the metric weights command was entered where the k2 value was changed from 0 to a value of 1. The command sequence is as follows:

```
router eigrp 200
 metric weights 0 1 1 1 0 0
```

In this instance, the load metric becomes the deciding variable. The load value for an interface is calculated every five seconds based on a five-minute weighted average. The load is presented as a fraction of 255, where 255 is a saturated link experiencing 100 percent utilization. Setting the k2 coefficient to a 1 can result in a route going into holddown when there is an increase in load due to a large FTP or some other type of action that can consume bandwidth over a period of time. The net effect is that the metric ends up doubling the bandwidth, which can possibly make a slow link artificially inflated with bandwidth. In such a scenario, not only can a load rise fast enough to make a route unstable, it can also fall just as fast. A falling load can change the metric to become favorable and cause the Cisco router send link state updates.

In addition to the formula noted above involving k values k1-k3, k5 and k4 come into play if the k5 value specified on the metric weights EIGRP router command is not zero. When this occurs, the following formula is also calculated to create the final metric for the route:

metric = metric x [k5 / (reliability + k4)]

A long-hand method for determining the metric can be performed for a specific router by performing the following:

1. Determine the smallest of all the bandwidths on the route from outgoing interfaces and divide 10,000,000 by that number. (The bandwidth is scaled by 10,000,000 in kilobits per second.)

2. Add all the delays on the route from the outgoing interfaces and divide this number by 10. (The delay is in 10s of microsecs.)

The path with the smallest metric is the best path. These values can be found by issuing the router operation command show interfaces. An example for this is as follows:

```
R1#sh int e 0
 Ethernet 0 is up, line protocol is up
 Internet address is 172.16.9.22, subnet mask is 255.255.255.0
 MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec, rely 255/255, load
   1/255
Serial 0 is up, line protocol is up
 Internet address is 172.18.19.9, subnet mask is 255.255.255.224
 MTU 1500 bytes, BW 784 Kbit, DLY 20000 usec, rely 255/255, load
   20/255

 R2#sh int s 0
 Serial 0 is up, line protocol is up
 Internet address is 172.19.42.1, subnet mask is 255.255.255.224
 MTU 1500 bytes, BW 448 Kbit, DLY 20000 usec, rely 255/255, load
   1/255
Serial 1 is up, line protocol is up
 Internet address is 172.18.19.8, subnet mask is 255.255.255.224
```

```
MTU 1500 bytes, BW 224 Kbit, DLY 20000 usec, rely 255/255, load
  3/255
R1# sh ip route `R2-s0`
 Total delay is 40,000 microseconds, minimum bandwidth is 448 Kbit
 Known via "eigrp 2022", distance 100, metric 26321
```

The calculation:

Metric = bandwidth + delay

= 10,000,000/448 + (20,000 + 20,000)/10

= 26321

```
R2# sh ip route `R1-e0`
Total delay is 21,000 microseconds, minimum bandwidth is 224 Kbit
Known via "eigrp 2022", distance 100, metric 46742
```

Metric = bandwidth + delay

= 10,000,000/224 + (20.000 + 1000)/10

= 46742

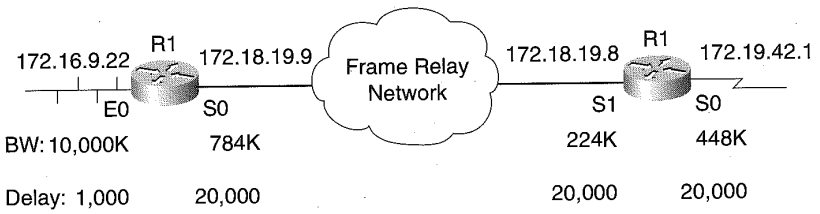


Figure 16-7 Router configuration for determining metric values.

Table 16-1 Default Delay and Bandwidth Values Used by EIGRP for Specific Media Types

Media Type	Delay	Bandwidth
Satellite	5,120 (2 seconds)	5,120 (500 Mbits)
Ethernet	25,600 (1 ms)	256,000 (10 Mbits)
1.544 Mbps	512,000 (20,000 ms)	1,657,856 bits
64 kbps	512,000 (20,000 ms)	40,000,000 bits
56 kbps	512,000 (20,000 ms)	45,714,176 bits
10 kbps	512,000 (20,000 ms)	256,000,000 bits
1 kbps	512,000 (20,000 ms)	2,560,000,000 bits

Routing Between Disconnected Networks with EIGRP

Route summarization is enabled by default when using both IGRP and EIGRP routing protocols. The Cisco IOS software will summarize the IGRP or EIGRP routes to the classful network boundary across classful network boundaries. Route summarization is the process of summarizing routes with long masks to routes with shorter masks that still meet the initial routing requirement. Summarization only uses the full prefixes of each IP network classification, meaning 8 bits for a Class A, 16 bits for a Class B, and 24 bits for a Class C address.

Summarization is useful in EIGRP due to its support of VLSM. For example, in Figure 16-8, router R1 connects to router R2 through serial interface 0 and router R3 through serial interface 1.

The networks connected to router R2 and router R3 are summarized in the EIGRP update packets to router R1 from router R2 and R3. EIGRP will summarize the following network addresses:

IP Subnet	Subnet Mask
172.16.8.0	255.255.255.0
172.16.16.0	255.255.255.0
172.16.24.0	255.255.255.0
172.16.32.0	255.255.255.0
172.20.8.0	255.255.255.0
172.20.16.0	255.255.255.0
172.20.24.0	255.255.255.0
172.20.32.0	255.255.255.0

into two routing entries using EIGRP. Using the router sh ip route command on router R1, the summarized routing entries will be displayed as

```
D 172.16.0.0/16 [120/1] via 172.30.1.1, 00:00:01, Serial0/0
D 172.20.0.0/16 [120/1] via 172.31.1.1, 00:00:01, Serial0/1
```

Using route summarization with EIGRP reduces router overhead and bandwidth required to send routing updates.

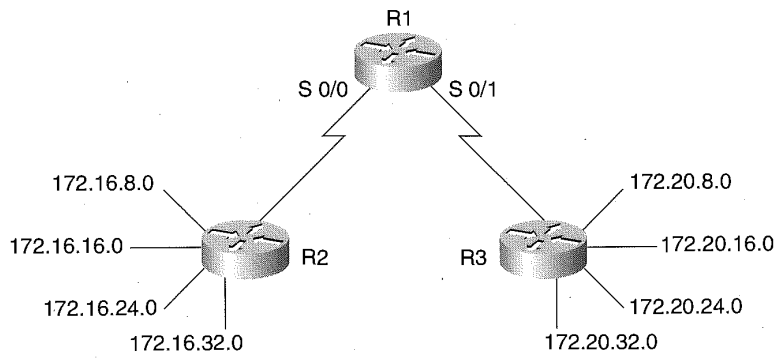


Figure 16-8 EIGRP network configuration used for route summarization.

Route summarization is the desired approach when using EIGRP. If a subnet is disconnected, however, the routers will not be able to correctly advertise the subnets. Figure 16-10 illustrates this issue. Router R2 in the figure has subnet 172.20.8.0 and router R3 has subnet 172.20.16.0, causing a disconnect of the subnets. Router R1 receives summarized entries from router R2 and R3 will indicate two paths to the 172.20.0.0 network, which is not really the case.

The sh ip route command displays the following on router R1 based on router summarization updates from R2 and R3:

```
D 172.20.0.0/16 [120/1] via 172.31.1.1, 00:00:01, Serial0/0
D 172.20.0.0/16 [120/1] via 172.30.1.1, 00:00:01, Serial0/1
```

The resulting routing table will cause connectivity problems for two reasons. First, due to the fact that the hop count is the same, the router will load balance packets between serial interface 0/0 and serial interface 0/1. Secondly, this will cause only 50 percent of the packets to successfully reach their true destination. This type of topology is corrected by disabling route summarization using the following router configuration mode command:

no auto-summary

Entering this in the configuration for routers R2 and R3 will disable summarization and cause the full subnet and host routing information to be transmitted to router R1. The sh ip route command issued on router R1 will now display the following entries:

```
D 172.20.8.0/24 [120/1] via 172.30.1.1, 00:00:01, Serial0/0
D 172.20.16.0/24 [120/1] via 172.31.1.1, 00:00:01, Serial0/1
```

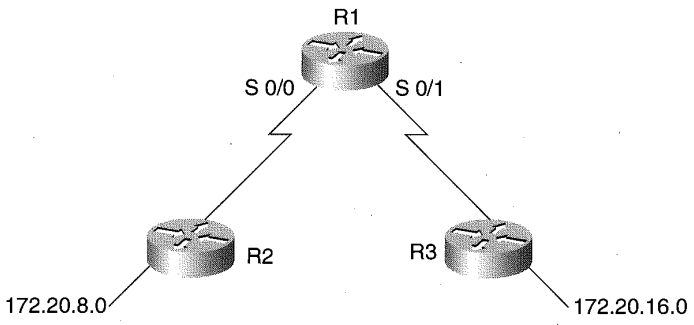


Figure 16-9 Disconnected networks and EIGRP route summarization.

The no auto-summary command on router R2 and R3 forces the EIGRP updates to advertise the full mask for each of the subnets. Now packets through router R1 for resources on subnet 172.20.8.0 will only be directed out serial interface 0/0. Likewise, packets through router R1 for resources on subnet 172.20.16.0 will only be directed out serial interface 0/1.

Summarizing Routes for Advertisements out Specific Interfaces

Further summarization of routes is possible through the use of the following interface configuration command:

ip summary-address eigrp *autonomous-system-number address mask*

The *autonomous-system-number* variable specifies a valid EIGRP AS number in use by this router. The *address* variable is the IP address being summarized. The *mask* variable is the subnet mask that is applied to the value of the *address* variable. Applying this interface command to any interface definition causes the EIGRP process to summarize routes out of this interface that may not be normally summarized. This command is normally not used since EIGRP defaults to route summarization. In instances such as disconnected networks where auto-summarization is disabled, however, the ip summary-address interface command is useful to summarize routes to portions of the network not involved with the disconnected networks.

Figure 16-10 diagrams such a situation. In this figure, network 172.16.0.0 is subnetted using 255.255.252.0 as the subnet mask. In default mode, EIGRP would summarize the route update to router R2 and R3 from router R1. Because auto-summarization is disabled to handle the disconnected network 172.20.0.0, however, each subnet associated with the 172.16.0.0 network would be sent to the neighboring routers. Using the ip summary-address eigrp interface command, the 172.16.0.0 network can be summarized to router R2 and R3, thus reducing routing table overhead and creating a more efficient path for router R2 and R3 to route packets to the 172.16.0.0 network.

The router configuration for aggregating the 172.16.0.0 routes follows:

Router R1 Configuration:

```
interface serial 0/0
 ip address 172.30.1.1 255.255.255.0
 ip summary-address eigrp 200 172.16.0.0 255.255.0.0
!
interface serial 0/1
 ip address 172.31.1.1 255.255.255.0
 ip summary-address eigrp 200 172.16.0.0 255.255.0.0
!
interface Ethernet 0
 ip address 172.16.4.1 255.255.252.0
!
interface Ethernet 1
 ip address 172.16.8.1 255.255.252.0
!
interface Ethernet 2
 ip address 172.16.12.1 255.255.252.0

router eigrp 200
 network 172.16.0.0
 network 172.30.0.0
 network 172.31.0.0
 no auto-summary
```

Tuning Hello Packet and the Hold Time intervals

Hello packets are sent periodically over interface connections between routing devices to enable the discovery of neighbor EIGRP routers. The hello packets are sent only over interface connections whose IP address network is specified under the router eigrp command using the router configuration network command. These hello packets are also used to determine if a neighbor has become unreachable or inoperative.

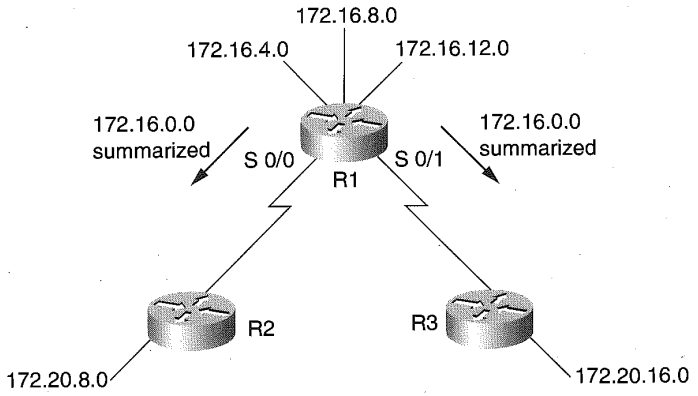
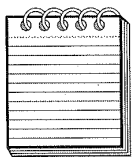


Figure 16-10 Aggregating route summarization for a specific network.

Every five seconds the EIGRP process sends a hello packet out EIGRP interfaces by default for multi-access networks and high-speed point-to-point interfaces. For low-speed serial interfaces and NBMA networks, the hello packet is sent every 60 seconds by default. In high-speed NBMA networks, the default is also five seconds.



NOTE: *Frame Relay and SMDS networks are considered NBMA networks only when the interface is not configured to use physical multicasting.*

Because EIGRP uses the bandwidth value assigned to an interface to determine the speed of a link, it can also alter the number of hello packets sent over the connection.

To help avoid sending too many hello packets over slow links due to inflated bandwidth or too few over-artificially low bandwidth values, use the following interface command:

ip hello-interval eigrp *autonomous-system-number seconds*

The *autonomous-system-number* variable specifies a valid EIGRP AS number in use by this router. The variable is the interval in seconds for sending hello packets out the specified interface.

The hold time is the amount of time advertised to the neighbor that the sender on this link is operational, even if hello packets have not been received. The hold time value defaults to 180 seconds for slow serial links

and slow NBMA networks. For high-speed NBMA and multiaccess networks, the default hold time is three times the hello-packet interval. These defaults may not be a long enough interval in congested and very large networks, but because the hold time is affected by the value of the bandwidth interface command and its direct affect on route convergence within the network, great care and planning is advised. If electing to modify the hold time interval, the following interface configuration command must be entered:

ip hold-time eigrp *autonomous-system-number seconds*

The *autonomous-system-number* variable specifies a valid EIGRP AS number in use by this router. The variable is the interval in seconds allowed without receiving hello-packets prior to determining that the neighbor router and routes pointing to it are unavailable.

Split-Horizon and EIGRP

A router advertising routes out of an interface that the routing information was received on frequently causes routing loops. The split-horizon feature employed for distance-vector routing protocols like RIP reduces the possibility of routing loops by blocking routing updates to connected routers from which the routing update was originally received.

As an added insurance, EIGRP also uses the split-horizon algorithm. Figure 16-11 diagrams this scenario. This type of scenario is most prevalent on broadcast type IP networks such as LANs. As shown in the figure, the routing advertisements received by router R1 for networks connected to router R2 are not sent to router R2 from router R1 in its routing update packets. Likewise, router R2 does not send routing advertisements to router R1 for networks attached to router R1.

Spilt-horizon is disabled using the following interface configuration command:

no ip split-horizon eigrp *autonomous-system-number*

Enabling split-horizon for a specific interface is accomplished by entering the following interface command:

ip split-horizon eigrp *autonomous-system-number*

Non-broadcast networks like Frame Relay, however, require split-horizon to be disabled. The Cisco IOS automatically disables split-horizon for

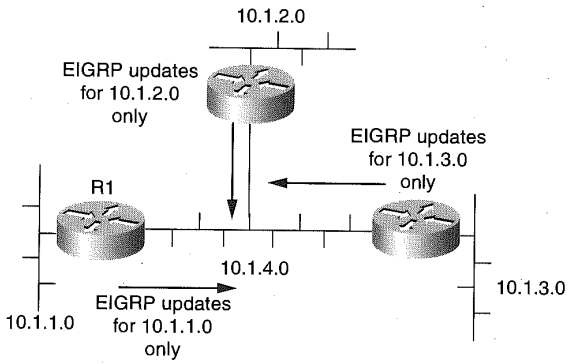


Figure 16-11
The effect of split-horizon on broadcast networks with EIGRP.

Frame Relay and SMDS networks when EIGRP is used as the routing protocol for those connections. Split-horizon is enabled for all other network connections. If split-horizon is disabled on an interface connecting a packet-switched network like X.25, it must be disabled on all other routers connecting to the same packet-switched network.

MD5 Authentication with EIGRP

EIGRP supports router authentication as a means of providing security for routing table updates. Using MD5 authentication, Cisco routers are protected from advertising routes to and receiving routes from promiscuous unauthorized routers. Authentication is defined on the interface level and hence provides for discreet selection of which interfaces must use authentication.

Authentication is determined by the configuration of a key chain specific to the interface. The key chain is identified by using the following interface command:

ip authentication key-chain eigrp *autonomous-system key-chain*

The *key-chain* operand is the name given to the valid group of keys. Specifying the **ip authentication key-chain eigrp** command enables EIGRP authentication. If this command is not specified on an interface using EIGRP, then authentication is not performed.

Along with enabling authentication, the mode of authentication can be specified. The mode of authentication defaults to plain, clear text that

becomes a security risk on connections and can be deemed unsecured. Encrypted authentication must be specifically specified. The following interface command is used to define the type of mode used for authentication on an interface:

ip authentication mode eigrp *autonomous-system* **md5**

The *autonomous-system* value is the AS number of an EIGRP process. The **md5** authentication technique is the only valid keyed authentication algorithm used with EIGRP.

Authentication is enabled using the **ip authentication key-chain eigrp** interface command, but the authentication keys must be defined in order for authentication to occur. Authentication keys themselves are defined using the following global configuration commands:

key chain *name-of-chain*

key *number*

key-string *text*

accept-lifetime *start-time* {**infinite** | *end-time* | **duration** *seconds*}

send-lifetime *start-time* {**infinite** | *end-time* | **duration** *seconds*}

The value of the *key-chain* variable on the key-chain command must match a defined **ip authentication key-chain eigrp** command on an interface to exercise the authentication key chain. The *number* variable of the key command identifies the number of the key being defined. The *text* variable on the key-string command is the authentication text identified by the key number used for authentication. Multiple keys can be configured as well.

The key numbers supplied in the key chain definition are searched from lowest to highest, and the first valid key found is then authenticated. The router sends only one authentication path despite multiple key definitions. The key number value, in combination with the interface associated with the packet, is used by the authentication algorithm and MD5 authentication to uniquely identify the authentication key.

The keys can be programmed to be active during specific times of a given day. This is done with the **accept-lifetime** and **send-lifetime** commands. The *start-time* variable of the accept-lifetime and send-lifetime commands defines when the key will be activated. The *end-time* of the accept-lifetime and send-lifetime commands defines the end on a given day when the key will no longer be in use. The format of the *start-time* and *end-time* variable is

hh:mm:ss month day year

The *hh:mm:ss* variable is the two-digit 24-hour clock representation of the time of day. The month variable is the first three letters of the month, the day is the number of the day of the month, and the year variable is the four-digit number for the year. The *start-time* and *end-time* variables default to an infinite time period starting from January 1, 1993. The **infinite** keyword can be used as the end time value, indicating that the key never expires after the value of the *start-time* variable has been reached. The **duration** keyword, followed by a valid value for the *seconds* variable, can also be used instead of the end-time variable value to define the usage length for the key. The following is an example of a configuration using RIP Version 2 and authentication between two Cisco routers.

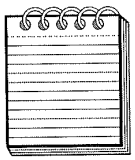
Router R1 Configuration:

```
interface Ethernet 0
 ip address 10.16.8.1 255.255.255.0
 ip authentication mode eigrp 200 md5
 ip authentication key-chain eigrp 200 colors
!
router eigrp 200
 network 10.0.0.0
!
key chain colors
 key 1
 key-string blue
 accept-lifetime 07:00:00 Feb 28 1999 duration 7200
 send-lifetime 08:00:00 Feb 28 1999 duration 3600
 key 2
 key-string green
 accept-lifetime 00:00:00 Mar 1 1999 infinite
 send-lifetime 00:30:00 Mar 1 1999 infinite
```

Router R2 Configuration:

```
interface ethernet 0
 ip address 10.16.8.2 255.255.255.0
ip authentication mode eigrp 200 md5
 ip authentication key-chain eigrp 200 colors
!
router eigrp 200
 network 10.0.0.0
 version 2
!
key chain colors
 key 1
 key-string blue
 accept-lifetime 07:30:00 Feb 28 1999 duration 7200
 send-lifetime 06:30:00 Feb 28 1999 duration 3600
 key 2
 key-string green
 accept-lifetime 23:30:00 Feb 28 1999 infinite
 send-lifetime 00:00:00 Mar 1 1999 infinite
```

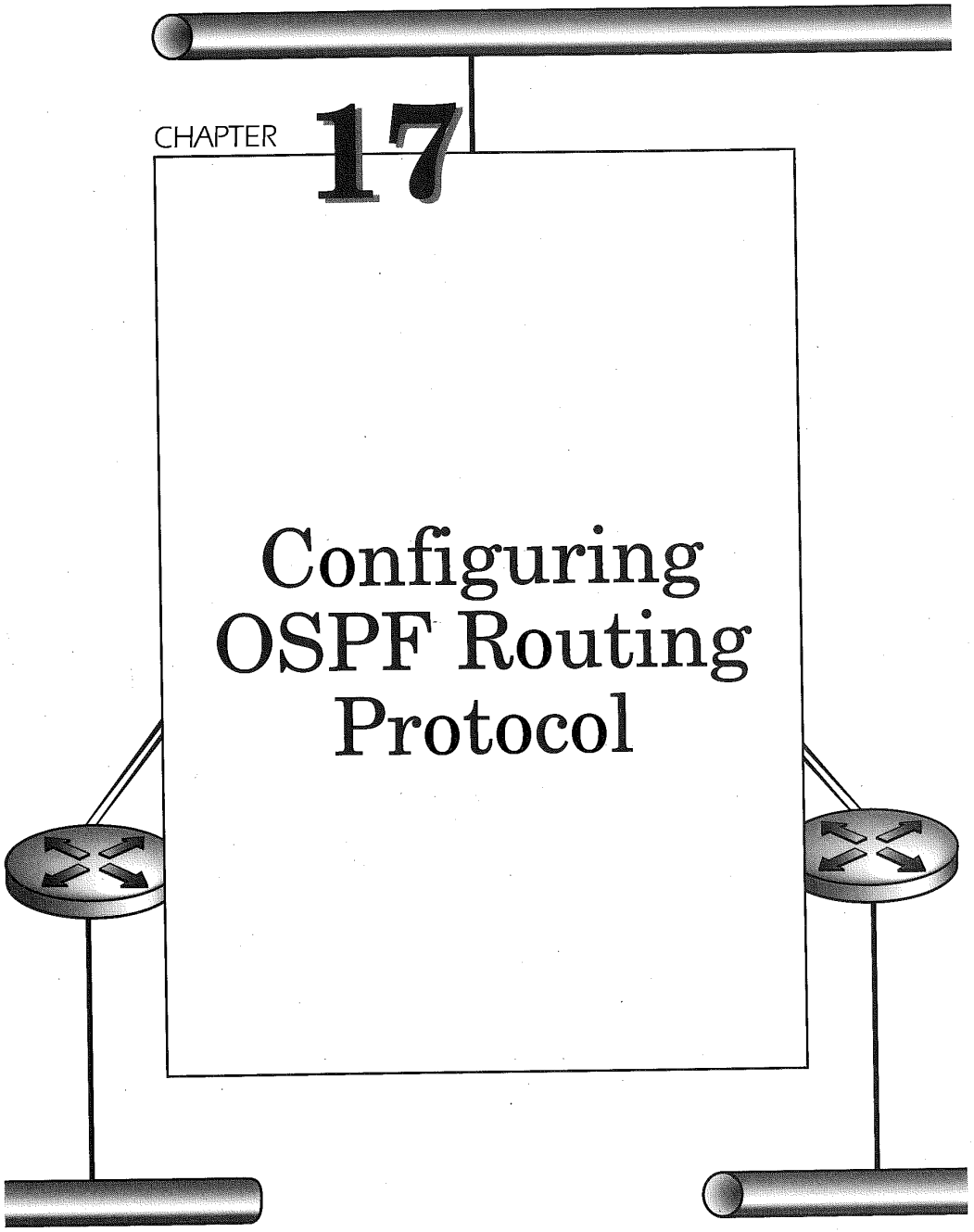
In these configurations, the two routers first use the keys identified by number 1 because the search order is from lowest to highest. The routers then use the number 2 key to authenticate routing updates between them. The start time on router R1 is defined to ensure the acceptance and delivery of EIGRP updates to router R2, given a 30-minute cushion on the time differences between the routers. The second key (number 2) on each router takes effect once the clocks on each router have met the start-time criteria and last indefinitely.



NOTE: *Both routers must be using EIGRP and have authentication enabled, in addition to the accept and send lifetime values being coordinated. Both routers must also specify the same key-string that will then force the router to examine the accept and send lifetime values.*

CHAPTER 17

Configuring OSPF Routing Protocol



Open Shortest Path First (OSPF) is a link-state IP routing protocol. Developed in 1988 by the Internet Engineering Task Force (IETF) and recently updated in RFC 2178, OSPF addresses the scalability concerns of large global IP networks that are not addressable by distance-vector-based IP routing protocols such as RIP and IGRP. OSPF meets the scalability concerns of distance-vector routing protocols by addressing the following:

- *Convergence:* OSPF route convergence uses a flooding mechanism to update neighboring OSPF routers within an area of new routes. Only the affected routes are updated and, due to route summarization, if the route is part of a summarized route to another area, the update remains only within the area affected.
- *Variable Length Subnet Mask (VLSM) support:* RIP1 and IGRP do not support VLSM, which makes OSPF a viable alternative for managing large IP networks within a restrictive address range.
- *Network Diameter:* In RIP, IGRP, and EIGRP networks the diameter (hops) between networks has a limit. The concept of areas and building hierarchical networks based on the areas enables OPSF to have a virtually limitless diameter.
- *Efficient bandwidth use:* Because OSPF is a link-state routing protocol, only updates of affected routes are sent to neighboring routers, instead of the entire routing table. OSPF uses a multicast link-state update (LSU) packet to accomplish the routing updates and only sends the LSU when a network change has occurred.
- *Advanced route selection:* OSPF uses bandwidth as a deciding factor in determining the optimal path between two networks. RIP1 and RIP2 rely on hop counts.

OSPF routers establish adjacencies upon the activation of the connections defined to the OSPF routing protocol process. The adjacencies are established using the Hello protocol. There are nine variables used in the Hello protocol that enable an OSPF router to form bidirectional communication with adjacent OSPF routers. Table 17-1 lists a table identifying these nine variables.

OSPF routers have a unique identifier for the router, called the router ID. Each router selects the highest dotted-decimal IP address on any active interface as the router ID. The exception to this rule is when the loopback interface is specified on the router. When loopback interfaces are defined, the highest IP address assigned to any of the loopback interfaces is chosen as the router ID. The router ID is important in the selection of the designated and backup routers for the OSPF network. If the interface goes down,

Table 17-1

OSPF Adjacency Variables Found in the Hello Packet

OSPF Adjacency Variable	Variables that Must Match Between Adjacencies
Router-ID	
Hello/Dead intervals	■
Neighbors	
Area-ID	■
Router priority	
Designated router (DR) IP address	
Backup DR (BDR) IP address	
Authentication password	■
Stub area flag	■

the router is unreachable. To avoid this situation, it is prudent to define a loopback interface as the forced OSPF router ID.

The hello/dead intervals are the predetermined timers used by all adjacent routers for specifying the frequency in seconds that a router will send hello packets or determine that an adjacent neighbor router is declared down. Neighboring routers must be using the same timing values for these intervals. If the values do not agree, the neighbors may believe that a router has gone down but is, in fact, still active on the network. The default of hello messages is 10 seconds and for dead intervals it is 40 seconds on non-NBMA interfaces.

The value for neighbors in the Hello packet is the list of adjacencies created by the router sending the hello packet. In initial contacts, this field is empty.

The area-ID is a definition that creates a segmented OSPF network. Using the same area-ID in all OSPF routers denotes a single commonly shared network, including the IP address subnet and mask. Each router on the shared segment with the same area-ID will have the same link-state database.

The router priority field of the hello packet indicates to the neighboring router(s) whether this router is eligible to become a designated or backup router. The router with the highest priority is the selected designated router (DR) and the router with the second highest priority is the backup designated router (BDR). If the highest priority values are equal, the router with

the highest router-ID becomes the DR and the router with the second highest router id becomes the BDR. A router specified with a priority of 0 can never be eligible as a DR or BDR, and Cisco IOS defaults the priority value to 1. The DR and BDR fields in the hello packet are the IP addresses of the current DR and BDR.

The authentication password field is filled in only when authentication is enabled on the router. The password provided by the Cisco IOS for authentication must be the same on the routers exchanging the hello packets. If authentication is set for one router and not another, an adjacency is not established. To ensure adjacencies with authentication, all peer routers participating on the OSPF network should use the same authentication password.

The last field used in the hello packet is the stub area flag. This field indicates to the peer router that the router sending the hello packet is in a stub area. A stub area is generally defined as a network with a single router having a single exit point from a router into the OSPF network. It is the exchange of the information in the hello packets that enables an OSPF network to function appropriately.

OSPF with Cisco IOS

Cisco IOS fully supports the OSPF RFC 2178, Not-So-Stubby-Area (NSSA) RFC 1587, and OSPF over-demand circuit RFC 1793. Cisco IOS at this point in time does not support Multicast-OSPF (MOSPF) Link State Advertisement (LSA) Type 6 packets. The enhancements to OSPF, specific to Cisco IOS features, include the following:

Totally Stubby Areas: This Cisco IOS-specific feature works only when Cisco routers are involved between a stub area and an OSPF area. The feature reduces OSPF routing information from the network link between the OSPF Area Border Router (ABR) that connects to the stub area. The Cisco ABR forwards the default summary link 0.0.0.0 into the stub area. In this way, as shown in Figure 17-1, bandwidth on the link to the totally stubby area router is saved for data traffic. The totally stubby area feature blocks external type5 LSAs and summary type 3 and 4 intra-area LSAs from being sent into the stub area.

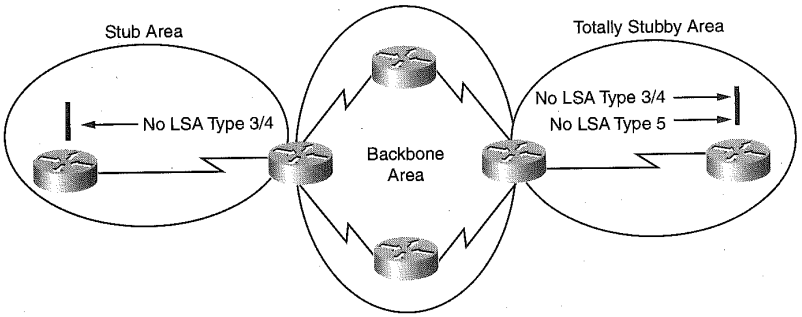
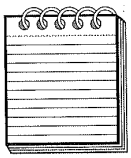


Figure 17-1
Totally stubby area feature on Cisco routers for reducing route table size.



NOTE: *Use Totally Stubby Areas* only *with Cisco routers. In a mixed Cisco and non-Cisco environment, specify the area configuration as a stub network.*

Cisco further increases OSPF network stability and scalability by employing an interface output cost, retransmission timer, and interface transmit delay timers.

Link output cost: Cisco IOS allows the router administrator to apply a cost to a specific link. This is useful in forcing OSPF route selection over preferred physical connections to destination networks from the router.

LSA retransmit interval: Using this feature, the router administrator can increase or decrease the default retransmission of LSAs to the neighbor router. The retransmit is used if the neighbor route has not responded to the LSA previously sent.

LSA transmit delay: The value specified here is used in conjunction with the retransmit interval in determining an unsuccessful delivery of an LSA over a link.

Specifying OSPF on Cisco Routers

Taking the defaults for all router configurations within an OSPF area's internal routers, area border routers, and autonomous system boundary

routers (ASBR) will most certainly provide an operational network. Much of the decision making is up to the OSPF neighbor adjacency negotiations for determining the DR and BDR, however, with no authentication of peer routers joining the network, and no default values that protect valuable bandwidth and processor utilization. Defining network-specific values for the OSPF area will enable the network engineer to create a deterministic OSPF configuration versus an undetermined topology.

Creating the OSPF Routing Process

The Cisco IOS software initializes an OSPF routing protocol process by the router administrator entering the global configuration command:

router ospf *process-id*

Defining **router ospf**, followed by a *process-id* number, spawns an OSPF routing process with the assigned process-id. Unlike IGRP and EIGRP, the value here is specific to the router and is not used to identify different OSPF autonomous systems. Multiple OSPF processes can be executed on any given router using unique process-ids for each. It is more common practice, however, to have one OSPF process running on any given router. The value for the process-id parameter ranges from 1 to 65,535.

The OSPF process defined must be associated with an active IP interface on the router in order for OSPF to begin building neighbor adjacencies and routing tables. The process becomes aware of which interfaces are using OSPF by defining the network area command in the following format under the router ospf global command:

network *address wildcard-mask* area *area-id*

The *address* parameter can be either the IP address of the interface, the subnet, or network address of the interface to which OSPF routing is to be applied.

Paired with the *address* parameter is the *wildcard-mask* parameter. The value specified for the *wildcard-mask* parameter identifies which bit of the *address* parameter value is used for interpreting the address parameter value. The *wildcard-mask* is expressed in dotted-decimal format. Using the following example,

```
network 172.16.0.0 0.0.255.255 area 0
```

the network address value, 172.16.0.0, is used with an applied mask of 0.0.255.255. All interfaces on the router using the Class B network address

of 172.16.x.x will have LSA sent and received, as well as Hello protocol packets being sent for neighbor adjacency establishment and DR/BDR determination. If the example

```
network 172.16.8.0 0.0.0.255 area 0
```

were used, the wildcard-mask implies that only interfaces using the 172.16.8.0 subnet would use OSPF as the routing protocol. The mask applied to the address is an entity unto itself and it does not have to match the mask used for the IP address defined on the interface. For example, if the router were configured as follows,

```
interface serial 0
ip address 172.16.8.16 255.255.255.248
!
interface serial 1
ip address 172.16.8.24 255.255.255.248
!
router ospf 1
network 172.16.8.0 0.0.0.255 area 0
```

the OSPF process would use both interfaces for OSPF routing since the first three bytes of the address parameter value are met by the wildcard-mask. If the following configuration were in use,

```
interface serial 0
ip address 172.16.8.1 255.255.255.0
!
interface serial 1
ip address 172.16.9.1 255.255.255.0
!
router eigrp 100
network 172.16.0.0
!
router ospf 1
network 172.16.9.1 0.0.0.0 area 0
```

then interface serial 0 would use only EIGRP routing protocol, while interface serial 1 would use both EIGRP and OSPF as routing protocols. When defining a specific interface to use the OSPF routing protocol, the full IP address assigned to the interface can be specified. In such a case as this example illustrates, the *wildcard-mask* must specify all 0s, indicating an exact match is required.

The *area-id* parameter value identifies to which OSPF area the specified network is associated. The *area-id* can be a decimal number in the range of 0 to 4,294,967,295, or it can be written in the dotted-decimal format of an IP address. Using an IP network or subnet dotted-decimal format to represent the area is often used to assist in identifying the attached area with a true network meaning. In the following example, the IOS configuration

```
router ospf 1
 network 10.0.0.0 0.255.255.255 area 10.1.0.0
```

is particularly useful because the area-id value specified identifies the IP subnet of the attached network. For the most part, installations use decimal values. If the OSPF network being defined has a single area, the area-id value must equal 0. This is because OSPF treats area 0 as the backbone area that connects all other OSPF areas.

The address parameter value must specify the primary IP address, subnet, or network of an active interface to enable OSPF for the interface. Specifying the IP address, subnet, or network of a secondary IP address for an active interface will not enable OSPF for the interface. Here's an example:

```
interface Ethernet 0
 ip address 172.16.1.1 255.255.255.0 secondary
 ip address 10.1.1.1 255.255.255.0
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

The Ethernet connected to interface Ethernet 0 will not be accessible from networks connecting to this router from other routers because the primary IP network 10.0.0.0, the subnet 10.1.1.0, and the exact IP address 10.1.1.1 are not specified on the network area command.

The Cisco IOS performs an OR operation when determining whether an interface is to have OSPF enabled. The process is as follows:

1. The *wildcard-mask* value is logically ORed with the IP address of each active interface.
2. The *wildcard-mask* value is then logically ORed against the address parameter value of the network area command.
3. The resulting ORed values are compared.
4. If the resulting values match, then the Cisco IOS enables OSPF for the interface and attaches the interface to the OSPF area specified by the area-id parameter value.

For example, if the IP address of an interface is 8.2.2.1 and the *wildcard-mask* is 0.0.255.255, the ORed value is

```
00001000.00000010.00000010.00000001
00000000.00000000.11111111.11111111
00001000.00000010.11111111.11111111
```

If the **network area** command *address* parameter specifies 8.2.0.0 with a *wildcard-mask* of 0.0.255.255, the ORed value is

```
00001000.00000010.00000000.00000000
00000000.00000000.11111111.11111111
00001000.00000010.11111111.11111111
```

Comparing the resulting values, we find that the results match and hence any interface using 8.2.x.x in the first two octets of the IP address

field will have OSPF enabled. Now suppose the IP address of the interface is 8.3.1.1 and the *wildcard-mask* is 0.0.255.255:

```
00001000.00000011.00000001.00000001
00000000.00000000.11111111.11111111
00001000.00000011.11111111.11111111
```

Now we compare this interface logically ORed result with the network area result found earlier:

```
00001000.00000011.11111111.11111111 IP address 8.3.1.1 logically ORed result
00001000.00000010.11111111.11111111 network area 8.2.0.0 logically ORed result
```

The resulting values do not match; hence, the interface using the 8.3.1.1 IP address will not have OSPF enabled.

In Figure 17-2, a single OSPF area topology is shown. The three routers within the figure are all connected using serial attached lines. Since serial point-to-point links are connecting only two routers, there is no use for a DR or BDR.

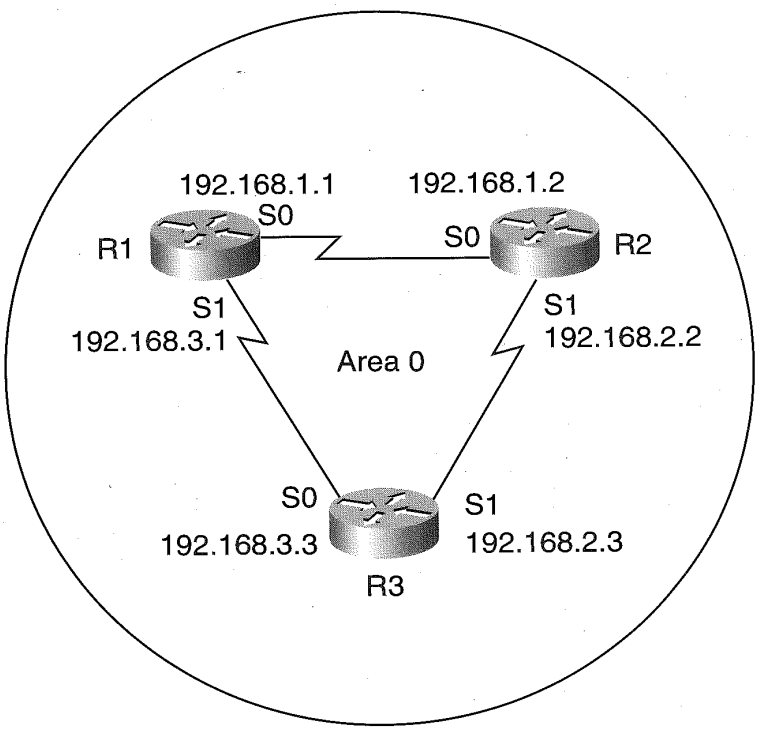


Figure 17-2
A simple point-to-point OSPF single area network.

The following are the configurations used to enable OSPF in Figure 17-2.

Router R1 Configuration:

```
interface serial 0
 ip address 192.168.1.1 255.255.255.0
!
interface serial 1
 ip address 192.168.3.1 255.255.255.0
!
router ospf 1
 network 192.168.0.0 0.0.255.255 area 0
```

Router R2 Configuration:

```
interface serial 0
 ip address 192.168.1.2 255.255.255.0
!
interface serial 1
 ip address 192.168.2.2 255.255.255.0
!
router ospf 1
 network 192.168.0.0 0.0.255.255 area 0
```

Router R3 Configuration:

```
interface serial 0
 ip address 192.168.3.3 255.255.255.0
!
interface serial 1
 ip address 192.168.2.3 255.255.255.0
!
router ospf 1
 network 192.168.0.0 0.0.255.255 area 0
```

In these configurations, the OSPF process in each router is specified as 1. This is done simply to ease configuration effort. The OSPF process-id in each router could have been defined using different values in each router, which must specify the same area-id value. If the area-id values are not in agreement, the exchange of hello packets would not result in the formation of neighbor adjacencies and hence each router would not be able to reach the networks attached to the adjacent routers. Since this is a single OSPF area network, the area-id must have a 0 specified in order for OSPF to create a routing table.

DR and BDR Election Using a Loopback Interface

Suppose the network were a data center backbone configuration, as shown in Figure 17-3, where each router on the Ethernet backbone is participating in the OSPF area. In this configuration, the routers would be sitting on a broadcast network and would therefore go through a DR and BDR election process.

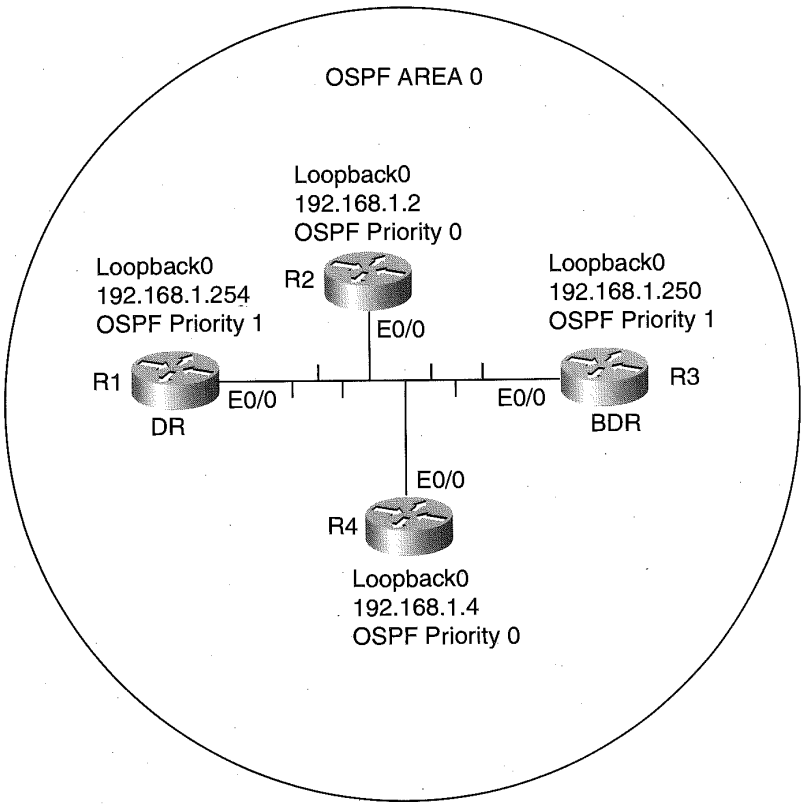


Figure 17-3 A single area OSPF network over Ethernet using DR and BDR.

The following router configurations detail the OSPF parameters:

Router 1 Configuration:

```
interface Ethernet 0/0
 ip address 172.16.1.1 255.255.255.0
!
interface loopback 0
 ip address 192.168.1.254 255.255.255.252
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 network 192.168.1.254 0.0.0.0 area 0
```

Router 2 Configuration:

```
interface Ethernet 0/0
 ip address 172.16.1.2 255.255.255.0
 ip ospf priority 0
!
interface loopback 0
 ip address 192.168.1.2 255.255.255.252

router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 network 192.168.1.4 0.0.0.0 area 0
```

Router 3 Configuration:

```
interface Ethernet 0/0
 ip address 172.16.1.3 255.255.255.0
!
interface loopback 0
 ip address 192.168.1.250 255.255.255.252

router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
network 192.168.1.250 0.0.0.0 area 0
```

Router 4 Configuration:

```
interface Ethernet 0/0
 ip address 172.16.1.4 255.255.255.0
ip ospf priority 0
!
interface loopback 0
 ip address 192.168.1.4 255.255.255.252


router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 network 192.168.1.4 0.0.0.0 area 0
```

In this configuration example, the use of the loopback interface, along with optional OSPF interface commands, is shown in determining the DR and BDR for the OSPF area. The DR election process is determined by first discovering which router on the OSPF broadcast network has the highest router priority. The second-highest router priority value presented by a router in the OSPF broadcast network becomes the BDR.

The OSPF process in Cisco IOS defaults the router priority value to 1. Since this automatically generates a tie with any other participating OSPF router, the tie is broken by determining the highest router ID. The router ID within a router is determined by the OSPF process of selecting the highest IP address assigned to any enabled OSPF interface, which is only superseded by the configuration of a loopback interface.

Allowing the OSPF election process to decide on the DR and BDR can become critical in busy backbone areas and creates an unknown topology. A Cisco router administrator can modify the OSPF DR/BDR election process by using the interface command ip ospf priority. The format of the command is as follows:

ip ospf priority *number*

The *number* parameter value can range from 0 to 255, where 0 is the default value. The value 255 is the highest value allowed. As shown in the configurations above that detail the topology diagrammed in Figure 17-3, we can see how using the ip ospf priority interface command, along with the IP address assignment of a loopback interface, can predetermine the DR and BDR on a broadcast network.

The OSPF process for Cisco automatically selects a loopback interface IP address as the preferred router ID over other interfaces. This is because the loopback interface is always active. If multiple loopback interfaces are defined, the OSPF process selects the highest IP address of all the defined loopback interfaces as the router ID.

Combining this knowledge with the ip ospf priority command, router R1 becomes the DR, and router R3 becomes the BDR for the sample network configuration. The election of router R1 as the DR is due to the fact that the ip ospf priority value defaults to 1, and the router ID presented in the Hello packet is the IP address 192.168.1.254 on the loopback 0 interface of router R1. Router R3 defaults the ip ospf priority value to a 1, making it DR-eligible. Because the router ID presented by router R3 is the IP address 192.168.1.250, however, which is found on the loopback 0 interface of router R3, it is elected as BDR since its IP address is lower than that presented by router R1.

Routers R2 and R4 each have loopback 0 interface definitions, yet the ip ospf priority value of 0 defined on the Ethernet interface connecting the router to OSPF area 0 indicates that these routers are not DR-eligible and therefore will not participate in the election process.

Predetermining the DR and BDR on a broadcast network, as described above, enables reduced link-state updates because LSAs are only delivered to the assigned DR and BDR routers. This allows for added design enhancements so that these routers only take care of the OSPF routing tables and are not part of delivering the data traffic to end user networks. Secondly, if all the routers are DR-eligible, an election storm process could occur on large router backbone networks. If the DR were to become inoperative, the BDR becomes the DR. In the above example, there is no BDR once this happens. However, once the router R1 connects to the network again, it becomes the new BDR until router R3 disconnects from the network.

Multi-Area OSPF Networks

The concept of areas within an OSPF network enables extremely large scalable network topologies. Using OSPF hierarchical area topologies addresses the scalable issues found in a single large OSPF area network. The advantages to a hierarchical topology are as follows:

- CPU overhead is reduced, due to frequent Shortest Path First (SPF) calculations.
- Routing table size is kept to a minimum.
- Route summarization minimizes the LSU overhead, protecting bandwidth.

Figure 17-4 illustrates a simple OSPF hierarchical topology, which is also referred to as hierarchical routing. A hierarchical routed network is a single autonomous system broken into smaller, more manageable networks termed areas. Routing between the areas is called inter-area routing, and routing within an area is termed intra-area routing. Because OSPF views each area as a network unto itself, SPF calculations for changes within an area are performed only by the routers within the area. In designing an OSPF hierarchical routing network, architecting a well-thought out IP

addressing scheme can further reduce routing table updates between areas by using route summarization. Because of this summarization, fewer LSUs are required to update the entire OSPF network.

OSPF Route Summarization

Summarizing routes between OSPF areas is the key to achieving a scalable hierarchical routing topology. OSPF summarizes two types of routes: intra-area (IA) routes and external routes. The IA routes are summarized by Area Border Routers (ABR) and external routes are summarized by Autonomous System Border Routers (ASBR). A router can perform both the ABR and ASBR functions simultaneously. Proper route summarization requires a contiguous IP address space for each OSPF area. Using discontiguous IP addressing between areas can cause an OSPF router to forward packets erroneously.

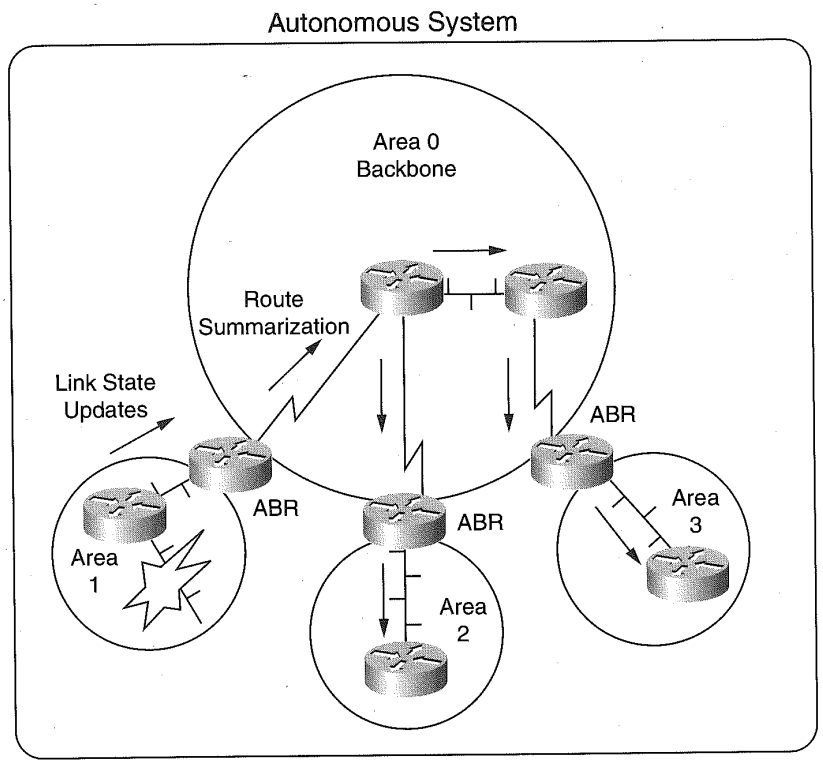


Figure 17-4 OSPF hierarchical routing topology.

Figure 17-5 illustrates the use of route summarization between OSPF areas. Both router R1 and R2 are ABR routers connecting OSPF area 0 to OSPF area 1 and area 2. Router R1 connects both area 0 and area 1, while router R2 connects area 0 to area 2. Summarizing routes between these areas enables summarization in both directions. Area 0 has reserved the address range 172.20.96–127.0, using 255.255.255.0 as the mask. Area 1 has reserved the address range 172.20.32–63.0, using 255.255.255.0 as the mask. Area 2 has reserved the address range 172.20.64–95.0, using 255.255.255.0 as the address range.

The following configuration example highlights the commands for the topology in Figure 17-5:

Router 1 Configuration:

```
interface Ethernet 0
 ip address 172.20.96.1 255.255.255.0
!
interface serial 0
 ip address 172.20.32.1 255.255.255.0
!
router ospf 1
 network 172.20.96.1 0.0.0.0 area 0
 network 172.20.32.1 0.0.0.0 area 1
 area 0 range 172.20.96.0 255.255.224.0
 area 1 range 172.20.32.0 255.255.224.0
```

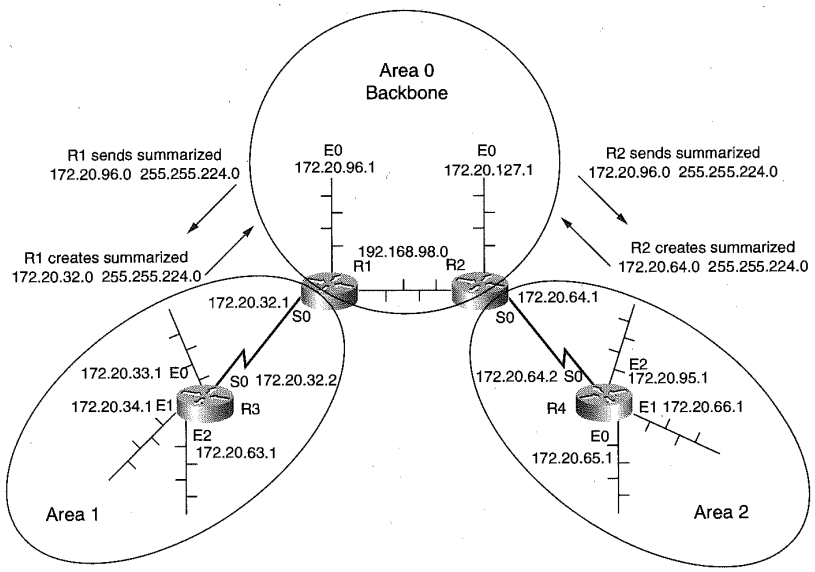


Figure 17-5 OSPF router summarization between areas.

Router 3 Configuration:

```
interface Ethernet 0
 ip address 172.20.33.1 255.255.255.0
!
interface Ethernet 1
 ip address 172.20.34.1 255.255.255.0
!
interface Ethernet 2
 ip address 172.20.63.1 255.255.255.0

interface serial 0
 ip address 172.20.32.2 255.255.255.0
!
router ospf 1
network 172.20.0.0 0.0.255.255 area 1
```

Router R2 Configuration:

```
interface Ethernet 0
 ip address 172.20.127.1 255.255.255.0
!
interface serial 0
 ip address 172.20.64.1 255.255.255.0
!
router ospf 1
 network 172.20.127.1 0.0.0.0 area 0
 network 172.20.64.1 0.0.0.0 area 2
 area 0 range 172.20.127.0 255.255.224.0
 area 2 range 172.20.64.0 255.255.224.0
```

Router R4 Configuration:

```
interface Ethernet 0
 ip address 172.20.65.1 255.255.255.0
!
interface Ethernet 1
 ip address 172.20.66.1 255.255.255.0
!
interface Ethernet 2
 ip address 172.20.95.1 255.255.255.0
!
interface serial 0
 ip address 172.20.64.2 255.255.255.0
!
router ospf 1
 network 172.20.0.0 0.0.255.255 area 2
```

The summarization of the routes between the areas is accomplished by specifying the **area range** router command. The format of the command is as follows:

area ***area-id*** **range** ***address mask***

The *area-id* parameter value denotes the OSPF area to which the summarization is being applied. The *address* parameter value is the IP address value used to define the range of addresses being summarized. The *mask* parameter value is the bit mask applied to the IP address specified by the address parameter.

In the above example configuration, the router R1 and R2 are ABR routers. Route summarization is performed only on ABR routers. The command area 0 range 172.20.96.0 255.255.224.0 on both router R1 and R2 denotes a summarized route. The applied summarization mask is a full 19-bit mask, which applies to the range of addresses for the subnets assigned to the three areas.

Suppose the subnet range in area 1 began at 8 with a mask of 255.255.252.0 applied to the IP addresses. In this case, multiple area range commands would be necessary to summarize the routes. For example, the IP networks coming from area 1 would consist of the following:

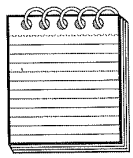
IP subnet address	Binary	Summarized mask
172.20.8.0	00001000	255.255.248.0
172.20.12.0	00001100	255.255.248.0
172.20.16.0	00010000	255.255.240.0
172.20.20.0	00010100	255.255.240.0
172.20.24.0	00011000	255.255.240.0
172.20.28.0	00011100	255.255.240.0
172.20.32.0	00100000	255.255.224.0
172.20.36.0	00100000	255.255.224.0
172.20.60.0	00100000	255.255.224.0

The resulting area range commands used in router R1 to summarize these new routes are

```
area 1 range 172.20.8.0 255.255.248.0
area 1 range 172.20.16.0 225.255.240.0
area 1 range 172.20.32.0 255.255.224.0
```

All 14 subnets are summarized into these three summary statements. It must be restated here that the IP addressing scheme deployed is paramount in using effective route summarization. If overlapping subnet masks are used, then summarization would be impossible and would cause undue SPF calculations and LSUs to traverse the links.

It is not prudent to summarize routes between OSPF areas when more than one ABR connects the two areas. Sending summarized routes between areas in such a topology reduces routing table size, but can also lead to the selection of a suboptimal path. This is especially important to note for ABR connectivity to OSPF area 0.



NOTE: *In an OSPF hierarchical routing topology, the* **router ospf** process-id *parameter is used to identify the autonomous system number of the OSPF network. Every OSPF router participating in an OSPF routing protocol must be using the same autonomous system number in order to exchange link state databases.*

In very large OSPF networks, it may be advantageous to divide the single OSPF autonomous system into smaller hierarchical networks with multiple autonomous systems defined. For example, in Figure 17-6, a large corporation has multiple areas within a large geographical area. The OSPF routing tables, due to expansion, have grown excessively large. Dividing the network into multiple autonomous systems would allow summarization between autonomous systems.

The ASBR routers R4 and R5 summarize the entire AS 1 routing table and forward the summarized route to router R2 and R6, which are part of AS 2. Likewise, R4 and R5 summarize the entire AS 2 routing table and forward the summarized route to router R1 and R3 in AS 1. Routers R4 and R5 are configured as follows:

Router R4 Configuration:

```
interface Ethernet 0
 ip address 10.1.1.1 255.255.255.0
!
interface Ethernet 1
 ip address 20.1.1.1 255.255.255.0
!
router ospf 1
 network 10.1.1.1 0.0.0.0 area 0
 summary-address 20.0.0.0 255.0.0.0
!
router ospf 2
 network 20.1.1.1 0.0.0.0 area 0
 summary-address 10.0.0.0 255.0.0.0
```

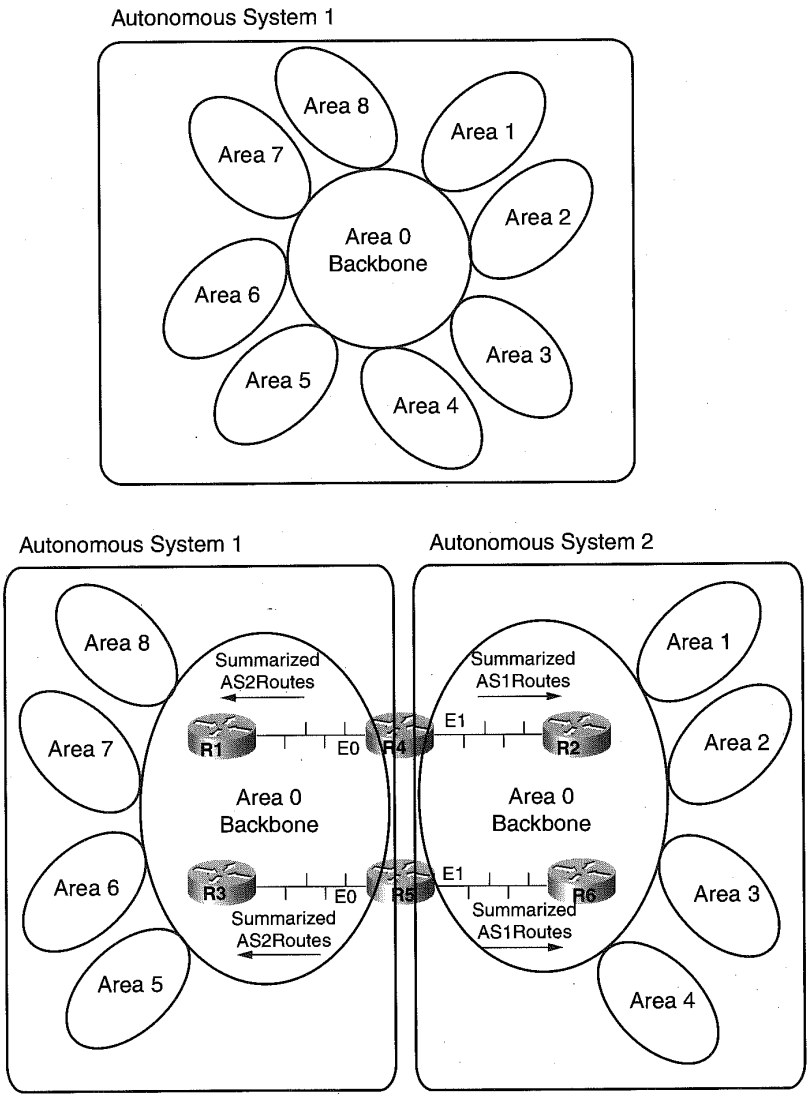


Figure 17-6
ASBR route summarization.

Router R5 Configuration:

```
interface Ethernet 0
 ip address 10.1.1.5 255.255.255.0
!
interface Ethernet 1
 ip address 20.1.1.5 255.255.255.0
!
router ospf 1
 network 10.1.1.5 0.0.0.0 area 0
 summary-address 20.0.0.0 255.0.0.0
!
router ospf 2
 network 20.1.1.5 0.0.0.0 area 0
 summary-address 10.0.0.0 255.0.0.0
```

The summary-address router command is used to summarize the OSPF routes from AS1 and AS2. The format of the command is

summary-address *address mask*

The *address* parameter value is the IP network address on which the mask will be applied. The *mask* parameter consists of the relative bits for performing the external route summarization.

Stub, Totally Stubby, and Not-So-Stubby-Area (NSSA) OSPF Areas

OSPF link-state advertisement (LSA) type-5 defining routes to external networks are not flooded into the stub area. Instead, the ABR connecting to the stub area sends a default route (0.0.0.0) for external networks into the stub area. This allows a router within the stub area to forward a packet destined for a network that is not found in the stub area router routing table. The stub area router will forward packets for networks not found in its own table to the ABR router that sent the 0.0.0.0 LSA. Figure 17-7 illustrates the topology of a stub area in OSPF.

Cisco IOS further reduces the size of a routing table within a stub area router by using the totally stubby area feature. A totally stubby area blocks OSPF LSA type-5, as does a stub area, but also blocks OSPF type 3 and 4 summary LSA for intra-area routes from being introduced into the stub area. Using this Cisco-specific feature, only intra-area stub routes and the default 0.0.0.0 routes are known in the stub area routers. An ABR forwards a summary LSA type 3 and 4 into the totally stubby area, which enables the stub area route to select the closest ABR gateway to all networks outside of the totally stubby area.

Stub and totally stubby areas are typically found in a hub and spoke topology. The stub or totally stubby area is the spoke. This is very common when using Frame Relay on the WAN. Further discussion on using Frame Relay with OSPF is found in Chapter 21, "Defining Frame Relay."

The OSPF Not-So-Stubby-Area (NSSA) option is defined in RFC 1587 and became a requirement under OSPF to enable a stub area router to use a non-OSPF routing protocol. This hybrid situation allows LSA type-7 (autonomous system external routes) to be forward to the OSPF network. Figure 17-8 illustrates the topology.

As shown in Figure 17-8, the NSSA connecting router R1 to router R2 must forward the routes from router R2 that are not OSPF routes. In this case, these routes are created from the RIP process on router R2, which is defined as an ASBR router. Router R2 is configured to perform RIP and OSPF routing. When router R2 receives an RIP update for networks 10.100.0.0/16, 10.110.0.0/16, and 20.20.0.0/8, it imports them into type-7 LSAs and forwards these LSAs to router R1. Router R1 is configured as an ABR, connecting to the OSPF backbone area 0 network. After router R1 performs SPF calculation, the type-7 LSAs are translated into type-5 LSAs and are flooded to area 0. Router R1 can also perform route summarization on these external routes received from the NSSA or it can filter these routes from the backbone area 0.

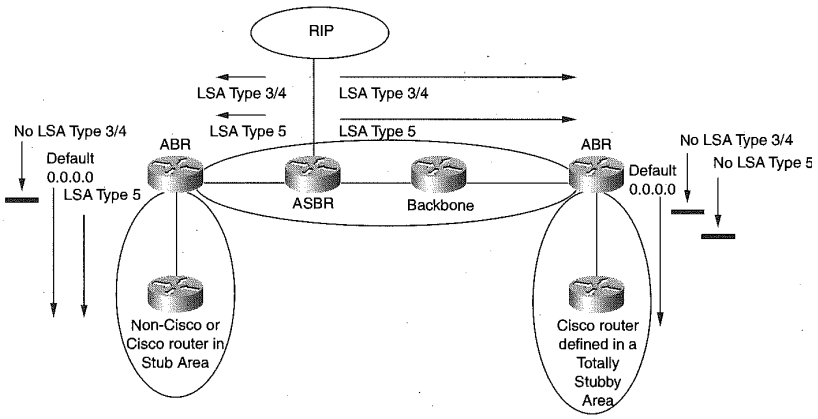


Figure 17-7 Stub and totally stubby network topology.

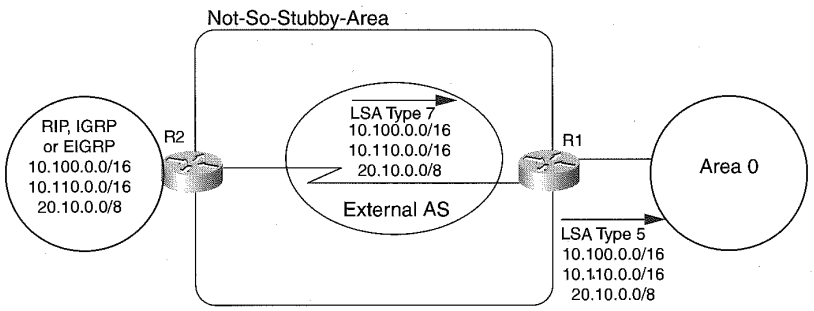


Figure 17-8
NSSA OSPF network topology.

Configuring a Stub and Totally Stubby Areas

A stub area is identified by specifying the area stub command on all routers participating in the OSPF stub area. The area stub command has the following format:

area *area-id* **stub [no-summary]**

The *area-id* parameter is the identifier associated with the stub or totally stubby area. It can be either a decimal value ranging from 0 to 4,294,967,295 or in a dotted-decimal format akin to an IP address. The optional no-summary keyword is used to define the stub area as a totally stubby area, eliminating the type 3 and 4 LSAs, along with LSA type 5, from entering the area.

In Figure 17-9, a typical topology for using a stub or totally stubby area is illustrated. The following configuration details the configuration for routers R1, R2, and R3 in forming the stub area.

Router R1 Configuration:

```
interface Ethernet 0
 ip address 10.10.10.1 255.255.255.0
!
interface serial 0
 ip address 10.20.20.1 255.255.255.0
!
interface serial 1
 ip address 10.30.30.1 255.255.255.0
!
router ospf 1
 network 10.10.10.1 0.0.0.0 area 0
 network 10.20.20.1 0.0.0.0 area 1
 network 10.30.30.1 0.0.0.0 area 2
 area 1 stub
 area 2 stub
```

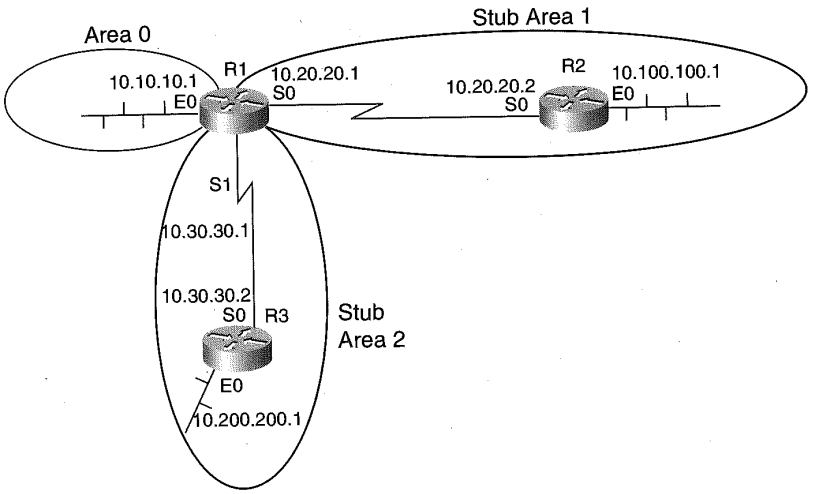


Figure 17-9
Stub network topology example.

Router R2 Configuration:

```
interface Ethernet 0
 ip address 10.100.100.1 255.255.255.0
!
interface serial 0
 ip address 10.20.20.2 255.255.255.0
!
router ospf 1
 network 10.0.0.0 0.0.0.0 area 1
 area 1 stub
```

Router R3 Configuration:

```
interface Ethernet 0
 ip address 10.200.200.1 255.255.255.0
!
interface serial 0
 ip address 10.30.30.3 255.255.255.0
!
router ospf 1
 network 10.0.0.0 0.0.0.0 area 2
 area 2 stub
```

The stub area 1 in the example configuration is changed to a totally stubby area by changing the router R1 configuration to include the no-summary keyword on the area stub command. This causes router R1 to forward a default route of 0.0.0.0 for all external routes out of the stub area. The resulting IOS configuration for router R1 is as follows:

Router R1 configuration for totally stubby areas:

```
interface Ethernet 0
 ip address 10.10.10.1 255.255.255.0
!
interface serial 0
 ip address 10.20.20.1 255.255.255.0
!
interface serial 1
 ip address 10.30.30.1 255.255.255.0
!
router ospf 1
 network 10.10.10.1 0.0.0.0 area 0
 network 10.20.20.1 0.0.0.0 area 1
 network 10.30.30.1 0.0.0.0 area 2
 area 1 stub no-summary
 area 2 stub
```

Defining the area stub with the no-summary command enables the totally stubby Cisco-specific feature. This command is needed only on the ABR router that connects the OSPF standard area with the stub area.

Configuring a NSSA

NSSA configurations are most useful in situations where the OSPF hierarchical routing network must connect to a non-OSPF network. This is typically used in Internet Service Provider (ISP) connections and in merged corporate networks due to corporate acquisitions or during migration from non-OSPF to OSPF hierarchical routing.

Shown in Figure 17-10 is an example NSSA network configuration with router R1 and R2 forming the NSSA OSPF area. Router R1 performs the non-OSPF-to-OSPF translation of non-OSPF routing updates to OSPF external link-state advertisement type-7. Router R2 translates these received type-7 LSAs to type-5 LSAs and can further summarize or filter routes to the NSSA area. Prior to using NSSA, this type of configuration required the non-OSPF and OSPF routing protocols to be executing on the standard OSPF area, which in this case is router R2.

The following is the configuration for router R1 and R2 referencing Figure 17-10:

Router R1 Configuration for NSSA:

```
interface serial 0
 ip address 192.168.1.1 255.255.255.252
!
```

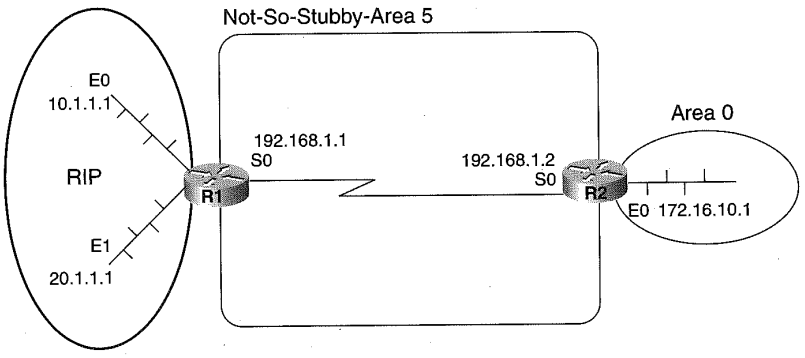


Figure 17-10
NSSA topology connecting a RIP network to an OSPF hierarchical network.

```
interface Ethernet 0
 ip address 10.1.1.1 255.255.255.0
!
interface Ethernet 1
 ip address 20.1.1.1 255.255.0.0
!
router rip
 network 10.0.0.0
 network 20.0.0.0
router ospf 1
 redistribute rip subnets
 network 192.168.1.1 0.0.0.0 area 5
 area 5 nssa
```

Router R2 Configuration for NSSA:

```
interface serial 0
 ip address 192.168.1.2 255.255.255.252
!
interface Ethernet 0
 ip address 172.16.10.1 255.255.255.0
!
router ospf 1
 summary-address 10.0.0.0 255.0.0.0 tag 16
 summary-address 20.0.0.0 255.0.0.0 tag 8
 network 192.168.1.2 0.0.0.0 area 5
 network 172.16.10.1 0.0.0.0 area 0
 area 5 nssa
```

The router R1 configuration indicates that the Ethernet interfaces 0 and 1 are using RIP as the routing protocol to the external network. This is determined by the network commands following the router rip command. The serial interface 0 on router R1 uses OSPF as the routing protocol and OSPF knows to use serial 0 interface because of the network area command found under the router ospf command. The redistribute rip subnets command under the router ospf command in router R1 directs the OSPF process on router R1 to translate the RIP updates into OSPF external link-state advertisement type-7 LSAs. The translation is to type-7 LSAs based on the area nssa command. In the router R1 configuration example, the NSSA area-id is defined as the decimal number 5. The format of the area nssa command is

area *area-id* nssa [no-redistribution]
[default-information-originate]

The *area-id* parameter can be either a decimal number ranging from 1 to 4,294,967,295 or a dotted-decimal format in the form of an IP address. The optional **no-redistribution** keyword is used only on the NSSA ABR router, in our example router R2, and directs the OSPF process to import routes only into the standard OSPF areas and not the NSSA areas. The **default-information-originate** optional keyword may be used to tell the NSSA ABR to send a default type-7 LSA into the NSSA area.

An optional command under the router ospf command in the router R2 configuration is included to discuss route aggregation. The command is summary-address and the format is as follows:

summary-address *address mask* [*prefix mask*] [not-advertise]
[tag *tag*]

The *address* parameter value is the IP address used to identify the range of addresses. The first *mask* parameter value is the bit mask applied to the address value to identify the summarized route. The optional parameter *prefix* is the IP address of a destination network that can be summarized. The following optional *mask* parameter value is the mask applied to the *prefix* value. The optional keyword **not-advertised** is used to suppress the routes that match the prefix/mask pairing so the routes are not advertised at all. The optional **tag** keyword and parameter *tag* are used to determine a match value in redistributing routes using route maps. The value used for *tag* ranges from 0 to 4,294,967,295.

Using OSPF Virtual Links

OSPF hierarchical routing is based on the premise that all areas connect to the backbone area 0 network. There may be times in the course of network implementation when operational cost requires the connectivity of a new OSPF area to the backbone area 0 network through another standard area. OSPF virtual links perform this task, connecting a disconnected area to the OSPF backbone area 0 network.

OSPF virtual links provide a logical path from the disconnected area to the backbone area. Virtual links have multiple purposes. The first is to connect a remote area with no physical connectivity to the backbone area. The second is to add a connection to a discontinuous backbone area. A third use is to provide redundancy when a router failure could cause a divided backbone area. The logical path connecting the disconnected areas is that the virtual link must be defined on the two routers that can share the common area and one of these routers must be connected to the backbone. The common area used for making the logical connection cannot be a stub area. The common area used for establishing the virtual link to the backbone is termed the transit area.

Configuring Virtual Links

In Figure 17-11, a virtual link topology connecting a remote location to the OSPF backbone area over serially connected routers is illustrated. Router R1 connects to the backbone area 0 network as well as to area 1 over a serial line to router R2. Router R2 connects over a serial line to router R3, which has been defined with area 2 attached to it. This type of topology is very plausible when the cost of establishing a physical link from router R3 to the backbone ABR is excessive over the connection to router R2.

In this figure, router R3 connects area 1 and area 2. It is then an ABR router and meets the virtual link requirement. Secondly, router R1 is connected to the backbone area and it is also an ABR router. Router R2, being in area 1, participates in the virtual link as a pass-through node, delivering the LSUs between router R1 and router R3. Area 1 becomes the transit area for the virtual link.

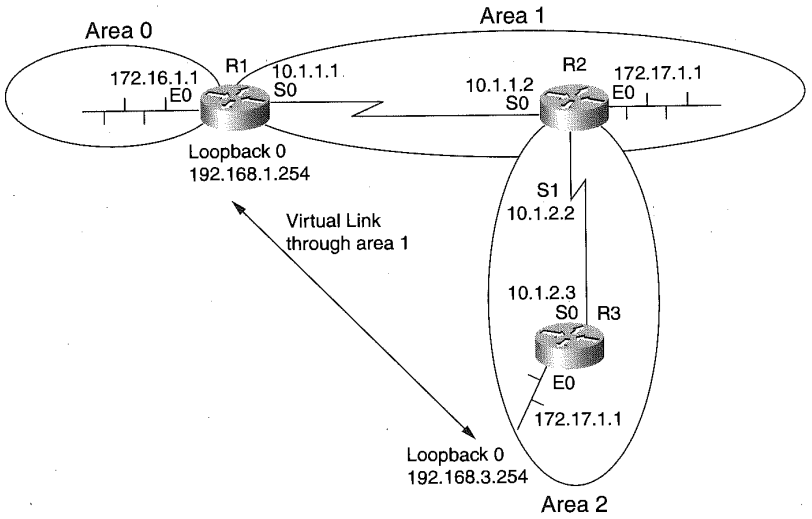


Figure 17-11 Connecting a physically disconnected area to the OSPF backbone area using a virtual link.

The following example configurations detail the virtual link connection for Figure 17-11:

Router R1 Configuration for Virtual link to router R3:

```
interface serial 0
 ip address 10.1.1.1 255.255.255.0
!
interface loopback 0
 ip address 192.168.1.254 255.255.255.252
!
interface Ethernet 0
 ip address 172.16.1.1 255.255.255.0
!
router ospf 1
 network 10.1.1.1 0.0.0.0 area 1
 network 172.16.0.0 0.0.255.255 area 0
 area 1 virtual-link 192.168.3.254
```

Router R2 Configuration for Virtual link to router R3:

```
interface serial 0
 ip address 10.1.1.2 255.255.255.0
!
interface serial 1
 ip address 10.1.2.2 255.255.255.0
!
```

```
interface loopback 0
 ip address 192.168.2.254 255.255.255.252
!
interface Ethernet 0
 ip address 172.17.1.1 255.255.255.0
!
router ospf 1
 network 10.1.0.0 0.0.255.255 area 2
 network 172.17.0.0 0.0.255.255 area 1
```

Router R3 Configuration for Virtual link to router R1:

```
interface serial 0
 ip address 10.1.2.3 255.255.255.0
!
interface loopback 0
 ip address 192.168.3.254 255.255.255.252
!
interface Ethernet 0
 ip address 172.18.1.1 255.255.255.0
!
router ospf 1
 network 10.1.2.0 0.0.0.255 area 1
 network 172.18.0.0 0.0.255.255 area 2
 area 1 virtual-link 192.168.1.254
```

In the example configurations, the router ID of router R3 is defined in the configuration for router R1 as the end point of the virtual link, connecting through transit area 1. In router R3, the router ID of router R1 is defined as the end point of the virtual link through transit area 1. The router IDs for the example use the IP address of the loopback interface of each router. The configuration in router R1 does not have any special definitions to allow the virtual connection. Although it may appear as a stub area, it is not defined as such and can therefore be used for establishing the virtual link.

The format of the area virtual-link command is as follows:

area *area-id* **virtual-link** *router-id* [**hello-interval** *seconds*] [**retransmit-interval** *seconds*] [**transmit-delay** *seconds*] [**dead-interval** *seconds*] [[**authentication-key** *key*] | [**message-digest-key** *keyid* **md5** *key*]]

The *area-id* parameter is the decimal or IP address dotted-decimal format identifier for the area being used as the transit area for the virtual link. The *router-id* parameter is the router-ID of the end point to which the router is defining the virtual link. The optional keyword **hello-interval** parameter *seconds* defaults to a value of 10 seconds and can range from 1 to 8,192, measured in seconds. The value specified here must be defined on

all participating routes on the common network. The value specifies the number of seconds the router will wait between sending hello-packets on the virtual link.

The optional **retransmit-interval** keyword parameter *seconds* defaults to five seconds with a range of 1 to 8,192. This value specifies the time in seconds for retransmitting LSAs to adjacency routers. The value must be greater than the expected round-trip delay between the routes on the virtual link. The optional **transmit-delay** keyword parameter *seconds* defaults to one second and can range from 1 to 8,192. The value specified increments the age count on the LSU packet before transmitting on the virtual link.

The optional keyword **dead-interval** parameter *seconds* defaults to four times the hello interval and also ranges from 1 to 8,192. The value specified here must be the same on all routers attached to the same common network. This interval is the time in seconds that must expire without the router receiving hello packets from the far end of the virtual link in order to declare the far-end router down. The optional authentication-key parameter key value is the password used on hello packets to the far end of the virtual link to authenticate the far-end router. The value is a continuous string of up to eight characters long. The far end must use the same password so that the routers establish a full state between them.

The configuration global command service password-encryption affects the key value, making the value from the **sh config**, **sh run**, **sh start**, and **wr terminal** display output encrypted for the key value. The optional **message-digest-key md5** parameter *keyid* is a decimal value ranging from 1 to 255, and the *key* parameter is an alphanumeric string up to 16 characters long. These values are used by the far-end router on the virtual point connection when MD5 authentication is in use. Again, all routes on the common network must use the same *keyid* and *key* values for OSPF to operate properly.

Non-Broadcast and Broadcast Network Configurations

OSPF, as of Cisco IOS Release 12.0, can configure NBMA networks as broadcast networks and broadcast networks as NBMA networks. This capability allows you to override the default media type for an interface. This

capability is especially useful in point-multipoint networks like Frame Relay. However, the point-multipoint feature available with Cisco IOS can also be applied to an Ethernet, Token Ring, or FDDI network as well. If a network is to be viewed as an NBMA-type network, a neighbor must be specified for each router connection that requires OSPF routing information. Defining a link as an OSPF point-to-multipoint connection enables broadcasting on NBMA networks. The point-multipoint feature has the following advantages:

- Explicit configuration of neighbor routers is not required.
- Only one IP subnet is used for the entire point-to-multipoint connection.
- Routing between the end points is accomplished through the common point connecting them.
- No DR election is required.
- It does not require a fully meshed network topology.

In some broadcast topologies there will be routers that do not support multicasting. In these networks, even if they are broadcast-based (such as Ethernet), a Cisco router can view the network as a non –broadcast network and therefore require neighbor routers be defined.

Configuring OSPF Point-to-Multipoint

Three OSPF commands are specific to the point-multipoint feature. The first of these is the ip ospf network command. The format of the command is as follows:

ip ospf network {broadcast | non-broadcast | {point-to-multipoint [non-broadcast]}}

This command is entered under the interface on which the default media type is being overridden for OSPF. The broadcast keyword uses an NBMA or point-to-point connection as if it were actually a broadcast network. The non-broadcast keyword modifies the use of a broadcast network to that of a NBMA network, requiring the router administrator to define the router neighbors connected to the network. The point-to-multipoint keyword enables an NBMA network connection to be used by OSPF as if it were a

broadcast network. The optional non-broadcast keyword of the point-multipoint keyword forces OSPF to use the point-to-multipoint connection it is defined on as a non-broadcast connection, requiring the definitions of the routing neighbor at the far end of the connection.

The neighboring routers on non-broadcast configurations are defined using the neighbor command under the router ospf command. The format of the neighbor command is as follows:

neighbor *ip-address* [priority *number*] [poll-interval *seconds*] [cost *number*]

The ip-address parameter of the neighbor command is the interface IP address assigned to the router at the other end of the connection. The optional priority keyword and parameter number are used in assigning the eligibility of this neighbor as a DR or BDR router. The default is 0 and can range from 1 to 255. This keyword does not apply when the ip ospf network point-to-multipoint command is specified for the neighbors interface on this router. The optional poll-interval keyword and its parameter seconds indicates the number of seconds between hello packets being sent to the neighbor after the dead-interval has expired and the neighboring router is declared down. The optional cost keyword and parameter number allows the router administrator to apply an OSPF cost to the link connecting to the neighbor. This keyword is ignored when used for an interface that is defined as an NBMA connection.

Figure 17-12 illustrates the use of a point-to-multipoint connection over a Frame Relay network. Router R1 in Figure 17-12 uses a single Frame Relay serial link connection to router R2 and router R3. All three routers define the Frame Relay connection as point-to-multipoint and specify the broadcast keyword on the Frame Relay map ip command defined for the interface. The Frame Relay-specific commands are discussed in Chapter 21, "Defining Frame Relay."

The following is the example router configuration for Figure 17-12:

Router 1 Configuration:

```
interface serial 0
 ip address 172.16.10.1 255.255.255.0
 ip ospf network point-to-multipoint
 encapsulation frame-relay
 frame-relay map ip 172.16.10.2 30 broadcast
 frame-relay map ip 172.16.10.3 31 broadcast
!
router ospf 1
 network 172.16.10.1 0.0.0.255 area 0
```

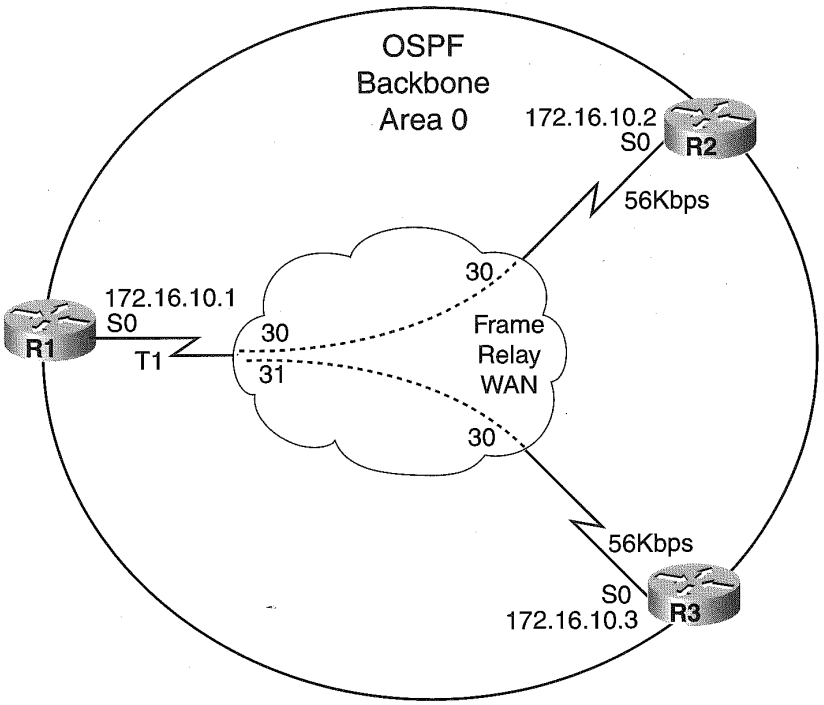


Figure 17-12
An OSPF point-to-multipoint Frame Relay connection as a broadcast network.

Router 2 Configuration:

```
interface serial 0
 ip address 172.16.10.2 255.255.255.0
 ip ospf network point-to-multipoint
 encapsulation frame-relay
 clockrate 56
 frame-relay map ip 172.16.10.1 30 broadcast
!
router ospf 1
 network 172.16.10.2 0.0.0.0 area 0
```

Router 3 Configuration:

```
interface serial 0
 ip address 172.16.10.3 255.255.255.0
 ip ospf network point-to-multipoint
```

```
 encapsulation frame-relay
 clock rate 56
 frame-relay map ip 172.16.10.1 30 broadcast
!
router ospf 1
 network 172.16.10.3 0.0.0.0 area 0
```

The above configuration listed for Figure 17-12 illustrates the use of the point-to-multipoint configuration for OSPF over Frame Relay as an NBMA network. In the next example, the same network topology is used, but the network is defined as a non-broadcast point-to-multipoint configuration. In this instance, the neighbor command is required to identify the router peer at the far end of the Frame Relay connection.

Router 1 configuration using point-to-multipoint non-broadcast:

```
interface serial 0
ip address 172.16.10.1 255.255.255.0
ip ospf network point-to-multipoint non-broadcast
encapsulation frame-relay
frame-relay local-dlci 200
frame-relay map ip 172.16.10.2 30
frame-relay map ip 172.16.10.3 31
!
router ospf 1
network 172.16.10.1 0.0.0.0 area 0
neighbor 172.16.10.2 cost 15
neighbor 172.16.10.3 cost 20
```

Router 2 configuration using point-to-multipoint non-broadcast:

```
interface serial 0
 ip address 172.16.10.2 255.255.255.0
 encapsulation frame-relay
 ip ospf network point-to-multipoint non-broadcast
 frame-relay local-dlci 30
 frame-relay map ip 172.16.10.1 30
!
 router ospf 1
 network 172.16.10.2 0.0.0.0 area 0
```

Router 3 configuration using point-to-multipoint non-broadcast:

```
interface serial 0
 ip address 172.16.10.3 255.255.255.0
 encapsulation frame-relay
 ip ospf network point-to-multipoint non-broadcast
 frame-relay local-dlci 31
```

```
 frame-relay map ip 172.16.10.1 31
!
 router ospf 1
 network 172.16.10.3 0.0.0.0 area 0
```

In the non-broadcast configuration, the neighbor command is used only on the hub router, which in this scenario is router R1. A cost is applied to each individual neighbor on the physical serial interface of router R1, enabling OSPF to calculate a different cost for routes to each neighboring router. Prior to IOS Release 12.0, the ip ospf cost interface command had to be used to apply a cost to the connection, but now the ip ospf cost value is applied to all the virtual circuits connecting to the physical Frame Relay connection. Using the neighbor command allows unique costs to be applied to each DLCI of a Frame Relay connection. Frame Relay DLCI connections, defined using subinterfaces, can safely use the ip ospf cost command as a means of applying a cost to the DLCI connection without defining neighbors. The format of the ip ospf cost command is

ip ospf cost *cost*

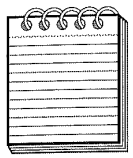
The cost parameter is an unsigned decimal value ranging from 1 to 65,535. Applying this command to an interface overrides the default OSPF cost for the media type. The default cost of an OSPF link is determined by using the following formula:

10^8 / bandwidth value

The bandwidth-value is the default bandwidth of the connected interface. The default OSPF path cost for the following links are listed here:

- 56-kbps serial link: 1785
- 64-kbps serial link: 1562
- T1 (1.544-Mbps serial link): 65
- E1 (2.048-Mbps serial link): 48
- 4-Mbps Token Ring: 25
- Ethernet: 10
- 16-Mbps Token Ring: 6
- FDDI and ATM (OC-3 and higher): 1

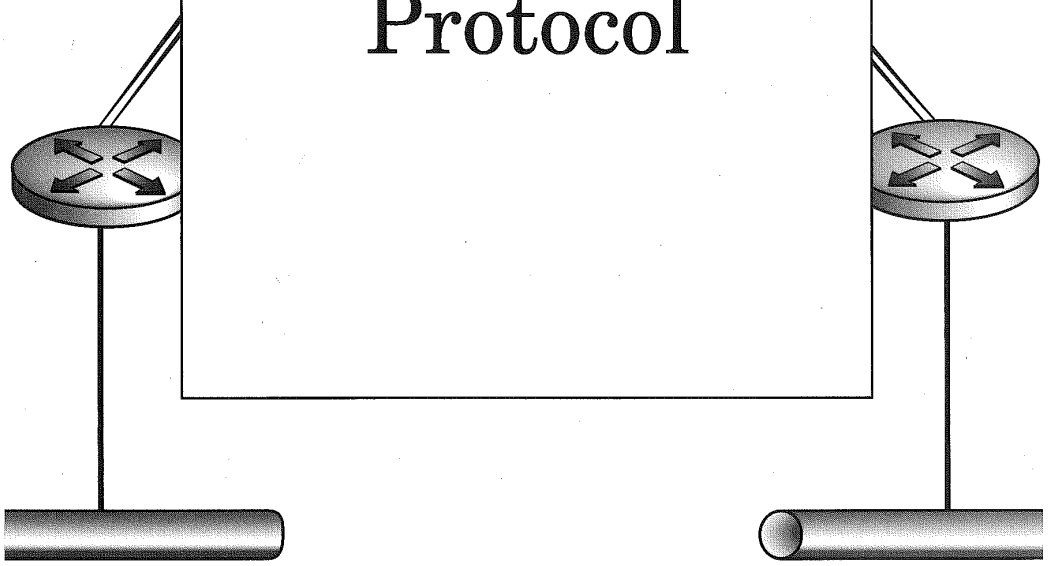
The ip ospf cost interface command can be used for any OSPF network connection and is not specific to NBMA or broadcast topologies.



NOTE: *The interface command bandwidth changes the OSPF calculated cost of a link when the ip ospf cost command is not specified for the interface.*

CHAPTER 18

Configuring BGP Routing Protocol



The Border Gateway Protocol (BGP) is an IETF standard defined in RFCs 1163, 1267, 1654, 1655, and 1771, with the latest standard being termed BGP4 for version 4. Cisco IOS supports versions 2, 3, and 4. BGP provides a loop-free routing environment for autonomous systems (AS). Under BGP, an autonomous system is a unique IP network of routers under control of a specific entity. Each AS can potentially use multiple interior gateway protocols (IGPs) for the exchange of routing tables within the AS, which uses an exterior gateway protocol (EGP) to send packets outside of the AS.

BGP is not always a requirement for connecting between autonomous systems. In most cases, the driving force for implementing BGP is the routing policies enacted between the connecting AS networks. If the routing policy of the one AS is the same as the connecting AS network, BGP is not even necessary. For example, a static route and default networks can be defined within the corporate network to provide connectivity to the connecting AS, which uses BGP to connect to other networks. This is often used when connecting a corporate network to an ISP provider that uses BGP to connect to the Internet. Many corporations use two ISPs or connections to the Internet to facilitate redundancy, load sharing, and cost-reduction through connection options that enable lower tariffs for off-hours use. In the redundancy case, using backup link-static and default network routes may be suitable, instead of implementing BGP. Configuring the ISP connections for load sharing will most likely require BGP.

A typical BGP network topology is illustrated in Figure 18-1. The corporate network has two ISP connections that enable the company to have redundancy, availability, and load sharing. In Figure 18-1, ISP1 connects to the corporate network in San Diego, while ISP2 connects to the corporate network in Atlanta. The policy of the corporate network in agreement with the ISP vendors is to have the majority of Internet traffic traversing the connection in Atlanta during the off-peak hours for the West Coast and the San Diego connection used during off-peak hours for the East Coast. This provides the lowest tariff rates for the corporation while maintaining redundancy, availability, and load sharing.

Routing policy plays an important role in determining the use of BGP. In Figure 18-2, a new AS (AS 60) is connected to the existing network. Router R1 is attached to AS 10 and uses BGP to communicate to AS 20 and AS 30. Router R6 initially connects to router R3 using a static route definition in the routing tables. The routes to networks in AS 60 will be viewed from router R1 as having the same policy as the networks existing in AS 20. This is because the AS 60 is not differentiated using BGP. AS 60 is also seen as an extension of AS 20. Therefore, router R1 cannot use a unique routing policy to networks from AS 10. By enabling BGP between router R6 and

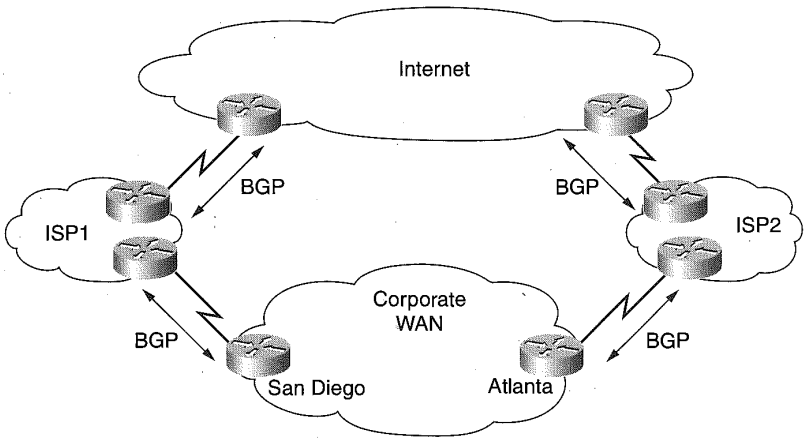


Figure 18-1 High availability using BGP on two ISP networks connecting to the Internet.

Router R3, AS 60 is announced into the AS 20 network as a unique AS connection. This is propagated to AS 10 and AS 30, thereby enabling a clarified routing policy to networks residing in AS 60.

Traffic on BGP is carried using a reliable transport. The Transmission Control Protocol (TCP) carries the BGP traffic over an established TCP connection that is formed over TCP port 179. BGP routers that form a connection are referred to as peer routers or neighbors. The connection enables the exchange of connection parameters and full BGP routing tables. The entire BGP routing table is exchanged at the onset of a TCP connection. After the full exchange, only incremental updates are sent between neighbors. The BGP table has a version number assigned to it that is used by all of the BGP peers. The version number increments after a BGP update occurs. The neighbors keep their connection alive by sending keepalive packets between the peers. If errors occur in the transmission of packets or a special condition occurs on a peer notification, the packets are still sent.

Exterior and Interior BGP Sessions

More than one BGP connection can occur in a single AS. In such a scenario, the AS with multiple BGP connections can be used as a transit AS. Two types of BGP sessions can exist when implementing BGP: Exterior BGP (EBGP) and Interior BGP (IBGP) sessions. Figure 18-3 highlights the EBGP and IBGP topology.

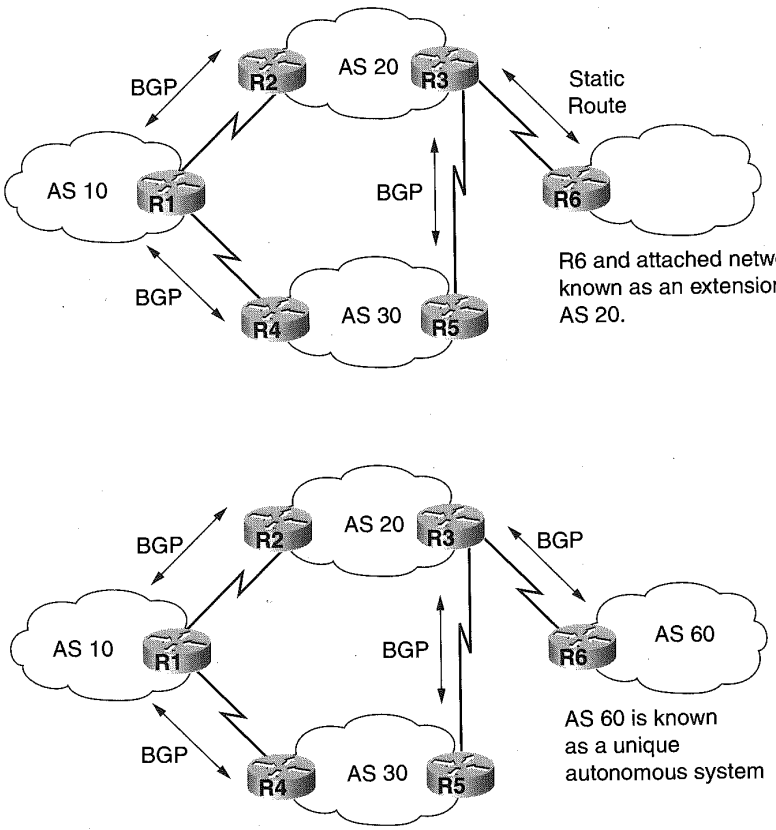


Figure 18-2
Using BGP for creating a routing policy.

The EBGP sessions are established between two unique AS networks. In Figure 18-3, AS 10 and AS 30 establish an EBGP session with AS 20. Adjacent routes sharing the same media and subnet connect EBGP sessions. The network in AS 20 also implements BGP, yet since the connection is between multiple routers within its network, it uses IBGP for communicating BGP routing tables. An IBGP session occurs between two routers of the same AS. This allows the IBGP routers to coordinate and synchronize routing policies for the AS. The IBGP peer connections do not have to be adjacent to each other. The peer connections for IBGP can be several hops away but must remain within the AS.

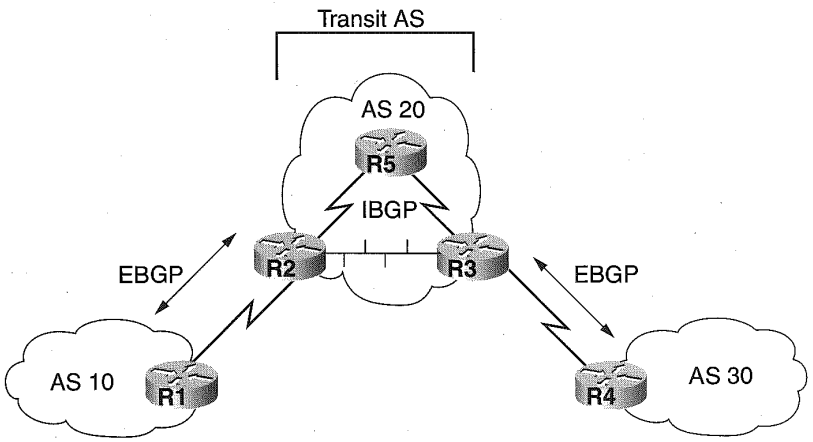


Figure 18-3
EBGP and IBGP sessions and transit AS topology.

Path Selection Under BGP

BGP has two mechanisms for selecting a path. The first mechanism is used for choosing a single path. BGP defaults to a single path in the IP routing table. In a single path selection scenario, the following process is implemented:

- Inaccessible next-hop routes are not considered.
- The routers with the highest BGP administrative weights are considered first.
- The routers with a high local preference in a tie for the highest BGP administrative weight are preferred.
- When the local preference is equal, the route originated by the local router is preferred.
- If the local router did not originate the route, the shortest autonomous system path is selected.
- Paths having the same autonomous system path length are then arranged by the lowest origin code first. IGP routes have a lower origin code than EGP routes, which are lower than incomplete routes.
- If all the paths are from the same autonomous system and the origin code list results in equal origin codes, then the tiebreaker is the path with the lowest Multi Exit Discriminator (MED) metric value. If the path has a missing metric value, it is assumed to be equal to 0.

- In the case where the MED values are equal, external paths are selected over internal paths.
- If only internal paths are remaining in the selection list and IGP synchronization is disabled, the BGP selects the closest neighbor as the best path.
- If, at this point, there is a tie as to which neighbor is closest, BGP selects the neighbor with the lowest BGP router ID value.

In situations when multiple paths are designed into the topology for EBGP connections to a neighboring AS, the path that the lowest BGP router-id is presented on is the preferred single path. Enabling multipath BGP support for EBGP paths determined from the same neighboring autonomous system allows for up to six paths installed in the routing table. BGP will then either load balance over the multiple paths on a per-packet or per-destination basis.

Defining the BGP Process

Four basic commands are required to initialize the BGP process on Cisco routers. These commands define the BGP process, what networks originate in this router, what BGP neighbors establish a connection to this router, and a means of reinitializing the BGP connections and routing table after configuration changes.

Enabling the BGP Process

The Cisco IOS begins a BGP process when the router administrator enters the router bgp global configuration command. The format of the command is

router bgp *autonomous-system*

The AS parameter value ranges from 1 to 65,535. The value specified must be the same for all routers participating in the IBGP network connections in order for the routers to exchange BGP routing updates within the same AS. The AS value code is also used as a tag for the information exchanged between the BGP peers. For example, in the two configurations below, the BGP specific parameters are shown:

Router R1 BGP Configuration:

```
router bgp 10
```

Router R2 BGP Configuration:

```
router bgp 20
```

These configurations identify two unique AS networks. Router R1 is defined as a router belonging to AS 10 and Router R2 is defined as a router belonging to AS 20.

Establishing BGP Peer Connections

The BGP process is made aware of which BGP routers will be peers with the local router through the definition of neighbors. Once defining the neighbors, the BGP process attempts to establish a BGP peer connection with the defined neighbor. BGP routing table updates will only be sent after a successful TCP peer connection has been established.

Upon establishing the peer connection, the peers send "open messages" to each other that identify the AS number, BGP version number, BGP route ID, and keepalive hold-time values, among others. The neighbor connection must be in an established state for the exchange of routing updates. If the neighbor connection state is not shown as established, the BGP update exchange has not occurred and BGP routing over these routers will not take place.

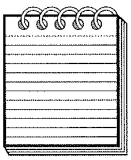
Identifying the neighbors is accomplished using the router bgp command neighbor remote-as. The format of the command is

neighbor {*ip-address* | *peer-group-name*} **remote-as** *number*

The *ip-address* parameter is the value of the IP address used to locate a BGP neighbor. The value must be in the dotted-decimal IP address format. The parameter *peer-group-name* value is a label given to a group of BGP peers that are recognized collectively as a group. The *ip-address* and *peer-group-name* parameter values are mutually exclusive. The *number* parameter following the remote-as keyword is the AS number to which the neighbor being defined belongs. BGP peer groups are discussed later in this chapter.

If the remote-as *number* value assigned to the neighbor is the same as the AS number assigned on the router bgp command, then the BGP process will be performing IBGP to the neighbor. If the remote-as number value is

different form the local router's AS number assigned on the router bgp command, then the BGP process will perform EBGP to the neighbor.



NOTE: *To verify IP connectivity between two neighbors prior to definition, use the Cisco IOS extended PING command, placing the IP address of the neighbor as the source address in the command.*

BGP implementation with Cisco IOS defaults to using BGP Version 4 protocol and then negotiates sequentially downward to determine the appropriate version on the neighbor. However, Cisco IOS enables the router administrator to define the BGP version being used by the neighbor using the neighbor version command. The format of the command is

neighbor {*ip-address* | *peer-group-name*} **version** *value*

The *ip-address* parameter is the IP address value of a defined BGP neighbor. The value must be in the dotted-decimal IP address format. The parameter *peer-group-name* value is a label given to a group of BGP peers that are recognized collectively as a group. The *ip-address* and *peer-group-name* parameter values are mutually exclusive. The *value* parameter following the version keyword defines the BGP version number to use for the TCP connection with the identified neighbor. The *value* parameter can be either 2, 3, or 4, with 4 being the default.

In Figure 18-4, the following configuration is applied:

Router R1:

```
router bgp 10
neighbor 172.21.1.1 remote-as 20
```

Router R2:

```
router bgp 20
neighbor 172.21.1.2 remote-as 10
neighbor 192.168.1.2 remote-as 20
```

Router R3:

```
router bgp 20
neighbor 192.168.3.1 remote-as 20
neighbor 192.168.4.1 remote-as 30
```

Router R4:

```
router bgp 30
neighbor 192.168.4.3 remote-as 20
```

All the routers in this configuration are assumed to be using BGP Version 4, since the neighbor version command was not explicitly entered. Since the AS number defined on the router bgp command for router R1 is different than the AS number defined on the neighbor command in the router R1 configuration, EBGP will be used from router R1 to the neighbor at 172.21.1.1. Router R2 uses IBGP to communicate with router R3. Recall that BGP uses TCP connections to deliver BGP routing updates. As such, the topology in Figure 18-4 illustrates that the BGP peers do not necessarily need to be directly connected for IBGP. Router R5 in the figure is used as the routing node for establishing the BGP TCP connection between router R2 and router R3. The BGP TCP connection between router R3 and router R4 completes the illustration and uses EBGP since this connection is established using different AS numbers.

When making BGP configuration changes, it is good practice to always reinitialize the BGP routing table and BGP TCP connections with the defined neighbors to ensure that the change has taken place. This is done by entering the following command:

clear ip bgp {* | *address* | *peer-group name*} **[soft [in | out]]**

The * value used in the command causes the current BGP peer connections to reset and a new routing table is built. The address parameter value is used when a specific neighbor connection needs to be reset and the routes

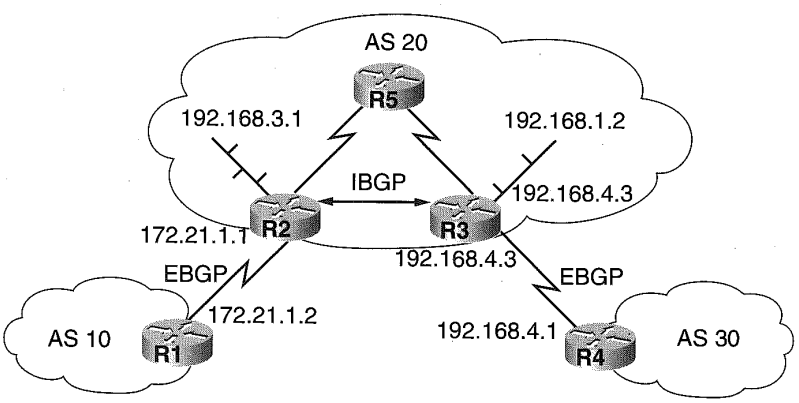


Figure 18-4 Simple BGP network topology.

received from that neighbor are updated. Using the peer-group-name value will reset all the connections from the local router to all members of the peer group. The *peer-group-name* must be the name assigned to a previously defined peer group in the local router.

The optional **soft** keyword indicates that the current BGP sessions are not reset, but the router will send routing updates. When using the **soft** keyword, the entire table is only updated if the **neighbor soft-reconfiguration** command is entered. Performing this function is memory-intensive and may cause long packet delays for connections traversing through the local router. The **in** variable of the optional **soft** keyword triggers only inbound reconfiguration, while the **out** variable of the soft keyword generates only outbound soft reconfiguration. Not specifying either variable triggers both inbound and outbound soft reconfiguration.

The clear ip bgp command should be used when the BGP configuration is changed in accordance with the following:

- BGP access list adds, changes, or deletions
- BGP weights have been altered
- BGP distribution-list adds, changes, or deletions
- BGP-specific timers
- Altering BGP administrative distance
- Using the route map command for BGP

By issuing the following command

```
clear ip bgp *
```

all the BGP peer connections are reset and the BGP peers will reestablish their connections and then exchange routing updates.

Forcing a Loopback Interface as the BGP Neighbor

BGP uses the best local address to a neighbor as the source for sending updates. This is usually the IP address of the interface with the best path to the neighbor. In cases where multiple links are available to connect to a neighbor, usually found in IBGP topologies, greater availability is possible by using a loopback interface as the BGP neighbor of the local router. To identify the use of a loopback interface as the BGP peer on the local router, the following router bgp command is used:

neighbor {*ip-address* | *peer-group-name*} **update-source** *interface*

The *ip-address* parameter is the value of the IP address of a defined BGP neighbor. The value must be in the dotted-decimal IP address format. The parameter *peer-group-name* value is a label given to a group of BGP peers that is recognized collectively as a group. The *ip-address* and *peer-group-name* parameter values are mutually exclusive.

The *interface* parameter of the **update-source** keyword identifies the loopback interface used as the neighbor. The IP address defined under the specified loopback interface becomes the IP address used in the neighbor remote-as command in the BGP peer router configuration. For example, in Figure 18-5, router R1 uses a loopback interface IP address as its neighbor address. Router R2 and Router R3 both use the loopback interface IP address defined on router R1 as the neighbor ip-address on their respective neighbor remote-as commands.

The following configuration examples illustrate the requirement of router R2 and router R3 defining the IP address of the loopback interface for router R1 as the neighbor IP address with which TCP connections are established.

Router R1 Configuration for illustrating the use of loopback interface on a single BGP router:

```
ip subnet-zero
!
interface serial 0
 ip address 192.168.1.1 255.255.255.252
!
interface serial 1
 ip address 192.168.1.5 255.255.255.252
!
interface loopback 0
 ip address 192.168.1.254 255.255.255.252
!
router eigrp 10
 network 192.168.1.0
!
router bgp 10
network 192.168.1.0 mask 255.255.255.252
neighbor 192.168.1.2 remote-as 10
neighbor 192.168.1.6 remote-as 10
neighbor 192.168.1.2 update-source loopback 0
neighbor 192.168.1.6 update-source loopback 0
```

Router R2 configuration for Figure 18-5:

```
ip subnet-zero
!
interface serial 0
```

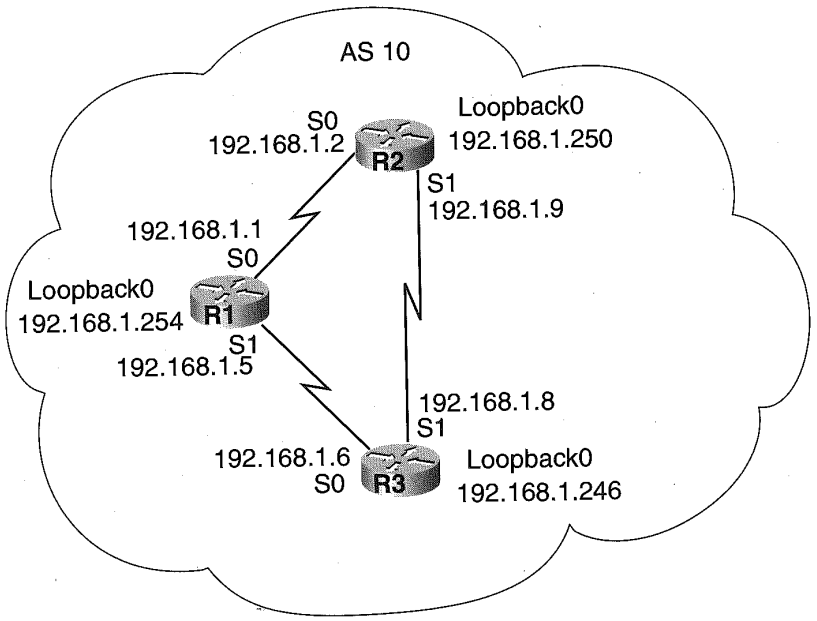


Figure 18-5
BGP topology for using a loopback interface as the BGP neighbor.

```
 ip address 192.168.1.2 255.255.255.252
!
interface serial 1
 ip address 192.168.1.9 255.255.255.252
!
router eigrp 10
 network 192.168.1.0
!
router bgp 10
network 192.168.1.0 mask 255.255.255.252
neighbor 192.168.1.254 remote-as 10
neighbor 192.168.1.8 remote-as 10
```

Router R3 configuration for Figure 18-5:

```
ip subnet-zero
!
interface serial 0
 ip address 192.168.1.6 mask 255.255.255.252
!
interface serial 1
 ip address 192.168.1.8 255.255.255.252
!
router eigrp 10
 network 192.168.1.0
!
```

```
router bgp 10
network 192.168.1.0 mask 255.255.255.252
neighbor 192.168.1.254 remote-as 10
neighbor 192.168.1.9 remote-as 10
```

As shown above, the neighbor update-source is only used for router R1 since it uses a loopback interface as the BGP peer IP address when communicating to both router R2 and router R3. The configuration for router R2 and router R3 uses the loopback IP address 192.168.1.254, defined on router R1 as the neighbor IP address on the respective neighbor remote-as commands for identifying router R1 as a neighbor. When using this configuration, should the link between R1 and R3 go down, R1 will not be able to establish a BGP TCP connection to R3, even if the link between R2 and R3 is active. This is because the IP address used as the BGP neighbor for R3 from R1 is the IP address associated with the link between R1 and R3.

A failsafe configuration is shown below, whereby if any physical interface goes down the BGP TCP connections can reroute their session through the other surviving connection because the loopback interface on all three routers is being used.

Router R1 Configuration for Figure 18-5:

```
ip subnet-zero
!
interface serial 0
 ip address 192.168.1.1 255.255.255.252
!
interface serial 1
 ip address 192.168.1.5 255.255.255.252
!
interface loopback 0
 ip address 192.168.1.254 255.255.255.252
!
router eigrp 10
 network 192.168.1.0
!
router bgp 10
network 192.168.1.0 mask 255.255.255.252
neighbor 192.168.1.250 remote-as 10
neighbor 192.168.1.246 remote-as 10
neighbor 192.168.1.250 update-source loopback 0
neighbor 192.168.1.246 update-source loopback 0
```

Router R2 configuration for Figure 18-5:

```
ip subnet-zero
!
interface serial 0
 ip address 192.168.1.2 255.255.255.252
!
interface serial 1
```

```
 ip address 192.168.1.9 255.255.255.252
!
interface loopback 0
 ip address 192.168.1.250 255.255.255.252
!
router eigrp 10
 network 192.168.1.0
!
router bgp 10
network 192.168.1.0 mask 255.255.255.252
neighbor 192.168.1.254 remote-as 10
neighbor 192.168.1.246 remote-as 10
neighbor 192.168.1.254 update-source loopback 0
neighbor 192.168.1.246 update-source loopback 0
```

Router R3 configuration for Figure 18-5:

```
ip subnet-zero
!
interface serial 0
 ip address 192.168.1.6 255.255.255.252
!
interface serial 1
 ip address 192.168.1.8 255.255.255.252
!
interface loopback 0
 ip address 192.168.1.246 255.255.255.252
!
router eigrp 10
 network 192.168.1.0
!
router bgp 10
network 192.168.1.0 mask 255.255.255.252
neighbor 192.168.1.254 remote-as 10
neighbor 192.168.1.250 remote-as 10
neighbor 192.168.1.254 update-source loopback 0
neighbor 192.168.1.250 update-source loopback 0
```

The network mask command is used under the router bgp command to identify the networks of this AS that we want BGP to distribute to peers. This command is used to quantify which networks the BGP process will advertise. The format of the command is

network *network-number* [**mask** *network-mask*]

The *network-number* is the full class network to be advertised by BGP. Using the optional *network-mask* parameter enables BGP to advertise specific subnets or supernets, instead of just the full class network address. A maximum of 200 network commands is allowed for each BGP process executing in a router. The networks included on the command do not have to be directly connected networks associated with the local router. These networks can be directly connected, dynamically learned, or from static route definitions.

For instance, if router R1 has a network, 192.168.20.0, directly connected to it and this network is advertised using an IGP to a BGP router, the BGP router can include the learned network in its advertisements using the network command. In the example above, since we are using the 192.168.1.0 Class C network with a 255.255.255.252 subnet mask on the loopback interface, we would want the BGP process to advertise the network to its peers to enable connectivity.

Multihop EBGP Connections

Some instances occur when the EBGP TCP connection is not the directly connected IP address. This may happen when the external BGP AS uses a single router for connectivity to the internal BGP AS, or when the external BGP is using a loopback interface as the neighbor IP address for the TCP connection to the EBGP router.

Figure 18-6 illustrates a scenario in which the connection between AS 10 and AS 20 is accomplished using a router acting as the conduit to the internal network of AS 20. In such a scenario, the router administrator of the BGP router R2 in AS 20 has decided that the critical point is the Ethernet connection to their internal network on router R2. If the Ethernet 0 interface on router R2 goes down, then there is no reason for BGP TCP connections to AS 10. This type of situation is termed a multihop configuration.

The following configuration example applies to the topology illustrated in Figure 18-6:

Router R1 Configuration:

```
interface serial 0
 ip address 192.168.1.1 255.255.255.0
!
router bgp 10
network 192.168.1.0
neighbor 172.16.1.1 remote-as 20
neighbor 172.16.1.1 ebgp-multihop
```

Router R2 Configuration:

```
interface serial 0
 ip address 192.168.1.2 255.255.255.0
!
interface Ethernet 0
 ip address 172.16.1.1 255.255.255.0
!
router eigrp
 network 172.16.0.0
 network 192.168.1.0
```

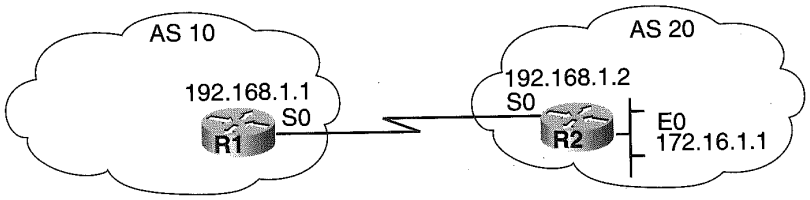


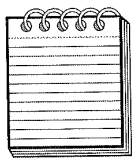
Figure 18-6 EBGP multihop topology when a single Ethernet connection is critical.

```
!
router bgp 20
network 172.16.0.0
neighbor 192.168.1.1 remote-as 10
```

The neighbor defined on router R1 references the Ethernet 0 address of router R2. The EBGP peer connection is established by the inclusion of the neighbor ebgp-multihop command in router R1. The format of the neighbor ebgp-multihop command is

neighbor {*ip-address* | *peer-group-name*} **ebgp-multihop** [*ttl*]

The *ip-address* parameter is the IP address value of a defined BGP neighbor. The value must be in the dotted-decimal IP address format. The parameter *peer-group-name* value is a label given to a group of BGP peers that is recognized collectively as a group. The *ip-address* and *peer-group-name* parameter values are mutually exclusive. The optional *ttl* variable following the **ebgp-multihop** keyword provides the mechanism for determining the timeout of the TCP connection created for the neighbor IP address defined. The range for the optional *ttl* variable value is 1 to 255.



NOTE: *Cisco IOS does not establish the multihop peer connection if the only available route to the defined neighbor is the default 0.0.0.0 route.*

Load balancing over parallel links between two routers of different autonomous systems is made possible by using the loopback interface as the neighbor IP address. Because the loopback interface is not a directly connected EBGP, multihop is required. Figure 18-7 illustrates this scenario. In the standard EBGP connection, BGP will select one of the links for sending packets. However, by using EBGP multihop and loopback interfaces, the

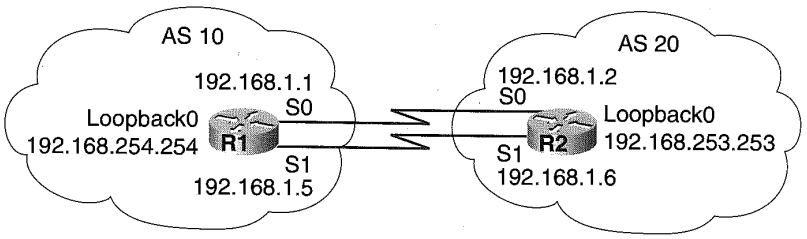


Figure 18-7 Forcing load balancing over parallel links using EBGP multihop and loopback interfaces.

BGP process is manipulated into using both physical connections and thereby performs load balancing.

The following configuration example details Figure 18-7:

Router R1 configuration:

```
ip subnet-zero
!
interface serial 0
 ip address 192.168.1.1 255.255.255.252
!
interface serial 1
 ip address 192.168.1.5 255.255.255.252
!
interface loopback 0
 ip address 192.168.254.254 255.255.255.252
!
router bgp 10
network 192.168.254.0
neighbor 192.168.253.253 remote-as 20
neighbor 192.168.253.253 ebgp-multihop
neighbor 192.168.253.253 update-source loopback 0
!
ip route 192.168.253.0 255.255.255.0 192.168.1.2
ip route 192.168.253.0 255.255.255.0 192.168.1.6
```

Router R2 configuration:

```
ip subnet-zero
!
interface serial 0
 ip address 192.168.1.2 255.255.255.252
!
interface serial 1
 ip address 192.168.1.6 255.255.255.252
!
interface loopback 0
 ip address 192.168.253.253 255.255.255.252
!
router bgp 20
network 192.168.253.0
neighbor 192.168.254.254 remote-as 10
```

```
neighbor 192.168.254.254 ebgp-multihop
neighbor 192.168.254.254 update-source loopback 0
!
ip route 192.168.254.0 255.255.255.0 192.168.1.1
ip route 192.168.254.0 255.255.255.0 192.168.1.5
```

Static routes are introduced into the configuration using the ip route command in each router. This allows each router to create two equal cost paths to reach the BGP neighbor addresses. An IGP can also be used to establish the equal cost path.

Controlling Routing Information Using Route Maps

The **route-map** configuration command is used in the context of BGP to define conditions for redistributing routing information to other routing protocols or for controlling routing information inbound or outbound from the BGP autonomous systems. The format of the command is

route-map *map-tag* **[[permit | deny]** | [*sequence-number*]]

The *map-tag* variable value is a name that is associated with the route map. More than one instance of the same *map-tag* value can be defined in the router. The *sequence-number* variable value defines the position of the route-map entry in the list of route maps configured with the same *map-tag* value.

As an example, the following configuration identifies a single route-map named MAP1 with two instances. The first instance is defined using sequence-number 10 and the second is defined using sequence number 20:

route-map MAP1 permit 10

()

()

()

route-map MAP1 permit 20

()

()

()

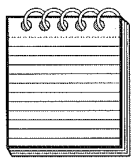
The statements following the **route-map** command describe the set of conditions applied to the *sequence- number* of the *map-tag*. In this example,

if the conditions following the MAP1 instance 10 are not met, the IOS proceeds to the next higher instance of the same *map-tag* value. In this case, MAP1 instance 20 and its associated conditions are applied. The **permit** and **deny** keywords of the **route-map** command indicate the action taken on the on the routing information.

The conditions that follow a **route-map** command are defined using match and set configuration commands. The **match** command defines the criteria used to determine if the routing information is of interest, and the **set** command specifies the action to take against the routing information. For example, if the route-map conditions are

```
match ip address 10.1.20.100
set metric 10
```

and the **permit** keyword is specified on the **route-map** command, the routes are redistributed or controlled in accordance with the set command following the **match** command. In this case, the routing information will have its distance metric set to 10. If the **route-map** command has the **deny** keyword applied, the route is not redistributed or controlled and the IOS breaks out of the list. On the other hand, if the match is not met at all and either **permit** or **deny** is specified on the **route-map** command, the next instance of the **route-map** is checked. This checking continues until a match is made to the list. Perchance that a match is not found in the list, the route will not be accepted, nor will it be forwarded.



NOTE: *route-map, when used for filtering BGP updates instead of redistributing between protocols, is not capable on filtering inbound when using a match on the ip address. Only outbound filtering when using a match on ip address is allowed. Route-map functions are available to all other routing protocols.*

Table 18-1 lists all the match and set conditions available using the route-map configuration command for BGP redistribution and routing information control.

In the following router configuration example, a typical configuration exists in which one AS is running RIP and the connection to another AS is done using BGP. Shown in Figure 18-8, router R1 and router R2 are using RIP within their AS. Router R2 of AS 100 and router R3 of AS 200 utilize BGP for routing information updates. Router R2 redistributes the BGP updates received from router R3 to RIP for distribution to router R1. If the routes to 172.16.0.0 are to be favored through router R2 versus some other

Table 18-1

"match" and "set" Commands for the route-map Configuration Command

"match" on	"set" Variable
match as-path	set as-path
match community	set clns
match clns	set automatic-tag
match interface	set community
match ip address	set interface
match ip next-hop	set default interface
match ip route-source	set ip default next-hop
match metric	set level
match route-type	set local-preference
match tag	set metric
	set metric-type
	set next-hop
	set origin
	set tag
	set weight

connection, we can set the metric to a value of 2 and all other routers with a metric of 10.

The following router configurations can be used to accomplish the setting of the metrics discussed above. The configurations only display the pertinent route configuration commands:

Configuration for router R2 in Figure 18-8:

```
R2#
router rip
network 172.16.0.0
network 172.18.0.0
network 172.20.0.0
passive-interface Serial0
redistribute bgp 100 route-map METRIC2
router bgp 100
neighbor 172.18.2.3 remote-as 200
network 172.20.0.0
route-map METRIC2 permit 10
```

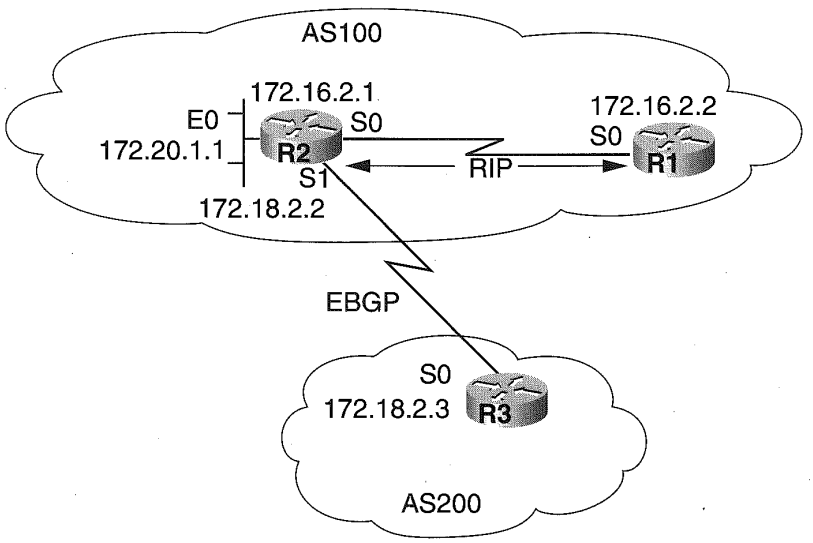


Figure 18-8 Using route-map conditions to set metrics for redistributed RIP routes.

```
match ip-address 1
set metric 2
route-map METRIC2 permit 20
set metric 10
access-list 1 permit 172.17.0.0 0.0.255.255
```

In this example, the first instance of METRIC2 uses the access-list 1 for IP access-list conditions to set the metric to 2. If the route matches the IP address 172.17.0.0, it will have a metric of 2 applied to the route and the IOS will break out of the list. If the route does not match, however, the next sequential instance is compared.

In this case, instance 20 is compared. Instance 20 of METRIC2 does not even provide a match command and therefore provides a default metric value of 10 for all other routes that do not provide routing information for the 172.17.0.0 network. This example also demonstrates the requirement to ask what will happen to route information that does not match any of the conditional criteria. If a second instance for all other routes was not included, the route would be dropped and not forwarded to any other routers.

Since the matching on IP addresses for inbound updates is not possible, outbound route maps must be applied. In the above example, route updates for 172.17.0.0 can be filtered from AS 100 by applying an outbound route map to router R3. Using the same topology as in Figure 18-8, router R3 can

filter updates about 172.17.0.0 to router R2 and AS 100 using the following router configuration:

Router R3 configuration for filtering 172.16.0.0 updates to AS 100:

```
router bgp 200
network 172.17.0.0
neighbor 172.18.2.2 remote-as 100
neighbor 172.18.2.2 route-map NO17217 out
route-map NO17217 permit 10
match ip-address 1
access-list 1 deny 172.17.0.0 0.0.255.255
access-list 1 permit 0.0.0.0 255.255.255.255
```

In this configuration example, the IP access-list number 1 applies a deny to the IP network 172.17.0.0 and permits any other network. The neighbor route-map command under the router bgp command applies the route-map tag named NO17217 to outbound routing updates. The route-map command map-tag named NO17217 indicates that the IOS will permit routing updates that pass through the access-list number 1 successfully. For example, if the 10.0.0.0 network is in AS 200 and a route for this network is being sent to router R2 in AS 100, the access-list number 1 in router R3 would allow the route, and the route-map command would then permit the route to be sent to the neighbor BGP router at address 172.18.2.2 in AS 100.

Redistributing IGP Routes into BGP

Interior Gateway Protocols (IGPs) such as IGRP, RIP, OSPF, and EIGRP can be advertised using BGP through route redistribution. This means that the internal routes of your AS will be advertised as BGP routes. In some cases, this may cause duplication of routes since BGP may have already learned of these IGP routes. In this case, using filters will help avoid the duplication of routes to the Internet and also enable the advertising of only those routes required. A further discussion of route redistribution is found in Chapter 19, "Route Redistribution."

In Figure 18-9, we see three AS's in which router R1 in AS 100 is advertising 192.168.1.0 and router R3 is advertising network 172.22.0.0. Using router R3 as an example, the router is using EIGRP as the IGP and redistributing EIGRP routes into BGP AS 200.

The router R3 configuration using only the network command AS:

router eigrp 10

```
network 172.22.0.0
redistribute bgp 200
```

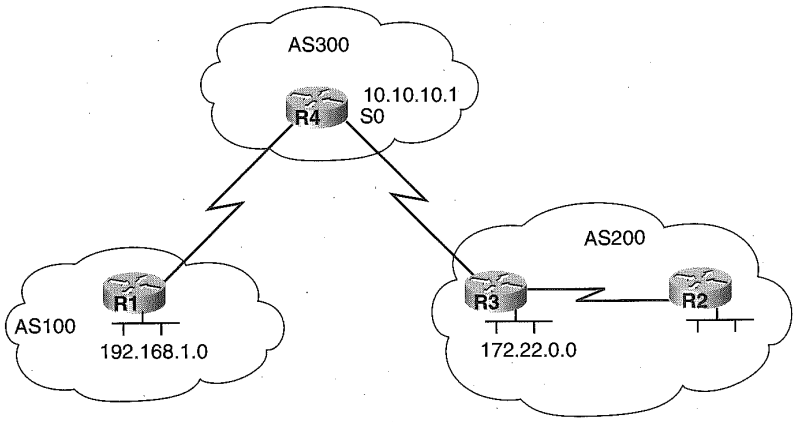


Figure 18-9 Using route redistribution between EIGRP and BGP.

```
default-metric 1000 100 250 100 1500
router bgp 200
neighbor 10.10.10.1 remote-as 300
network 172.22.0.0 mask 255.255.0.0
```

Using only the network command under the router bgp definition limits the networks originated by AS 200 to 172.22.0.0. However, applying the redistribute command under the router bgp definition, as follows

```
router eigrp 10
network 172.22.0.0
 redistribute bgp 200
default-metric 1000 100 250 100 1500
router bgp 200
neighbor 10.10.10.1 remote-as 300
redistribute eigrp 10
```

causes 192.168.1.0, known by EIGRP, to be injected into BGP. Since AS 200 is not the source of the address, filters are needed to stop the 192.168.1.0 network from being sourced by AS 200. This is accomplished using access-lists. The following configuration for router R3 in Figure 18-9 illustrates the uses of the filter to achieve the route redistribution without sourcing 192.168.1.0 from AS 200.

Router R3 configuration using access-list filters for route redistribution:

```
router eigrp 10
network 172.22.0.0
redistribute bgp 200
default-metric 1000 100 250 100 1500
```

```
router bgp 200
neighbor 10.10.10.1 remote-as 300
neighbor 10.10.10.1 distribute-list 1 out
redistribute eigrp 10
access-list 1 permit 172.22.0.0 0.0.255.255
```

Using this configuration, the neighbor distribute-list command under router bgp 200 applies an outbound filter that points to access-list 1, which permits only 172.22.0.0 routes to be advertised. All other routes are denied due to the implicit deny of an access list.

Static routes can also be used to identify originating networks or subnets within a router. BGP, however, considers static routes as having an unknown origin. This is also termed as incomplete origin. Using static routes to accomplish the routing in the previous example, router R3 could have been configured as

```
router eigrp 10
network 172.22.0.0
redistribute bgp 200
default-metric 1000 100 250 100 1500
router bgp 200
neighbor 10.10.10.1 remote-as 300
redistribute static
ip route 172.22.0.0 255.255.255.0 null0
```

The static route defined using the ip route command forces the route into the routing table; however, the actual packets are discarded due to the route pointing to the null0 interface. Using this type of method, a supernet can be safely advertised. Remember that the routes generated from the above methods are in addition to BGP routes learned from BGP neighbors either internal or external. Using redistribution, static, or the network command generates routes as originating from the AS of the router.

Nexthop Attribute and Multiaccess/NBMA Networks

In Figure 18-10, two autonomous systems are connected using router R1 and R3. The next-hop address is always the IP address specified on the neighbor command for EBGP connections. In the figure, router R3 uses 172.16.2.2 as the next-hop address when advertising route 172.16.0.0 to router R1. Router R1 will use the next-hop address 172.16.2.1 when advertising the network 172.31.0.0 to router R3. The IBGP protocol requires that the next-hop advertised by EBGP be carried into IBGP. This forces router R1 to advertise 172.16.0.0 to router R2, its IBGP peer with a next-hop of

172.16.2.2. Because of this, router R2 uses the next-hop address of 172.16.2.2 and not 172.31.2.1 to reach the 172.16.0.0 network.

A key to this configuration is that router R2 must be able to reach the next-hop address of 172.16.2.2 using the IGP of the AS. One way to accomplish this is to run the IGP on router R1 network 172.16.0.0 and make the IGP link passive to router R3 so only BGP information is exchanged. The following configuration demonstrates this technique as applied against Figure 18-10:

Router R1 configuration for Figure 18-10:

```
router bgp 100
neighbor 172.16.2.2 remote-as 300
neighbor 172.31.5.1 remote-as 100
network 172.31.0.0
```

Router R2 configuration applied to Figure 18.10:

```
router bgp 100
neighbor 172.31.3.1 remote-as 100
```

Router R3 configuration as applied to Figure 18-10:

```
router bgp 300
neighbor 172.16.2.1 remote-as 100
network 172.16.0.0
```

The result of these configurations is the advertisement of 172.16.0.0 from router R3 to router R1 will use 172.16.2.2 as the next-hop address. Router R1 will use 172.16.2.2 as the next-hop address when router R1 advertises 172.16.0.0 to router R2.

On multiaccess networks, as illustrated in Figure 18-11, the EBGP router advertises the actual address of the router originating the advertised network, instead of its own, since the EBGP routers and the target IBGP router are on the same multiaccess network. As shown in Figure 18-11, router R1 in AS 100 connects to AS 200 using a common Ethernet segment on which router R3 and R4 are attached. Router R3 is the EBGP peer-to-router R1.

The router R3 sends a BGP update to router R1 using the next-hop address of 172.16.2.3 for network 172.30.0.0, instead of its own IP address 172.16.2.2 for the common Ethernet. This is done to avoid an unnecessary hop to access the networks originating on router R4.

In NBMA networks where connections between BGP AS's are not provided, using a full-mesh routing fails because the next-hop IP address advertised is not reachable by some of the routers connected to the NBMA network. This is diagrammed using Figure 18-12. Router R3 advertises a

next-hop address of 172.16.2.3 to router R1 for the network 172.31.0.0, which is connected to router R4. Router R3 knows of this next-hop address because it has a Frame Relay PVC connecting to router R4. However, router R1 does not have a PVC to router R4 and will not be able to route packets to the next-hop address specified by router R3 in its advertisement of the 172.31.0.0 network. The Cisco IOS solves this dilemma by incorporating a command feature called nexthopself.

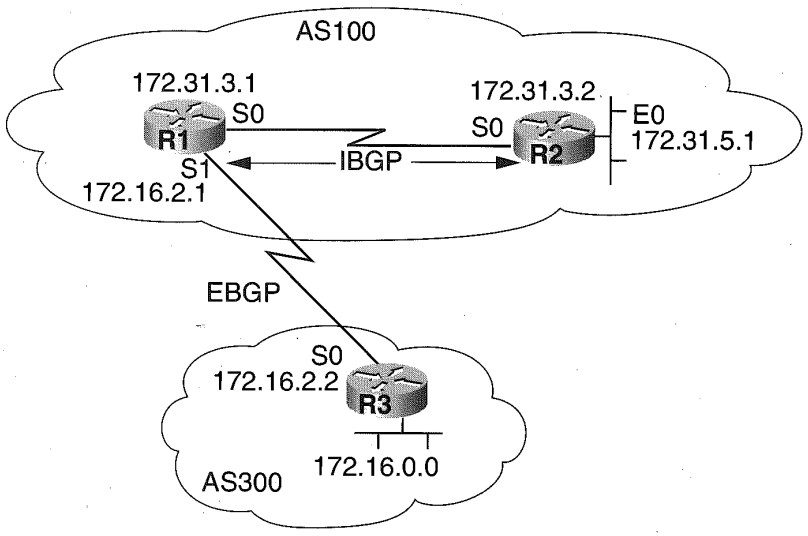


Figure 18-10 Next-hop attribute assignment BGP router configuration.

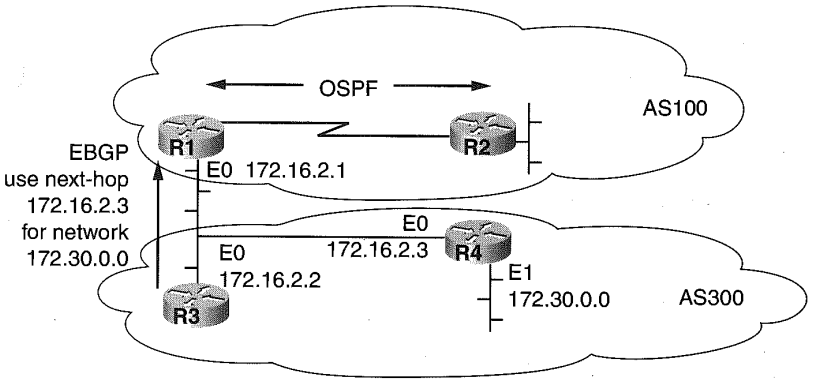


Figure 18-11 Next-hop addressing on multiaccess networks.

The nexthopself feature is applied to the neighbor command as follows:

neighbor {*ip-address* | *peer-group-name*} **next-hop-self**

The *ip-address* variable value is the IP address of the neighbor. A *peer-group-name* can be used instead of an IP address to identify a group of BGP peers to which this command applies. The command is most useful on NBMA networks where the BGP neighbors do not have direct access to each other even though they are on the same IP network or subnet. Using this command causes the router to override the true default next-hop IP address and places instead in the advertisement its own IP address for the interface connecting to the neighbor identified by the *ip-address* or *peer-group-name* values.

In the configuration laid out in Figure 18-12, the following is entered on router R3 to identify itself as the next-hop address, instead of 172.16.2.3 as the next-hop address for network 172.31.0.0:

Router R3 configuration as applied to Figure 18-12:

```
router bgp 300
neighbor 172.16.2.1 remote-as 100
neighbor 172.16.2.1 next-hop-self
```

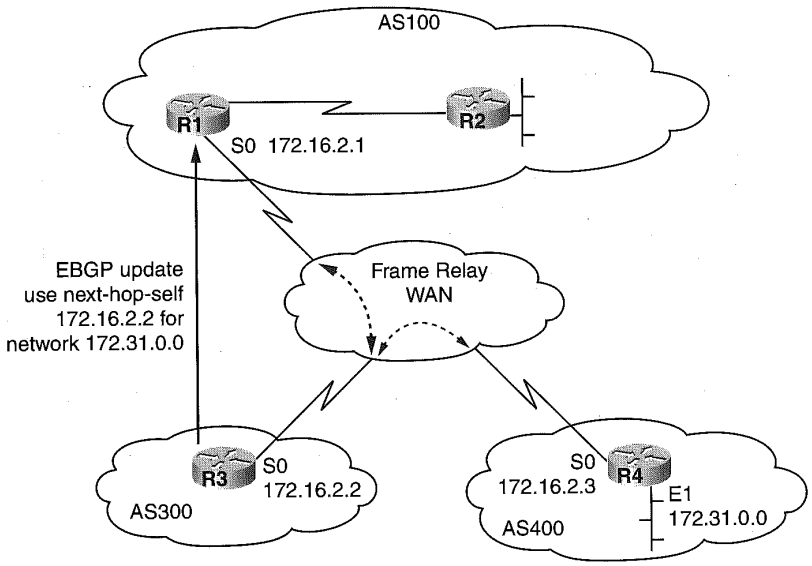


Figure 18-12 Next-hop addressing configuration for NBMA networking example.

Using this configuration, router R3 sends a EBGP update to router R1, specifying its own IP address of 172.16.2.2 as the next-hop address for network 172.31.0.0 and enabling connectivity for AS100 to AS 400.

Forcing BGP to Prefer a IGP Route Using the Backdoor Command

In some cases, AS connectivity may be a combination of EBGP and IGP. In Figure 18-13, Router R1 in AS 100 connects to router R2 in AS 200 and uses an IGP (IGRP, EIGRP, OSPF, RIP) to build routing tables between the two networks. Router R1 and Router R2 both use EBGP to communicate routing information with router R3 of AS 300. In the figure, router R1 will receive two routing updates for network 172.16.0.0 (160.10.0.0), which is connected to router R2 in AS 200.

You can indicate which networks are reachable by using a *backdoor* route that the border router should use. A backdoor network is treated as a local network, except that it is not advertised. To configure backdoor routes, use the following command in router configuration mode:

network *address* **backdoor**

The *address* variable value for the network backdoor configuration command is the IP network reachable using the IGP. Using Figure 18-13, a backdoor command can be added to the configuration for router R1, enabling a route to be used for the more direct path to network 172.16.0.0 as follows:

Configuration for router R1 using the network backdoor command applied to Figure 18-13:

```
router eigrp 10
network 172.16.0.0
router bgp 100
neighbor 10.20.20.1 remote-as 300
network 172.16.0.0 backdoor
```

Router R1 learns of network 172.16.0.0 from router R2 using EIGRP with a default distance of 90. R1 also receives a routing update from router R3 via EBGP with a default distance of 20. The EBGP route, because of the lower distance value, is normally the preferred route, but because the network backdoor command is used in the configuration, the EIGRP route becomes the preferred route.

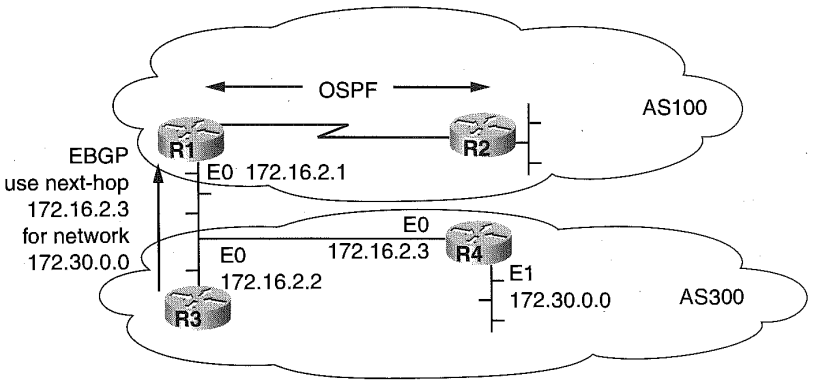


Figure 18-13 EBGP and IGP network configuration and backdoor routing.

Synchronization of Routes

BGP does not advertise a route until all routes within the AS have learned of the route through the IGP. Once the IGP has propagated the route to all other routers within the AS, BGP then advertises it to the external peers. This mostly happens when traffic is passing from one AS to another AS using the local AS. Figure 18-14 illustrates this concept.

The synchronization between BGP to IGP routes is demonstrated using Figure 18-14. In the figure, router R3 in AS300 sends routing updates about its connected network, 172.16.0.0. Router R1 and R2 use IBGP, enabling R2 to receive updates for network 172.16.0.0 by using 10.20.20.1 as the next hop, and are not redistributing routes into IGP. For R2 to reach the next-hop address 10.20.20.1, it sends traffic to router R5. This creates a problem because router R5 is not aware of the 172.16.0.0 network. Router R2 using BGP synchronization will not advertise the 172.16.0.0 route until it is learned using the IGP. Once learning of the route, router R2 updates router R4. To facilitate the process, a static route in router R2 can be added, forcing router R2 to add the 172.16.0.0 network into the IGP routing table.

This synchronization process is enabled under BGP by default, but in some scenarios synchronization is not warranted. An example would be when the AS is not acting as a transient AS for traffic between two other autonomous systems, or when all routes within the AS are running BGP. In these cases, synchronization can be disabled, which reduces the routes in the IGP routing table while providing quicker convergence for BGP. Disabling BGP synchronization is accomplished using the following BGP router configuration command:

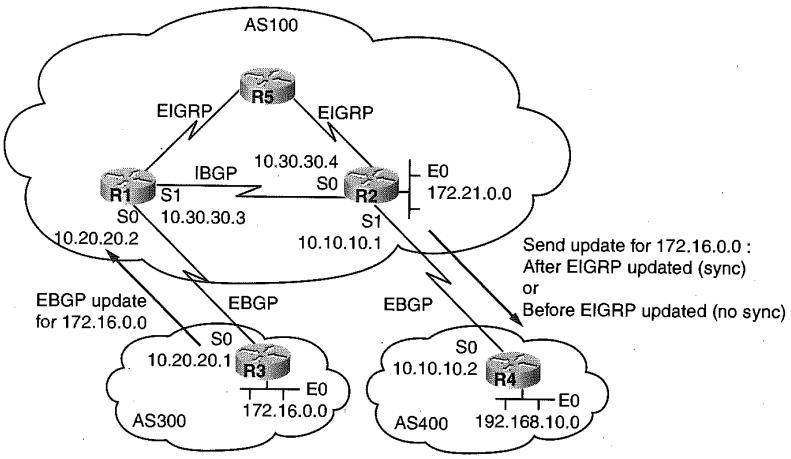


Figure 18-14 Network configuration illustrating route synchronization.

no synchronization

Employing this command under the router bgp configuration command enables BGP to advertise routes to peers without synchronizing the route to the IGP. Using the Figure 18-14 for the following sample configuration, the no synchronization command can be added to router R2, which enables R2 to place the 172.16.0.0 network in the IP routing table and advertises to router R4 without the route being included in the IGP routing table.

Configuration for Router R2 in Figure 18-14 to disable synchronization:

```
router bgp 100
network 172.21.0.0
neighbor 10.10.10.2 remote-as 400
neighbor 10.30.30.3 remote-as 100
no synchronization
```

Router R1 configuration for Figure 18-14:

```
router bgp 100
network 172.21.0.0
neighbor 10.30.30.4 remote-as 100
```

Router R4 configuration for Figure 18.14

```
router bgp 400
neighbor 10.10.10.1 remote-as 100
network 192.168.10.0
```

In order for the no synchronization command to take effect, the following

command must be entered in EXEC operator mode to reset the BGP peer connections:

clear ip bgp *address*

The address variable value is the IP address of the neighbor for which the BGP routes are cleared. In the example, the following commands entered sequentially will clear the neighbor routes and reset the BGP peer connection for the R1 and R4 routers in Figure 18-14:

```
clear ip bgp 10.30.30.3
clear ip bgp 10.10.10.2
```

Best Path Selection Using the Weight Attribute

Cisco IOS Release 10.0 introduced a Cisco-specific BGP attribute to identify preferred paths when multiple paths for a network exist. The router bgp configuration command neighbor weight command is as follows:

neighbor {*ip-address* | *peer-group-name*} **weight** *weight*

The *ip-address* variable is the IP address of the neighbor to which the *weight* is applied. The *peer-group-name* can be used instead of the *ip-address* variable and identifies a BGP peer group to which the *weight* will be applied to all the neighbors in the group. The *weight* variable of the **weight** keyword can range from 0 to 65,535. The preferred path is the weight with the higher value.

Normally, without coding the neighbor weight command routes learned from other BGP peers have a default weight of 0. Routes originating within the local router use 32,768 as the default weight.

Using Figure 18-15, router R1 knows of the 172.21.0.0 network from AS400. Router R1 provides router R3 with this information through routing updates. Router R2 also knows of the 172.21.0.0 network via AS400 and sends routing updates to router R3. Router R3 now has in its routing tables two viable paths to reach the 172.21.0.0 network. By setting the weight attribute higher on router R3 for updates from router R1 over updates from router R2, the preferred path to the 172.21.0.0 network for router R3 is through the router R1 BGP peer connection with a next hop of 172.21.0.0. The following configuration illustrates the commands used in router R3 to achieve the preferred path through router R1:

Router R3 configuration to prefer R1 in Figure 18-15:

```
router bgp 300
neighbor 10.10.10.1 remote-as 100
neighbor 10.10.10.1 weight 200
neighbor 10.20.20.2 remote-as 200
neighbor 10.20.20.2 weight 100
```

The result of the configuration causes the connection from router R3 to router R1 to be the selected route since its defined weight is higher (200) than that of the weight assigned to the peer connection between router R3 and router R2.

Forcing a Preferred AS Exit Path

In many network topologies, multiple paths can be selected from among the AS's. A preferred path out of an AS can be identified as a preferred path by modifying the local preference attribute of BGP. The local preference attribute is set using the Cisco IOS command:

bgp default local-preference *value*

The value variable is a valid decimal integer ranging from 0 to 4,294,967,295. The higher the number, the greater preference is given to the route being advertised to reach the AS. If the local preference attribute is not set through IOS commands, it defaults to 100. In Figure 18-16, a typical configuration is illustrated, showing the local preferences for the various BGP routers within the network.

In Figure 18-16, AS200 receives routing information for the 172.16.0.0 network from router R1 of AS100 and router R2 of AS300. The local preference attribute set in the routes enables the BGP process to determine the best path to exit AS200 when reaching the 172.16.0.0 network. The following configuration examples sets router R4 as the preferred exit from AS 200.

Configuration for router R3 in Figure 18-16:

```
router bgp 200
neighbor 10.10.10.1 remote-as 100
neighbor 192.168.11.2 remote-as 200
bgp default local-preference 1500
```

Configuration for Router R4 in Figure 18-16:

```
router bgp 200
neighbor 10.30.30.4 remote-as 300
neighbor 192.168.11.1 remote-as 200
bgp default local-preference 2000
```

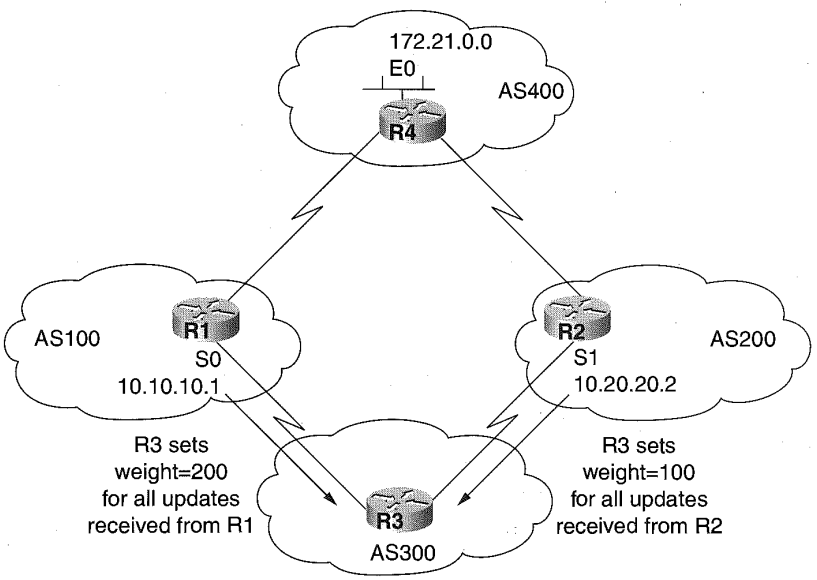


Figure 18-15
Influencing BGP path selection using the neighbor weight attribute.

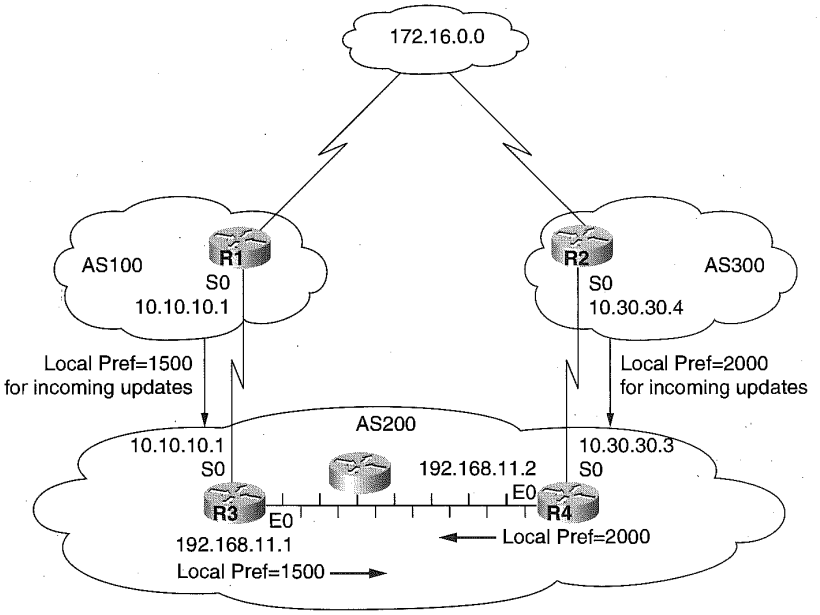


Figure 18-16
The effect of the local preference attribute for selecting the preferred AS exit path.

Because the BGP local preference attribute value is exchanged between BGP peers within an AS, router R3 and R4 of AS200 determine that the 172.16.0.0 network is given a higher local preference when connectivity is made from AS300 over AS100. All traffic destined for the 172.16.0.0 network, including traffic originating at networks on router R3, will traverse the path from router R4 to router R2.

Influencing the Preferred Path into an AS

Many configurations use multiple paths between the autonomous systems for redundancy and availability. Modifying the metric attribute allows an AS to influence the preferred path dynamically between the autonomous system connections when multiple entry points exist. If BGP4 is being used, the metric attribute is known as the Multi-exit discriminator (MED) attribute. In BGP3, the metric attribute is called the Inter-As attribute. Unlike the local preference, which is specific to the BGP routing process within a Cisco router, the metric attribute is exchanged with the external BGP neighbors. However, the metric received is not sent to another AS. BGP defaults the metric attribute value to 0 and prefers the lowest metric attribute presented in the compared routing information updates received for a given network. Figure 18-17 illustrates the use of the metric attribute.

Routers R2, R3, and R4 are sending routing information to AS 100 for the 192.168.180.0 (180.10.0.0) network. R2 is in AS 400, and R3 and R4 are in AS 300. Each router influences the preferred path into AS 100 by setting the metric attribute using the following configurations:

Router R2 configuration for Figure 18-17:

```
router bgp 400
neighbor 10.40.40.4 remote-as 100
neighbor 10.40.40.4 route-map setmetricout out
route-map setmetricout permit 10
set metric 50
```

Router R3 configuration for Figure 18-17:

```
router bgp 300
neighbor 10.20.20.2 remote-as 100
neighbor 10.20.20.2 route-map setmetricout out
neighbor 10.10.10.2 remote-as 300
route-map setmetricout permit 10
set metric 120
```

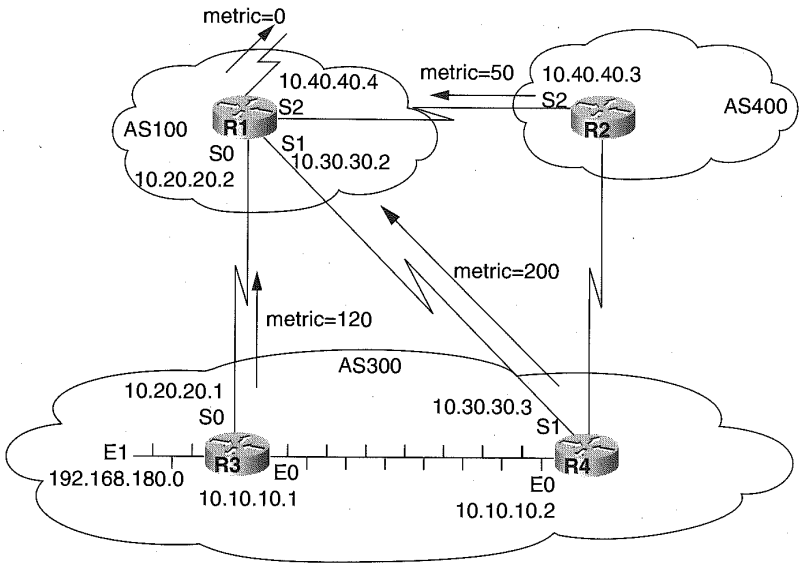


Figure 18-17 Influencing the preferred path into an AS using the metric attribute.

Router R4 configuration for Figure 18-17:

```
router bgp 300
neighbor 10.30.30.2 remote-as 100
neighbor 10.30.30.2 route-map setmetricout out
neighbor 10.10.10.1 remote-as 300
route-map setmetricout permit 10
set metric 200
```

Routers R2, R3, and R4 each set the metric attribute using the **route-map setmetricout** and **set metric** commands. Router R2 sets the metric attribute for routes coming from R2 to 50. Routes coming from router R3 are set to 120 and the ones from router R4 are set to 200.

By default, routers compare only the metrics from neighbors of the same AS. In the example, the routing information provided by routers R3 and R4 are compared by router R1. The routing information from router R2 is not compared because router R2 is in a different AS than router R3 and R4. Router R1 also receives a routing update for the 192.168.180.0 network from router R2. This metric attribute shows a lower value of 50, but router R1 cannot compare the routing information received from routers R3 and R4 with that of R2 because R2 is in a different AS than routers R3 and R4.

Router R1 then chooses the preferred route to 192.168.180.0 based on other BGP attributes. R1 can be configured, however, to compare routing

information received from all connected autonomous systems in an effort to compare the metric attribute of a network to all external BGP neighbors by specifying the following command under the router bgp command:

bgp always-compare-med

Using this command in the configuration for router R1 causes the BGP process to compare the metric attribute of all alternate paths to a destined network. In Figure 18-17, router R1 is configured to take advantage of this enhancement as follows:

Router R1 configuration for Figure 18-17:

```
router bgp 100
neighbor 10.20.20.1 remote-as 300
neighbor 10.30.30.3 remote-as 300
neighbor 10.40.40.3 remote-as 400
bgp always-compare-med
```

By including the **bgp always-compare-med** command in router R1, the preferred path to 192.168.180.0 network is router R2. Without the inclusion of the **bgp always-compare-med** command in router R1, the preferred path to the 192.168.180.0 network is to router R3. In this example, the extra hop through router R2 is warranted because of the low line speeds connecting router R3 and R4 to R1. The R1-R2 and R2-R4 connections use T1 line speeds and hence will have a greater probability of throughput.

Grouping Destinations That Share a Common Attribute

There are many network configuration instances in which multiple destinations can be grouped by shared common attributes. By grouping these destinations into a community, routing decisions and other BGP attributes can be applied to all the destination networks belonging to the community. The communities attribute is optional and is not sent to a neighbor, but if the set community command is used in conjunction with a route-map and match command, a destination is assigned all the attributes belonging to the community as a whole. The format of the **set community** command is

set community {*community-number* **[additive]}** | **none**

The *community-number* variable value can be in the range of 1 to 4,294,967,200, or the keyword **no-export** or **no-advertise.** The default *community-number* value is the well-know community name **internet.**

Every BGP router belongs to the **internet** community. When using the **no-export** value for *community-number*, the route is not advertised to EBGP peers. The **no-advertise** keyword value for the *community-number* variable indicates to the BGP process that the route in question is not advertised to either an internal or external BGP peer.

The **additive** keyword following the *community-number* value indicates that the network route being interpreted will have the specified community added to its routing information. Without the **additive** keyword, the specified community will replace all previously defined communities for the destination network in any further route information updates. The final keyword **none** directs the BGP process to strip away the communities attribute from the prefix of any route that passes the route-map criteria.

In order for BGP neighbors to learn of the different communities for destination networks, the neighbor send-community command must be entered into the router configuration. The format of the command is

neighbor {*ip-address* | *peer-group-name*} **send-community**

The *ip-address* variable is the IP address of the BGP neighbor to which the communities attribute is sent. The *peer-group-name* variable can be used in place of the *ip-address* variable, resulting in the same action.

The following router configuration provides an example for using the communities attribute.

```
router bgp 100
 neighbor 192.168.171.50 remote-as 200
 neighbor 192.168.171.50 send-community
 neighbor 192.168.171.50 route-map set-community out
!
route-map community 10 permit
 match address 1
 set community no-export
!
route-map community 20 permit
 match address 2
```

In this example, the route map named **community** is applied to the outbound routing updates of the neighbor using the 192.168.171.50 IP address. The actions are applied to the outbound updates based on the out keyword specified on the **neighbor route-map** command. As the BGP routes are processed, they pass through access list 1 and 2. The routes passing access list 1 are set to the well-known community name of no-export. Any routes that fail the criteria for instance 10 are processed by instance 20 of the community route-map where they are advertised as normal since no community attribute is applied to their routing information. Routes meeting the

criteria for instance 10 of the community route-map will not be advertised to EBGP peers.

Suppose in Figure 18-18, the BGP routes advertised from router R2 to R3 are not to be sent by router R3 to any other external peers. The following configuration is used:

Router R2 configuration for Figure 18-18:

```
router bgp 200
network 172.16.0.0
neighbor 10.30.30.1 remote-as 300
neighbor 10.30.30.1 send-community
neighbor 10.30.30.1 route-map setnoexport out
route-map setnoexport
match ip address 1
set community no-export
access-list 1 permit 0.0.0.0 255.255.255.255
```

Router R2 sets all outbound routing updates to router R3 with the no-export communities attribute forcing router R3 to not forward the routes received from router R2 to router R1. Again, this is indicated by using the neighbor send-community command in concert with the neighbor route-map out command.

The communities attribute can also be used to set other attributes like weight and metric. In the following configuration examples applied to Figure 18-18, router R2 now sets the communities attribute to include the 300 and 400 communities with any other communities attribute found on the route. We can also have router R3 set the weight for these routes.

Router R2 configuration using the additive keyword applied to Figure 18-18:

```
router bgp 200
network 172.16.0.0
neighbor 10.30.30.1 remote-as 300
neighbor 10.30.30.1 send-community
neighbor 10.30.30.1 route-map set-community out
route-map set-community
match ip address 2
set community 300 400 additive
access-list 2 permit 0.0.0.0 255.255.255.255
```

Router R3 configuration applied to Figure 18-18:

```
router bgp 300
neighbor 10.30.30.3 remote-as 200
neighbor 10.30.30.3 route-map community-check in
route-map community-check permit 10
match community 1
set weight 20
route-map community-check permit 20
```

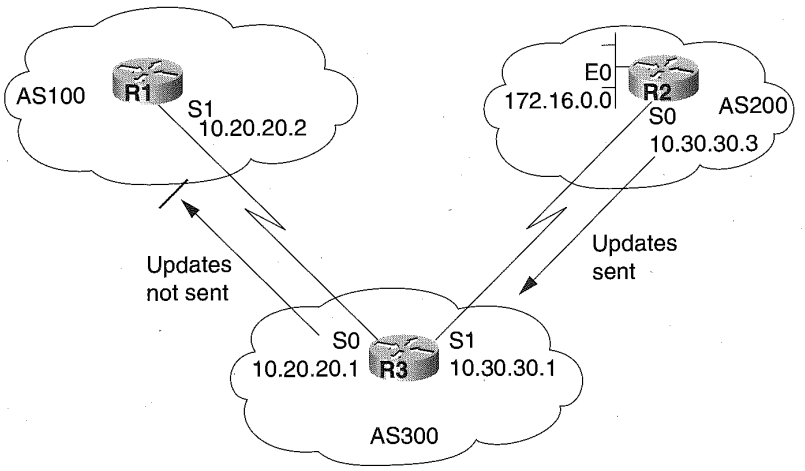


Figure 18-18
Using the community attribute for route filtering.

```
match community 2 exact
set weight 10
route-map community-check permit 30
match community 3
ip community-list 1 permit 300
ip community-list 2 permit 400
ip community-list 3 permit internet
```

The result of these configurations matches any route with 300 in the communities attribute by match list 1. The action taken in match list 1 sets the weight of the route to 20. Match list 2 applies to any route that only has the 400 community name in the communities attribute because of the exact keyword used on the match community statement. If only the 400 community is assigned to the communities attribute, it will be set to a weight of 20. The last instance of the community-check route map catches any other type of route and assigns the well-known community name of internet to the communities attribute.

This filtering is made possible by the use of the ip community-list command. This command groups communities together for use in a match clause statement of a route-map command. The format of the ip community-list command is

ip community-list *community-list-number* **{permit | deny}** *community-number*

The *community-list-number* variable value ranges from 1 to 99 and identifies one or more permit or deny community groups. The **permit** keyword

grants access for the matching condition specified, while the **deny** keyword prohibits access when a matching condition is found. The *community-number* variable value is a valid *community-number* defined on a previous **set community** command.

It is important to remember that an implicit deny will follow the first permit community list. It is therefore always necessary to permit any routes that do not meet any of the previous match criteria. This is shown in the sample configuration above for router R3.

Route and Path Information Filtering

As in all other routing protocols, managing and controlling the number of routing updates is an important practice for conserving network resources. Three basic filtering techniques are used for controlling BGP routing updates. These filtering methods are based on route information, path information, and communities. In the previous section, we discussed filtering on communities. This section concentrates on filtering using route and path information.

Routing information updates can be managed by using access-lists that are defined and applied to the BGP routing updates both received and sent. Cisco IOS achieves this filtering by using the neighbor distribute-list command under the router bgp command. The format of the neighbor distribute-list command is

neighbor {*ip-address* | *peer-group-name*}
distribute-list {*access-list-number* | *name*} {**in** | **out**}

The *ip-address* variable value is the IP address of a BGP neighbor. The *peer-group-name* variable can be used instead of the *ip-address* variable. The *access-list-number* variable is a valid integer from 1 to 199, identifying a standard or extended access list that is applied to the routing updates to or from the identified neighbor. The *access-list-name* can be used instead of the *access-list-number*. The **in** keyword applies the identified access list to the incoming routing updates and the **out** keyword applies the identified access list to the outgoing routing updates.

Suppose in Figure 18-19 updates for network 172.16.0.0 originating from router R2 and received by router R3 are filtered to stop these updates from being propagated into AS100. Using the neighbor distribute-list command, router R3 can filter these updates by applying the filter for routing information to router R1.

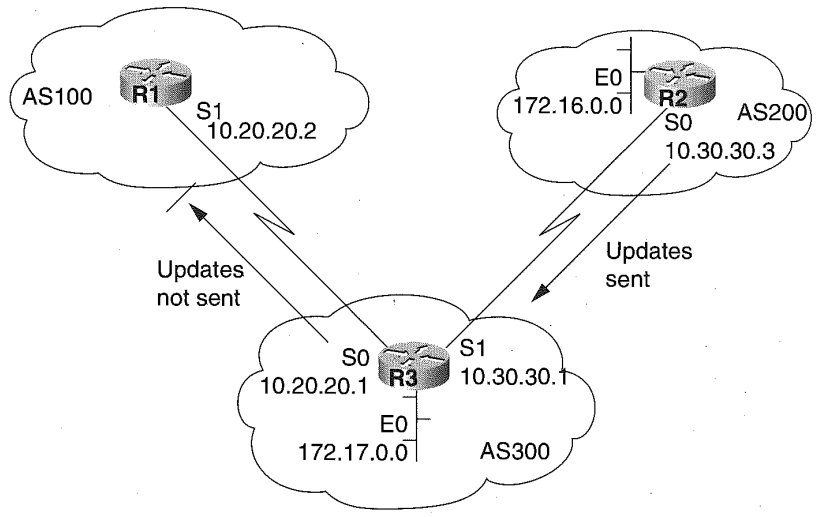


Figure 18-19
Using an access list to control updates to/from BGP neighbors.

Router R3 configuration applied for Figure 18-19:

```
router bgp 300
network 172.17.0.0
neighbor 10.30.30.3 remote-as 200
neighbor 10.20.20.2 remote-as 100
neighbor 10.20.20.2 distribute-list 1 out
access-list 1 deny 172.16.0.0 0.0.255.255
access-list 1 permit 0.0.0.0 255.255.255.255
```

The router R3 configuration filters outbound routing updates to the neighbor identified by IP address 10.20.20.2 using access-list number 1. This is denoted on the neighbor 10.20.20.2 distribute-list 1 out command. The access-list 1 defined in router R3 first denies all routes specific to the 172.16.0.0 network. Any network IP address that does not meet this criterion is then permitted by using the access-list 1 permit 0.0.0.0 255.255.255.255 command.

In many BGP network configurations, supernets (CIDR) addressing is used. Taking the above example, suppose router R2 has many different 172.16.0.0 subnets and that the network 172.16.0.0 is a CIDR network with a mask of 255.240.0.0, creating a supernet. To meet the original objective, the filter must be applied to an 8-bit mask or 172.0.0.0 network with a 255.0.0.0 subnet mask. The first impulse is to apply an access-list permit command to the 172.0.0.0 network with a 0.255.255.255 mask applied, but this permits all the network addresses beginning with 172.0.0.0. Further

filtering must be applied to identify the specific network address when using supernets. This is done using the extended access list. In applying the extended access-list permit, only the 172.0.0.0 8-bit subnet mask network must be applied:

```
access-list 101 172.0.0.0 0.255.255.255 255.0.0.0 0.0.0.0
```

Using this access list and specifying the neighbor distribute list command that identifies the 101 access list, only the network routing updates for 172.0.0.0 255.0.0.0 are permitted. All other networks are denied. For more information on extended access lists, consult Chapter 13, "IP Configuration."

The route filtering controls routing updates to specific neighbors based on network numbers. Path filtering applies filters for both inbound and outbound routing updates based on autonomous system paths in a BGP network configuration. Path filtering is applied using the Cisco IOS feature called regular expressions.

Regular expressions are a means of defining a pattern or a string that must match the given input. For BGP, the string consists of the path information. Table 18-2 lists the variable uses in a regular expression.

Using regular expressions path filtering can be applied to the whole AS. For example, in Figure 18-20, a network topology illustrates four autonomous systems serially connected. Suppose, as shown the figure, that AS100 does not require updates about the 172.16.0.0 (160.10.0.0) network. Using access lists again in the router R3 configuration, we can stop routing updates with AS200 as the origin from entering AS100. This path filtering is accomplished using two configuration commands. The first of these is the ip as-path access-list global command. The format of this command is

ip as-path access-list *access-list-number* **{permit | deny}** *as-regular-expression*

The *access-list-number* variable value points to a regular expression access list number that defines the filtering. The **permit** keyword grants access on the matching condition, while the **deny** keyword stops access on the matching condition. The *as-regular-expression* variable is the matching criteria used in the access list of the autonomous system. The **access-list** filter can be applied to both inbound and outbound flows. The Cisco IOS process examines the regular expression specified on the **ip as-path access-list** command against the autonomous system path of the route as an ASCII string. If there is a match, the defined condition is applied. An autonomous system path will not have the AS of the local router.

The second router configuration command required to use path filtering is the **neighbor filter-list** command. This command follows the **router bgp** command to which the filter is to be applied. The format of the command is

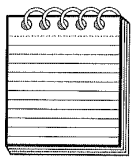
Table 18-2

Regular Expressions

Ranges	Ranges are written as a sequence of characters that are contained within left and right square brackets, such as [10ABC].
Atoms	An atom is a single character being matched in the following manner: . — The character in this position matches any single character ^ — The characters match the beginning of the input string $ — The characters match the end of the input string \ — An exact match to the character _ — The character matches a comma (,), left brace ({), right brace (}), the beginning of the input string, the end of the input string, or a space e.g.: _400_ routes via AS400 ^400$ routes with AS400 as the origin ^400.* routes coming from AS400 ^$ routes originating from this autonomous system
Pieces	A piece is an atom followed by either a * — matching 0 or more sequences of the atom, such as a*, in which there is the letter a or a followed by anything else + — matches 1 or more sequences of the atom, such as a+, in which one occurrence of a or more should be found ? — matches the entire atom or the null string, such as ab?a, which matches abaa or aba
Branch	A branch is a concatenation of pieces.

neighbor {*ip-address* | *peer-group-name*}
filter-list *access-list-number* {**in** | **out** | **weight** *weight*}

The *ip-address* variable is the IP address of the neighbor to which the filter is to be applied. The *peer-group-name* variable can be used instead of the *ip-address* variable. If the *peer-group-name* variable is used, then the filter is applied to all the neighbors associated with the peer-group. The *access-list-number* variable is the access-list number defined on the **ip as-path access-list** command. The **in** keyword applies the filter to the incoming routes and the **out** keyword applies the filter to the outbound routes. The **weight** keyword and its associated *weight* variable allow the router administrator to assign a value from 0 to 65,535 to incoming routes matching the AS paths. This value provides a relative importance to the matching incoming AS path.



NOTE: The weight values specified on the **neighbor weight** *and* **neighbor filter-list** *commands are overridden by the weight values specified on the* **match as-path** *and* **set weight** *route-map commands.*

Following through on the supposition, router R3 in Figure 18-20 is configured as follows to prevent R3 from sending updates on the 172.16.0.0 network to router R1.

Configuration for router R3 in Figure 18-20:

```
router bgp 300
neighbor 10.30.30.3 remote-as 200
neighbor 10.20.20.2 remote-as 100
neighbor 10.20.20.2 filter-list 1 out
ip as-path access-list 1 deny ^200$
ip as-path access-list 1 permit .*
```

The access-list number referred in the neighbor filter-list command identifies the ip as-path access list to be applied to the routing updates to or from the identified neighbor. The configuration denies any updates that have AS path information that starts with 200 (^200) and ends with 200 (200$). Updates for the 172.16.0.0 network sent from router R2 having path information beginning and ending with 200 match the filter and are then denied.

The ip as-path permit command in router R3 configuration uses the ".*" regular expression. The use of the dot (.) indicates that any character will be a match. The "*" represents the repetition of the character found in the dot position. These characters together indicate that any path information is allowed. This statement is needed to override the implicit deny found for all access lists.

Verifying the paths that have matched a regular expression for BGP is possible using the command, **sh ip bgp regexp** *regular-expression.* The *regular-expression* value is the expression used on the ip as-path access-list command. In the above example, to verify that the AS paths beginning and ending with 200 are being filtered, the command would be entered as **sh ip bgp regexp ^200$**.

The **ip as-path access-list** command can also be used in conjunction with the **neighbor route-map** command when matching routing information using the **match as-path** statement. The topology shown in Figure 18-21 illustrates this combination.

Suppose in Figure 18-21 router R3 is to learn only of local networks of AS200. In addition, the weight of the permitted routes are set to 20. The

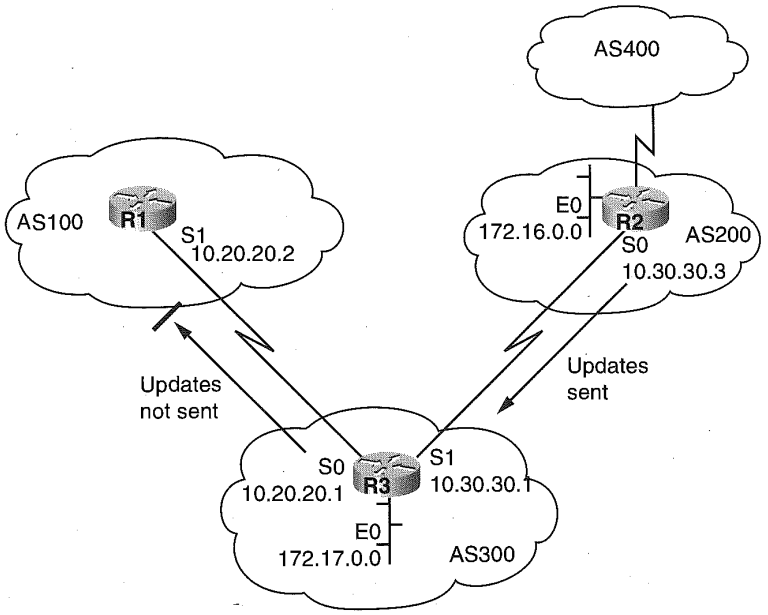


Figure 18-20
A path-filtering topology example for controlling updates to/from BGP neighbors.

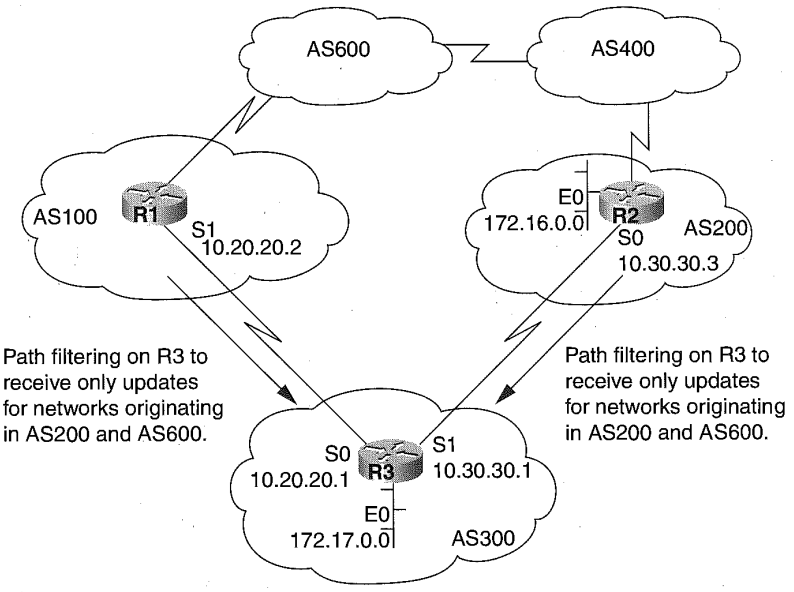


Figure 18-21
Network topology for using neighbor route-map and ip as-path access-list commands to control BGP routing updates.

filter to perform this task is applied to router R3 and has the following configuration:

Configuration for router R3 of Figure 18-21:

```
router bgp 300
network 172.17.0.0
neighbor 10.30.30.3 remote-as 200
neighbor 10.30.30.3 route-map only200 in
route-map only200
match as-path 1
set weight 20
ip as-path access-list 1 permit ^200$
```

Updates that originate from AS200 will have the AS path information starting and ending with 200. Applying the ip as-path access-list filter with ^200$ as the regular expression ensures that only this path is permitted. All other updates are dropped. The ip as-path access-list is identified under the route-map named only200 by the match as-path statement. The only200 route-map is identified by the **neighbor route-map** command under router bgp. The routes accepted will be given a weight of 20.

To complicate the scenario, all routes other than paths originating in AS200 will be given a weight of 10 and updates originating from AS400 are to be dropped. Again the filter is applied to router R3 in the following manner:

Configuration for router R3 in Figure 18-21 dropping AS400 and applying a weight of 10 to all other paths:

```
router bgp 300
network 172.17.0.0
neighbor 10.30.30.3 remote-as 200
neighbor 10.30.30.3 route-map drop400 in
route-map drop400 permit 10
match as-path 1
set weight 20
route-map drop400 permit 20
match as-path 2
set weight 10
ip as-path access-list 1 permit ^200$
ip as-path access-list 2 permit ^200 600 .*
```

In this expanded example, paths originating from AS200 are still set to a weight of 20, but all other permitted paths are set to a weight of 10. Setting the other permitted paths to a weight of 10 is accomplished by the second instance of the drop400 route-map specification. The second instance of the drop400 route-map applies ip as-path access-list number 2 to the AS path information. This filter permits only paths that begin with 200 followed by 600, thereby causing router R3 to accept paths for updates behind AS400 while dropping AS400 paths.

Manipulating the Path Information Order

The BGP decision process can be influenced through changing the order of the AS path information in an update. This is a common practice to influence path load balancing. The path information of a BGP update can be manipulated using the set as-path prepend command following a route-map statement. The format of the set as-path prepend command is

set as-path {tag | prepend *as-path-string*}

The **tag** keyword directs the IOS to convert the tag associated with a route into an AS path. The **tag** keyword can only be used for redistributing routes into BGP. The **prepend** keyword and its associated variable named *as-path-string* appends the value of *as-path-string* to the AS path information of the route matched by the route-map definitions. The changing of the AS path in this manner applies to both inbound and outbound BGP route maps. The **set as-path** command artificially increases the length of the AS path information.

In Figure 18-22, AS300 is getting all its traffic from AS100 because the AS path length received from AS100 is shorter (100, 300) than the AS path length received from AS400 (400, 200, 300). This is verified by interrogating router R3 in Figure 18-22. Router R3 advertises its local network 172.17.0.0 to both AS100 and AS200. AS600 receives this information and determines reachability about 172.25.0.0 via two routes. The first is from AS100 and the second from AS400. If all the other attributes are the same, AS600 will use the shortest path to router R3, in this case the route through AS100. Although this makes sense from a BGP-routing perspective, it may not be preferred based on bandwidth availability and router performance.

Suppose in Figure 18-22 the bandwidth connecting AS100 to AS300 is 56 Kbps and the router R1 is a Cisco 4500 series router. The physical connection from AS100 to AS600 is also a 56-Kbps line, but router R6 is a Cisco 7500 series router using an RSP4 and VIP-2 processors. The connection from AS600 through AS400 to AS200 through to router R3 in AS300 is T3 (45 Mbps) with low utilization. In such a case, it is prudent to manipulate the AS path in such a way as to increase the AS length of the path received from AS100 to a length greater than that received from AS200. The configuration used in router R3 to produce this desired effect is as follows:

```
router bgp 300
network 172.17.0.0
neighbor 10.20.20.2 remote-as 100
neighbor 10.20.20.2 route-map prefer-AS400 out
route-map prefer-AS400
set as-path prepend 300 300
```

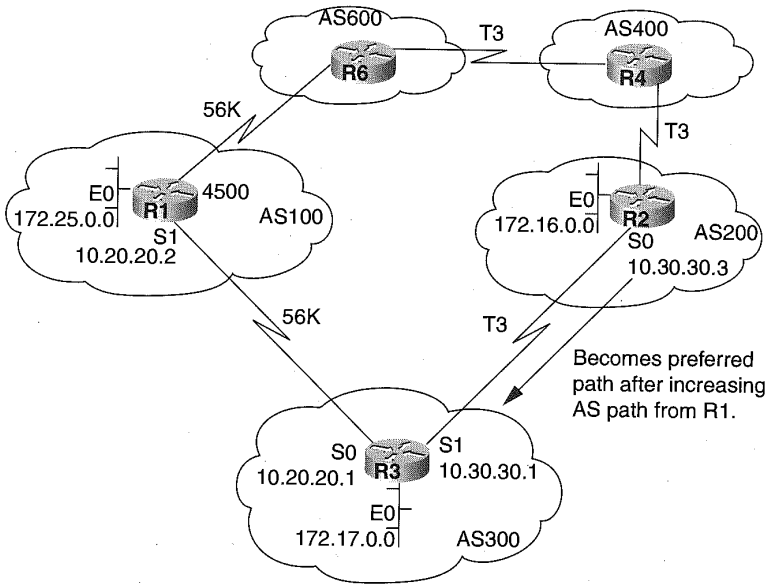


Figure 18-22
Network topology for using the set as-path prepend command to increase the AS path length.

A common way of increasing the AS path is to replicate the local AS number in the path information. This is done in the above example using the **set as-path prepend 300 300** command. By implementing this statement, AS600 receives updates on network 172.17.0.0 from AS100 with path information such as 100, 300, 300, 300. This now creates a path length longer than the natural path length generated from AS400 (400, 200, 300).

Using BGP Peer Groups

In many of the previous discussions, the peer-group-name variable is discussed on many of the commands affecting BGP. A BGP peer group is a list of BGP neighbors having the same update policy requirements. Peer groups enable the router administrator to assign the same update policies, such as route maps, distribute-lists, and filter-lists, to the group, rather than defining the same policies to the individual neighbors. The peer group is created using the neighbor peer-group routing configuration command. The format of the command is

neighbor *peer-group-name* **peer-group**

The *peer-group-name* variable is the name assigned to the peer group. Neighbors are assigned to a peer group using the following router configuration command:

neighbor *ip-address* **peer-group** *peer-group-name*

The *ip-address* variable is the value of the IP address for a BGP neighbor belonging to the peer group. The *peer-group-name* variable following the **peer-group** keyword is the BGP peer group to which the neighbor is a member.

Each member of the peer group inherits the following configuration options:

```
remote-as (only if the peer-group-name is used on a neighbor
remote-as command)
version
update-source
out-route-map
out-filter-list
out-dist-list
minimum-advertisement-interval
next-hop-self
```

Using Figure 18-23, peer groups can be applied for internal and external BGP neighbors. When defining internal BGP peer groups, the neighbor peer-group-name remote-as command format is useful since the configuration options are affecting only the routers within the internal BGP network. For example, router R3 in Figure 18-23 can be configured using a BGP peer group and applies filter-lists to inbound and outbound updates. The filters being applied are not pertinent to the example so they are not coded. However, they are included to illustrate that the lists affect the entire peer group.

Configuration for router R3 for internal BGP neighbors in Figure 18-23:

```
router bgp 300
neighbor IBGPmap peer-group
neighbor IBGPmap remote-as 300
neighbor IBGPmap route-map SETMETRIC out
neighbor IBGPmap filter-list 1 out
neighbor IBGPmap filter-list 2 in
neighbor 10.50.50.2 peer-group IBGPmap
neighbor 10.60.60.2 peer-group IBGPmap
neighbor 10.30.30.2 peer-group IBGPmap
neighbor 10.30.30.2 filter-list 3 in
```

In the sample configuration for router R3, the peer group named IBGPmap has enlisted routers R5, R6, and R7 as members of the peer

group. A neighbor peer-group command is used to associate each internal BGP neighbor with the IBGPmap peer group. A separate filter list is defined for neighbor 10.30.30.2 (router R5) that will override the filter-list 2 applied to the entire peer group. When using peer groups, options can only be overridden that affect inbound updates.

External BGP peer groups can also be defined to further control the effects of options to multiple neighbors. In the following router configuration example, router R3 has defined external BGP neighbors into a peer group named EBGPmap. The **neighbor** *peer-group-name* **remote-as** command format is not used in defining external BGP peer groups because multiple autonomous system numbers must be defined.

Configuration for router R3 using external BGP peer group definition applied to Figure 18-23:

```
router bgp 300
neighbor EBGPmap peer-group
neighbor EBGPmap route-map SETMETRIC
neighbor EBGPmap filter-list 1 out
neighbor EBGPmap filter-list 2 in
neighbor 10.20.20.2 remote-as 100
neighbor 10.20.20.2 peer-group EBGPmap
neighbor 10.40.40.2 remote-as 600
neighbor 10.40.40.2 peer-group EBGPmap
neighbor 10.10.10.2 remote-as 200
neighbor 10.10.10.2 peer-group EBGPmap
neighbor 10.10.10.2 filter-list 3 in
```

For each external BGP neighbor, a **neighbor remote-as** and **neighbor peer-group** command pair is specified.

Aggregate Addresses

BGP4 is the only version of BGP that supports Classless Interdomain Routing (CIDR). CIDR removes the notion of IP class type addressing by allowing what is termed supernets. Supernets are mostly used on Class B and Class C address format. For example, the class C address of 192.168.200.0 with a 255.255.255.0 subnet mask can become a supernet using just 192.168.0.0 with a 255.255.0.0 subnet. Using CIDR supernets is very useful in reducing the size of the routing updates, and hence the routing tables.

BGP allows for network address aggregation by propagating the most common bits of an IP network address together so that a single route is advertised for building routing tables to the supernet. Using Figure 18-24, the networks originating from AS200 router R2 are 172.16.0.0, 172.17.0.0, and 172.30.0.0. Aggregate addressing and the notion of supernets allows

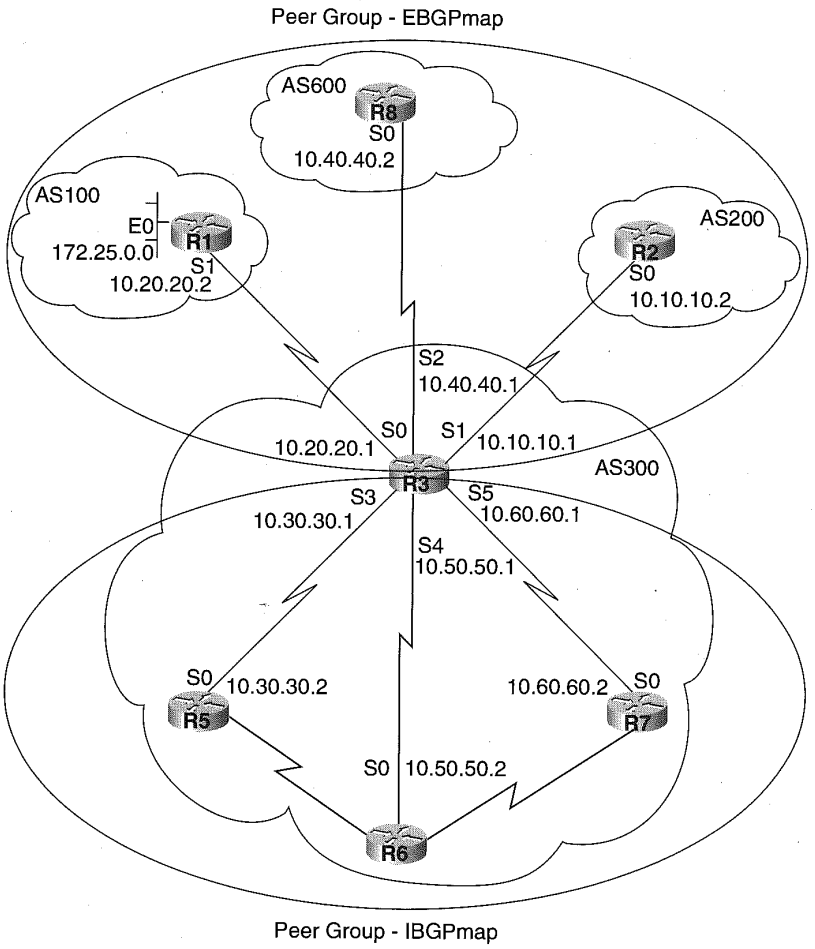


Figure 18-23
Network topology for defining a BGP peer group example.

the router administrator to reduce the three route advertisements into the 172.0.0.0 supernet.

In Figure 18-24, router R2 sends routing updates for the 172.16.0.0, 172.17.0.0, and 172.30.0.0 networks. Router R3 can be configured as follows to forward a supernet route advertisement of the 172.0.0.0 network to router R1. The following configuration demonstrates the specification of simple aggregate addressing:

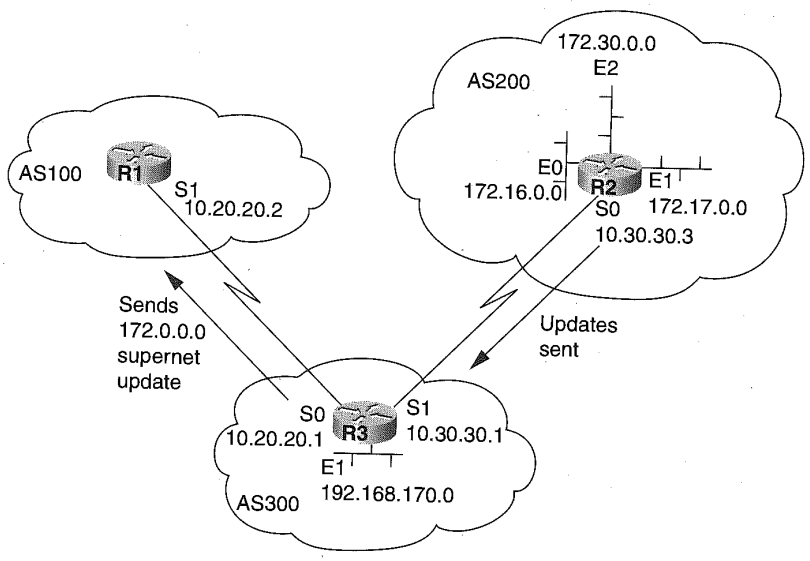


Figure 18-24
A simple topology for BGP route aggregation.

Router R2 configuration applied to Figure 18-24:

```
router bgp 200
neighbor 10.30.30.1 remote-as 300
network 172.16.0.0
network 172.17.0.0
network 172.30.0.0
```

Router R3 Configuration applied to Figure 18-24:

```
router bgp 300
neighbor 10.30.30.3 remote-as 200
neighbor 10.20.20.2 remote-as 100
network 192.168.170.0
aggregate-address 172.0.0.0 255.0.0.0
```

The above sample configuration uses the simplest form of the aggregate-address command. The full format of the command is as follows:

aggregate-address *address mask* **[as-set] [summary-only] [suppress-map** *map-name*] **[advertise-map** *map-name*] **[attribute-map** *map-name*]

The *address* variable is the aggregate or supernet address being used to summarize the routes. The *mask* variable is the bit mask applied to the *address* variable to determine the supernet. Specifying only the *address*

and *mask* variables on the command causes BGP to advertise the supernet, also called the prefix route in addition to the more specific routes.

In the previous example, not only will the 172.0.0.0 network be advertised but also the more specific 172.16.0.0, 172.17.0.0 and 172.30.0.0 networks are advertised to router R1. Aggregate addressing is only possible if a more specific route is found in the BGP routing table. For example, router R2 cannot aggregate the 172.x.x.x networks if it does not have a more specific entry for the network in its own BGP routing table. Specific entries can be included in the BGP routing table from either incoming routing updates received from other autonomous systems, the redistribution of IGP static routes into BGP, or using the network command.

Based on this discussion, the above configuration example did not reduce routing updates or the routing table but actually increased it by one route. To accomplish the actual reduction of routing updates and minimizing the size of the BGP routing table, the **summary-only** keyword is applied to the **aggregate-address** *address mask* command. Specifying the **aggregate-address** command with the **summary-only** keyword is formatted as

aggregate-address *address mask* **summary-only**

where the *address* variable is the IP address of the network to aggregate and the *mask* is the bit mask applied to the *address* variable. The **summary-only** keyword causes BGP to advertise only the prefix address and not the more specific addresses. Attention, however, must be given to how the networks are included in BGP. Defining networks on the **network** statement causes the network address to be included in BGP updates even when the **summary-only** keyword is specified.

Including the optional **as-set** keyword causes the BGP process to create AS set path information for both the prefix address advertisement and the more specific routes. The format for using the **as-set** keyword is

aggregate-address *address mask* **as-set**

again where *address* is the IP address of the supernet and the *mask* is the bit mask applied to the *address* variable.

The **as-set** keyword of the aggregate-address command is used primarily on aggregate route information where the path attribute has lost information due to the function of the AS-SETS process when using aggregation. The AS-SETS process reduces the path information length by including an AS number only once, even though it may be in multiple paths used to build the aggregate.

In Figure 18-25, router R3 is to aggregate using network 172.0.0.0 255.0.0.0 and send the path to router R4. Router R4 will not be aware of the

origin of this route if router R3 does not use the **as-set** keyword on the aggregate address command to generate path information as a set. The set includes all the path information regardless of which path was received first.

The router configuration used to accomplish the desired outcome is as follows:

Router R1 configuration applied to Figure 18-25:

```
router bgp 100
network 172.20.0.0
neighbor 10.20.20.1 remote-as 300
```

Router R2 configuration applied to Figure 18-25:

```
router bgp 200
network 172.16.0.0
neighbor 10.30.30.1 remote-as 300
```

Router R3 configuration applied to Figure 18-25:

```
router bgp 300
neighbor 10.30.30.3 remote-as 200
neighbor 10.20.20.2 remote-as 100
neighbor 10.40.40.4 remote-as 400
aggregate-address 172.0.0.0 255.0.0.0 summary-only
aggregate-address 172.0.0.0 255.0.0.0 as-set
```

Applying the above configurations, router R3 sends updates on the supernet 172.0.0.0 255.0.0.0 to router R4 with the AS-SET of ({100 200}) included in the path information. Without configuring the aggregate-address as-set command in router R3, router R4 would not be aware that the 172.0.0.0 supernet is actually originating from two different autonomous systems. Using the **as-set** keyword on a supplementary **aggregate-address** command protects router R4 from creating a loop to AS100 through an unknown backdoor connection.

The **suppress-map** keyword followed by its *map-name* variable enables the suppression of more specific routes when performing aggregate address summarization using a route map that is applied to the aggregate entry. For example, in Figure 18-25, when using **suppress-map** on the aggregate-address with associated route-map and access-list statements, the supernet 172.0.0.0 and 172.16.0.0 networks can be advertised while the specific routes for 172.20.0.0 can be suppressed. The following configuration demonstrates this for router R3:

Router R3 configuration suppressing the 172.20.0.0 network:

```
router bgp 300
neighbor 10.30.30.3 remote-as 200
```

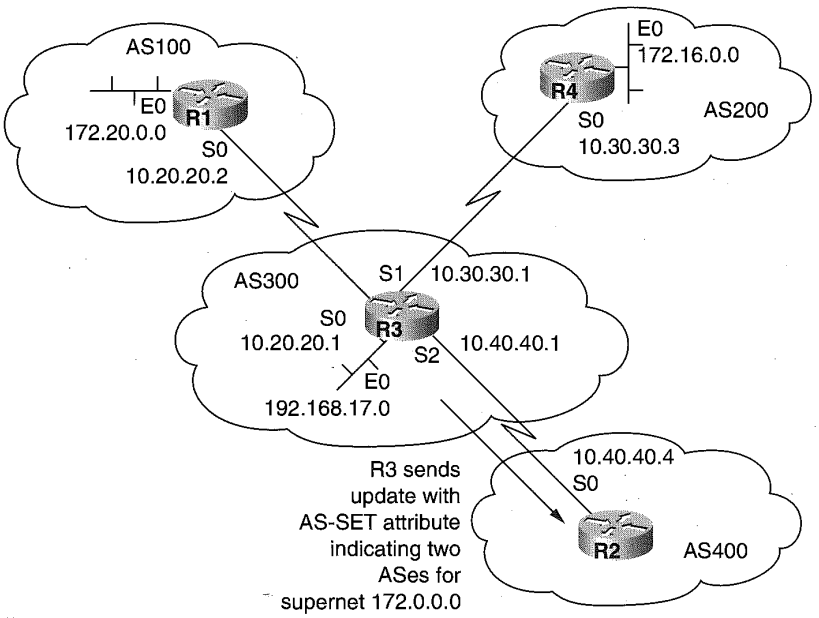


Figure 18-25 Network topology used for **as-set** aggregate command.

```
neighbor 10.20.20.2 remote-as 100
network 192.168.17.0
aggregate-address 172.0.0.0 255.0.0.0 suppress-map NO172-20

route-map NO172-20 permit 10
match ip address 1
access-list 1 deny 172.20.0.0 0.0.255.255
access-list 1 permit 0.0.0.0 255.255.255.255
```

The **advertise-map** keyword and its associated *map-name* identify the route map used for selecting routes when creating AS-SET origin communities. The **attribute-map** keyword followed by its *map-name* variable value identifies the route map used for setting the various attributes of the advertised aggregate route.

Grouping Multiple Autonomous Systems into a BGP Confederation

Many Internet Service Provider (ISP) networks have large IBGP mesh configurations within their AS. Using the concept of a confederation, a large AS

can be divided into smaller autonomous systems, grouped together as a single confederation that is assigned its own AS. Each of the new autonomous systems has an IBGP mesh with connections to the other new autonomous systems in the confederation.

Because they belong to a confederation, the EBGP peering between the new autonomous systems exchanges their routing updates as if they are connected using IBGP. This means that next-hop, metric, and local preference information is passed between the EBGP peers of the confederation, even though normally EBGP connections do not propagate this information. Connections from networks to the confederation use the confederation AS number despite the physical connection. Figure 18-26 illustrates a BGP confederation topology.

A BGP confederation is built using two bgp commands. The first of these defines the AS number associated with the confederation. The format for this command is

bgp confederation identifier *autonomous-system*

The value specified for the *autonomous-system* variable of the **bgp confederation identifier** command is the AS number associated with the BGP confederation. The value can range from 1 to 65,535 and must be unique within the network.

The second required definition for using BGP confederation is the bgp confederation peers command. BGP peers become members of the confederation by their specification of the bgp confederation peer command in their router configuration. The format of the command is

bgp confederation peers *autonomous-system* [*autonomous-system*]

The *autonomous-system* variable(s) identifies the BGP AS numbers of routers belonging to the confederation defined under the router bgp command.

In Figure 18-26, the AS500 consists of nine BGP routers with EBGP connections to other AS's. Creating a full mesh within AS500 requires nine peer connections for each router broken down into eight IBGP peers and one EBGP peer to an external AS. Using a confederation, the AS500 can safely be divided into three AS's, each having a full IBGP mesh. The following illustrates the BGP confederation configuration for routers R1, R3, and R4 in Figure 18-26:

Router R1 configuration for defining BGP confederations in Figure 18-26:

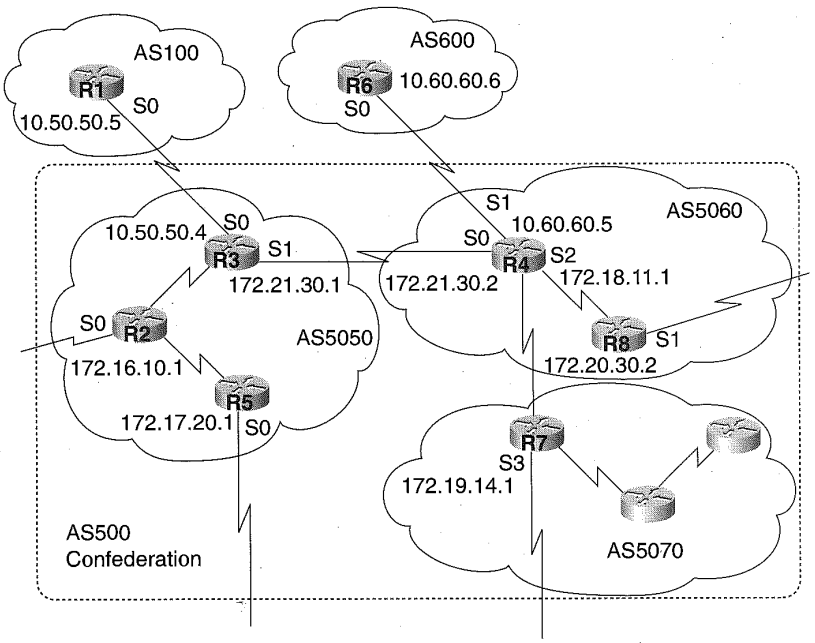


Figure 18-26
A BGP confederation topology.

```
router bgp 100
neighbor 10.50.50.4 remote-as 500
```

Router R3 configuration for defining BGP confederations in Figure 18-26:

```
router bgp 5050
bgp confederation identifier 500
bgp confederation peers 5060 5070
neighbor 172.16.10.1 remote-as 5050
neighbor 172.17.20.1 remote-as 5050
neighbor 172.18.11.1 remote-as 5060
neighbor 172.19.14.1 remote-as 5070
neighbor 10.50.50.5 remote-as 100
```

Router R4 configuration for defining BGP confederations in Figure 18-26:

```
router bgp 5060
bgp confederation identifier 500
bgp confederation peers 5050 5070
neighbor 172.20.30.2 remote-as 5060
neighbor 172.21.30.1 remote-as 5050
neighbor 172.19.14.1 remote-as 5070
neighbor 10.60.60.6 remote-as 600
```

Router R1 does not know of the new AS's 5050, 5060, and 5070, which were created for the confederation. Router R1 contacts peers using the 500 AS number. The 500 is the BGP confederation identifier defined in router R3 and R4. The bgp confederation peers command for router R3 identifies AS 5060 and 5070 as being a part of the confederation using identifier 500. Router R4 defines AS 5050 and 5070 as members of the confederation using the same identifier 500.

Reducing IBGP Peering Using Route Reflectors (RR)

Internal BGP peering does not have IBGP peers forward advertised routes learned by other IBGP peers to a third IBGP peer. IBGP peers providing learned routes to other IBGP routers within the AS are called route reflecting. An IBGP router acting as a route reflector (RR) performs a function akin to a hub and spoke configuration. The router using the RR feature has peer connections with all the routers in the AS and updates their tables. The non-RR routers have peer connections only with the RR router, as illustrated in Figure 18-27.

In this simple configuration, the normal full IBGP mesh for connecting router R1, R2, and R3 in AS100 is not needed because router R3 is performing the RR function. Router R1 and R2 do not peer with each other. Instead, they learn of the routes from only router R3. Router R3 "reflects" routing updates received from R1 to R2 and updates from R2 to R1. The router acting as the RR refers to the neighbors receiving updates from it as clients. The combination of the RR and its clients is called a cluster.

In Figure 18-27, router R1, R2, and R3 form the cluster with only router R3 as the single RR in the AS. IBGP peers of the RR that are not defined as clients are called non-clients. A router becomes an RR by including in its router bgp configuration the **neighbor route-reflector-client** command. The format of this command is

neighbor *ip-address* **route-reflector-client**

The *ip-address* variable value is the IP address of a BGP neighbor being established as a client of the router. Specifying this command causes the router to act as a route reflector.

More than one RR can exist in an AS, and if more than one RR is present then there can be multiple clusters. A RR communicates with other RRs as if they were IBGP peers. The multiple RRs can be defined in the same cluster

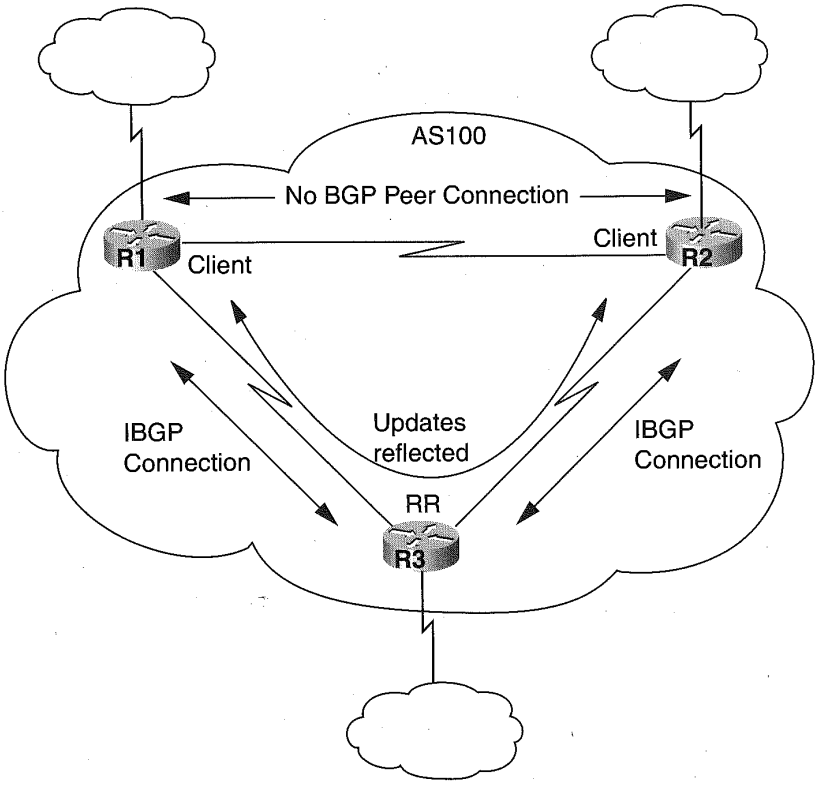


Figure 18-27 Simple network topology for defining RR.

or to other clusters. For example, in Figure 18-28, router R1, R2, and R3 are defined as a single cluster and R3 is the RR for the cluster. A second cluster is defined by the RR definition in router R4 whose clients are R5 and R6. Router R7 is an RR unto itself, forming a third cluster within the AS. The three RRs are fully meshed. The clients of each cluster, however, do not peer with the clients of other clusters.

When a RR receives a route from a non–client peer, it reflects the route to all the clients of its cluster. If the RR receives a route from a client peer, the route is reflected to all the non-client peers as well as the client peers. Finally, if the route received by a RR is from an external BGP peer, the RR sends the update to all client and non-client peers.

The following router configurations highlight the definitions for the BGP routers R2, R3, and R4 of Figure 18-28.

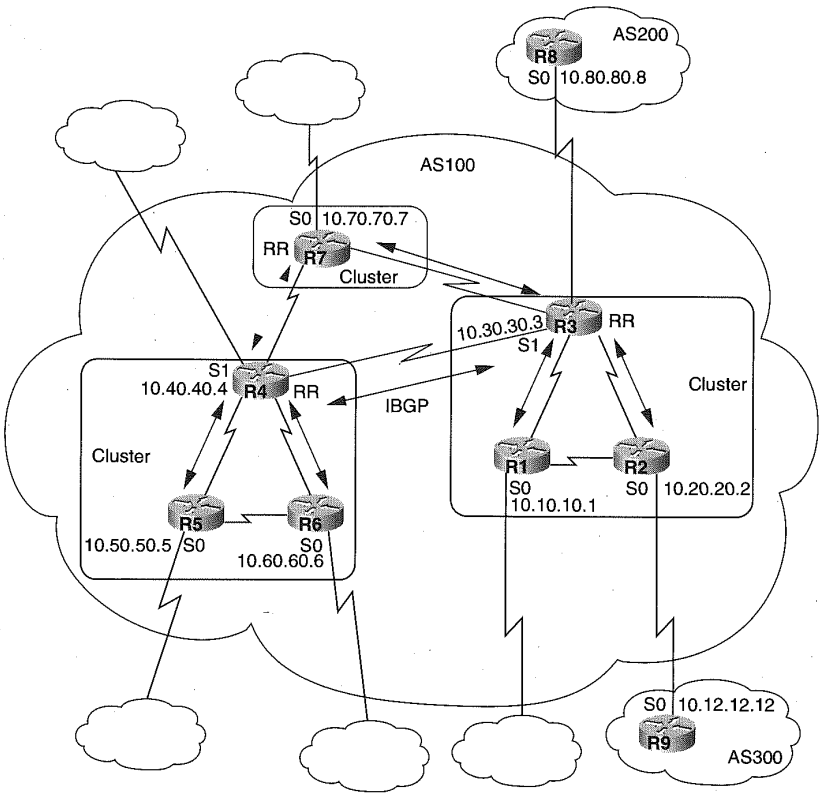


Figure 18-28
Multiple RR network topology.

Router R2 configuration applied to Figure 18-28:

```
router bgp 100
neighbor 10.30.30.3remote-as 100
neighbor 10.12.12.12 remote-as 300
```

Router R3 configuration applied to Figure 18-28:

```
router bgp 100
neighbor 10.20.20.2 remote-as 100
neighbor 10.20.20.2 route-reflector-client
neighbor 10.10.10.1 remote-as 100
neighbor 10.10.10.1 route-reflector-client
neighbor 10.70.70.7 remote-as 100
neighbor 10.40.40.4 remote-as 100
neighbor 10.80.80.8 remote-as 200
```

Router R4 configuration applied to Figure 18-28:

```
router bgp 100
neighbor 10.60.60.6 remote-as 100
neighbor 10.60.60.6 route-reflector-client
neighbor 10.50.50.5 remote-as 100
neighbor 10.50.50.5 route-reflector-client
neighbor 10.70.70.7 remote-as 100
neighbor 10.30.30.3 remote-as 100
```

Because routes are reflected, it is possible the learned internal BGP routes will end up causing a routing information loop. The schema used for route reflecting, however, employs two methods to avert a routing information loop. The first method uses the originator-id optional attribute that is four bytes in length and is created by the RR. The attribute value is actually the route-id of the route originator in the local AS. This enables the route originator to determine if the route has come back to itself, thereby enabling the originator of the route to ignore the routing update.

The second method of route reflecting is the use of cluster lists. The cluster list is also a non-transitive BGP optional attribute as well as a sequence of cluster-ids that a route has passed through. A RR will append its local cluster ID to the cluster list when reflecting a route from a client to non-client peers.

Using a cluster list, a RR can determine if a loop exists by discovering its own cluster ID in the cluster list of the router advertisement. If the RR finds its own cluster ID in the cluster list, the advertisement is ignored. If the cluster list is empty, the RR creates begins a new cluster list. Cluster IDs are defined using the bgp cluster-id command. The format of the command is

bgp cluster-id *cluster-id*

where the cluster-id variable value is the ID assigned to the cluster by the RR. The value can be up to four bytes.

The cluster-id is important in configurations employing redundancy to avoid a single point of failure for the RR of a cluster. Pictured in Figure 18-29 is such a network topology. Multiple RRs are defined within a cluster to provide redundancy. Each RR of a cluster must use the same cluster-id in order to utilize the cluster-list method of avoiding loops.

Routers R4, R5, R6, and R8 in Figure 18-29 are part of a single cluster in which R4 and R8 provide the services of a RR for the cluster. Each RR in the cluster must use the same cluster-id in order to determine if a route advertisement is causing a loop. In the following router configuration examples, router R4 and R8 define a cluster-id of 1111. Router R3 does not have a cluster ID defined since it is the only RR within its cluster.

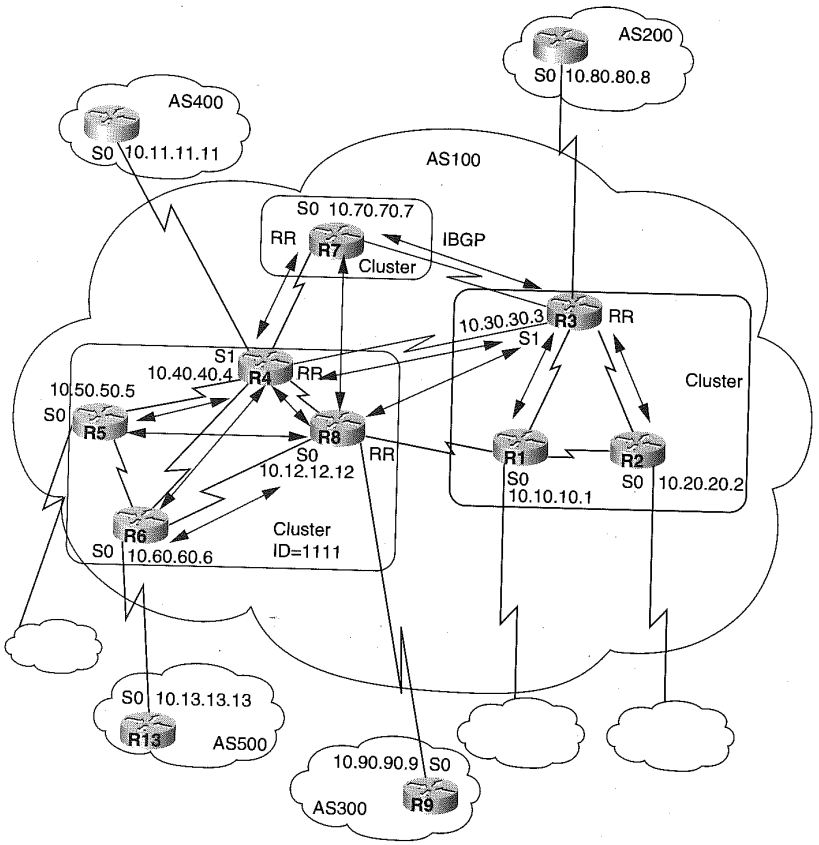


Figure 18-29
Multiple RRs in a cluster to avoid a single point of failure.

Router R3 configuration applied to Figure 18-29:

```
router bgp 100
neighbor 10.10.10.1 remote-as 100
neighbor 10.10.10.1 route-reflector-client
neighbor 10.20.20.2 remote-as 100
neighbor 10.20.20.2 route-reflector-client
neighbor 10.40.40.4 remote-as 100
neighbor 10.70.70.7 remote-as 100
neighbor 10.12.12.12 remote-as 100
neighbor 10.80.80.8 remote-as 200
```

Router R4 configuration applied to Figure 18-29:

```
router bgp 100
neighbor 10.12.12.12 remote-as 100
```

```
neighbor 10.50.50.5 remote-as 100
neighbor 10.50.50.5 route-reflector-client
neighbor 10.60.60.6 remote-as 100
neighbor 10.60.60.6 route-reflector-client
neighbor 10.70.70.7 remote-as 100
neighbor 10.30.30.3 remote-as 100
neighbor 10.11.11.11 remote-as 400
bgp route-reflector 1111
```

Router R6 configuration applied to Figure 18-29:

```
router bgp 100
neighbor 10.12.12.12 remote-as 100
neighbor 10.40.40.4 remote-as 100
neighbor 10.13.13.13 remote-as 500
```

Router R8 configuration applied to Figure 18-29:

```
router bgp 100
neighbor 10.40.40.4 remote-as 100
neighbor 10.50.50.5 remote-as 100
neighbor 10.50.50.5 route-reflector-client
neighbor 10.60.60.6 remote-as 100
neighbor 10.60.60.6 route-reflector-client
neighbor 10.70.70.7 remote-as 100
neighbor 10.30.30.3 remote-as 100
neighbor 10.90.90.9 remote-as 300
bgp route-reflector 1111
```

If clients of a cluster are meshed, route reflection is not required. By default, an RR has client-to-client reflection enabled. To disable this, enter the no bgp client-to-client reflection command. It is important to note that peer groups cannot be used if client-to-client reflection is enabled. Any clients of a route reflector cannot therefore be members of a BGP peer group. Having the clients as part of a peer group could cause invalid routes reflected to clients that are not part of the peer group.

In Figure 18-30, a topology is in in which where router R4, R5, and R6 understand router reflection, while routers R1, R2, and R3 are not configurable as RRs. Routers R1, R2, and R3 can be configured in a full IBGP mesh along with R4. When any of the routers R1, R2, or R3 are upgraded to support RR, the other two remaining traditional routers can become clients of the RR. In our example, R3 becomes the RR, and R1 and R2 become the clients of the R3 cluster. Once R3 is made an RR with R1 and R2 as clients, the full IBGP mesh that exists between the three can be removed.

The following router configurations coincide with routers R4 and R3 in Figure 18-30:

Router R3 configuration applied to Figure 18-30:

```
router bgp 100
neighbor 10.40.40.4 remote-as 100
```

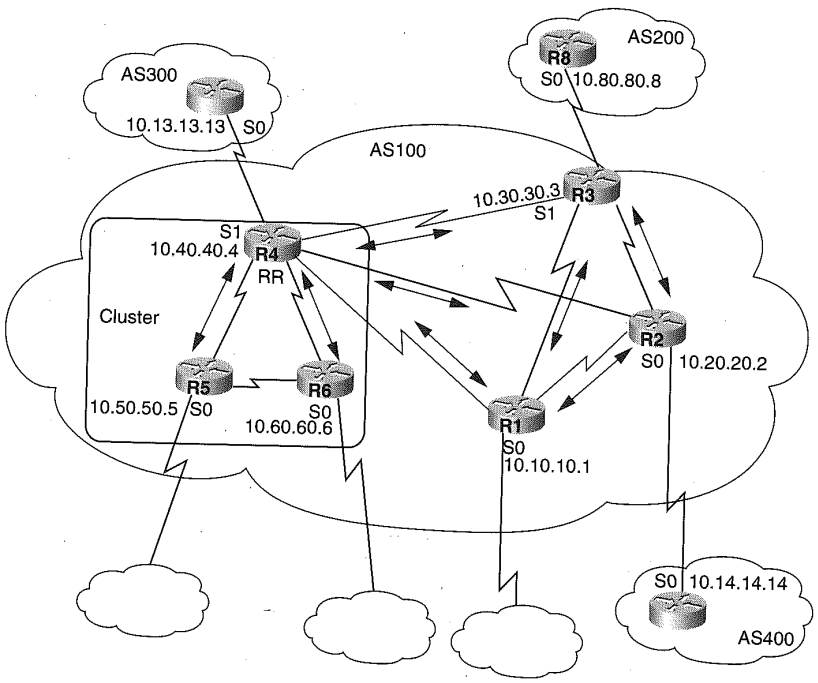


Figure 18-30
RR migration strategy topology.

```
neighbor 10.20.20.2 remote-as 100
neighbor 10.10.10.1 remote-as 100
neighbor 10.14.14.14 remote-as 400
```

Router R4 configuration applied to Figure 18-30:

```
router bgp 100
neighbor 10.60.60.6 remote-as 100
neighbor 10.60.60.6 route-reflector-client
neighbor 10.50.50.5 remote-as 100
neighbor 10.50.50.5 route-reflector-client
neighbor 10.30.30.3 remote-as 100
neighbor 10.20.20.2 remote-as 100
neighbor 10.10.10.1 remote-as 100
neighbor 10.13.13.13 remote-as 300
```

Two features must be watched when employing RRs and avoiding routing loops. The first is the set clause command. When specified for outbound route-maps, it does not affect routes that are reflected to IBGP peers by a RR. The second item is the nexthop-self feature for RRs. Using the nexthop-self on a RR router only affects the nexthop of learned EBGP routes because the nexthop of a RR does not change.

Managing Unstable Routes

Unstable routes can greatly affect network performance. Each time a connection somewhere in the route fails or has an adverse performance, routes can be recalculated and then propagated throughout the network. As of Cisco IOS version 11.0, a feature was introduced to protect BGP from unstable routes known as route dampening.

A route that flip-flops between being available and not available is considered to be a flapping route. When a route flaps for the first time, the IOS BGP process places the route into a history state. When a route is placed in a history state, the route is not perceived to be the best route to the destination. Consecutive route flaps cause the IOS BGP process to place a penalty on the route. A penalty is valued at 1,000 and is cumulative. The value of penalty is stored in the BGP, routing travel around the route in question until the penalty. The penalty is halved after a half-life period. This is performed and adjusted every five seconds in an effort to allow the route to establish itself as a stable route once again.

Over the course of time, the cumulative penalty placed on the route may become greater than a specified suppress limit. The suppress limit is the cumulative total of penalties weighed against a route. Once the suppress limit is exceeded, the state of the route changes from history to damp.

A route can only be considered suppressed for a maximum amount of time. This time defaults to four times the half-life value. In the damp state, the route is no longer advertised to the BGP neighbors. The router IOS places a route back into the BGP table and forwards the route to neighbors once the penalty value has dropped below a reuse-limit level. The process to determine if a route has fallen below the reuse-limit level occurs every 10 seconds. If a route is found to have its penalty below the reuse-limit level, the BGP process advertises the route to all of its neighbors.

This capability to dampen routes is disabled by default. The format for enabling route dampening is

bgp dampening [*half-life reuse suppress max-suppress-time*] [**route-map** *map*]

Specifying only **bgp dampening** enables route dampening using default values. The *half-life* optional variable defaults to 15 minutes. The *half-life* period default is specified when this positional variable is specified. The value for *half-life* ranges from one to 45 minutes.

The optional *reuse* variable defines the reuse-limit value for determining the penalty value under which the route can be unsuppressed. The *reuse* value can range from 1 to 20,000 and defaults to 750.

The *suppress* variable is the value at which the cumulative penalty must exceed to change the state of the route from history to damp. The range for the *suppress* variable is 1 to 20,000 and defaults to 2,000. The *max-suppress-time* variable value is defined in minutes and defaults to the half-life value multiplied by four. The *max-suppress-time* value can range from 1 to 20,000.

The **route-map** keyword followed by the *map* variable identifies the route map enabling route dampening. If any of the default values for the positional variables *half-life*, *reuse*, *suppress*, and *max-suppress-time* are altered to meet the network requirements, they must all be specified since they are positional variables.

Suppose, as shown in Figure 18-31, router R2 is taking the default values of the bgp dampening command. The two routers, R2 and R4, are defined as follows:

Router R2 configuration applied to Figure 18-31:

```
 interface Serial0
 ip address 10.203.250.2 255.255.255.252
 interface Serial1
 ip address 10.192.208.6 255.255.255.252
router bgp 100
 bgp dampening
 network 10.203.250.0
 neighbor 10.192.208.5 remote-as 300
```

Router R4 configuration applied to Figure 18-31:

```
interface Loopback0
ip address 10.192.208.174 255.255.255.192
interface Serial0/0
 ip address 10.192.208.5 255.255.255.252
router bgp 300
 network 10.192.208.0
 neighbor 10.192.208.6 remote-as 100
```

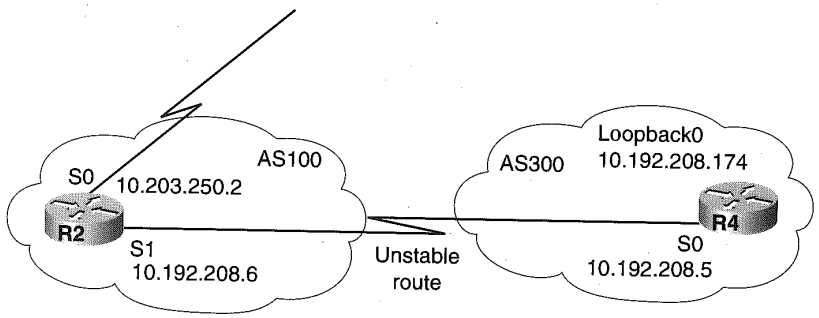
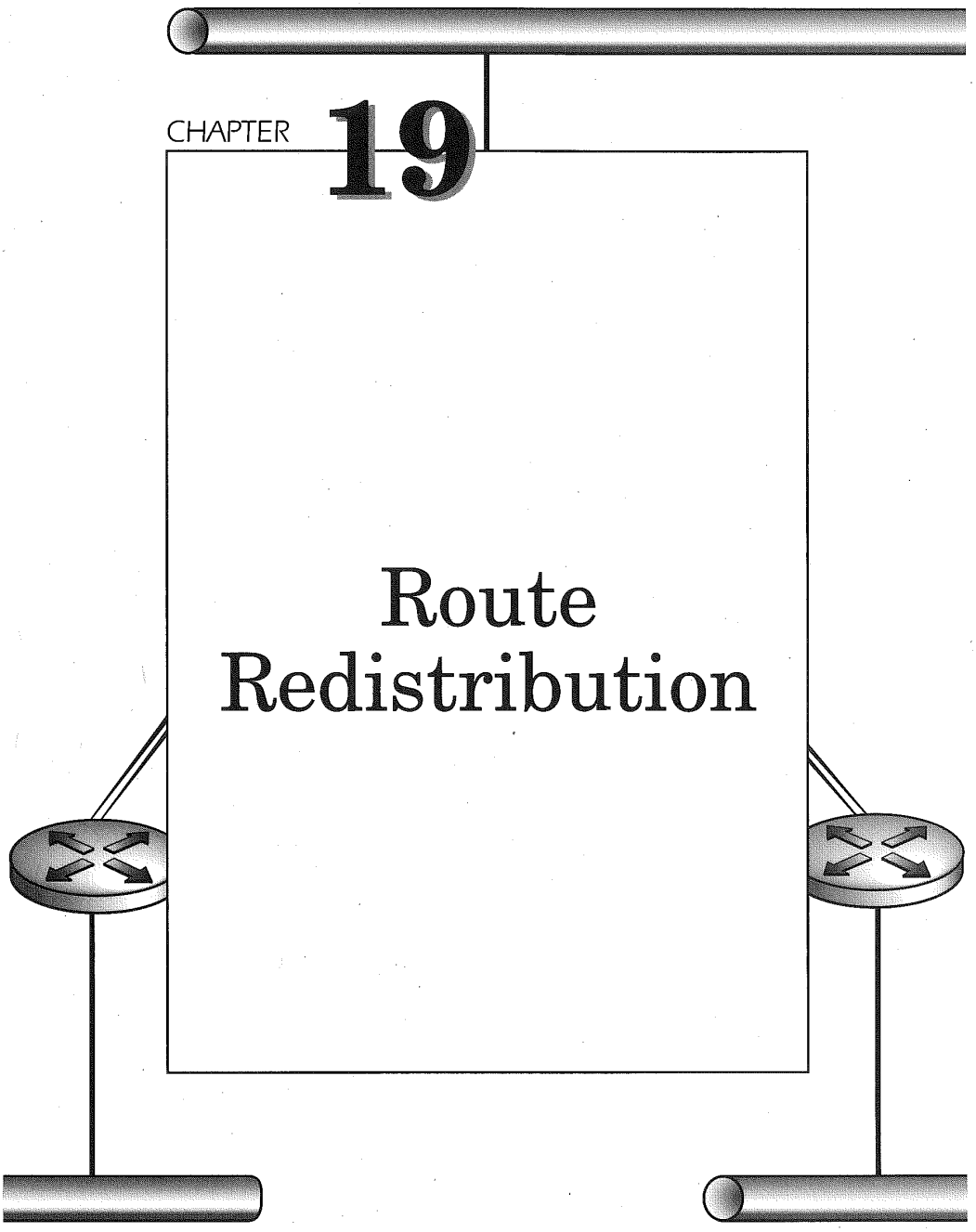


Figure 18-31 Route flapping network topology example.

If the EBGP link to router R4 from router R2 becomes unstable, router R2 places the BGP entry for 10.192.208.0 into a history state. After several more flaps over a given time period, the route becomes dampened. Only after the penalty backs down below the reuse limit will the route again be advertised.

CHAPTER 19

Route Redistribution



A single IP routing protocol is the preferred method for managing IP routes within the network. Circumstances will arise, however, when more than one routing protocol is required. In previous chapters, we have alluded to some of these circumstances. More than likely, multiple routing protocols will be necessary during a migration from one interior gateway protocol (IGP) to another, such as migrating from RIP to EIGRP or RIP to OSPF. Another example is a historical network design where a Sun workstation is acting as a RIP router for a LAN but requires connectivity to the WAN that is using Cisco EIGRP or OSPF.

Another potential requirement for running multiple routing protocols is the mix of vendor router hardware and software. An example would be the merger of two companies. One company using non-Cisco router platforms running RIP and the other company running Cisco router platforms running EIGRP require both routing tables to be merged. Cisco IOS enables the simultaneous use of multiple routing protocols by using the route redistribution feature.

Understanding Route Redistribution

Cisco IOS enables the execution of multiple IP routing protocols. Each routing protocol and the networks serviced by the routing protocol are referred to as an autonomous system. Cisco IOS using the route redistribution feature enables the exchange of routing information built by one protocol with another. In a matter of speaking, Cisco IOS translates the routing information of one routing protocol into another.

For example, Figure 19-1 displays two AS's. The AS100 uses the Cisco IGRP routing protocol and the AS200 is using the Cisco EIGRP routing protocol. At least one router within the network must be using both routing protocols for each AS to learn of the routes between the AS's. A router using multiple routing protocols and redistributing the routes between the routing protocols is termed an Autonomous System Boundary Router (ASBR). The ASBR interconnects the two unique AS's so that the routes of both are advertised to each AS.

The Class C network 192.168.140.0 address used in AS200 with EIGRP is advertised to router R1 by the ASBR, router R2. The resulting IGRP routing table in router R1 includes the route to 192.168.140.0 with a 255.255.255.0 mask. The router R1 IGRP table will direct any 192.168.140.x routing request to serial interface S1 of router R1. Router R2

will receive the request and forward this request over serial interface S0 of router R2 to reach the 192.168.140.0 network. Router R3, in its EIGRP routing table, will then interpret the destination address of the request and match it against its routing table. The route for destination IP address 192.168.140.10 falls into the network address of 192.168.8.0 with a 255.255.248.0 mask. The EIGRP routing table in AS200 for router R3 forwards the packet to the Ethernet 0 interface on router R3, as indicated by the routing table.

The sending host in Figure 19-1 is IP address 172.16.1.2 with a 255.255.255.0 mask on Ethernet interface 1 of router R1 in AS200. Router R3 has included in its routing table a route for the 172.16.0.0 network that it has learned from the ASBR router R2 advertisement. The reply back in this simple configuration uses the reverse path. Both router R1 and R3 receive a summarized route advertisement from router R2.

The Redistribution Router Configuration Command

The router configuration command redistribution enables the Cisco IOS to translate the routes learned from one routing protocol into another routing protocol. The command format is as follows:

redistribute *protocol* [*process-id*] {**level-1** | **level-1-2** | **level-2**} [**metric** *metric-value*]

[**metric-type** *type-value*] [**match** {**internal** | **external 1** | **external 2**}]

[tag *tag-value*] [**route-map** *map-tag*] [**weight** *weight*] [**subnets**]

The *protocol* variable identifies the source-routing protocol. The routes derived from this routing protocol are the routes translated into the routing protocol under which this **redistribute** command is specified. The values available for the *protocol* variable are

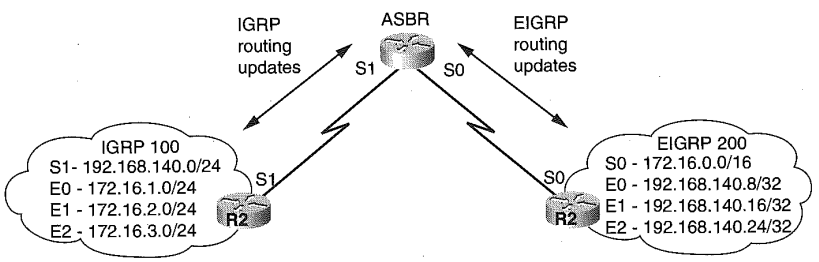


Figure 19-1 A simple route redistribution configuration.

- **bgp**
- **eigrp**
- **igrp**
- **isis**
- **ospf**
- **static [ip]**
- **connected**
- **rip**

Specifying the **static [ip]** keywords together as the value for *protocol* is used to redistribute IP static routes into intersystem-to-intersystem (IS-IS). Specifying the **connected** keyword as the value for *protocol* is used when interfaces are enabled for IP but are not specified under a router configuration command using the **network** command. This situation is often seen when there is no reason to have the router send hello messages or routing update broadcasts onto LANs that do not have another router on the LAN.

If **connected** is used, OSPF and IS-IS redistribute these routes as external routes to the AS. The *process-id* variable is optional and is used to identify the bgp, eigrp, or igrp AS number of the BGP, EIGRP, or IGRP process, having its routes redistributed or the OSPF process number assigned to one of potentially multiple OSPF processes within the router. RIP does not require a *process-id* value. The **level-1, level-1-2,** and **level-2** keywords are used only for IS-IS.

The optional **metric** keyword, followed by its variable named *metric-value,* is a value ranging from 0 to any positive integer. The value used on the **redistribute** command for the **metric** *metric-value* variable overrides the *metric* value used on a **default-metric** command. The *metric-value* used is assigned as the seed metric for the redistributed route. (The seed metric will be covered later in Table 19-2.)

Redistributing routes in OSPF without specifying the *metric-value* variable on the **metric** keyword causes OSPF to use a default cost of 20 for the imported route for all redistributed routes, except those derived from BGP. In the case of redistributing BGP into OSPF with the **metric** keyword specified without a **metric-value**, BGP redistributed routes receive a cost metric of one. Redistribution of OSPF routes from one OSPF AS to another OSPF AS without the *metric–value* coded after the specification of the **metric** keyword causes the importing OSPF process to use the derived OSPF route cost metric.

The **metric-type** optional keyword followed by its *type-value* variable, when used for OSPF, defaults to a type 2 external route as the default route being advertised into the OSPF AS. Using a value of **1** indicates that the default route is a Type 1 external route. A value of **2** for the **metric-type** *type-value* variable specifies a default route advertisement of Type 2 external routes into the OSPF AS.

The optional keyword **match** and its arguments **internal**, **external1**, or **external2** are specifically for OSPF routes being redistributed into other routing protocols. Specifying the **internal** argument of the optional **match** keyword indicates that the routes are internal routes of the AS. Using the **external1** argument indicates that the routes are external to the AS but were calculated by OSPF as type 1 external routes. The **external2** argument indicates that the routes are external to the AS and that OSPF had created them as type 2 external routes.

The optional **tag** keyword and its associated *tag-value* variable assign a 32-bit decimal value to external routes. The *tag-value* specified is not used by the OSPF routing protocol but can be used by the ASBR. If a tag is not defined, the default tag used when redistributing BGP routes is the remote AS number of routes coming from BGP. The other routing protocols default the tag to 0.

The optional **route-map** keyword and its following *map-tag* variable identify the filter to apply to routes imported from the source routing protocol. Not specifying a **route-map** allows all routes to be redistributed, while not specifying a *map-tag* value is a means of filtering all the routes being redistributed. The **weight** keyword and its variable *weight* assigns an integer from 0 to 65,535 to routes being redistributed into BGP. BGP then uses the *weight* value to determine the best path to the destination if multiple paths exist. The final optional keyword is the **subnets** keyword. This keyword is used for redistributing routes into OSPF, enabling granular redistribution versus summarized redistribution.

The following configuration code is an example of the redistribution command between RIP and OSPF:

```
router rip
 redistribute ospf 109 metric 10
router ospf 109
 redistribute rip metric 200 subnets
```

In this example, the OSPF-derived routes are redistributed into RIP routes having a hop count of 10. The RIP-derived routes being redistributed into OSPF routes as type 2 external routes are given an OSPF cost of 200.

Selecting the Best Path Based on a Routing Protocol

Each of the routing protocols discussed has a variance on the way common metrics between them can be used due to the metric structure and/or the algorithm used to derive the metric value. This makes redistribution difficult to implement. Cisco IOS utilizes a methodology for determining the best path to a destination when two or more routing protocols are used, resulting in two or more different routes to the destination. The methodology is based on an administrative distance and a default metric.

Altering Trusted or Believable Routes Using the Distance Command

The administrative distance is a metric value that determines the validity or believability of a routing protocol. The various routing protocols available in Cisco IOS are prioritized in the order of most to least believable. This is the first criterion used by the Cisco IOS in determining which of the multiple routes is the best to reach the destination. The default administrative distance values applied to the various routing protocols are listed in believability order in Table 19-1.

Judging from the table, if a router running RIP and EIGRP received a route advertisement for the 172.16.0.0 network from both of the routing protocols, the EIGRP route would be the most believable. Therefore, it is this route that is placed in the routing table.

Although the table displays the default values used by Cisco IOS for administrative distances, occasions will occur when the distance may need to be altered. For example, when migrating from RIP to EIGRP, the EIGRP routes can be given a higher administrative value than the default RIP, or RIP can be given a lower administrative distance value than the default EIGRP. This enables both routing protocols to build their own tables but provide a preference as to which routing protocol provides the best path. Once the EIGRP routing tables are built, the default distances can be reinstated and the EIGRP routing information for the connected networks will be used as the best path to the destination networks.

The administrative distance of a routing protocol can be altered by using the **distance** router configuration command. The format of the **distance** command is as follows:

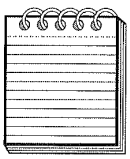
distance ***weight*** **[*address mask* [*access-list-number* | *name*]] [ip]**

Table 19-1 The Default Administrative Distances Used by Cisco IOS

Route Source	Default Distance
Connected interface	0
Static route	1
Enhanced IGRP summary route	5
External BGP	20
Internal Enhanced IGRP	90
IGRP	100
OSPF	110
IS-IS	115
RIP	120
EGP	140
External EIGRP	170
Internal BGP	200
Unknown	255

The *weight* variable is the actual administrative distance that is assigned to the routes built by the routing protocol to which the **distance** command is specified. The value of the *weight* variable can be a number ranging from 10 to 255. The administrative value is the default metric used by Cisco IOS when there is no other metric available for the route. A value of 255 specified for the *weight* variable instructs the Cisco IOS to not build an entry in the routing table for the routing protocol to which this **distance** command is applied. Assigning a value to the *weight* variable is purely a subjective versus quantitative approach. The *weight* value selected should be chosen to meet the objective for using the **distance** command.

The optional *address* variable is the network address to which the *weight* value is to be applied. The following *mask* variable, written in four-part dotted-decimal form, is the bit mask applied to the *address* variable value. Using these variables, only routes that match the *address/mask* variable value pair are given the administrative distance value specified by this command. The optional *access-list-number* or *name* variable value specifies the access-list number or name of a standard IP access list that is applied to the incoming routing update. The optional **ip** keyword is for IS-IS routing protocols, enabling the routing tables to build IP derived routes for IS-IS.



NOTE: When applying the distance command to BGP, the value sets the administrative distance for the external BGP routing entries and not the internal BGP routing entries.

The use of the access-list arguments applies to the insertion of a route into the routing table. Using this argument on the **distance** command enables the router administrator to filter networks based on the sending IP address supplying the routing update.

Multiple instances of the **distance** command can be used under the router configuration mode. As an example

```
router igrp 10
 network 192.168.31.0
 network 172.28.0.0
 distance 255
 distance 90 192.168.31.0 0.0.0.255
 distance 120 172.28.1.1 0.0.0.0
```

the router AS for this IGRP definition is 10. The IGRP router configuration includes the 192.168.31.0 and 172.28.0.0 networks. The first **distance** command applied to the routes received indicates that all the routes are ignored when the routing update does not have an explicit distance set. The second **distance** command applies an administrative distance of 90 to all routing updates from routers advertising the 192.168.31.0 network. The final **distance** command sets the administrative distance to 120 for all routing updates received from the router with the source address of 172.28.1.1.

Modifying the Seed Metric Using the default-metric Command

An ASBR uses the administrative distance value to determine which routing protocol to use for selecting the best route. The next step of the ASBR is to translate the metric(s) of the received routing update into a routing update appropriate for the other routing protocol. As an example, RIP uses a hop count as its metric. Redistributing RIP into OSPF requires that the

hop count be assigned an OSPF cost value. Recall that cost is the determinant metric of OSPF. The determinant metric of each routing protocol is termed as the seed or default metric.

The seed metric, once established, increments normally within the AS. A router administrator can influence the route selection process by specifying the default metric value. The default metric value is modifiable using the router configuration mode command **default-metric**. There are two formats for the command. The first format, shown below, is used for settling the default metric value during route redistribution with the BGP, OSPF, and RIP-routing protocols. The format of the command when used for these routing protocols is

default-metric *number*

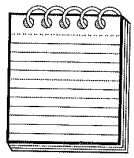
The value used for *number* variable ranges from 0 to any positive integer that reflects the values used by the seed metric of the routing protocol importing the redistributed route. The seed metric is listed in Table 19.2. The value of the *number* variable in the **default-metric** command is used as the metric value for the seed metrics of each respective routing protocol.

As an example, if the following were defined in a router running both RIP and OSPF, the OSPF-derived routes would be translated into RIP routes using 4 as the hop-count metric for the RIP routing information on all redistributed OSPF routes. The RIP routes translated into OSPF are assigned a cost value of 10 based on the **default-metric** command found under the router ospf configuration command:

```
router rip
 default-metric 4
 redistribute ospf 100
router ospf 100
 default-metric 10
 redistribute rip
```

Table 19-2 Seed Metrics for RIP, OSPF, and BGP

Routing Protocol	Seed Metric
RIP	Hop count
OSPF	Cost
BGP4	Multi-exit discriminator (MED) metric
BGP2 and BGP3	INTER_AS



NOTE: *Redistribution of OSPF routes between OSPF processes does not cause the OSPF processes to maintain route metrics. A default-metric statement is needed to ensure the proper metric assignment.*

When defining the default metric for route redistribution involving IGRP or EIGRP, the command format is different. This is because these routing protocols have five metrics. The format of the command is as follows when used with IGRP or EIGRP:

default-metric *bandwidth delay reliability loading mtu*

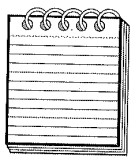
The variables of the **default-metric** command used for IGRP and EIGRP are positional and must all be specified when changing only one of the variables. The *bandwidth* variable is the minimum bandwidth in kilobits per second of the **default-metric** command. The *bandwidth* variable value can be set to 0 or any positive integer.

The *delay* variable value is measured in tens of microseconds. The *delay* variable can be set to 0 or any positive number. The actual value is the result of multiplying the *delay* variable value by 39.1 nanoseconds.

The *reliability* metric variable value indicates the capability of the route to successfully transmit packets. The number value for the *reliability* metric variable ranges from 0 to 255. The higher the value, the more reliable the route. A value of 255 indicates that the transmission of packets is considered to have a 100-percent success rate. A value of 0 for the reliability variable means the route is completely unreliable for transmitting packets.

The *loading* variable indicates the effective bandwidth, or percentage, on the route. The value for *loading* is a number ranging from 0 to 255, with 255 indicating that the bandwidth on the route is saturated or 100 percent utilized.

The final variable is the *mtu* (maximum transmission unit) size allowed on the route in bytes. The value for *mtu* can be set to 0 or be any positive integer.



NOTE: *When redistributing routes from BGP, OSPF, or RIP to EIGRP and IGRP, a default metric statement is needed. Only connected and static routes can be redistributed without a default metric. Redistributing routes between IGRP and EIGRP does not require a default metric command.*

As an example, the following router configuration redistributes RIP-sourced routes into EIGRP routes, using a bandwidth metric of 1,000, a delay of 50, a reliability of 255, a loading of 50, and a standard MTU size (mtu) of 1,500 bytes:

```
router rip
 network 131.110.0.0
router eigrp 100
 network 131.111.0.0
 redistribute rip
 default-metric 1000 50 255 50 1500
```

This example demonstrates a one-way redistribution. RIP-derived routes are being redistributed into EIGRP routes using the default-metric command. A two-way redistribution would involve the EIGRP-derived routes being redistributed into RIP.

Filtering Redistributed Routes Using the distribute-list Command

The filtering of the redistributed routes provides added control on which routes are allowed to be redistributed once received as routing updates and on being advertised into the AS. The filtering is accomplished using the **distribute-list** command. Two formats support the control functions. The first format controls which of the received routing updates are to be translated into the routing protocol process:

distribute-list {*access-list-number* | *name*} in [*type number*]

The **distribute-list in** command enables filtering of received updates prior to translation. The **distribute-list in** command is applicable to all routing protocols except IS-IS and OSPF. The use of the **distribute-list in** command is effective in preventing routing loops from being propagated.

Specifying the *access-list-number* or *name* variable identifies a Cisco IOS access list filter that is applied to the received routing update. The values specified must point to a standard IP access list number or name. The list specifies which networks can be received or which can be suppressed prior to redistribution.

The optional *type number* variable pair identifies which router interface the distribute-list applies. Not using this optional pair enables the **distribute-list in** to all interfaces on which the routing protocol executes. In the following example configuration, the EIGRP routing protocol permits only the default network of 0.0.0.0 and the network 172.16.0.0. Any network on incoming routing updates not matching these criteria are suppressed:

```
router eigrp 10
 network 172.16.0.0
 distribute-list 1 in
!
access-list 1 permit 0.0.0.0
access-list 1 permit 172.16.0.0
access-list 1 deny 0.0.0.0 255.255.255.255
```

The filtering mechanism required that controls routing updates during redistribution for OSPF and all other routing protocols is accomplished by using the distribute-list out router configuration command. The format of the command is

distribute-list {*access-list-number* | *name*} out
[*interface-name* | *routing-process* | *autonomous-system-number*]

Using the **distribute-list out** command network, advertisements can be suppressed during the sending of routing updates. Specifying the *access-list-number* or *name* variable identifies a Cisco IOS access list filter that is applied prior to sending routing updates. The values specified must point to a standard IP access list number or name. The list will specify which networks can be sent or which can be suppressed prior to sending the route updates. The out keyword is necessary to apply the **distribute-list** command to outbound routing updates.

The *interface-name* optional variable is used to identify a specific router interface on which the filter is applied for outgoing routing updates. The *routing-process* optional variable can be any of the following keywords:

- **bgp**
- **eigrp**
- **igrp**
- **isis**
- **ospf**
- **static**
- **connected**
- **rip**

The *autonomous-system-number* optional variable identifies the AS number of the routing process if an AS number is applicable. When using the optional *routing-process* optional variable, the filter is applied to only the routes derived from the *routing-process* value specified. It is only after the *routing-process*-derived routes are applied to the access-list filter that any other access list is applied to the *routing-process*-derived routes that are not suppressed by the **distribute-list out** access list.

In the following example, a **distribute-list out** command is used to allow only the 192.168.31.0 network of EIGRP AS 110 to be redistributed into EIGRP AS number 10:

```
router eigrp 10
 network 10.0.0.0
 redistribute eigrp 110
 distribute-list 1 out eigrp 110
!
router eigrp 110
 network 192.168.31.0
 network 192.168.32.0
!
access-list 1 permit 192.168.31.0
```

The **distribute-list 1 out eigrp 110** command under the router configuration for EIGRP AS 10 directs the routing update send process to use access-list 1. The access-list 1 permits only the 192.168.31.0 network to be sent from derived routes of EIGRP AS110 into the EIGRP AS 10 network.

Redistribution Considerations

Because route redistribution is such a powerful means for affecting routing within the network, it is important to understand its impact. When implementing redistribution, it is best to have a distinct boundary between the routing protocols in question. This enables a point of reference for determining routing problems, should they occur.

Using one-way route redistribution avoids routing loops and minimizes convergence time between the routing protocols. In a one-way route redistribution topology, it is suggested that in the reverse direction a default route be implemented, enabling reverse routing. Although two-way route redistribution provides a complete translation of routes between the routing protocols, it may require the use of default routes, route filters, or the specification of default metrics. Using one or many of these techniques will reduce the chances of creating routing loops between the AS's.

EIGRP and redistribution is a special case because EIGRP supports not only IP but Novell IPX and AppleTalk RTMP-routing protocols. Focusing in on IP-routing protocols, EIGRP can have its routes redistributed automatically with IGRP if both have the same AS number. If the EIGRP and IGRP AS numbers are different, however, the redistribution of routes with EIGRP by IGRP is treated just like the redistribution of the other IP-routing protocols. A distinct redistribute command must be entered, identifying the AS number of the routing protocol for routes to be redistributed. For example,

```
router igrp 10
 network 172.16.0.0
 router eigrp 10
 network 10.0.0.0
```

will have their routes redistributed automatically because they share the same AS number. In this case, the AS number is 10. However, if the two routing protocols were defined as

```
router igrp 10
 network 172.16.0.0
router eigrp 200
 network 10.0.0.0
```

then no route redistribution occurs automatically. In this case, the redistribute command is required. An example would be to code the following for two-way route redistribution:

```
router igrp 10
 network 172.16.0.0
 redistribution eigrp 200
router eigrp 200
 network 10.0.0.0
 redistribution igrp 10
```

Redistribution Examples

Route redistribution, as discussed earlier, may be required in many different scenarios. The most common are redistribution scenarios of RIP-to-OSPF, IGRP-to-EIGRP, and RIP-to-EIGRP. The following sections discuss these redistribution configurations.

RIP V1-to-OSPF Redistribution

Quite often, RIP V1-to-OSPF migrations are encountered to take advantage of the open-standard (such as interoperability between router vendors) and to utilize the minimized convergence time, VLSM support, and route summarization capabilities of OSPF over RIP Version 1. Shown in Figure 19-2 is a RIPO V1 network using a Class B network addressing scheme with a full 24-bit network mask (8-bit subnet mask).

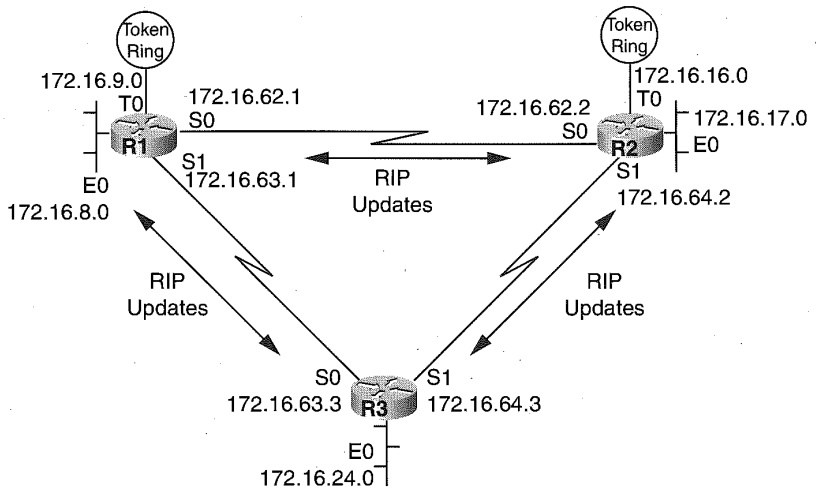


Figure 19-2 RIP V1 network configuration prior to implementing OSPF.

The network address assignments attaching to routers R1, R2, and R3 provide a range for use by each of the routes. For example, subnet 172.16.8-15.0 is for use on router R1 non-backbone connections. Router R2 has reserved subnets 172.16.16-23.0 for its non-backbone network connections. Router R3 is assigned subnets 172.16.24-31.0 for its non-backbone connections. The backbone connections (serial links) in Figure 19-2 are assigned subnet addresses of 172.16.62-64.0. For each of the routers in Figure 19-2, the following router configurations are in use prior to implementing OSPF:

Router R1 configuration for Figure 19-2 prior to implementing OSPF:

```
interface serial 0
ip address 172.16.62.1 255.255.255.0
interface serial 1
ip address 172.16.63.1 255.255.255.0
interface ethernet 0
ip address 172.16.8.1 255.255.255.0
interface tokenring 0
ip address 172.16.9.1 255.255.255.0
router rip
network 172.16.0.0
```

Router R2 configuration for Figure 19-2 prior to implementing OSPF:

```
interface serial 0
ip address 172.16.62.2 255.255.255.0
interface serial 1
ip address 172.16.64.2 255.255.255.0
```

```
interface ethernet 0
ip address 172.16.17.2 255.255.255.0
interface tokenring 0
ip address 172.16.16.2 255.255.255.0
router rip
network 172.16.0.0
```

Router R3 configuration for Figure 19-2 prior to implementing OSPF:

```
interface serial 0
ip address 172.16.63.3 255.255.255.0
interface serial 1
ip address 172.16.64.3 255.255.255.0
interface ethernet 0
ip address 172.16.24.3 255.255.255.0
router rip
network 172.16.0.0
```

Since the network address of all the interfaces on the routers are 172.16.0.0, only this network is required under the router rip configuration command of each router.

Typically, in a network, such as that shown in Figure 19-2, OSPF is first added to the backbone connections and given the assignment as OSPF area 0. Figure 19-3 illustrates the new network configuration utilizing OSPF Area 0 to connect the backbone routers R1, R2, and R3 as ASBRs.

The following router configurations apply to Figure 19-3 in support of using OSPF as the backbone-routing protocol and RIP as the edge-routing protocol. The configurations employ the use of the passive-interface, default-metric, and distribute-list out commands:

Router R1 configuration for Figure 19-3 with OSPF Area 0 on the backbone:

```
interface serial 0
ip address 172.16.62.1 255.255.255.0
interface serial 1
ip address 172.16.63.1 255.255.255.0
interface ethernet 0
ip address 172.16.8.1 255.255.255.0
interface tokenring 0
ip address 172.16.9.1 255.255.255.0
!
router rip
default-metric 10
network 172.16.0.0
passive-interface serial 0
passive-interface serial 1
redistribute ospf 100 match internal external 1 external 2
distribute-list 2 out ospf 100
```

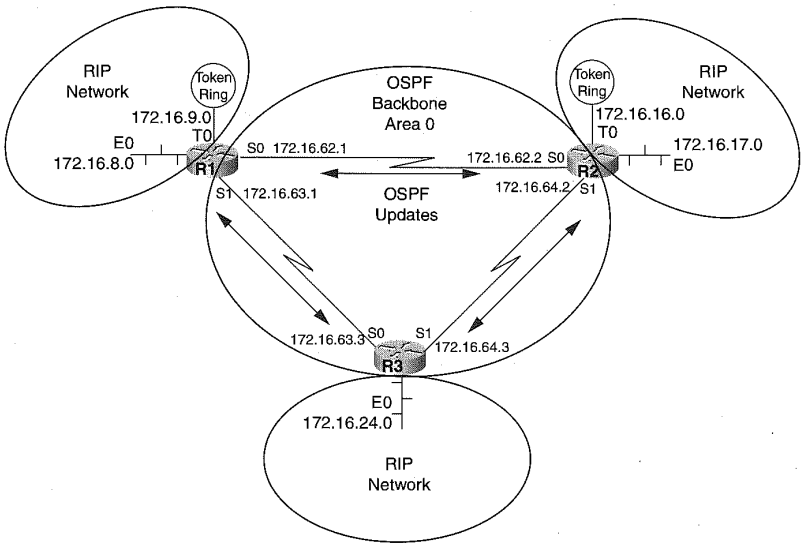


Figure 19-3
RIP V1 network configuration with OSPF Area 0 assigned to the backbone connections.

```
!
router ospf 100
default-metric 20
network 172.16.62.0 0.0.0.255 area 0
network 172.16.63.0 0.0.0.255 area 0
redistribute rip subnets
distribute-list 1 out rip
!
access-list 1 permit 172.16.8.0 0.0.7.255
access-list 1 deny 0.0.0.0 255.255.255.255
access-list 2 deny 172.16.8.0 0.0.7.255
access-list 2 permit 0.0.0.0 255.255.255.255
```

Router R2 configuration for Figure 19-3 with OSPF Area 0 on the backbone:

```
interface serial 0
ip address 172.16.62.2 255.255.255.0
interface serial 1
ip address 172.16.64.2 255.255.255.0
interface ethernet 0
ip address 172.16.17.2 255.255.255.0
interface tokenring 0
ip address 172.16.16.2 255.255.255.0
!
```

```
router rip
default-metric 10
network 172.16.0.0
passive-interface serial 0
passive-interface serial 1
redistribute ospf 100 match internal external 1 external 2
distribute-list 2 out ospf 100
!
router ospf 100
default-metric 20
network 172.16.62.0 0.0.0.255 area 0
network 172.16.64.0 0.0.0.255 area 0
redistribute rip subnets
distribute-list 1 out rip
access-list 1 permit 172.16.16.0 0.0.7.255
access-list 1 deny 0.0.0.0 255.255.255.255
access-list 2 deny 172.16.16.0 0.0.7.255
access-list 2 permit 0.0.0.0 255.255.255.255
```

Router R3 configuration for Figure 19-3 with OSPF Area 0 on the backbone:

```
interface serial 0
ip address 172.16.63.3 255.255.255.0
interface serial 1
ip address 172.16.64.3 255.255.255.0
interface ethernet 0
ip address 172.16.24.3 255.255.255.0
!
router rip
default-metric 10
network 172.16.0.0
passive-interface serial 0
passive-interface serial 1
redistribute ospf 100 match internal external1 external2
distribute-list 2 out ospf 100
!
!
router ospf 100
default-metric 20
network 172.16.63.0 0.0.0.255 area 0
network 172.16.64.0 0.0.0.255 area 0
redistribute rip subnets
distribute-list 1 out rip
access-list 1 permit 172.16.24.0 0.0.7.255
access-list 1 deny 0.0.0.0 255.255.255.255
access-list 2 deny 172.16.24.0 0.0.7.255
access-list 2 permit 0.0.0.0 255.255.255.255
```

Because OSPF is being used as a routing protocol on the backbone routers, the transmission of RIP-routing updates between routers R1, R2, and R3 is not needed. The RIP-routing updates are suppressed using the passive-interface command, specifying the applicable serial interface on each of the routers. Each of the three routers redistributes the RIP subnet

routing updates into OSPF routes by the inclusion of the subnets keyword on the redistribute command under the router ospf configuration command. Without the subnets keyword only the 172.16.0.0 network is redistributed.

The RIP-routing process on each router is updated with the routes learned by OSPF through the redistribution of the RIP routes. Router R1's RIP process learns of routes from router R2 RIP process from the redistribution of RIP routes in router R2. The same process is used for router R1 and R3.

The **ospf keyword** on the redistribute command under the router rip configuration command indicates that all routes originating in the AS, routes derived as intra-area OSPF routes (external1), and routes derived as inter-area routes (external2) are to be redistributed into RIP routes. The default-metric command on each of the routers applies a default hop count of 10 for the OSPF routes translated into RIP.

As mentioned earlier, it is always good practice to provide the capability of blocking a potential routing loop, even if there is not a known configuration that may cause a loop. In the above router configuration examples, the potential for routing loops is addressed using the distribute-list out command under the router ospf configuration command for each router. The **distribute-list out** command identifies that the access-list number 1 will be applied to outgoing RIP updates only by the use of the **rip** keyword on the command. The access-list 1 permits only the subnet address range assigned to each of the routers for their end user interfaces. Any other RIP routing updates not meeting this criteria are suppressed by the deny statement following the permit statement of the access list.

The result of the distribute-list out command stops each router from advertising the networks derived by the other RIP-routing processes back into the OSPF backbone, which protects the backbone from creating a loop. To complete the loop-free environment configuration, the RIP process of each router should employ a distribute-list out command applied to the OSPF routes being redistributed in the same manner as the one that was applied to the OSPF process. By filtering on OSPF routes that are derived by OSPF from being redistributed into RIP, the duplication of routes is avoided and a loop-free configuration is achieved.

IGRP-to-EIGRP Redistribution

IGRP-to-EIGRP redistribution in its simplistic form is accomplished using the same AS numbers for the two routing protocols. For example, in Figure 19-4, router R5 connects an EIGRP network that requires connectivity to resources in an IGRP network. Router R5 becomes the ASBR for the redistribution and is configured to use both IGRP and EIGRP. Recall that IGRP and EIGRP use the same routing metrics in determining the best route to a destination network. Because of this commonality, when IGRP and EIGRP use the same AS number the routes are automatically redistributed and hence no redistribution command is necessary.

In Figure 19-4, router R6 uses EIGRP as the only routing process. Because the EIGRP AS is supporting only EIGRP routers, the router configuration for router R5 uses the passive-interface command on the connection to the EIGRP network for the IGRP route process, disabling the IGRP-routing update broadcasts to the EIGRP network. The following router configurations apply to the IGRP-to-EIGRP redistribution configuration diagrammed in Figure 19-4:

Router R5 configuration applied to Figure 19-4 supporting a single EIGRP AS connection to an IGRP AS:

```
interface ethernet 0
ip address 192.168.42.121 255.255.255.248
interface ethernet 1
ip address 192.168.42.9 255.255.255.248
!
router igrp 42
network 192.168.42.0
router eigrp 42
network 192.168.42.0
```

Router R6 configuration applied to Figure 19-4 supporting a single EIGRP AS connection to an IGRP AS:

```
interface ethernet 0
ip address 192.168.42.17 255.255.255.248
interface ethernet 1
ip address 192.168.42.10 255.255.255.248
!
router igrp 42
network 192.168.42.0
router eigrp 42
network 192.168.42.0
```

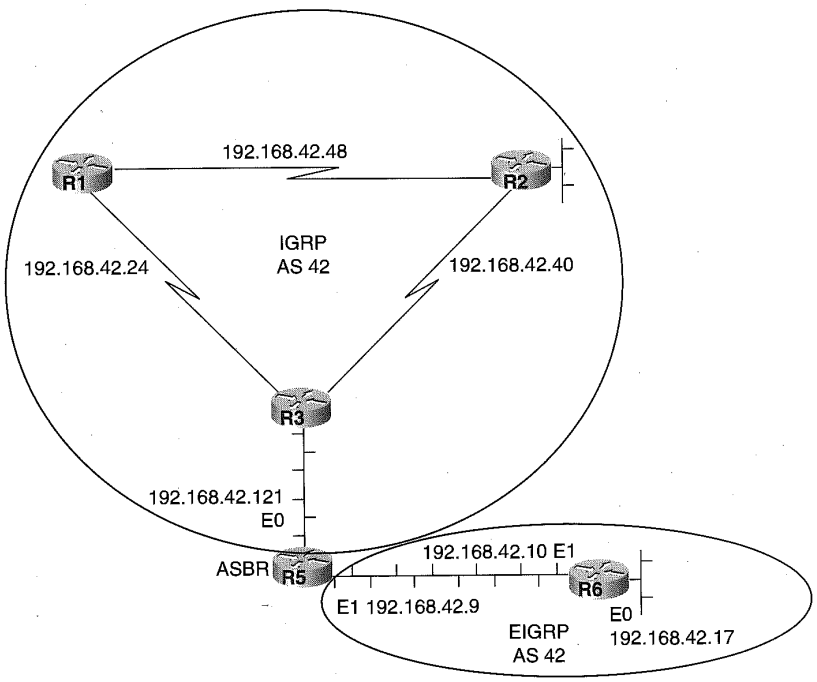


Figure 19-4
IGRP and a single EIGRP network configuration using the same AS number.

The subnet 192.168.42.120 is viewed as an internal route for router R6 because router R5 is also running EIGRP along with IGRP. The other routes by router R6 are considered external routes because they are derived from the IGRP redistribution occurring on router R5.

The network configuration shown in Figure 19-5 illustrates the implementation of redistributing EIGRP networks into IGRP networks. Routers R1, R2, and R3 all use IGRP AS 60. Router R1 also uses static routes to the network 192.168.9.0 .

The following router configurations apply to Figure 19-5 prior to implementing EIGRP:

Router R1 configuration for Figure 19-5 prior to implementing EIGRP:

```
interface serial 0
ip address 172.16.1.1 255.255.255.0
interface serial 1
ip address 172.16.2.1 255.255.255.0
interface ethernet 0
ip address 10.1.1.1 255.255.255.0
```

```
interface ethernet 1
ip address 10.1.2.1 255.255.255.0
router igrp 60
network 172.16.0.0
network 10.0.0.0
default-metric 1000 100 1 1 1500
redistribute static
ip route 192.168.9.0 255.255.255.0 e0
```

Router R2 configuration for Figure 19-5 prior to implementing EIGRP:

```
interface serial 0
ip address 172.16.1.2 255.255.255.0
interface serial 1
ip address 172.16.3.1 255.255.255.0
router igrp 60
network 172.16.0.0
```

Router R3 configuration for Figure 19-5 prior to implementing EIGRP:

```
interface serial 0
ip address 172.16.2.2 255.255.255.0
interface serial 1
ip address 172.16.3.2 255.255.255.0
interface ethernet 0
ip address 172.16.11.4 255.255.255.0
interface ethernet 1
ip address 172.17.12.1 255.255.255.0
router igrp 60
network 172.16.0.0
network 172.17.0.0
```

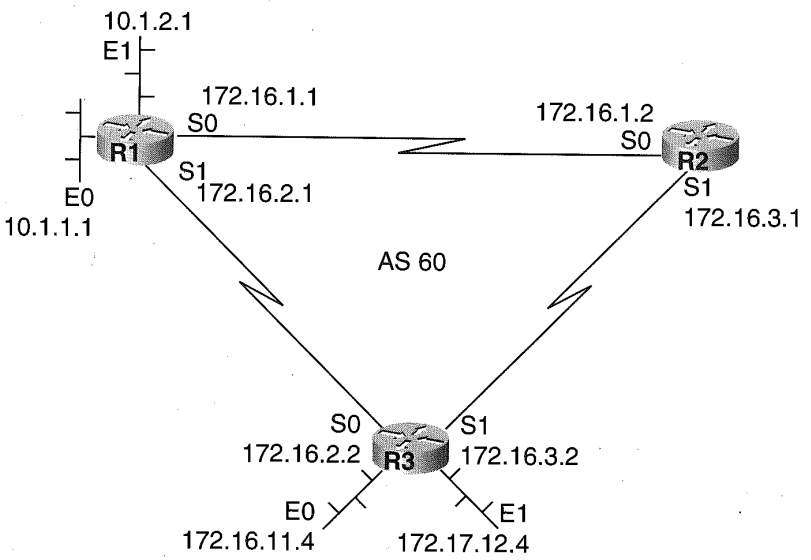


Figure 19-5 Implementing EIGRP into an IGRP network configuration.

The default-metric command used in router R1 assigns the metrics to the static routes as they are translated into IGRP-routing entries. By adding the router eigrp 60 command to router R1, IGRP routes are automatically included in the EIGRP routing table. The following configuration is now applied to router R1 in Figure 19-5:

Router R1 configuration for Figure 19-5 implementing EIGRP:

```
interface serial 0
ip address 172.16.1.1 255.255.255.0
interface serial 1
ip address 172.16.2.1 255.255.255.0
interface ethernet 0
ip address 10.1.1.1 255.255.255.0
interface ethernet 1
ip address 10.1.2.1 255.255.255.0
router igrp 60
network 172.16.0.0
network 10.0.0.0
default-metric 1000 100 1 1 1500
redistribute static
!
router eigrp 60
network 172.16.0.0
network 10.0.0.0
default-metric 1000 100 1 1 1500
redistribute static
!
ip route 192.168.9.0 255.255.255.0 e0
```

Router R3 is now updated to include the IGPR-routing process. After this process is defined to router R3, EIGRP routing updates are exchanged between router R1 and R3. Routes for network 10.0.0.0 and 172.16.0.0 are automatically summarized in the EIGRP updates. Because EIGRP distance values are less than those derived by IGRP, the EIGRP routes learned now displace the IGPR routes in router R1 and R3 routing tables. Adding EIGRP routing to router R2 completes the inclusion of EIGRP into the network.

The following configurations show both IGRP and EIGRP routing in all three routers. At this point, the IGRP process can be removed from each router by entering no router igrp 60 in configuration mode. Connectivity is not disrupted because the EIGRP tables are being updated by the neighboring EIGPR routers. The following route configurations show the IGRP and EIGRP definitions as applied to Figure 19-5:

Router R2 configuration for Figure 19-5 with EIGRP:

```
interface serial 0
ip address 172.16.1.2 255.255.255.0
interface serial 1
 ip address 172.16.3.1 255.255.255.0
!
router igrp 60
 network 172.16.0.0
!
router eigrp 60
 network 172.16.0.0
```

Router R3 configuration for Figure 19-5 with EIGRP:

```
interface serial 0
ip address 172.16.2.2 255.255.255.0
interface serial 1
ip address 172.16.3.2 255.255.255.0
interface ethernet 0
ip address 172.16.11.4 255.255.255.0
interface ethernet 1
ip address 172.17.12.1 255.255.255.0
router igrp 60
 network 172.16.0.0
 network 172.17.0.0
!
router eigrp 60
 network 172.16.0.0
 network 172.17.0.0
```

RIP-to-EIGRP Redistribution

RIP-to-EIGRP redistribution requires the mapping of unlike routing metrics. For example, in Figure 19-6, a RIP network and an EIGRP network require connectivity to support the merger of two companies. The ASBR in this case is router R1, which must support two-way redistribution between RIP and EIGRP. The configuration must also protect the network routing tables from having EGIRP sending routes learned via RIP back into the RIP network. It must also prevent the RIP process from sending routes learned via EIGRP back into the EIGRP network. This "route feedback" is what typically causes unstable routing tables and results in erroneous routing information with a high potential for routing loops to occur.

Router R1, acting as the ASBR, implements the stated requirements above. The router configuration for router R1 follows:

Router R1 configuration applied to Figure 19-6:

```
interface Ethernet 0
 ip address 172.16.1.1 255.255.255.0
interface Ethernet 1
 ip address 172.16.2.1 255.255.255.0
interface serial0
 ip address 172.16.10.1 255.255.255.0
!
router rip
 network 172.16.0.0
 redistribute eigrp 10
 default-metric 2
 passive-interface serial 0
!
router eigrp 10
 network 172.16.0.0
 redistribute rip
 default-metric 1544 100 255 1 1500
 distribute-list 1 in
 passive-interface ethernet 0
 passive-interface Ethernet 1
access-list 1 permit ip 172.16.1.0 255.255.255.0
access-list 1 permit ip 172.16.10.0 255.255.255.0
access-list 1 deny ip
```

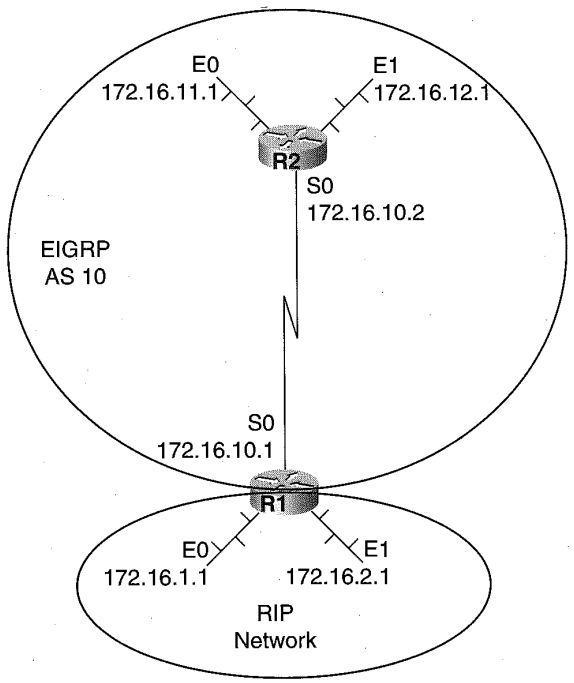


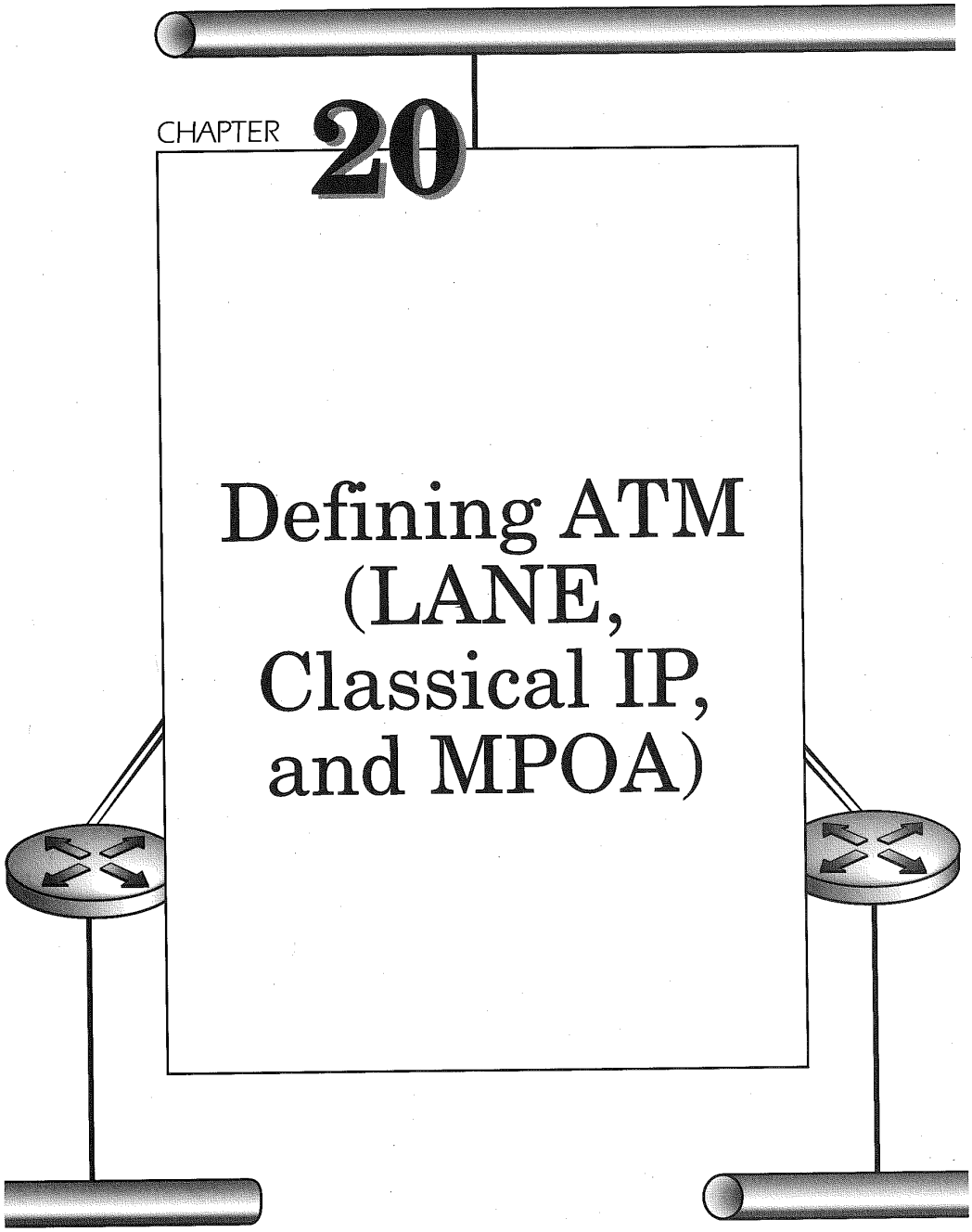
Figure 19-6 RIP and EIGRP redistribution network configuration.

In the above router configuration, router R1 stops routing updates on serial 0 for the RIP-routing protocol and on Ethernet 0 for the EIGRP-routing protocol by using the passive-interface command. The distribute-list in command is used under the router eigrp process to identify the routes that are filtered prior to being translated from RIP into EIGRP. In the example given, the 172.16.1.0 and 172.16.10.0 networks are redistributed, but the 172.16.2.0 network on ethernet 1 is suppressed from being redistributed because it does not pass through the filter applied by access-list 1.

The derived RIP routes are assigned EIGRP metric values by the default-metric router configuration command. The first value represents the minimum bandwidth assigned to the route. In this case, it describes the bandwidth as 1.544 Kbps. The second value of 100 specifies a route delay in tens of microseconds. The third value indicates that there is a 100 percent reliability of the link based on the 255 value being specified. The fourth value indicates that the effective bandwidth of the route is a 1 and the final value defines the MTU size available on the route.

CHAPTER 20

Defining ATM (LANE, Classical IP, and MPOA)



As discussed in Chapter 6, "ATM Internetworking Design," ATM was developed with the idea of supporting multiple services. The ATM transmission protocol could be a network solution due to its versatility or because it made more sense before Synchronous Optical Network (SONET) was offered for wide area connectivity.

Regardless of the reason for using ATM, Cisco provides solutions for implementation. The implementation could be on the Lightstream series ATM switch products, an ATM-capable router (see Chapter 6 for routers that are ATM-capable), or the Catalyst series switches. In this section on configuration for LANE services, Classical IP and MPOA (multiprotocol over ATM) will be discussed.

Configuring LANE

All components of LANE services can be configured on an ATM-capable router. A router can be a LAN emulation client (LEC), a LAN emulation server/broadcast unknown server (LES/BUS) pair, or a LAN emulation configuration server (LECS). A LEC participates in LANE by joining an ELAN, which involves the contacting of the LECS for ELAN definitions. Each ELAN definition includes a LES server and a BUS server that provide address resolution and broadcast capability within an ELAN. The LECS contains the address locations of the LES/BUS servers. From the LECS, a LEC can find its designated LES server to become a member of a particular ELAN. The configuration of these different components will be addressed in the following sections.

Initial Configuration for LANE

Before any of the components are configured on the router, a few parameters must be set first. There must be some ATM switch cloud to connect to first. The ATM prefix address for this switch cloud must be determined and set. This is not too important in the case of a private network where the preconfigured addresses of the devices can be used, but if a Private Network-Network Interface (PNNI) hierarchy is going to be used or if the network is going to link to a public ATM network, then a unique ATM prefix should be obtained.

After a proper ATM prefix is configured in the ATM switch cloud, two permanent virtual circuits (PVC) need to be set up for interim local management interface (ILMI) and switched virtual circuit management. ILMI

is the network management system that provides configuration, performance, and fault management within the user-to-network interface (UNI). The UNI is the connection between a LEC device and a PNNI device.

The two PVCs that are needed are virtual path indicator (VPI)/virtual circuit indicator (VCI) 0/5 and 0/16. VPI/VCI 0/5 is for signaling control to set up and tear down switch virtual circuits. VPI/VCI 0/16, the ATM forum circuit, is to communicate with ILMI. These PVCs are configured on the main ATM interface of the router.

The format for the configuration of the PVCs on the main ATM interface is as follows:

interface atm (*slot/0* | *slot/port-adapter/0* | *number*)

The **interface** keyword specifies that this will be an interface command and the interface is of type **ATM**. The slot number indicates where in the router the ATM module is located. Since the interface is the main interface, the number 0 or just the number of the interface is listed. If the interface is a subinterface, the syntax of the command would be 0.subinterface-number. The command for configuring a PVC on an ATM interface is

atm pvc *vcd vpi vci* **qsaal**

The **atm pvc** keywords are used to indicate that the router will set up a PVC. The *vcd* is the circuit identifier, which has the characteristics of being represented by virtual path identifier (*vpi*) and virtual channel identifier (*vci*). For each physical port on an ATM device, there can be 256 virtual paths. Each path contains 4,096 virtual channels. Each link between two ATM devices is a virtual circuit that is mapped to a virtual path. The **qsaal** indicates the type of ATM signaling. The command to configure the LANE setup PVC is

atm pvc *vcd vpi vci* **ilmi**

The other PVC definition is for ILMI, which is used for this initialization of LANE. Again, a vcd is used to indicate the circuit number. vpi is used to give the circuit a virtual path number and vci tells which virtual circuit in the virtual path to use. The ilmi informs router that the circuit is being used for interim local management interface traffic to configure and monitor LANE.

The example configuration for this is

```
interface ATM8/0
 atm pvc 1 0 5 qsaal
 atm pvc 2 0 16 ilmi
```

This example shows the configuration of the PVCs for LANE operation on a router with an AIP in the eighth slot. The configuration for the PVC

has to be on the main interface, so a 0 is used for the interface number. The qsaal pvc of vpi 0 and vci 5 is the ATM Forum standard pvc for ATM signaling, which is used for building circuits as well as tearing them down. The ilmi pvc of vpi 0 and vci 16 is also an ATM forum standard for ILMI signaling, which includes communication between LANE components. Once the PVC is set up on the router and the router has a connection to an ATM switch, LANE components can be configured on the ATM interface of the router.

Configuring a LAN Emulation Client (LEC)

A router can be configured to be a LEC in the instance of performing routing for a specific emulated LAN (ELAN). This includes the routing for Ethernet-emulated LANs and Token Ring-emulated LANs. Some of the protocols specific to Ethernet are IP, IPX, AppleTalk, DECnet, Banyan VINES, and XNS-routed protocols, along with bridging between emulated LANs. Some of the protocols specific to Token Ring are IP from Token Ring to Ethernet, IPX from Token Ring to Ethernet, two-port and multiport source route bridging between Ethernet and Token Ring, IP and IPX multiring, SRB, SR/TLB, SRT, AppleTalk DECnet, Banyan VINES, and XNS protocols. Since this is a handbook dedicated to the entire router, the configuration examples for LANE will not cover every protocol but will include the most widely used implementations.

For a router to become a LEC, there must be a LAN Emulation Configuration Server (LECS) and LAN Emulation Server (LES) configured first. These components can be configured on a Catalyst switch, a Lightstream switch, or a router. Refer to documentation for configuration of the Catalyst or Lightstream switches.

The configuration for each of these LANE services on a router will be given in following sections. A LEC becomes operational after it contacts the LECS and finds the location of the LES of the ELAN the LEC is trying to become part of. All these components are interdependent of each other, so a sound design, along with fault tolerance, should be considered.

The first example will be a router participating in an Ethernet ELAN as a LEC. This gives the router the capability to route between its own emulated LAN and other emulated LANs that it is joined to or other legacy LANs that are defined on the router's other interfaces. For instance, if an ATM switch such as a Lightstream 1010 is at the core of the network providing the LANE services, the router could be used strictly for routing, as shown in Figure 20-1. As shown in the figure, the ATM switch has an LEC definition for two ELANs and the router can join both ELANs and route between them.

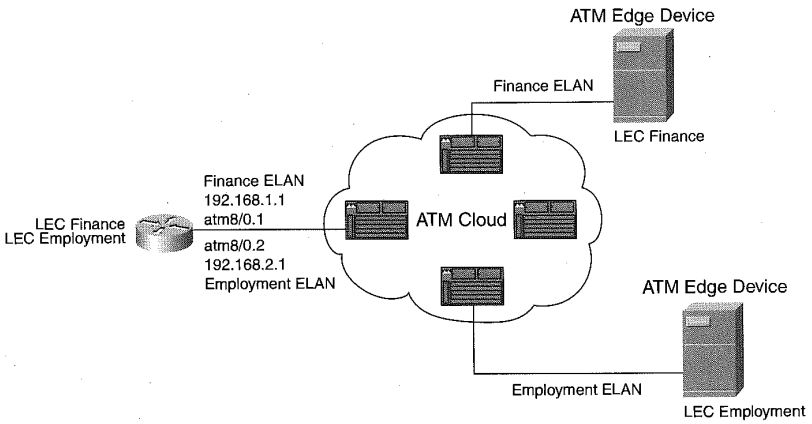


Figure 20-1
Cisco router acting as a LEC on an Ethernet ELAN.

Usually the configuration for a LEC is done as a subinterface on the router's ATM interface. Before the router can become part of the ELAN, a few important notes must be made clear before continuing. The name of the ELAN (case-specific) has to be known as well as the correct medium access sublayer (Ethernet or Token Ring). If the ELAN will eventually participate in MPOA, then the ELAN id should also be known. Once these parameters are determined, the configuration is

interface atm (*slot/**0**.subinterface-number* | *slot/port adapter/**0**.subinterface-number* | *number.subinterface-number*)

The definition for a LEC is usually defined on the subinterface of an ATM interface. To define the subinterface, the slot where the ATM interface is located in the router is specified. The subinterface where the definition is going to reside is given by following the main interface number 0 by a period and then the subinterface number. As is shown, it is different with a port adapter or an interface number on the 4000 series routers. Once the subinterface is created, a LANE client can be bound to the subinterface with the following command:

lane client (**ethernet** | **tokenring**) ***elan-name*** [**elan-id** *id*]

To define the LANE client, the keyword **lane** is used and **client** tells which component of LANE is being configured. The media type used for the ELAN follows the client keyword. Either the **ethernet** or **tokenring** keyword is used to define the media type being emulated. The ***elan-name*** variable should be the name of the ELAN indicated in the LECS that the router is going to join. The **elan-id** *id* variable is used for configuring MPOA.

On the same subinterface of the ATM interface, the protocol should be added. If IP, IPX, or other protocols are needed, then they are added as if the interface was an Ethernet or Token Ring interface.

The example configuration relative to Figure 20-1 is

```
interface atm8/0.1 multipoint
 lane client ethernet finance elan-id 1
 ip address 192.168.1.1 255.255.255.0
!
interface atm8/0.2 multipoint
 lane client ethernet employment elan-id 2
 ip address 192.168.2.1 255.255.255.0
```

In the command sequence, subinterface number one of an ATM interface in the eighth slot is being configured. Subinterface number one is going to be a LANE client of the finance ELAN. The finance elan has an elan identifier of 1. The IP protocol will run on subinterface one, which has an IP address of 192.168.1.1 and a mask of 255.255.255.0. The command sequence also has another subinterface that is number two. Subinterface number two will be a LANE client of the employment ELAN. The IP protocol will run on subinterface number two. The IP address of the subinterface is 192.168.2.1 and the mask is 255.255.255.0. The router will have the capability to route IP packets between the two subinterfaces.

Once the interface is enabled, the following sequence takes place. The LEC establishes a direct virtual channel connection (VCC) usually though the ATM Forum's well-known address to the LECS. The LEC attempts to find the ATM address of the LES. If the address is found, then the LEC creates a direct VCC with the LES and attempts to join. If the join is successful, then the LEC does a LE_ARP_REQUEST to the LES to establish a multicast send VCC with the BUS. Now the LEC has address resolution and multicast capabilities as part of the ELAN.

LANE may not look attractive at first because it appears to lack redundancy. This is not the case since Cisco's HSRP can be implemented within LANE. LANE emulates layer 2 of the OSI Reference Model, so the configuration is the same as if two routers were Ethernet-attached. An example of HSRP within LANE is shown in Figure 20-2.

The configuration for the HSRP example of Figure 20-2 is as follows:

Router R1 configuration applied to Figure 20-2:

```
interface atm8/0.1 multipoint
 lane client ethernet finance elan-id 1
 ip address 192.168.1.2 255.255.255.0
 standby 1 ip 192.168.1.1
 standby 1 preempt
```

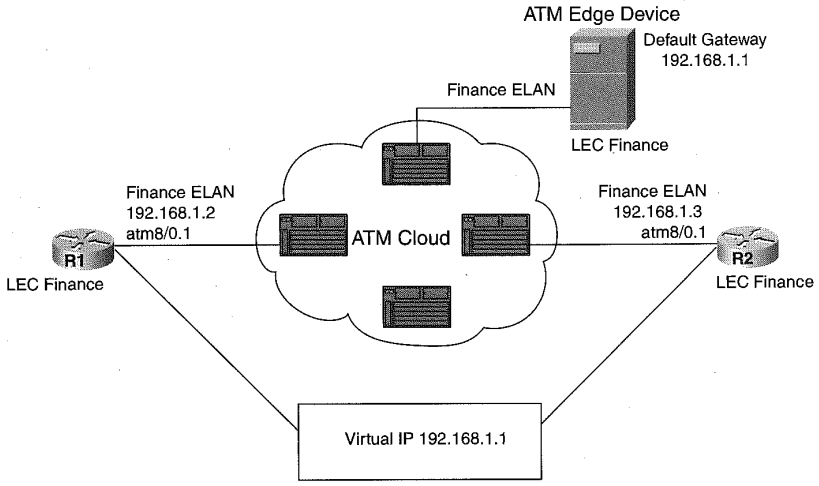


Figure 20-2
HSRP used in a LANE configuration.

```
standby 1 priority 150
standby 1 authentication corporate
```

Router R2 configuration applied to Figure 20-2:

```
interface atm8/0.1 multipoint
 lane client ethernet finance elan-id 1
 ip address 192.168.1.3 255.255.255.0
 standby 1 ip 192.168.1.1
 standby 1 preempt
 standby 1 authentication corporate
```

The configuration above is similar to that of HSRP between two Ethernet interfaces. LANE emulates what Ethernet does on layer two; as long as the two routers are part of the same ELAN, packets can flow between them as if they were physically connected. The connection now becomes more logical than physical. The HSRP hello packets can flow back and forth between the routers across the ATM cloud. The difference in the definition lies on both ATM subinterfaces. Each router becomes a LANE client of the same ELAN. Both interfaces have an IP assigned to them, as any Ethernet interface would, and the standby group definitions specified for HSRP are the same as those used for Ethernet interfaces.

Of course, the router does more than just route between two Ethernet ELANs. The following examples show the configuration of a LEC on a Token Ring ELAN and routing between ELANs and legacy networks.

A Token Ring LEC is needed for routing between Token Ring interfaces. This would be done in the case of having multiple physical rings on separate routers that become part of the same logical ring. An example of this is in Figure 20-3. If a router has more than one physical Token Ring interface, then a virtual ring is used. This way the traffic can flow within its ring and bridge to the virtual ring. The logical definition of the LANE Token Ring interface is applied to an ATM subinterface.

The configuration for a token ring ELAN is as follows:

interface atm (*slot* / **0**.*subinterface-number* | *slot*/*port adapter* / **0**.*subinterface-number* | *number*.*subinterface-number*)

The definition of a Token Ring ELAN is also defined on the subinterface of an ATM interface. The subinterface is defined the same as an Ethernet ELAN is defined on a subinterface. The **interface** keyword indicates that the configuration is applied to an interface. The **atm** keyword signifies that the interface is an ATM interface. *Slot* is a number that tells which slot the ATM module is in, or *number* tells what interface number. The **0** is the indication of the main interface. The period followed by *subinterface* describes what logical subinterface the definition will be on. The command for configuring a Token Ring client is as follows:

lane client tokenring e*lan-name* **elan-id** *id*

The Token Ring ELAN definition differs from Ethernet by indicating the ELAN type from the keyword **tokenring**. The *elan-name* is defined being case-sensitive and identifies the ELAN being joined. The **elan-id** *id* is used for future configurations of MPOA.

The same protocols that can be configured on a physical Token Ring interface can also be configured to operate on the LANE definition of Token Ring. IP, IPX, or other protocols can be defined.

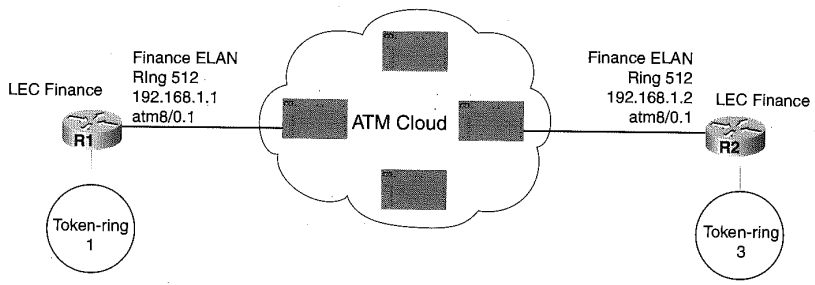


Figure 20-3 Cisco routers participating in a Token Ring ELAN configuration.

The example configuration relative to Figure 20-3 is as follows:

Router R1 configuration applied to Figure 20-3:

```
interface atm 8/0.1 multipoint
 lane client tokenring finance elan-id 1
 ip address 192.168.1.1 255.255.255.0
 source-bridge 512 1 1
 source-bridge spanning
!
interface tokenring 9/0
 ring-speed 16
 source-bridge 1 1 512
 source-bridge spanning
```

Router R2 configuration applied to Figure 20-3:

```
interface atm 8/0.1 multipoint
 lane client tokenring finance elan-id 1
 ip address 192.168.1.1 255.255.255.0
 source-bridge 512 3 3
 source-bridge spanning
interface tokenring 9/0
 ring-speed 16
 source-bridge 3 3 512
 source-bridge spanning
```

In Figure 20-3, the two separate routers each have a physical Token Ring interface. The separate Token Ring can be bridged into one logical ring between the routers. As can be seen, both Router R1 and Router R2 join the same Token Ring Elan called finance. Each router Token Ring physical interface is bridged with the Virtual Ring defined on the ATM subinterface. This is done by using the source-bridge command. The command specifies that the first number indicate the local ring, which is unique to the interface. The second number indicates what bridge is going to be used. The third number indicates the target ring for the traffic.

On Router R1, the local Ring 512 on the ATM interface sends traffic through bridge 1 to the target ring 1. Traffic flows to the physical Token Ring interface by being the local ring, which is the target of the Token Ring ELAN. The traffic comes through bridge 1. The traffic also flows in the other direction from local ring 1 through bridge 1 to the target ring 512. The traffic can then flow on layer 2 between the routers by the Token Ring ELAN. Because all the traffic flows between the routers, one interface can be given an IP address to route IP packets in the logical ring.

If one of the routers has two physical Token Ring interfaces, then a virtual ring could be created to do source-route bridging between the two interfaces. This is shown in Figure 20-4 and the configuration is as follows:

Router R1 configuration applied to Figure 20-4:

```
source-bridge ring-group 4
!
interface atm8/0.1 multipoint
 lane client tokenring finance
 ip address 192.168.2.1 255.255.255.0
 source-bridge 1024 1 4
 source-bridge spanning
interface tokenring 9/0
 ring-speed 16
 source-bridge 1 1 4
source-bridge spanning
interface tokenring 9/1
 ring speed 16
 source-bridge 3 3 4
 source-bridge spanning
```

The configuration above is useful when the router is connected logically to other devices that have Token Ring running on them. A Catalyst or another router may have an uplink to an ATM cloud. The ATM subinterface provides the versatility to configure the upper-layer protocols such as AppleTalk or IP as well the capability to make a network more organized on layer two. As was shown, this all can be done through LEC definitions on the router.

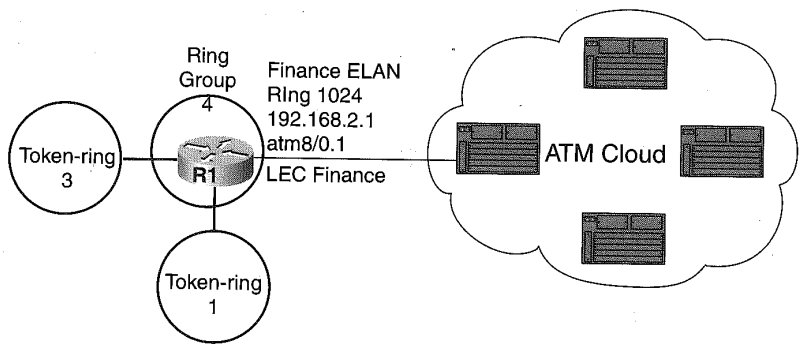


Figure 20-4
Bridging two Token Ring interfaces and a Token Ring ELAN using a virtual ring group.

Configuring a LAN Emulation Configuration Server (LECS)

As previously described, a LEC joins an ELAN after finding the LES. The LEC does this by setting up a direct VCC to a LECS through ILMI. The LECS gives the LEC the ATM address of the LES server it is looking for. Without the LECS there would be a need for the broadcast mechanism to get address resolution for the LES. Since ATM is connection-oriented, this is not part of the protocol for initial LANE setup. It is easier for the LECS to be the central control point for LEC devices to find their designated LES.

The LECS is actually a database of ATM addresses of the LES that can be indexed by LEC devices. The database contains the address of the LES associated with ELAN names. This gives a LEC the capability to join an ELAN by name. Recall that the configuration of a client is

lane client ethernet finance

with finance being the name of the ELAN.

The LEC, which is the router, has to find the address for the LES of finance. The way the LEC finds this address is by connecting to the LECS through ILMI. Then from the address associated with finance in the LECS the LEC establishes a VCC to the LES to attempt to join the ELAN.

A router, a Catalyst switch, and a Lightstream switch can all be configured to run the LECS. Of course, the example here will use the router. The first thing to do is define the database, and then bind it to an interface. Once the LECS database is bound to an interface, an ATM address needs to be associated with the LECS. Cisco provides the capability to automatically compute the address of the LECS or define an explicit address. This is not important unless you are looking for redundancy, which will be explained a little later. The address assigned to the LECS has to be registered in the ATM switch cloud so the switches know the location of the LECS.

The format of a single LECS with a few ELAN definitions will be given just as a setup to build on later. Redundancy with multiple LECS and LES will be described later in the section. Initially, the database will need a name and then, upon naming the database, the ELAN definitions will be defined. The naming of the database is as follows:

lane database *database-name*

This command is done in global configuration mode. The **lane** keyword specifies that it is a LANE configuration command. The **database** keyword indicates that the definition is a LECS database with a specific name assigned by the *database-name* variable value. To add entries to the new database, the following commands are needed:

name *elan-name* **server-atm-address** *atm-address* **restricted** [**index** *number*]

name *elan-name* **local-seg-id** *segment-number*

ELAN definitions are the records in this database. The *elan-name* variable value defines the name of an ELAN. The address of the LES/BUS services for that ELAN is specified using the *atm-address* variable value. If the ELAN is going to be limited to just a few specified clients, then the keyword **restricted** is used. The index *number* optional operand is used to define which LES/BUS will be the primary for the ELAN in the case of Simple Server Redundancy Protocol (SSRP). If the ELAN is a Token Ring, then a *segment-number* is used.

To define a database with a LES/BUS for an Ethernet ELAN called finance and a LES/BUS for a Token Ring ELAN called employment, the sequence is

```
lane database Mycompany
 name finance server-atm-address 11111111111111111111111111111111111111111
 name employment server-atm-address 22222222222222222222222222222222222222222
 name employment local-seg-id 1250
```

This defines the database, but it will need to be bound to an interface for use. To bind the database to an interface, the following command is used:

interface atm (*slot*/**0** | *slot*/*port adapter*/**0** | *number*)

A LANE database must be defined on the main interface of the ATM module. The previous commands display the syntax for reaching the main ATM interface. This differs from a subinterface in that a subinterface number is not specified. Once the ATM main interface is located, the binding of the LANE database to that interface can be done by the following interface configuration command:

lane config database *database-name*

This command directs the IOS to build a LECS database with the name of the *database-name* variable value. The *database-name* should match the name of the database that was defined in the global command. After the

database is bound to the interface, it now needs an ATM address. The ATM address should be the same addresses that the ATM switches in the ATM cloud point to for the LECS. The following commands define the ATM address:

lane config auto-config-atm-address | lane config fixed-config-atm-address | lane config config-atm-address *atm-address*

Specifying **lane config auto-config-atm-address** indicates that the LECS will run in SSRP and the ATM address of the LECS will be automatically given by the switch.

Entering the **lane config fixed-config-atm-address** command causes the LECS to run in SSRP and the ATM address of the LECS will be automatically given by the device or it will not run in SSRP and only the well-known address is used.

Configuring the **lane config config-atm-address** atm-address directs the LECS to run in SSRP and the address of the LECS to be designated by a 20-byte ATM address.

To finish up the example started above, the definitions are as follows:

```
interface atm 8/0
 lane config database Mycompany
 lane config auto-config-atm-address
```

Using the above configuration example, the device will determine the ATM address.

The above example is given in Figure 20-5 and explains a simple definition of setting up the LECS. Cisco provides redundancy with the LECS and the LES. The redundancy that Cisco provides is called Simple Server Redundancy Protocol (SSRP). The nice feature of SSRP in implementing the LECS is that you can have multiple LECS defined so if one fails, there is a backup LECS.

Cisco's implementation is that of having one master LECS based on a priority. If the master goes down, other LECS are there to backup the master. The definition of the LECS addresses is located on the ATM switches. When a LEC requests the LECS from the ATM switch, it gives the list of LECS addresses and the device should use the LECS with the highest priority. The priority is given by the definition of the LECS address on the ATM switch. This is determined by an index number associated with a LECS address. No matter what address scheme is being used to have the necessary redundancy, each LECS defined should have identical databases.

If the well-known address is the choice for LECS to run, then a few rules should be followed first. There should never be more than one LECS configured on the same switch. This will confuse the ATM switches that LECS refer to because they have the same ATM address. Use the lane config fixed-config-atm-address command for the LECS to listen to the well-known address.

It may be a good idea to use the well-known address scheme so that other third-party products will work with the configuration. The example of using SSRP for LECS redundancy is given in Figure 20-5 and the configuration for this is as follows:

Router R1 configuration applied to Figure 20-5:

```
lane database Mycompany
name finance atm-server-address 11111111111111111111111111111111111111111
name employment server-atm-address 22222222222222222222222222222222222222222
name employment local-seg-id 1250
!
interface atm 8/0
 lane config database Mycompany
 lane config fixed-config-atm-address
```

Router R2 configuration applied to Figure 20-5:

```
lane database Mycompany
name finance atm-server-address 11111111111111111111111111111111111111111
name employment server-atm-address 22222222222222222222222222222222222222222
name employment local-seg-id 1250
!
interface atm 8/0
 lane config database Mycompany
 lane config fixed-config-atm-address
```

The example shows two separate routers acting as the LECS with an ATM address being the well-known address. Both databases on both routers have the same database definition. The priority of which router will be the primary LECS is defined on the ATM switches.

This shows redundancy for the LECS but SSRP also provides for redundancy amongst the LES/BUS servers. The redundancy for the LES/BUS is done through the configuration on the LECS. To implement SSRP for the LES/BUS servers, multiple devices are running the LES/BUS service and are defined in the LECS. In the ELAN definition, each device running a LES/BUS for that ELAN is given by its address. An example of this is shown in Figure 20-6.

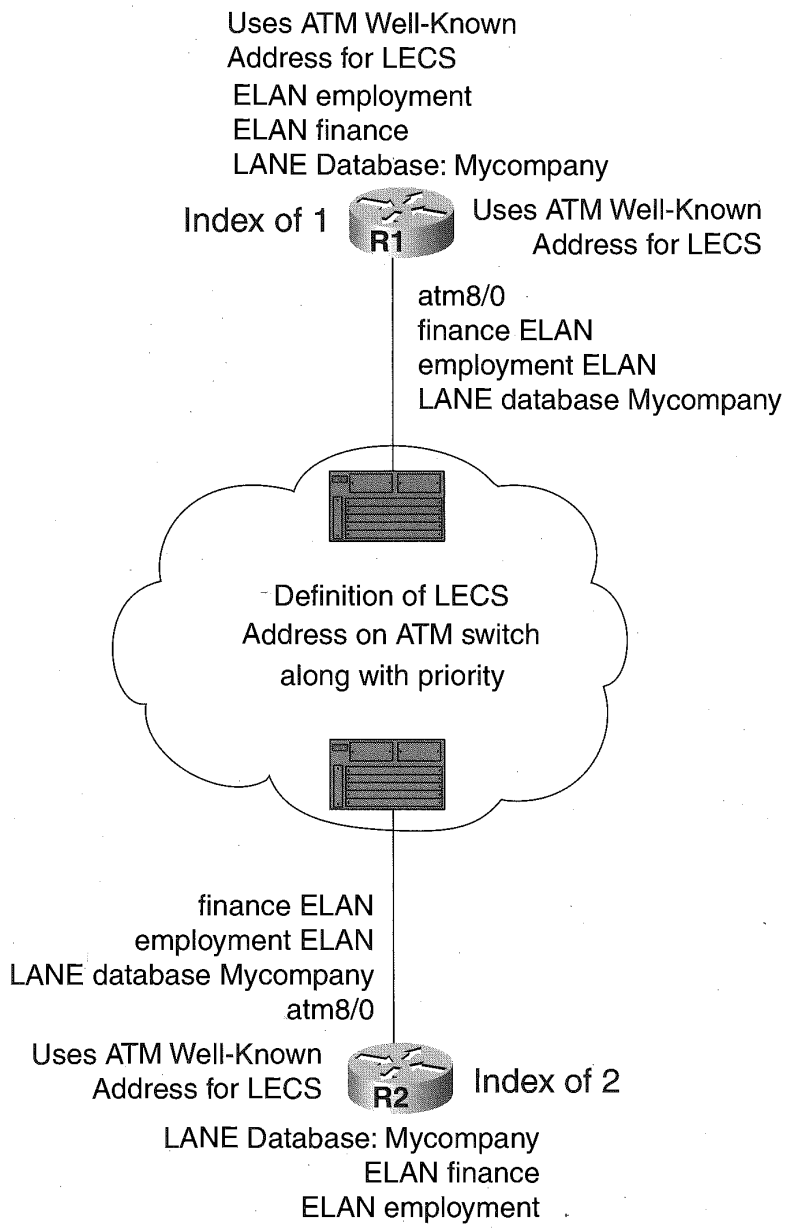


Figure 20-5
Using SSRP for LECS redundancy.

The router configurations for Figure 20-6 follow:

Router R1 configuration applied to Figure 20-6:

```
lane database Mycompany
name finance atm-server-address 33333333333333333333333333333333333333333 index 1
name finance atm-server-address 44444444444444444444444444444444444444444 index 2
!
interface atm8/0
 lane config database Mycompany
 lane config fixed-config-atm-address
```

The index value specified identifies the LES/BUS server priority. The higher priority identifies which LES/BUS server is the master server.

Figure 20-6 contains separate devices denoted by the different ATM addresses. The different devices have LES/BUS definitions for the same ELAN named finance. When a LEC queries this LECS, the LECS responds with the first devices defined with an index of 1. If that device is not available, the LEC is directed to the other device with an index of 2.

There is no restriction to the number of LECS or LES/BUS that can be run. Cisco recommends two, or three at most, for the LECS. The network topology assists in determining the correct number of LECS and LES/BUS that are needed.

The last feature that can be determined in a LECS is restricting which device can join a certain ELAN. If there is a need for only certain devices to join a particular ELAN, that can be specified in the LECS. The configuration is as follows:

lane database *database-name*

name *elan-name* **server-atm-address** *atm-address* **restricted** [**index** *index-number*]

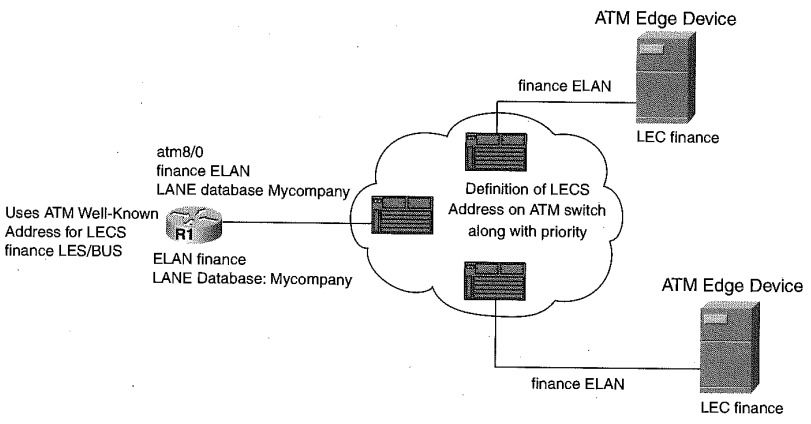


Figure 20-6 LES/BUS services on a Cisco router in support of LANE.

The previous two statements are the same as when defining any LECS database. The only thing that is added is the keyword **restricted** to indicate that an ELAN will be restricted to list LEC devices. To list the LEC devices that will only be able to participate in the ELAN, the following command is used:

client-atm-address *atm-address* **name** *elan-name*

This command defines the clients that can join by ATM address. The keywords **client-atm-address** are given, followed by the actual ATM address of the LEC device and the ELAN that the LEC is trying to join. The *elan-name* should be the same as the *elan-name* used in the LES/BUS definition. Here is an example of this:

```
lane database Mycompany
name finance atm-server-address 555555555555555555555555555555555555555555 restrict
client-atm-address 47.000000000ffe21085643223400.0800.0234.2345.01 name finance
```

This example restricts any other client from joining this ELAN named finance, except the device assigned to that ELAN. This may be useful in some situations, but the management of restricted ELANs could become cumbersome. Now that a way to join an ELAN has been defined, along with the facilities to join an ELAN, there is a need to know how to define the services on which an ELAN operates. The next section will discuss the definition of a LES server.

Configuring a LAN Emulation Server (LES)

In the previous sections, we saw how to configure a LEC for a router to participate in an ELAN. From the configuration, we saw that a LEC has to know the LES/BUS definition associated to the ELAN name. The LECS, which can be configured on a router, provides the definition of the location of the services for an ELAN.

The services of the ELAN are the LES and BUS servers. The LES provides address resolution within the ELAN. The BUS provides the broadcast facility, along with the unknown address resolution. Although the LES and BUS services are different entities, the definition on the router is a single definition for both servers. Like the LEC, the LES is done on an ATM subinterface. The configuration is as follows:

interface atm (*slot*/**0**.*subinterface-number* | *slot*/*port-adapter*/**0**.*subinterface-number* | *number*.*subinterface-number*)

This is the same command given above to get to the subinterface of an ATM interface. This is indicated by providing the slot number of the ATM module followed by the subinterface number that is being created. Once the subinterface is created, the LES/BUS definition can be put in place by using the following interface command:

lane server-bus (**ethernet** | **tokenring**) *elan-name*

This statement defines the LES/BUS server on the router. It distinguishes whether the ELAN will emulate ethernet or tokenring. It also defines the ELAN by name. The name is what is used to configure the LEC devices that will attempt to become part of the ELAN.

An example configuration is

```
interface atm 8/0.1 multipoint
 lane server-bus ethernet finance
```

This is an example of the router acting as the LES/BUS for the ELAN named finance. Using this router configuration, the router R1 in Figure 20-6 can act as both a LEC and a LES/BUS for the finance ELAN.

The design in Figure 20-6 is a feasible configuration of an ELAN, but the single router now becomes the single point of failure. A better implementation would be to use SSRP and HSRP for redundancy. For each LES/BUS definition on a separate router, the LECS has to be updated to show the additional server as well as which server has the highest priority. The redundant design is shown in Figure 20-7.

The router configurations for Figure 20-7 are as follows:

Router R1 configuration applied to Figure 20-7:

```
lane database Mycompany
name finance atm-server-address 11111111111111111111111111111111111111111index 1
name finance atm-server-address 22222222222222222222222222222222222222222 index 2
!
interface atm 8/0
 lane config database Mycompany
 lane config fixed-config-atm-address
!
interface atm8/0.1 multipoint
 lane server-bus ethernet finance
 lane client ethernet finance
 ip address 192.168.1.2 255.255.255.0
 standby 1 ip 192.168.1.1
 standby 1 preempt
 standby 1 priority 125
 standby 1 authentication corporate
```

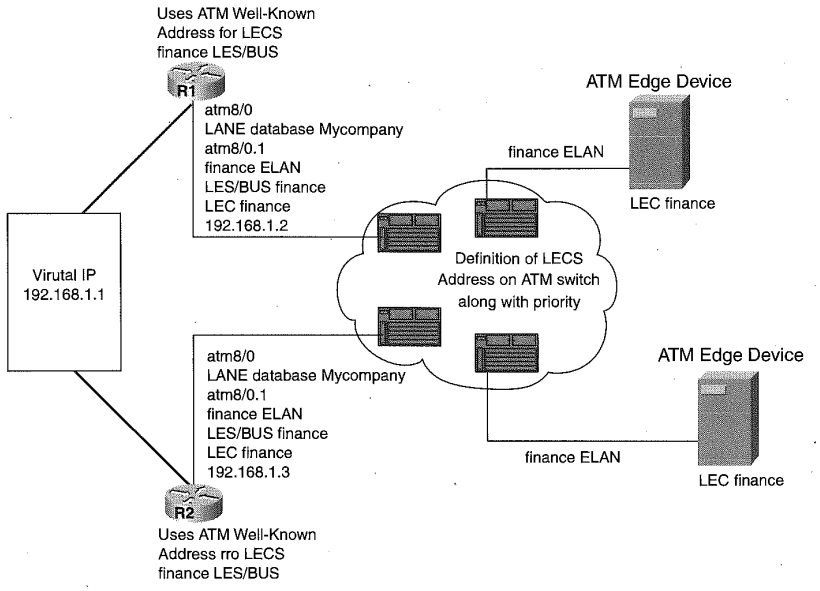


Figure 20-7
Redundancy using SSRP and HSRP.

Router R2 configuration applied to Figure 20-7:

```
lane database Mycompany
name finance atm-server-address 11111111111111111111111111111111111111111index 1
name finance atm-server-address 22222222222222222222222222222222222222222 index 2
!
interface atm 8/0
lane config database Mycompany
lane config fixed-config-atm-address
!
int atm8/0.1 multipoint
 lane server-bus ethernet finance
 lane client ethernet finance
 ip address 192.168.1.3 255.255.255.0
 standby 1 ip 192.168.1.1
 standby 1 preempt
 standby 1 authentication corporate
```

The router configuration examples for Figure 20-7 illustrate not only the use of SSRP for the LES/BUS but also the use of SSRP for the LECS. As can be seen, both databases are identical on both routers. From the example, the priority LECS could not be determined without seeing which router is

defined as the priority LECS in the ATM switches. Both routers also provide SSRP for the LES/BUS. Each router has a LANE server definition for finance ELAN. This is indicated in the LECS database by router R1 having an ATM address of 11111111111111111111111111111.111111111111 and router R2 with 22222222222222222222222222222.222222222222, as the ATM address.

If a LEC is joined to the LES/BUS on R1 and R1 loses connectivity to the ATM cloud, the LEC will join the LES/BUS on R2.

The configurations also employ the use of HSRP for establishing routing redundancy between R1 and R2. If R1 loses connectivity, R2 becomes the primary routing interface for the ELAN. The LEC devices in the ELAN will be able to communicate outside of the ELAN through the router.

LANE gives the capability to utilize existing routing infrastructure to upgrade existing LAN technologies. In this section, we see how an existing router with ATM modules can do all the services that would be needed in a multiprotocol environment. Due to its dynamic nature, ATM may be a solution for backbone upgrade or wide-area connectivity. We will see other ATM protocols in the next sections that may or may not provide the services that LANE does.

Configure ATM over a Serial Interface

In an environment where the majority of the network is ATM, a remote site may be connected to the ATM network using a serial or high-speed serial interface connected to an ATM service data unit (ADSU). The connection is configured for communication through ATM-Data Exchange Interface (ATM-DXI) encapsulation. ATM-DXI gives the capability to transfer multiprotocols through a serial interface using encapsulation. This section will explain the configuration for setting up ATM-DXI encapsulation.

For multiprotocol traffic to flow from a serial interface through an ADSU, ATM-DXI encapsulation must be enabled and a permanent virtual circuit (PVC) must be set up. Once these parameters are in place, each of the protocols that are going to be transmitted out the serial interface have to be mapped to the PVC. When the PVC is initially configured, there are three types of encapsulation to distinguish from:

- snap, the encapsulation of multiprotocol LLC/SNAP compatible with RFC 1483
- nlpid, the network layer protocol identification multiprotocol encapsulation compatible with RFC 1490
- mux, the multiplex encapsulation that sets the interface to encapsulate only one protocol

The configuration of setting up the serial interface for ATM-DXI encapsulation is

interface serial [*slot* /] *number*

The ADSU is physically connected to a serial interface on the router so the configuration is done on through an interface command. The above command specifies the serial interface on which the ATM-DXI configuration will be run.

Example: **ip address** 192.168.1.1 255.255.255.0

Like any other interface, the protocol commands define the type of traffic flow. The protocol configured to operate on the serial interface has to be encapsulated to run through the ADSU. The following commands set up the encapsulation:

encapsulation atm-dxi

The type of encapsulation for a serial interface connected to an ADSU is specified in this command. The encapsulation is an ATM data exchange interface. The connection from the serial interface to the ATM cloud at the remote end will be done through a PVC. The PVC will be created and the different protocols will be encapsulated through it. The command for creating a PVC on a serial interface that is connected to the ATM cloud via a ADSU is

dxi pvc *vpi vci* (**snap** | **nlpid** | **mux**)

The type of the PVC is data exchange interface (**dxi pvc**). The PVC is identified by the virtual path identifier (*vpi*) followed by the virtual circuit identifier (*vci*). This enables the devices in the ATM cloud to build a virtual channel from the router serial interface to the end destination through the ATM cloud. The **snap** encapsulation protocols indicate the compatible protocols given in RFC 1483. The **nlpid** protocol is also a multiprotocol encapsulation technique but is compatible with RFC 1490. RFC 1490 is similar to

RFC 1483, but RFC 1490 refers to Frame Relay and RFC 1483 refers to ATM encapsulation. Cisco uses **nlpid** for backward compatibility with older IOS software. Specifying the **mux** indicates that only one protocol will be encapsulated over the PVC.

dxi map *protocol protocol-address vpi vci* [**broadcast**]

The **dxi map** command maps a protocol defined on the serial interface to the PVC created on the interface. The *protocol* followed by the address of the protocol (*protocol-address*) is given to indicate which protocol will flow within the PVC. The PVC created on the interface is identified by a virtual path identifier (*vpi*) and a virtual circuit identifier (*vci*). Specifying the **broadcast** keyword indicates that broadcast traffic will flow over the PVC.

An example of this configuration is

```
interface serial 1
 ip address 192.168.1.1 255.255.255.0
 ipx network C0A80101
 encapsulation atm-dxi
 dxi pvc 0 22 nlpid
 dxi map ip 192.168.1.2 0 22 broadcast
 dxi map ipx C0A80102 broadcast
```

This example demonstrates the use of ATM-DXI encapsulation of IP and IPX over VPI 0 VCI 22. As you can see, the protocol is mapped to the address of the other end of the serial link. The **broadcast** statement enables broadcast packets over the serial link.

Configuring Classical IP

Classical IP over ATM offers the capability to transmit the IP protocol over ATM. The transmission uses an ATM ARP server for address resolution. Each node on the Logical IP Subnet (LIS) registers with the ARP server by including the server address in the nodes configuration. The router can be both a node and ARP server and it can perform routing and ARP services for each LIS with which it participates.

Classical IP does not provide redundancy that includes HSRP. You may want to use Classical IP if a vendor only supports Classical IP or in the event of remote data backup to an external source from a server. Classical IP also offers connectivity at SONET speeds.

An example of a router registering with an ARP server as a node is shown in Figure 20-8. In this example, the router performs the routing for the LIS. Before configuring the router as a node, the ATM NSAP address of the ARP server must be known. An end system identifier (ESI) also must be chosen for the router. This can be the MAC address with a one-byte identifier from 00 to FF. The MTU size for Classical IP is 9180. This will cause a problem if the MTU size of the main ATM interface is less than 9180. If Classical IP is going to be configured on a router, the main ATM interface should be set to 9180. The configuration for the router participating in a Classical IP LIS is

interface atm (slot/**0**.subinterface-number | slot/port adapter/**0**.subinterface-number | number.subinterface-number)

Similar to LANE, the configuration of Classical IP is usually done as a subinterface. The command above shows the syntax for creating the subinterface for configuring Classical IP. The keywords interface and atm are used to indicate which type of interface. The location is specified by indicating the slot number followed by the subinterface number:

atm esi-address *esi-address*

An ATM device receives an NSAP address through ILMI or the device already has an NSAP address associated with it from the manufacturer. Since Classical IP does not use ILMI, it is good to identify the Classical IP interface with an end system identifier followed by two selector bytes. The *esi-address* value is seven bytes, or 14 hexadecimal digits. A good practice is to define the end-system identifier as the MAC address followed by a selector byte. This provides a unique address since it is MAC-based:

atm arp-server (**self** | **nsap** *server-nsap-address*)

This command specifies where the ARP server is located. If the router is going to act as the ARP server, then the keyword **self** is used. If another device is acting as the ARP server, then the NSAP of that address is given so the router can register with that device. A network configuration example is diagrammed in Figure 20-8.

The following configuration for the router as a registered node, as shown in Figure 20-8, is

```
interface atm8/0.1 multipoint
 ip address 192.168.1.1 255.255.255.0
 atm esi-address 0010A6A5F820.01
 atm arp-server nsap 47.00011111111111111111111111.111111111111.00
```

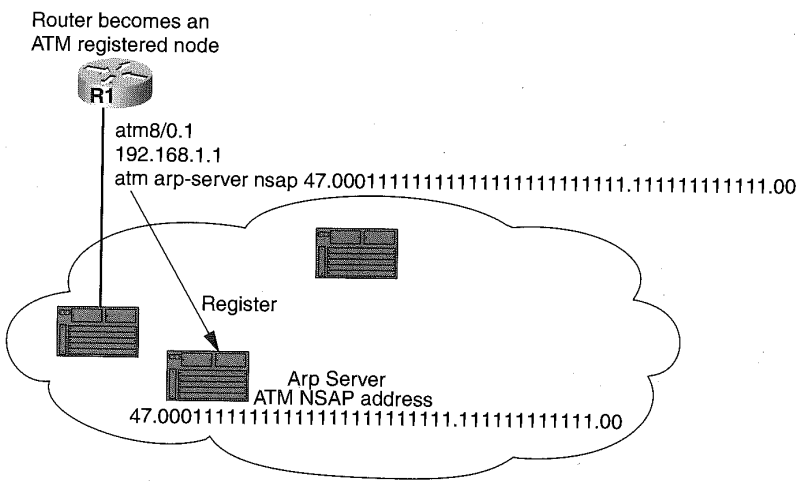


Figure 20-8
Router registering with an ARP server to support Classical IP.

In the example, the router registers with a device that has a different NSAP address specified in the configuration. If the router acts as the ARP server, then the keyword **self** would be used instead.

The router can also act as the ARP server and perform the routing for the LIS. This is shown in Figure 20-9.

The following router configuration is applied to Figure 20-9:

```
interface atm8/0.1 multipoint
 ip address 192.168.1.1 255.255.255.0
 atm esi-address 0010A6A5F820.01
 atm arp-server self
```

These examples assume that Classical IP is configured in an environment with switch virtual circuits (SVC). In an environment where the ATM connections are permanent virtual circuits, the configuration is different. There must be PVCs set up to get reverse ARP datagrams. The default time that reverse ARP datagrams will be sent is every 15 minutes. The configuration for a PVC in this environment is

interface atm (*slot*/**0** | *slot*/*port adapter*/**0** | *number*)

PVCs can only be configured on the main ATM interface of the router. The previous command specifies the interface by giving the interface type and the slot on which the interface resides. There is no subinterface, so 0 or just the number of the interface is used.

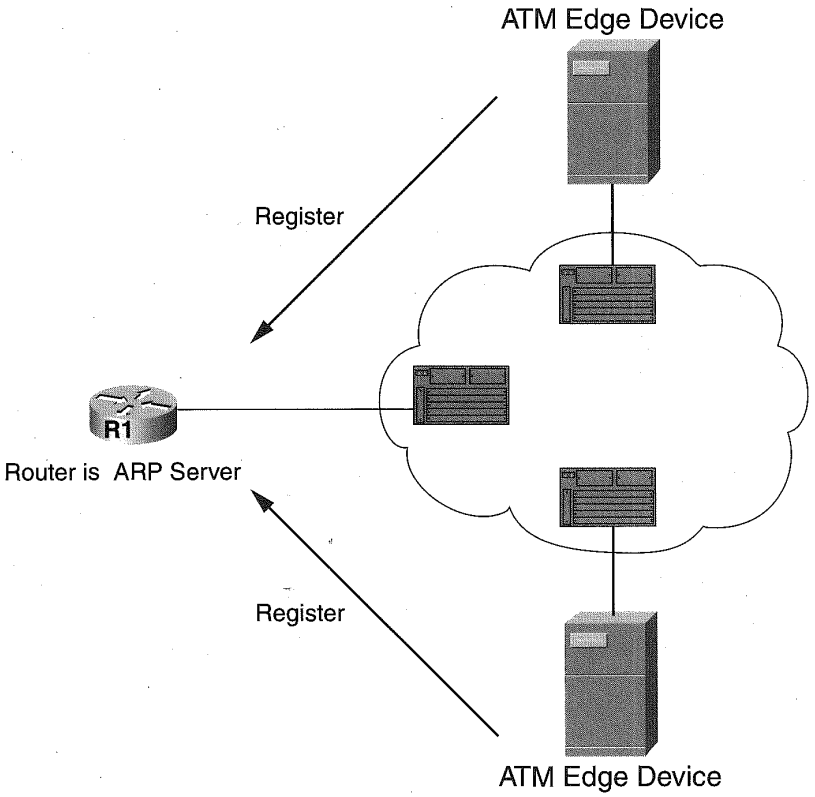


Figure 20-9
Router acting as the ARP server and routing node.

atm pvc *vpi vci* **aal5snap inarp** (*minutes*)

The configuration of the PVC for Classical IP is identified as an ATM PVC. The PVC is then represented by the virtual path identifier (*vpi*) and virtual circuit identifier (*vci*). The type of ATM traffic that will flow over the PVC will be ATM adaptation layer 5 snap (**aal5snap**). The **inarp** keyword enables reverse ARP updates to be received in intervals of so many minutes.

An example configuration for PVC environment is

```
interface atm8/0
 ip address 192.168.1.1 255.255.255.0
 atm pvc 1 0 20 aal5snap inarp 10
 atm pvc 2 0 21 aal5snap inarp
```

This example defines PVC 1 as receiving reverse ARP datagrams every 10 minutes and PVC 2 will respond to reverse ARP datagrams. A map list does not need to be created because this is IP only.

Configuring Multiprotocol over ATM (MPOA)

Why use MPOA in an ATM environment? What MPOA offers is the capability to circumvent the routing device so the edge devices can make the layer 3 decisions using cache entries. The operation of MPOA is simply that MPOA clients in the network query a MPOA server to find a direct connection to another client, circumventing the router. This process is not done without an initial setup. Before explaining the MPOA process, a few terms need to be defined.

MPOA client (MPC): A multiprotocol over an ATM client (such as an edge device or ATM-attached device that is MPOA-compatible).

MPOA server (MPS): A multiprotcol over an ATM server (such as a router or ATM switch that contains the information of the MPOA clients).

Ingress edge device: The location in the network where traffic flows into the MPOA system (an MPOA edge device like a Catalyst).

Egress edge device: The location in the network where traffic flows out of the MPOA system (an MPOA edge device like a Catalyst).

Nonbroadcast mulitiaccess (NBMA): A network that has multiprotocol capabilities but does not have the facility for broadcasts. This is usually a connection-oriented environment.

Next Hop Resolution Protocol (NHRP): The protocol that routers use to find the device connections in a non-broadcast multi-access (NBMA) environment.

Using these terms, the traffic flow in an MPOA system will be explained. The flow of the MPOA traffic can be seen in Figure 20-10. As shown in the figure, a host would like to communicate with its server. The host is attached to a Catalyst edge device on one end of the network and the server is attached to a Catalyst edge device on the other end of the network. Both

Catalyst devices are MPOA clients (MPC) that have a LEC defined on different ELANS. The Catalyst with the host attached is the ingress edge device because the host is initiating traffic to the server, so the traffic is flowing into the MPOA system. The Catalyst with the server attached is the egress edge device because it is receiving traffic from the MPOA system. The router acts as the MPOA server (MPS), which has the connection information of both clients. As the figure shows, this is an non-broadcast multiaccess environment because it is ATM.

Normally, if both Catalysts are defined to join separate ELANs, then a router would have to route the traffic between both ELANs. With MPOA, a direct VCC can be set up between the Catalysts to do the work of layer 3 by the MPS giving the MPCs the connection information. The process is as follows:

1. The MPC of the hosts sends an MPOA request to the MPS.
2. The MPS finds either a different MPS through NHRP or sends a cache-imposition request to the other MPC of the server.

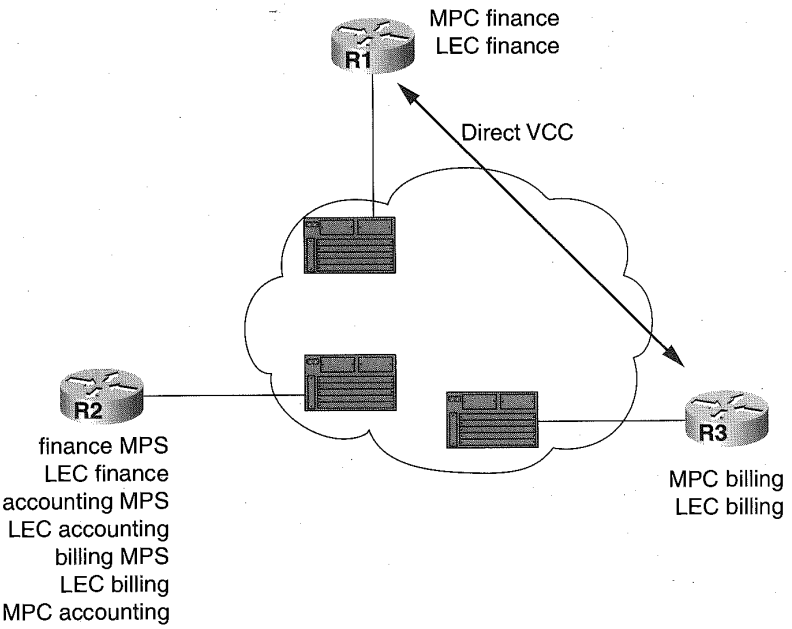


Figure 20-10
An MPOA network configuration.

3. The cache-imposition reply is then sent back to the MPS.
4. The MPOA resolution is then sent to the MPC of the host.
5. The MPC of the hosts then establishes a direct VCC to the MPC of the server instead of going through the router.

Each time the hosts want to communicate with the server the MPC of the hosts has the direct VCC to the MPC of the server cache for direct communication.

Configuring a MPOA Client (MPC)

Usually a router is not configured to be the MPC. Edge devices servicing end users would be the best candidates for MPCs. If LANE is the existing protocol on the ATM, the router would know of the other router in the network through LANE address resolution protocol. An instance when the router might be a feasible MPC is when a central MPOA server (MPS) is located among many wide area routers, as in Figure 20-10. If the wide area connectivity is SONET, then through the MPS the routers could all be MPCs, which communicate through the wide area via direct cache VCCs. This way the broadcast traffic can be reduced on the wide area.

The configuration for the MPC is listed here. In the LECS database, enter the following:

name *elan-name* **elan-id** *id*

When building the LECS database, the elan name is given, followed by an ELAN *id*. This statement is used after the *elan-name* variable value and the associated LES/BUS server ATM address is indicated. The **elan-id** is used for the MPOA configuration. On the subinterface of the router, the following is used:

lane server-bus ethernet *elan-name* **elan-id** *id*

The LES/BUS server for an ELAN is given the associated elan id that was given in the LECS. The **elan-id** is used for the operation of MPOA:

mpoa client config name *mpc-name*

After the ELANs are identified with ELAN identifiers, the MPOA client is configured under global configuration. The MPOA client is identified by a unique name. Once the MPOA client is identified, it needs to be bound to an interface. This is done with the following two commands:

interface atm (*slot/0* | *slot/port adapter/0* | *number*)

Since the MPOA client has to be bound to a hardware address, the main ATM interface is used. The main interface is identified by slot number followed by 0 or by an interface number:

mpoa client name *mpc-name*

On the main ATM interface, the MPOA is then bound to the hardware address by giving the MPOA client name. These statements associate a physical address with a MPOA client.

Now that the client has a hardware address, the MPOA client will be bound to a LEC. To bind a MPOA client to a LANE client, the following would be done:

interface atm (*slot*/**0**.*subinterface-number* |
slot/port adapter/**0**.*subinterface-number* | *number*.*subinterface-number*)

The LEC definition is usually done on a subinterface, which is identified by slot number followed by subinterface number. This can be seen in earlier sections of this chapter.

lane server-bus ethernet *elan-name*

The LES/BUS definition does not have to be defined on the same interface of the router as the LEC. This statement is shown so it is clear that a LES/BUS is needed for the LEC to function. This statement can be on another router or on an ATM switch. The command indicates that it is a LANE LES/BUS server that emulates Ethernet and is identified by a name. Once the server is set up, the client can be configured:

lane client ethernet *elan-name*

The client configuration is the same in a pure LANE configuration. The client is given with the type of emulation and identified with an *elan-name*. Only one LEC can be defined for each subinterface. The *elan-name* of the LEC has to match what is defined in the LECS database. After the LEC with the associated LES/BUS services is in place, the MPOA client is bound to the LEC client.

lane client mpoa client name *mpc-name*

The MPOA client is bound to the LEC by defining the MPOA client on the same subinterface as the LEC client to which it is being bound. The MPOA client is bound by giving the keywords, **lane client mpoa**. The bound MPOA client is then defined by the MPOA client name bound to the main ATM interface.

Using these configuration commands and Figure 20-10, the following is the example configuration:

The following router configuration are applied to Figure 20-10:

Router R1 configuration applied to Figure 20.10:

```
mpoa client config name widearea1
interface atm8/0
 mpoa client name widearea1
 atm pvc 1 0 5 qsaal
 atm pvc 2 0 16 ilmi
interface atm8/0.1 multipoint
 ip address 192.168.1.1 255.255.255.0
 lane client ethernet finance elan-id 100
 lane server-bus ethernet finance elan-id 100
 lane client mpoa client name widearea1
```

Router R2 configuration applied to Figure 20-10:

```
lane database SONET
name finance server-atm-address 47.000000001111111111111111.111111111111.11
name finance elan-id 100
name accounting server-atm-address 47.000000002222222222222222.2222222222222.22
name accounting elan-id 101
name billing server-atm-address 47.000000003333333333333333.333333333333.33
name billing elan-id 102
mpoa client config name widearea2
interface atm8/0
 atm pvc 1 0 5 qsaal
 atm pvc 2 0 16 ilmi
 lane config fixed-config-atm-address
 lane config database SONET
 mpoa client name widearea2
interface atm8/0.1 multipoint
 ip address 192.168.2.1 255.255.255.0
 lane client ethernet accounting elan-id 101
 lane server-bus ethernet accounting elan-id 101
 lane client mpoa client name widearea2
```

Router R3 configuration applied to Figure 20-10:

```
mpoa client config name widearea3
interface atm8/0
 atm 1 0 5 qsaal
 atm 2 0 16 ilmi
 mpoa client name widearea3
interface atm8/0.1 multipoint
 ip address 192.168.3.1 255.255.255.0
 lane client ethernet billing elan-id 102
 lane server-bus ethernet billing elan-id 102
 lane client mpoa client name widearea3
```

This configuration sets up the three MPOA clients. After router R2 is configured to be the MPS, the routers can set up direct VCC between each other via the MPS, instead of being directly connected by one ELAN. This eliminates broadcast traffic among the routers.

Configuring a MPOA Server (MPS)

The MPS does the work of finding the path from end to end through NHRP. The MPS converts MPOA requests and replies from the MPCs in the network. When the path is found, the MPOA sends the resolution to the MPC that initiated the request. The configuration of the MPS is
LANE configuration:

name *elan-name* **elan-id** *id*

The same is needed for the MPOA server as the MPOA client. In the LECS database, each ELAN that will participate in MPOA has to be assigned an ELAN identifier. This is done using the previous statement after the definition of the LES/BUS server. The structure of the LECS is given in an earlier section. The ELAN is identified by the same name as the LES/BUS server and the elan-id is assigned by an administrator based on a MPOA design:

lane server-bus (**ethernet** | **tokenring**) *elan-name* **elan-id** *id*

It is shown again that a LES/BUS service must be up and running for MPOA to operate with LANE. The LES/BUS configuration defines the server along with the type of ELAN. The ELAN is identified by name and given an identifier for MPOA.
MPS configuration:

mpoa server config name *mps-name*

The MPOA server configuration is similar to the client because it needs a globally defined name. This name is given in global configuration mode using the keyword **mpoa server config name** with the name that will represent the MPOA server in the ATM cloud. Once the MPOA server has been identified by name, it needs to be bound to a hardware address.

interface atm (*slot*/**0** | *slot*/*port adapter*/**0** | *number*)

The MPOA server is done on the main ATM interface. The main interface has a hardware address associated to it so the MPOA can inherit that address. To bind the MPOA server to the main ATM interface, the interface command is used. This is the main ATM interface, so the subinterface number is eliminated and 0 or the interface number is used.

mpoa server name *mps-name*

Once the configuration mode is created for the main ATM interface, the MPOA server is bound to the interface by identifying the MPOA server that

was defined in global mode. The correlation between the two statements is the mps-name. After the MPOA server has a bound hardware address, it can now be bound to a LANE client.

interface atm (*slot*/**0**.*subinterface-number* | *slot*/*port adapter*/**0**.*subinterface-number* | *number*.*subinterface-number*)

The lane client usually resides on a subinterface so the binding of an MPOA server to a lane client is done on a subinterface. The command for getting to an ATM subinterface is as shown above and has been explained in various other sections in this chapter.

lane client mpoa server name *mps-name*

The MPOA server is bound to a LANE client the same way an MPOA client is. The name of the LANE client is not specified and the definition of the MPOA server is placed on the same subinterface as the LANE client. This binds the MPOA server to the LANE client implicitly.
The following is the the additional configuration to finish the example in Figure 20-10.

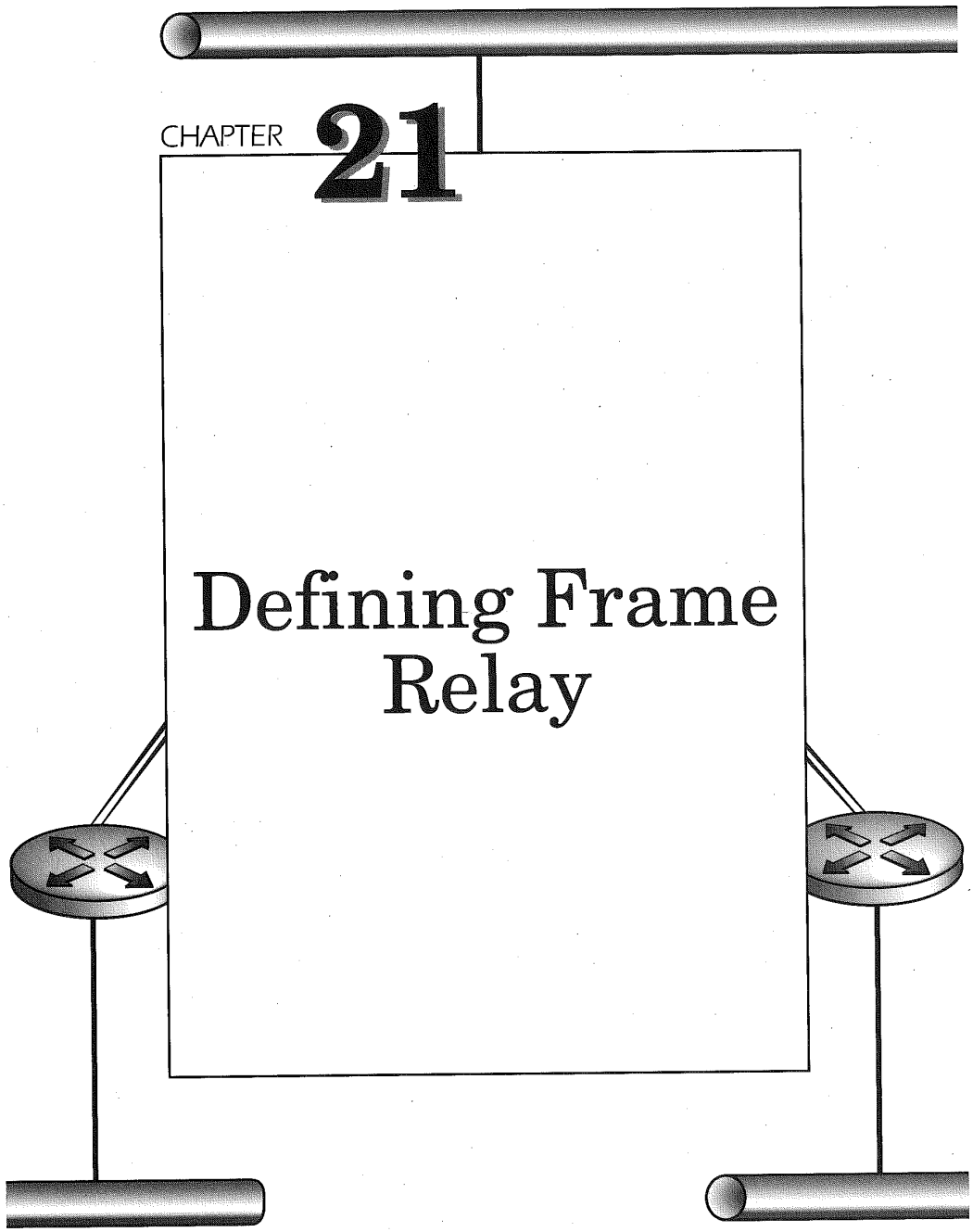
Additional configuration for router R2 in Figure 20-10:

```
mpoa server config name corporate
interface atm8/0
 mpoa server name corporate
interface atm8/0.1 multipoint
 lane client mpoa server name corporate
interface atm8/0.2 multipoint
 ip address 192.168.1.2 255.255.255.0
 lane client mpoa server name corporate
 lane client ethernet finance elan-id 100
interface atm8/0.3 multipoint
 ip address 192.168.3.2 255.255.255.0
 lane client mpoa server name corporate
 lane client ethernet billing elan-id 102
```

With this configuration, if router R1 wants a connection to router R3, the MPS on router R2 requests and negotiates the connection between the two. This way router R2 does not route between the two routers, but instead a direct VCC is set up between the two for traffic flow.

CHAPTER 21

Defining Frame Relay



Frame Relay has become an integral part of connecting remote locations over a WAN. Its cost-effectiveness and capability to provide traffic shaping using Cisco IOS has proven this technology to be a meaningful backbone infrastructure. In this chapter, the various Frame Relay configurations are discussed in concert with Frame Relay-specific characteristics.

A Simple Frame Relay Configuration

In its simplest form, Frame Relay provides a single circuit from one location to another using a public Frame Relay network. Figure 21-1 diagrams this simple configuration. Frame Relay-connected Cisco routers attach via synchronous line communications to a Frame Relay switch. The Frame Relay provider configures a permanent virtual circuit (PVC) through the Frame Relay public network connecting the two routers. To properly configure the routers, the router administrator must be given some information from the Frame Relay network provider.

First and foremost is the Local Management Interface (LMI) type used by the connecting frame relay switch. The LMI updates passed by the switch to the Cisco router will dynamically update the router to the data link connection identifier (DLCI) being used for the local router PVC connection. The Cisco IOS takes advantage of the Inverse ARP function of the Frame Relay switch to determine the remote IP address of the router on the far-end of the PVC. Using Inverse ARP allows the Cisco IOS to automatically map the local DLCI to the associated remote IP address.

The following router configuration example illustrates the Cisco IOS commands necessary to make the router connection in Figure 21-1.

Router R1 configuration applied to Figure 21-1:

```
interface Serial0
 ip address 172.20.12.2 255.255.255.0
 encapsulation frame-relay
!
router rip
 network 172.20.0.0
```

Router R2 configuration applied to Figure 21-1:

```
interface Serial0
 ip address 172.20.12.3 255.255.255.0
 encapsulation frame-relay
```

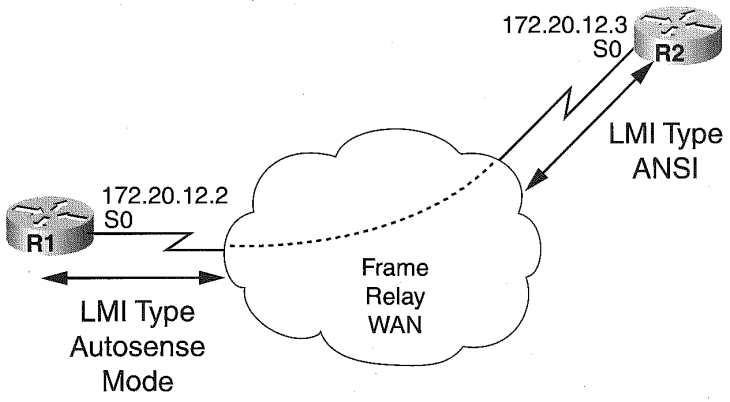


Figure 21-1
A basic Frame Relay network configuration.

```
 frame-relay lmi-type ansi
!
router rip
 network 172.20.0.0
```

The router configurations for Figure 21-1 take into account that inverse ARP is enabled by default, along with IP split-horizon being enabled by default. The Cisco IOS is made aware that the serial interface being used is a Frame Relay connection from the inclusion of the encapsulation frame-relay command. The format of the command to enable Frame Relay on a serial interface is as follows:

encapsulation frame-relay cisco | ietf]

The keyword cisco is used by default, should either cisco or ietf not be specified on the command. Using the Cisco keyword causes the Cisco IOS to encapsulate the data using a Cisco proprietary encapsulation technique that includes a four-byte header, which is made up of a two-byte DLCI field and a two-byte field identifying the type of data being encapsulated. The cisco keyword is enabled by default and can only be used with Cisco Systems Frame Relay hardware.

The second keyword ietf is used to allow the Cisco IOS to interact with non-Cisco equipment using the Internet Engineering Task Force (IETF) standard RFC 1490. If the encapsulation type needs to be changed, the serial interface should be modified to a shutdown state prior to changing the

encapsulation type. Issuing a no shutdown command after changing the encapsulation type ensures the interface will use the new method. Not performing this scenario causes the router to continue to use the current encapsulation type method. In Figure 21-1, both routers are Cisco routers and therefore the encapsulation default of Cisco is appropriate.

IP split-horizon on Frame Relay interfaces is disabled by default. This allows routing updates to be sent in and out of the same interface for the same destination. IP split-horizon works for the Figure 21-1 scenario because it only involves the use of IP protocol. If bridging or IPX were in use on this connection, split horizon would have to be enabled.

To enable the use of split-horizon on Frame Relay connections, the following command must be entered on the Frame Relay interface definition:

no ip split-horizon

The LMI type is specified on the router R2 configuration. As of Cisco IOS release 11.2, however, the LMI type is automatically determined during initial connectivity to the Frame Relay switch. The autosense capability of Cisco IOS release 11.2 or higher is disabled by explicitly configuring the LMI type on the interface.

frame-relay lmi-type {ansi | cisco | q933a}

The **ansi** keyword is used when the switch utilizes the Annex D ANSI standard T1.617 LMI protocol. The **cisco** keyword is used when communicating with a Cisco switch or any other Frame Relay switch that supports the Cisco LMI type. The **q933a** keyword is specified when the switch supports the ITU-T Q.922 Annex A standard. Not specifying the **frame-relay lmi-type** command on the serial interface definition enables the LMI autosense feature.

Along with the explicit specification of an LMI type, the keepalive command should be entered on the serial line interface to ensure connectivity between the devices. The format for the keepalive command is

keepalive *number*

The number variable is any positive integer that is less than the keepalive timer on the switch. The value is measured in seconds and defaults to 10. The example router configuration for router R2 in Figure 21-1 defaults the keepalive timer to 10 seconds because it is not coded.

Dynamic and Static Addressing

The Cisco router learns of protocol network addresses connecting to the Frame Relay network dynamically through the use of Frame Relay inverse ARP. Using inverse ARP, the router requests the next-hop protocol address connected to the DLCI. The responses received from the inverse ARP are placed in an address-to-DLCI table. Dynamic address mapping and inverse ARP are enabled by default for all supported network protocols. The protocols that are supported by inverse ARP are as follows:

- AppleTalk
- Banyan VINES
- DECnet
- IP
- Novell IPX
- Xerox XNS

Because Inverse ARP is enabled by default, no specific specification is required for dynamic addressing unless Inverse ARP is disabled by entering the following command under the specified interface configuration:

no frame-relay inverse-arp [*protocol*] [*dlci*]

The *protocol* optional variable allows the router administrator to disable inverse ARP for a specific network protocol while still using inverse ARP for the other supported protocols. The value for the *protocol* variable can be one of the following keywords:

- **appletalk**
- **decnet**
- **ip**
- **ipx**
- **vines**
- **xns**

The optional *dlci* variable identifies a specific DLCI to which the **no frame-relay inverse-arp** command applies. The value for the *dlci* variable is a valid DLC number for the interface ranging from 16 to 1,007. Specifying both the protocol and dlci variables together targets the protocol of a specific DLCI. This allows another DLCI running the same protocol to continue to use dynamic address mapping. Table 21-1 lists the results of the various combinations of the command.

Table 21-1 The Various Command Format Functional Results of the no frame-relay inverse-arp Interface Command

Entered Command	Result
no frame-relay inverse-arp	All protocols and all DLCIs on the interface are affected.
no frame-relay inverse-arp *protocol*	Only the specified protocol is affected on all the DLCIs used by the interface.
no frame-relay inverse-arp *dlci*	Only the specified DLCI has dynamic address mapping disabled for all the protocols on the DLCI.
no frame-relay inverse-arp *protocol dlci*	Only the specified protocol on the specific DLCI has dynamic address mapping disabled.

Dynamic address mapping is specific to multipoint Frame Relay configurations. In point-to-point configurations, there is only a single destination; hence, there is no need to discover addresses. Inverse ARP should be disabled for a specific protocol and/or DLCI when it is a known fact that the protocol is not being supported on the far end of the PVC.

Static address mapping requires the router administrator to be aware of the next-hop protocol address for each DLCI. Applying a static map is made possible by defining the frame-relay map interface command. Specifying this command automatically disables inverse ARP for any of the affected DLCIs. Static mapping is used when the far-end router does not support inverse ARP or when the specific protocol being used between the routers does not support inverse ARP. The format for specifying static address mapping is

frame-relay map *protocol protocol-address dlci* [broadcast] [ietf | cisco]

[payload-compress {packet-by-packet | frf9 stac [*hardware-options*]}]

The *protocol* variable is the network protocol to which this static address mapping is being applied. The values available for the protocol variable are one of the following:

- **appletalk**
- **decnet**
- **dlsw**
- **ip**

- **ipx**
- **llc2**
- **rsrb**
- **vines**
- **xns**

The *protocol-address* variable value following the *protocol* variable specifies the destination network address of the identified protocol. The value used here must be specified in accordance with the type of protocol being statically mapped.

The *dlci* variable value identifies which DLCI number is being used to connect the *protocol-address* on the Frame Relay interface. The optional keyword **broadcast** is used when the networking protocol functions as a broadcast protocol when multicasting is not applicable. This keyword is particularly important to use when using the OSPF routing protocol for IP networks.

The **ietf** optional keyword instructs the OSPF Frame Relay process to use the IETF Frame Relay RFC 1490 encapsulation method. This is used primarily when the Cisco router is communicating to a non-Cisco hardware platform over the Frame Relay network. The optional **cisco** keyword is used when the router is communicating with another Cisco hardware-based platform. Using either the Cisco or ietf keyword overrides the method specified by the interface configuration command **encapsulation frame-relay** for the interface being defined. Not specifying the **cisco** or **ietf** keywords causes the address mapping to inherit the attributes set by the interface configuration command **encapsulation frame-relay** for the interface being defined.

The optional **payload-compress packet-by-packet** keyword instructs the Cisco IOS to use the stacker method of compressing the payload on a packet-by-packet basis. This method is Cisco proprietary and will not be interoperated with non-Cisco routers. The optional keyword **payload-compress frf9 stac** instructs the IOS to compress using the RFC 1490 FRF.9 compression standard.

Compression of the payload may have performance considerations on the router, depending on the router configuration. If the router includes a compression service adapter (CSA), the compression is performed in the CSA. This is known as hardware compression. If a CSA is not available but the router (7200 series) contains a VIP2 card for the serial Frame Relay connection, the VIP2 software executing on the VIP2 card performs the compression. Using the VIP2 software to compress the payload is called software compression. The final configuration is where neither the CSA nor

the VIP2 are available on the router. In this situation, the router performs the compression using the router's main processor (route processor) and is also called software compression.

The *hardware-options* optional parameter of the optional **payload-compress** keyword enables the router administrator to direct which cards are to be used for the compression techniques. Specifying **distribute** as the *hardware-options* value indicates the compression is performed by the software of the VIP2 card. If the **distribute** option is specified but there is no VIP2 card for use, the router performs compression using the router's main processor. This option is specific to the 7500 series routers only. Using the **software** value as the *hardware-options* variable indicates that the router's main processor is to be used for compressing the payload. Specifying the **csa** *csa-number* keyword and variable instructs the IOS to perform the payload compression on a specific CSA card identified by the *csa-number* variable value. This specification applies to only the 7200 series router platform.

Use of the **frame-relay map broadcast** command in conjunction with the **ip ospf network broadcast** interface configuration command negates the need to define each OSPF neighbor. This enables OSPF to treat the non-broadcast multicast access Frame relay network as if it were a broadcast network.

Figure 21-2 diagrams a Frame Relay network connection using static address mapping. The serial interfaces of all the routers involved specify an IP address and identify the network-address-to-DLCI mapping using the frame-relay map command.

The following router configurations apply to Figure 21-2:

Router R1 configuration applied to Figure 21-2:

```
interface Serial0
 ip address 192.168.1.1 255.255.255.0
 encapsulation frame-relay
 frame-relay map 192.168.1.2 30
 frame-relay map 192.168.1.3 31 ietf
!
router rip
 network 192.168.1.0
```

Router R2 configuration applied to Figure 21-2:

```
interface Serial0
encapsulation frame-relay
ip address 192.168.1.2 255.255.255.0
 frame-relay map 192.168.1.1 32
 !
router rip
 network 192.168.1.0
```

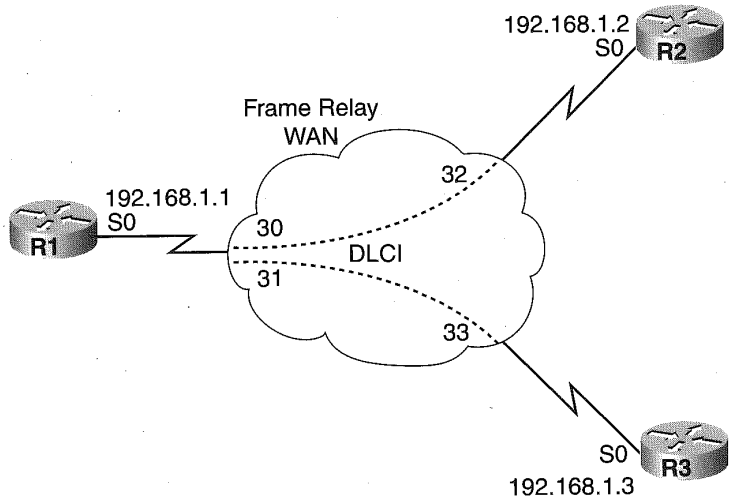


Figure 21-2
Specifying static address mapping.

Router R3 configuration applied to Figure 21-2:

```
interface Serial0
ip address 192.168.1.3 255.255.255.0
encapsulation frame-relay ietf
 frame-relay map 192.168.1.1 33
!
router rip
 network 192.168.1.0
```

The router configuration applying to Figure 21-2 illustrates the use of the frame-relay map command to create static network address mapping. The network address used by the frame-relay map is the IP address of the far-end router that is reached by using the DLCI specified on the command.

For example, router R1 communicates with router R3 using DLCI 31 addressing IP address 192.168.1.3 over this DLCI. Router R3 receives and sends packets to router R1 using the destination IP address 192.168.1.1 over its DLCI number 33. Note that router R2 also specifies the same destination IP address to communicate with router R1 as that used by router R3. However, R2 maps the destination IP address 192.168.1.1 to its DLCI number 32.

Frame Relay Subinterfaces

Subinterfaces are a Cisco IOS feature that enables a single physical interface to be viewed as multiple virtual interfaces. The use of subinterfaces enables a Cisco router to apply physical interface attributes to each virtual interface. For example, instead of having 50 DLCIs assigned to a specific physical interface, each DLCI can be assigned to a given virtual subinterface of the physical interface. This enables the Cisco IOS to apply specific interface requirements to meet the needs of each DLCI.

Point-to-Point Frame Relay Subinterfaces

Subinterfaces were initially created to handle split-horizon on NBMA networks like Frame Relay when using distance-vector routing protocols. Recall that split-horizon infers that routing updates received on an interface cannot be retransmitted back out the same interface. This still holds true for a Frame Relay connection with multiple PVCs defined. This is because the routing protocols use the physical interface as the determining factor in deciding if split-horizon is involved.

In Figure 21-3, router R2 and R3 cannot exchange routing information with each other because their only connection would be out the same physical interface on which they received routing updates about the other router from router R1. This is because router R1 enforces split-horizon for the single physical connection to the Frame Relay network.

Using the notion of subinterfaces for the diagram shown in Figure 21-3, the Cisco router can avoid the split-horizon rule on Frame Relay. Figure 21-4 illustrates this point. Subdividing the single physical interface into two virtual interfaces requires separate network addresses for each virtual interface and hence appears to the routing protocol as distinct interfaces. Because of the use of subinterfaces, routing updates from router 2 on one of the virtual point-to-point subinterface connections can be forwarded to router R3 on the other virtual point-to-point subinterface connection over the same physical link without breaking the split horizon rule.

Multipoint Frame Relay Subinterfaces

Cisco router serial interfaces are considered multipoint connections. The subinterfaces are considered point-to-point only if the point-to-point key-

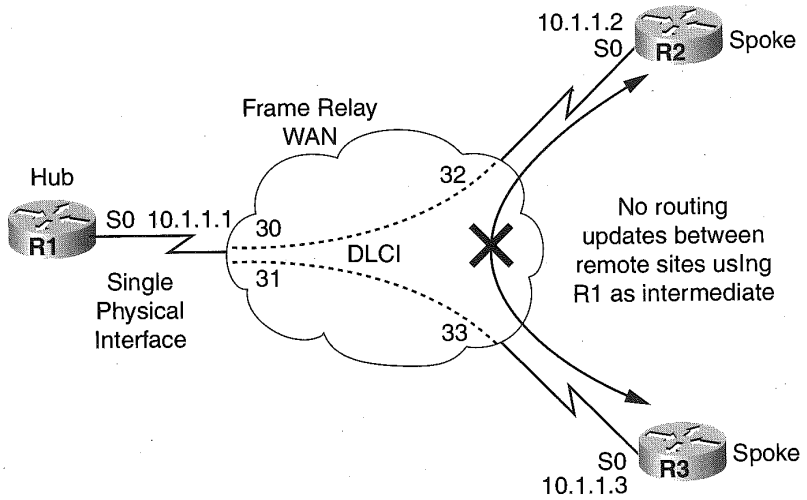


Figure 21-3
The effect of split-horizon on the physical frame relay interface.

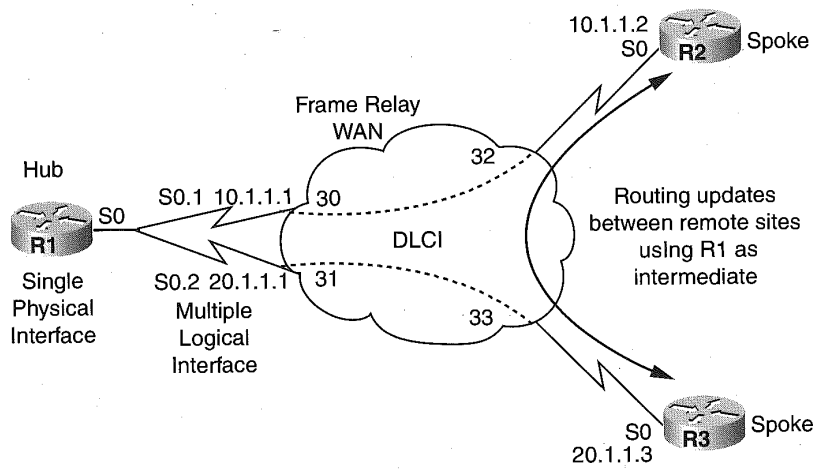


Figure 21-4
Using subinterfaces to account for the split-horizon rule.

word is specified on the interface configuration command. A subinterface can also be defined as a virtual multipoint connection. Although it is a virtual multipoint, it is still bound to the split-horizon rule because all the network resources attached to the virtual multipoint are connected to the same network number. Figure 21-5 illustrates such a configuration.

This type of configuration is most often seen when an existing multipoint frame relay network is being migrated to a subinterface point-to-point network. The virtual multipoint subinterface is utilized to maintain the single network address number for each of the remaining remote locations during the migration process.

Figure 21-5 shows a multipoint subinterface connecting three other remote locations. Each of these remote locations is considered to be connected to the same network number. Router R4 is no longer part of the virtual multipoint network and exists on its own point-to-point subinterface with its own network number. It is important to understand in this scenario that a single physical frame relay connection is still in use to accomplish this configuration.

Specifying a Frame Relay Subinterface

Specifying a Frame Relay subinterface first requires the specification of a physical interface. Once the physical interface is defined, the subinterface(s) can be defined. The order in which the definition is required follows:

interface serial {*number* | *slot/port* | *slot/adapter/port*}

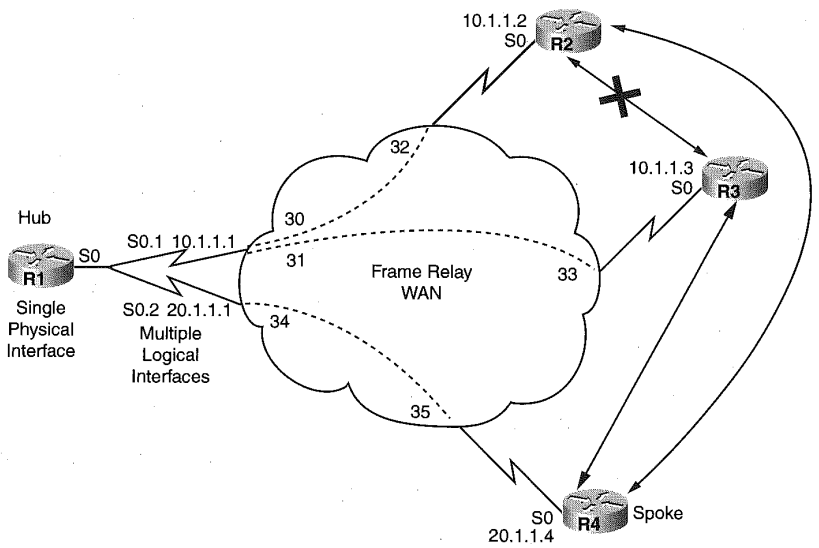


Figure 21-5 A virtual multipoint subinterface and point-to-point subinterface network configuration.

encapsulation frame-relay

interface serial {{*number* | *slot/port* | *slot/adapter/port*}.*subinterface-number* {**multipoint** | **point-to-point**}}

The **interface serial** configuration command specifies the physical Frame Relay interface that is to be used by all subsequent subinterfaces. The *number* variable is the value of the port for which this definition is being specified. The *slot/port* pair is used on routers that have multiple slots that support a number of ports on which a serial line can be connected. The *slot/adapter/port* variable is used on routers that support interface cards that have interchangeable adapters with each adapter having a number of ports. The encapsulation frame-relay command must be entered on the physical connection to ensure the appropriate encapsulation technique. The entire encapsulation frame-relay command set is available for use. Following the encapsulation frame-relay command of the physical interface definition, the subinterfaces can now be defined.

The interface serial command is again entered, but the assignment of the definitions to follow the command is applied to the subinterface number of the physical interface through the use of the subinterface-number variable. The subinterface-number variable is the virtual interface number belonging to the physical interface definition. The assignment of the subinterface-number ranges from 1 to 1,024. The number, slot/port, or slot/adapter/port variable on the subinterface interface configuration command denotes which interface the subinterface is using. For example, if the following were entered under the interface configuration mode

```
interface serial 1/1
encapsulation frame-relay
interface serial 1/1.1
encapsulation frame-relay
```

the subinterface would be denoted by the 1/1.1 variable of the interface serial command. If serial 1/0.1 were entered following the interface serial 1/1 command, the subinterface being defined is actually the serial interface on slot 1 and port 0, not slot 1 and port 1. Therefore, be careful when making configuration changes to subinterfaces. An innocent typo can cause an invalid configuration by directing changes to a different subinterface than the one desired.

The subinterfaces must also be provided with network addressing. Point-to-point subinterfaces require the use of the **frame-relay interface-dlci** command to identify the network address. Multipoint subinterfaces use inverse ARP and therefore learn that the network addresses also require the use of this command. However, multipoint subinterfaces can be mapped

statically when the frame-relay map command is used, as previously discussed. The format of the frame-relay interface-dlci command is

frame-relay interface-dlci *dlci* [ietf | cisco] [voice-encap *size*]

The dlci variable value is the DLCI number used by the Frame Relay provider for connecting the router to a remote location over a PVC. The DLCI and therefore the traffic carried by the DLCI are managed by the configuration commands applied to the subinterface. The **ietf | cisco** keywords indicate the type of Frame Relay encapsulation used for the DLCI associated with the subinterface. The voice-encap size optional pairing is strictly used for the Cisco MC3810 Access server to provide the encapsulation of voice over Frame Relay. The value specified for the size variable determines the data segmentation size of the voice message. The value for size varies depending on the Frame Relay port access rate. Table 21-3 lists the recommended size value, as compared to the port access rate.

The recommendation for the size value of the voice-encap keyword is based on back-to-back frame relay connection between two MC3810 routers. Extra header bytes may be requiered depending on the type of frame relay switch being used for connection over a public frame relay network.

The command **frame-relay interface-dlci** is required for all point-to-point subinterfaces and multipoint subinterfaces that have inverse ARP enabled. Multipoint subinterfaces using static addressing do not require this command.

Figure 21-6 diagrams a point-to-point subinterface configuration. Router R1 uses a single physical Frame Relay interface and two subinterfaces to connect routers R2 and R3 that are defined for transporting IP traffic. The remote destination address of the point-to-point subinterfaces on router R1

Table 21-3 Frame Relay Port Access Rate and the Recommended Size Value for voice-encap Keyword

Port Access Rate	Recommended Data Segmentation Size
64 kbps	80 bytes
128 kbps	160 bytes
256 kbps	320 bytes
512 kbps	640 bytes
1,536 kbps (full T1)	1,600 bytes
2,048 kbps (full E1)	1,600 bytes

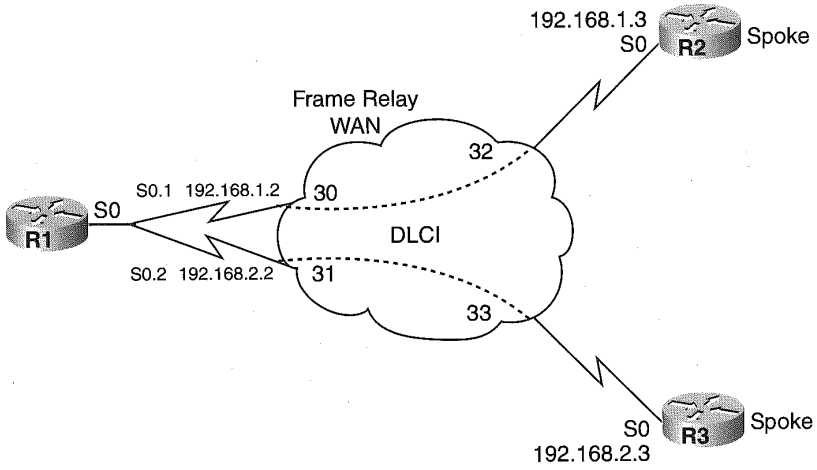


Figure 21-6
A point-to-point subinterface configuration.

are learned from the DLCI specification on the frame-relay interface-dlci commands. Since routers R2 and R3 are using the physical interface for connection to the Frame Relay network, the Cisco IOS defaults the interface to a multipoint configuration. Multipoint interfaces automatically use inverse ARP. Therefore, router R2 and R3 will dynamically learn of the network addressing for the Frame Relay network.

The following router configurations apply to Figure 21-6:

Router R1 configuration applied to Figure 21-6:

```
interface Serial0
 no ip address
 encapsulation frame-relay
!
interface Serial0.1 point-to-point
 ip address 192.168.1.2 255.255.255.0
 frame-relay interface-dlci 30
!
interface Serial0.2 point-to-point
 ip address 192.168.2.2 255.255.255.0
 frame-relay interface-dlci 31
!
router rip
 network 192.168.1.0
 network 192.168.2.0
```

Router R2 configuration applied to Figure 21-6:

```
interface Serial0
 ip address 192.168.1.3 255.255.255.0
 encapsulation frame-relay
!
router rip
 network 192.168.1.0
```

Router R3 configuration applied to Figure 21-6:

```
interface Serial0
 ip address 192.168.2.3 255.255.255.0
 encapsulation frame-relay
!
router rip
 network 192.168.2.0
```

In the router configurations for Figure 21-6, the **frame-relay lmi-type** command is not specified. This means that the routers are using Cisco IOS 11.2 or higher and that LMI autosense is enabled.

Hub and Spoke Configurations

In most Frame Relay networks, the configuration is a hub and spoke. In Frame Relay terms, this means a partially meshed network. The advantage of this is the capability to connect multiple remote locations over single communications links by virtue of using multiple PVCs. Using the subinterface feature of Cisco IOS, each PVC can be mapped to a specific subinterface of the physical interface.

Dynamic Addressing for IP-Only Connectivity

Figure 21-7 illustrates the use of Frame Relay subinterfaces in a partially meshed Frame Relay network. The use of subinterfaces in this configuration allows routing updates to be passed from router R2 to R3 and vice versa.

The use of the subinterfaces in the sample router configurations enables the ease of expansions and management of the PVCs. The subinterface assignment to specific PVCs enables a router administrator to selectively shutdown DLCI connections without affecting the entire Frame Relay physical connection. The following router configurations apply to Figure 21-7:

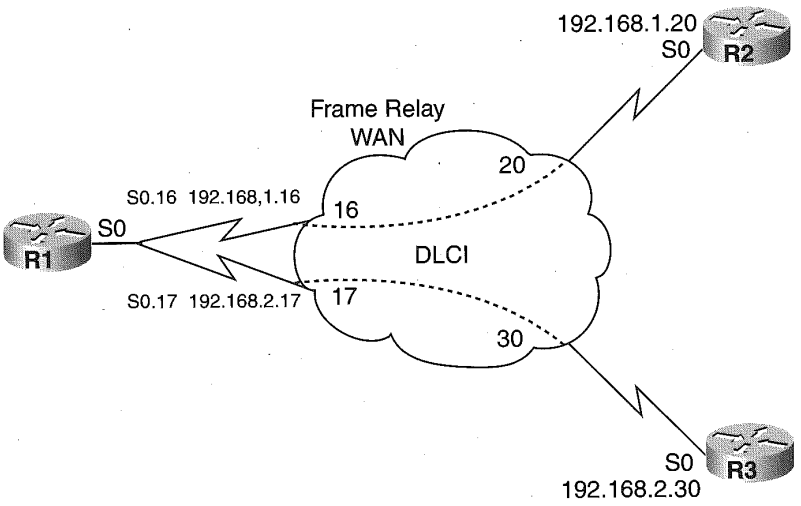


Figure 21-7
A partially meshed network configuration supporting IP traffic only.

Router R1 configuration applied to Figure 21-7:

```
version 11.2
!
 interface Serial0
no ip address
encapsulation frame-relay
!
interface Serial0.16 point-to-point
description Frame Relay to routerR2
ip address 192.168.1.16 255.255.255.0
frame-relay interface-dlci 16 broadcast
!
interface Serial0.17 point-to-point
description Frame Relay to routerR3
ip 192.168.2.17
frame-relay interface-dlci 17 broadcast
!
router rip
version 2
network 192.168.1.0
network 192.168.2.0
no auto-summary
```

Router R2 configuration applied to Figure 21-7:

```
version 11.2
!
interface Serial0
no ip address
```

```
encapsulation frame-relay
!
interface Serial0.20 point-to-point
 ip addrs 192.168.1.20 255.255.255.0
frame-relay interface-dlci 20 broadcast
!
router rip
version 2
network 192.168.1.0
no auto-summary
```

Router R3 configuration applied to Figure 21-7:

```
version 11.2
!
 interface Serial0
no ip address
encapsulation frame-relay
!
interface Serial0.30 point-to-point
 ip address 192.168.2.30 255.255.255.0
frame-relay interface-dlci 30 broadcast
!
router rip
version 2
network 192.168.2.0
no auto-summary
```

In the above router configurations, the router administrator has chosen to match the subinterface number to the DLCI number for the PVC being assigned to the subinterface. This is good practice as it keeps a logical connection between the subinterfaces and the PVCs. The configurations also conserve IP addressing by utilizing the ip unnumbered feature of Cisco IOS.

Static Addressing over a Multipoint Configuration

In Figure 21-8, a single physical router interface is used for connecting router R2 and R3. To ensure connectivity mapping, the router administrator has chosen to use static address mapping by specifying the frame-relay map ip command for the correct address assigned to each of the physical interfaces.

The following router configurations apply to Figure 21-8:

Router R1 configuration applied to Figure 21-8:

```
version 11.2
 interface Ethernet0
ip address 10.1.1.1 255.255.255.0
```

```
!
interface Serial0
ip address 192.168.1.1 255.255.255.0
encapsulation frame-relay
frame-relay map ip 192.168.1.2 16 broadcast
frame-relay map ip 192.168.1.3 17 broadcast
!
router rip
version 2
network 10.0.0.0
network 192.168.1.0
no auto-summary
```

Router R2 configuration applied to Figure 21-8:

```
interface Ehternet0
ip address 10.2.1.1 255.255.255.0
!
interface Serial0
ip address 192.168.1.2 255.255.255.0
encapsulation frame-relay
frame-relay map ip 192.168.1.1 20 broadcast
frame-relay map ip 192.168.1.3 20 broadcast
!
router rip
version 2
network 10.0.0.0
network 192.168.1.0
no auto-summary
```

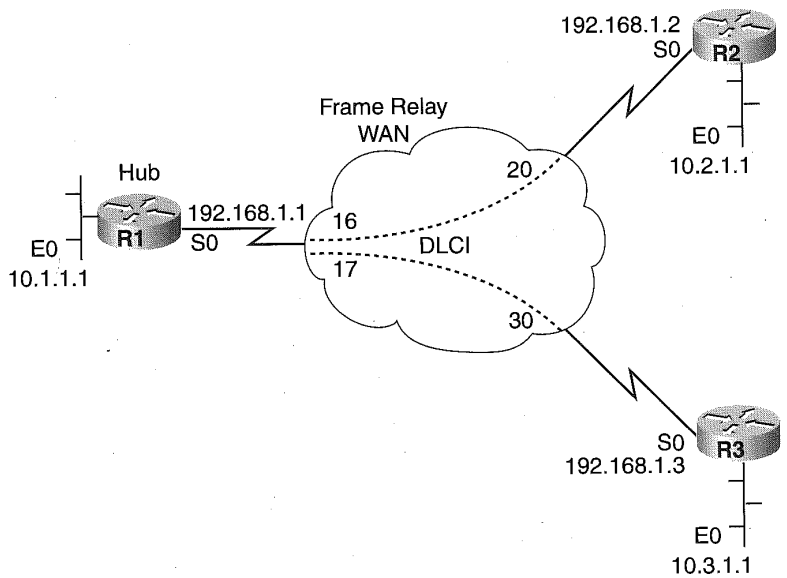


Figure 21-8 Static address mapping in a multipoint configuration.

Router R3 configuration applied to Figure 21-8:

```
version 11.2
!
interface Ethernet0
ip address 10.3.1.1 255.255.255.0
!
interface Serial0
ip address 192.168.1.3 255.255.0.0
encapsulation frame-relay
 frame-relay map ip 192.168.1.1 30 broadcast
frame-relay map ip 192.168.1.2 30 broadcast
!
router rip
version 2
network 10.0.0.0
network 192.168.1.0
no auto-summary
```

In the above configurations, routers R2 and R3 map each other's IP address over the DLCI number that is used to reach router R1. This is because inverse ARP is disabled by including the frame-relay map ip command on the physical interface of the routers. In large Frame Relay, partially meshed networks, static address mapping can become an administrative nightmare. It is highly suggested that dynamic address mapping be used whenever possible.

Traffic Shaping on Frame Relay

Version 11.2 of Cisco IOS greatly enhanced the capabilities of Frame Relay on Cisco routers by introducing traffic-shaping features. These features enable the router to perform the following functions:

- *Enforcing traffic rates for each virtual circuit:* This enables the router to ensure that the peak rate allowed for outbound traffic is available. The peak rate should match that of the CIR for the PVC or a value that provides the service required by the business.
- *The throttling of traffic dynamically for each virtual circuit:* If the router receives BECN packets, indicating congestion on the network, the outbound rate is reduced to allow congestion to ease. Once the congestion has eased, by virtue of not receiving the BECN packets anymore, the outbound traffic rate is increased.

- *Support for priority and custom queuing for each virtual circuit:* This allows a router administrator to apply custom or priority queuing to a virtual circuit to meet the business requirements.

Traffic-shaping for Frame Relay is enabled by the use of the interface configuration command:

frame-relay traffic-shaping

The traffic-shaping command does not have specific variables in its format. However, the shaping of the traffic for a virtual circuit comes about from the specification of several other frame-relay interface commands.

The following router configuration will be used to explain the various other related commands needed to establish traffic shaping. Let's start with a sample router configuration command employing traffic shaping on a serial interface:

```
interface serial 0
encapsulation frame-relay
bandwidth 64
ip address 10.10.10.1 255.255.255.0
frame-relay traffic-shaping
frame-relay map ip 10.10.10.2 16 broadcast
frame-relay interface-dlci 16
frame-relay class DS1
!
map-class frame-relay DS1
 frame-relay cir 64000
 frame-relay bc 8000
frame-relay be 0.
frame-relay mincir 16000.
```

The frame-relay class command in the sample configuration has the following command format:

frame-relay class *name*

This command is used to identify a map-class specification that sets the parameters that apply to shaping the traffic for the associated DLCI. In this case, the map-class providing this information is named DS1. The parameters set by the map-class specification are applied to all the virtual circuits defined to the interface or subinterface on which the frame-relay class command is defined. The parameters are applied to each virtual circuit in the following order:

1. Parameters set by the map-class defined for the virtual circuit
2. Parameters set by the map-class associated with the subinterface

3. Parameters set by the map-class specified on the physical interface
4. The default parameters of the physical interface

The name value identifies a map-class frame-relay global configuration command. The format of the command is

map-class frame-relay *map-class-name*

The *map-class-name* variable value must match the name of a **frame-relay class** *name* variable value in order for the subsequent Frame Relay parameters to affect the virtual circuit. Following a map-class frame-relay global command, the specific traffic shaping commands are encountered.

The first of these traffic shaping commands is the frame-relay cir command. The format of this command is

frame-relay cir {in | out} *bps*

The **in | out** keyword identifies the flow of traffic to which the committed information rate (CIR) is being defined. The default direction of the flow of traffic is in. The *bps* variable is the CIR in bits per second and defaults to 56,000. The value chosen here should be the value at which the router will provide the best performance for the applications using the connection without risking poor performance. The rate here is not the provider's CIR. In our example, the frame-relay cir command specifies that 64,000 bits per second (64 Kbps) is being assured for inbound traffic. A second frame-relay cir command can be entered to specify the desired rate for outgoing traffic by using the frame-relay cir out command with a bits-per-second (bps) value specified. The rates defined here are the normal rates assured without network congestion being present.

The frame-relay bc command following the frame-relay cir command in the sample router configuration is used to provide the committed burst rate of incoming and outgoing traffic. The format of this command is

frame-relay bc {in | out} *bits*

The **in | out** keyword restricts the command to the selected flow. If neither in nor out is specified, the committed burst rate is applied to inbound and outbound traffic. The default value for the bits variable is 7,000. The rule of thumb is to make the committed burst rate (Bc) one-eighth the rate of the CIR.

Associated with the frame-relay bc command is the frame-relay be command. The frame-relay be command is expressed in the following command format:

frame-relay be {in | out} *bits*

Again the **in | out** keywords restrict the bits value to only one flow. The default, if neither in nor out is defined, is to apply the bits value to inbound and outbound traffic flow. The bits value defaults to 7,000 bits per second and defines the excess or extra amount of traffic that can be sent if needed during a given interval. The sample router configuration has the CIR at the port speed, so there is no excess amount of bandwidth to allow extra burst.

The time interval is an important factor in the values used. The time interval (Tc) is equal to the committed burst rate (Bc) divided by the CIR value. In our example, the time interval is equal to

Tc = 8K(Bc) / 64K(CIR) = 1/8 seconds

This means that every eighth of a second the router can send up to the amount defined by the Bc and Be values.

The final command used in our example indicates how the router will respond after receiving BECN responses from the Frame Relay network. Cisco IOS release 11.2 and higher automatically have the BECN response enabled, and it is recommended that you do not disable it. Often the router is set to fall back to the actual CIR provided by the Frame Relay network configuration that is guaranteed for the circuit. This value is defined by using the frame-relay mincir command. The format of the command is

frame-relay mincir {in | out} *bps*

The **in | out** keywords restrict the bps value to only one flow. The default, if neither in nor out is defined, is to apply the bps value to inbound and outbound traffic flow. The bps value defaults to 56,000 bits per second and uses the value defined as the rate for when BECN responses are received.

Frame Relay traffic-shaping in Cisco IOS 11.2 or higher can take advantage of the Cisco IOS Quality of Service (QoS) features. For example, the QoS priority queue list can be specified for a Frame Relay interface using the following command:

frame-relay priority-group *list-number*

This command is used in conjunction with a map class to prioritize protocols over the Frame Relay virtual circuit. The list-number variable directs the Cisco IOS to apply the priority-list command set to the traffic on the virtual circuit. For example, the following router configuration assigns a higher priority to IP traffic on the Frame Relay virtual circuit over the other protocols:

```
interface serial 0
 encapsulation frame-relay
 ip address 10.10.10.10 255.255.255.0
 frame-relay interface-dlci 30
 class vc30
!
map-class frame-relay vc30
 frame-relay priority-group 1
!
priority-list 1 protocol ip high
```

In this example, the frame-relay priority-group command points to the priority-list 1. This list indicates that the IP protocol is to receive higher service than other protocols on the DLCI. The example also introduces the class command, which identifies a map-class definition for a specific DLCI. This is juxtaposed to the frame-relay class command, which affects a whole interface or subinterface. The command format for the class virtual circuit configuration command is

class name

in which name is the value of a valid **map-class frame-relay** definition.

The custom queuing feature of Cisco IOS QoS is enabled for frame-relay interfaces much in the same way as priority queuing. The following command is entered on the frame–relay interface or subinterface definition to identify a custom queuing list:

frame-relay custom-queue-list *list-number*

The *list-number* variable value identifies a valid **queue-list** number for which the **map-class frame-relay** command of this **frame-relay custom-queue-list** command is specified. The queues defined in the associated queue-list affect the traffic of the DLCI to which the map-class is applied.

Transparent Bridging

Transparent bridged traffic over Frame Relay is accomplished by applying the bridge-group command to the interface or subinterface for which bridged traffic is to be transmitted. As shown in Figure 21-9 and the associated router configurations, a typical hub and spoke configuration will route IP and IPX traffic and bridge all other traffic. The IP routes are dynamically resolved, in this case using RIP V2. The IPX routers and associated services are resolved dynamically using IPX RIP/SAP protocols. The use of subinterfaces enables the resolution of these routing protocols by avoiding the split-horizon rule.

Router R1 configuration applied to Figure 21-9:

```
version 11.2
 ipx routing 0000.0caa.1111
!
interface Ethernet0
ip address 10.1.1.1 255.0.0.0
ipx network 100 encapsulation SAP
ipx network 101 encapsulation NOVELL-ETHER secondary
bridge-group 1
!
interface Serial0
no ip address
encapsulation frame-relay
!
interface Serial0.16 point-to-point
description Frame Relay to R2
ip unnumbered Ethernet0
ipx network AAAA
frame-relay interface-dlci 16 broadcast
bridge-group 1
!
interface Serial0.17 point-to-point
description Frame Relay to R3
ip unnumbered Ethernet0
ipx network BBBB
frame-relay interface-dlci 17 broadcast
bridge-group 1
```

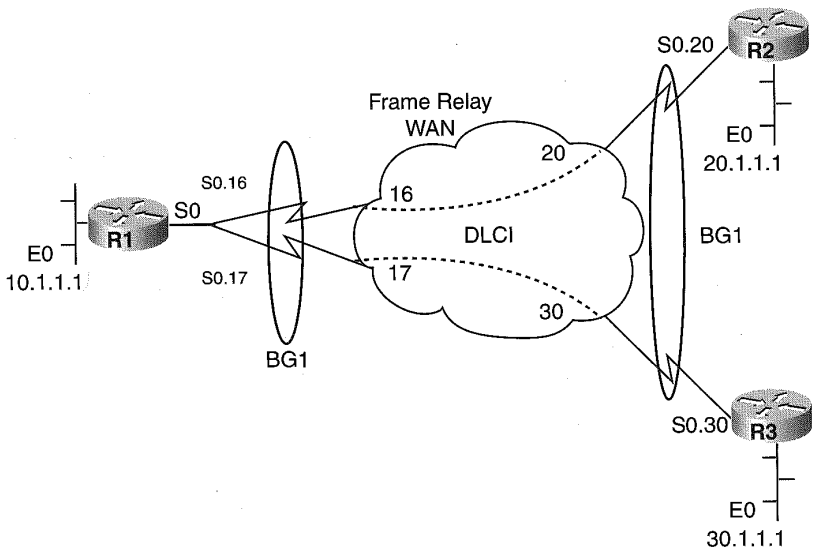


Figure 21-9
The hub and spoke configuration used for routing IP and IPX traffic while bridging all other traffic.

```
!
router rip
version 2
network 10.0.0.0
no auto-summary
!
bridge 1 protocol ieee
```

Router R2 configuration applied to Figure 21-9 :

```
version 11.2
 ipx routing 0000.0cbb.2222
!
interface Ethernet0
ip address 20.1.1.1 255.0.0.0
ipx network 200
bridge-group 1
!
interface Serial0
no ip address
encapsulation frame-relay
!
interface Serial0.20 point-to-point
description Frame Relay to R1
ip unnumbered Ethernet0
ipx network AAAA
frame-relay interface-dlci 20 broadcast
bridge-group 1
!
router rip
version 2
network 20.0.0.0
no auto-summary
!
bridge 1 protocol ieee
```

Router R3 configuration applied to Figure 21-9:

```
version 11.2
 ipx routing 0000.0ccc.3333
!
interface Ethernet0
ip address 30.1.1.1 255.0.0.0
ipx network 300
bridge-group 1
!
interface Serial0
no ip address
encapsulation frame-relay
!
interface Serial0.30 point-to-point
description Frame Relay to R1
ip unnumbered Ethernet0
ipx network BBBB
frame-relay interface-dlci 30 broadcast
bridge-group 1
!
```

```
router rip
version 2
network 30.0.0.0
no auto-summary
!
bridge 1 protocol ieee
```

In these transparent bridge configuration examples, each of the Frame Relay PVCs are grouped into bridge-group 1 and all use the IEEE bridge protocol.

Managing Performance Problems Using the Broadcast Queue

Bandwidth on Frame Relay networks is a valuable commodity. Performance issues may arise in Frame Relay networks that have a large number of DLCIs on a single router interface. These performance issues are caused by the replication of routing and service advertisement updates for each of the DLCIs. These updates not only consume bandwidth, but they end up using interface buffers that would normally be used for end-user traffic. In this type of configuration, high packet rate loss is probable for the Frame Relay interface.

Cisco IOS attempts to resolve this type of Frame Relay performance issue by establishing a special queue that maintains only broadcast traffic. This broadcast queue is managed independently of the normal interface queue. Consequently, the broadcast queue has its own buffers, size, and service rate.

The broadcast queue for Frame Relay is enabled by default. The format of the command is

frame-relay broadcast-queue ***size byte-rate packet-rate***

The size variable value determines the total number of packets the broadcast queue will hold. This size value defaults to 64 packets. The byte-rate variable value defaults to 256,000 bytes per second and defines the maximum number of bytes that can be transmitted in a second. The final variable is the packet rate. Its value defaults to 36 packets per second and defines the maximum number of packets that can be transmitted in a second. The broadcast queue holds only those packets that have been repli-

cated for transmission over the multiple DLCIs on an interface. The original routing packet and SAP packets are queued on the normal interface queue. Likewise, bridged and spanning-tree packets are sent on the normal queue.

The rule of thumb for setting the byte rate is

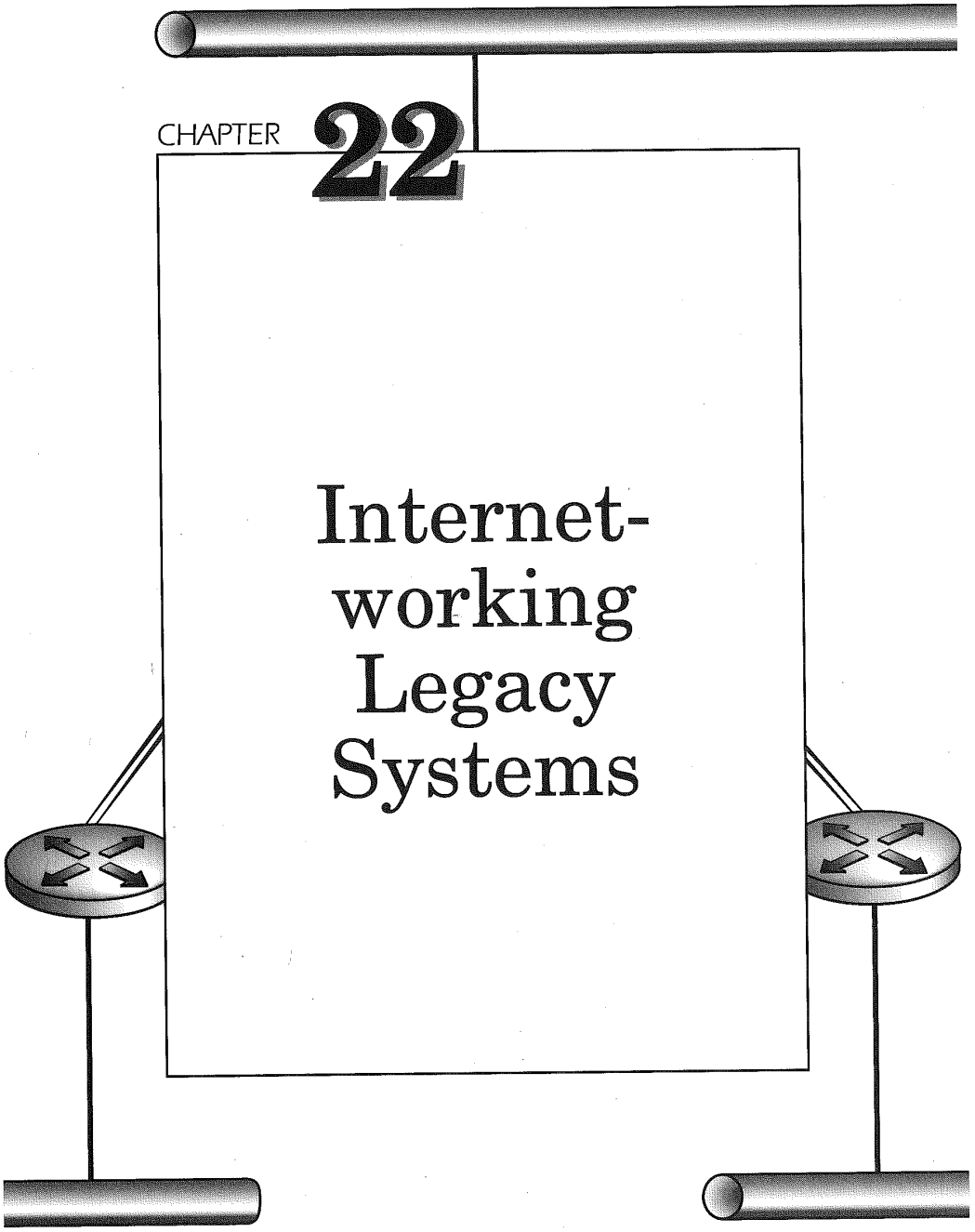
Less than n/4 times the minimum remote access rate (bytes/sec)

N is the number of DLCIs that the broadcast is replicated on and the local access rate is bytes/sec.

The packet rate rule of thumb is to set the rate by assuming packets will be 250 bytes.

CHAPTER 22

Internet-working Legacy Systems



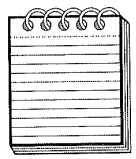
The speed at which Cisco Systems' routers and the powerful Cisco IOS software were embraced by corporate network architects was due in part to the support of legacy system networks. The term legacy systems is a term that was created by industry pundits to describe the mainframe networking and system environments. Varying estimates state that 60 to 80 percent of all corporate data in one form or another is either sourced or archived using mainframe systems, networks, and applications.

Typically, the mainframe environment employs two types of networking protocols for transporting data: the IBM Systems Network Architecture (SNA) and Internet Protocol (IP). The IBM SNA networks were first deployed using the IBM Synchronous Data Link Control (SDLC) protocol. This is a data link layer networking protocol used for point-to-point and multipoint communication lines between SNA front-end processors (FEPs) and end-user controllers. The advent of LANs has forced SNA to be supported on Token Ring and Ethernet LANs. Technologies such as Frame Relay and ATM have also caused SNA to incorporate these WAN technologies. Cisco IOS supports the transport of SNA data using the following:

- SDLC
- Source-Route Bridging (SRB)
- Data link switching (RFC 1795)
- Frame Relay RFC 1490 FRF.3
- APPN over ATM RFC 1484
- SMDS, PPP, and ISDN RFC 1661

Traditional SNA is a master-slave architecture. SNA was enhanced to allow the peering of network resources through the implementation of Advanced Peer-to-Peer Networking (APPN). Using APPN, network resources can establish connections with other resources as required. The SNA/APPN architecture is based on an applications requesting a conversation with another application by simply specifying the name of the partner application. Network addresses in APPN are not a requirement. This simplification on an application level requires a robust and complicated networking level. Cisco licenses the APPN code from IBM in support of the APPN networking requirement.

Mainframe computers can also be accessed using IP. Typically, the IP stack, associated transport, and applications such as TCP, FTP, SMTP, and DNS are installed on the mainframe. The three most commonly found in an IBM mainframe environment are IBM's TCP/IP, Interlink, and Cisco IOS/390.



NOTE: *Cisco IOS/390 is licensed from Interlink.*

Cisco routers have direct access to the mainframe computer using the Channel Interface Processor (CIP) on all Cisco 7000 series routers. The CIP enables SNA and IP traffic between the mainframe and the corporate enterprise network.

This chapter involves exploring the many different methods of transporting SNA and support of IP traffic to the mainframe using the CIP.

Serial Tunneling (STUN)

STUN is typically used to preserve equipment investment that continues to perform its function. A Cisco router is often used at the data center site to collect not only WAN internetworking traffic but to assist in the migration from a distinct SNA/SDLC network into a multiprotocol network.

In Figure 22-1, a simple STUN network configuration is deployed. The host computer is an IBM AS/400 with serial line connections supporting IBM SNA PU Type 1 control units at the remote locations. The topology illustrates the use of STUN using a single router and a remote router connection to the host location.

Router R1 and R2 use the following Cisco IOS router configurations to enable SNA SDLC connectivity through the router network:

Router R1 configuration applied to Figure 22-1:

```
stun peer-name 172.16.1.1.1
stun protocol-group 1 sdlcstun protocol-group 2 sdlc!
interface loopback 0
 ip address 172.16.1.1 255.255.255.0
!interface serial 0 encapsulation stun stun group 1 stun route address
C1 interface serial 2 direct!interface serial 1 encapsulation stun stun
group 2 stun route address C2 interface serial 3!interface serial 2
encapsulation stun stun group 1 stun route address C1 interface serial 0
direct!interface serial 3 no shutdown
```

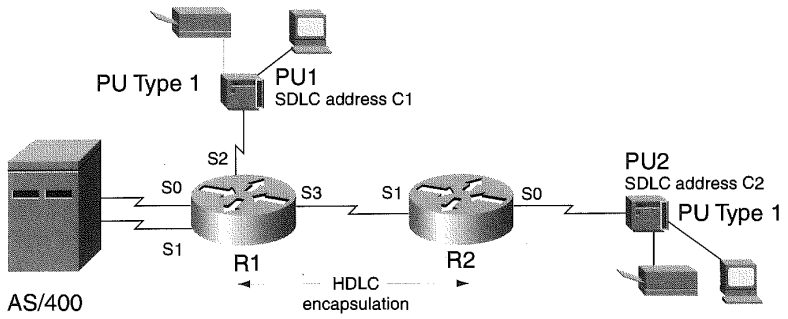


Figure 22-1
Simple STUN network configuration.

Router R2 configuration applied to Figure 22-1:

```
stun peer-name 172.16.2.1
stun protocol-group 2 sdlc!
interface loopback 0
 ip address 172.16.2.1 255.255.255.0
!interface serial 0 encapsulation stun stun group 2 stun route address
C2 interface serial 1 !interface serial 1no shutdown
```

These two router configurations introduce the basic STUN configuration commands required to support SNA/SDLC communications through Cisco routers. STUN uses IP for delivering SDLC frames. Each router must define the IP address of an interface that is used as the STUN peer for transporting the SDLC frame. The STUN peer is specified using the following global configuration command:

stun peer-name *ip-address*

The *ip-address* value is an interface on the router that supports the IP protocol and has an IP address assigned. For high availability, the loopback interface is very useful in this case because the loopback interface is always perceived by the router as being active on the network. In the example configurations for Figure 22-1, the loopback0 interface is used for each router as the IP address assigned to be the STUN peer.

Each STUN interface must be associated with a STUN protocol group. The STUN protocol group ensures that received packets are delivered to the appropriate interfaces on the router and forms a logical grouping of interfaces on the router. The global configuration command used to create the STUN protocol groups is

stun protocol-group *group-number frame-format*

The *group-number* variable may range from 1 to 255 and identifies the protocol grouping. *The frame-format* variable value identifies the type of protocol being used for this STUN protocol group. The possible values are

- **basic**
- **schema**
- **sdlc**
- **sdlc sdlc-tg**

The **basic** value indicates that the default frame format on the serial interface is to be used. The serial interface default is HDLC. Specifying **schema** enables the router administrator and the network engineers to use a unique protocol for the STUN protocol group. The **sdlc** value for *frame-format* is used when connecting the router to a device that supports IBM's SNA SDLC and is an SNA PU Type 1, 2, or 2.1 resource.

The **sdlc sdlc-tg** *frame-format* value is used when the Cisco router is used to connect two IBM FEPs over a router backbone network in support of SNA transmission groups between two SNA PU Type 4 resources. In our example, each serial interface is connecting an SNA/SDLC device; the serial interfaces s0, s1, and s2 on router R1; and the serial interface s0 on router R2.

The router configuration example for router R1 in Figure 22-1 uses two STUN protocol groups. Protocol group 2 is used for connectivity to the SNA device named PU1 and protocol group 3 is used for the SNA connection to the SNA device named PU2. The PU2 connection is supported remotely from router R1. Also take notice that although the router R1 may appear in Figure 22-1 as local to the FEP, it may indeed be remotely connected to the FEP.

A serial interface supporting the SNA/SDLC connection must be enabled to support STUN. The STUN method is enabled using the following serial interface configuration command:

encapsulation stun

Each interface using STUN must be associated with a STUN protocol group. The interface is associated with a STUN protocol group by having the following command specified under its interface configuration:

stun group *group-number*

The *group-number* variable must match a valid *group-number* variable value specified on the global **stun protocol-group** configuration command.

Each interface can use an encapsulation technique different from the other interfaces that use STUN. The stun route interface command identifies the type of encapsulation in use for the interface. The three encapsulation techniques available for STUN are

- Direct HDLC
- TCP
- Frame Relay

The router interface can support multiprotocol communications only if one of the three encapsulation techniques is also defined on the interface using STUN. Figure 22-1 demonstrates the use of the direct HDLC encapsulation form of the **stun route interface** command. The topology also supports IP and any other supported Cisco IOS networking protocol because the direct HDLC encapsulation is defined on the serial interfaces. The format of the command for direct HDLC encapsulation is as follows:

stun route all interface serial *number*

stun route all interface serial *number* **direct**

stun route address *address-number* **interface serial** *number*

stun route address *address-number* **interface serial** *number* direct

The **stun route all interface serial** *number* command is specified for the serial interface connecting to another appropriately configured router at the other end of the serial line. The *number* parameter value is the interface number of the outbound serial link used for forwarding the SDLC frames. When the command is specified as **stun route all interface serial** *number* **direct,** the **direct** keyword indicates that this router is routing the SDLC traffic internally without sending the frame out of the router to another one.

Using the **stun route address** *address-number* **interface serial** *number* interface command on the outgoing serial link forwards all SDLC frames for the specified SNA PU station address defined by the *address-number* parameter value using the serial interface identified by the *number* value. Adding the **direct** keyword to the end of this command indicates to the router that the STUN connection is made without sending the data to a remote router.

The router configuration for Figure 22-1 uses the **stun route address interface serial** form of **the stun route** command for the STUN connection to the AS/400, PU1, and PU2 devices. The use of HDLC encapsulation

for STUN is restricted to a dedicated point-to-point link between the two routers participating in the STUN connection or when the same router is used as the connection.

STUN can also be encapsulated using TCP. Figure 22-2 illustrates a configuration that uses TCP encapsulation for delivery over STUN. A Frame Relay network is used to provide the WAN connectivity. Again, in the sample configuration for the routers, the **stun peer-name** and **stun protocol-group** global commands must be specified. They are on the interface configurations where the TCP encapsulation technique is specified.

The network configuration diagrammed in Figure 22-2 also depicts the capability of Cisco IOS to support a multipoint line configuration using local acknowledgement and priority queuing for the delivery of SNA data to end resources.

The virtual multidrop of the remote SNA PU's is specified on the router connection to the FEP. The interface serial 0 of router R1 identifies the two remote PU's, PU1 and PU2, as being connected by this interface. This is most likely required because the Network Control Program (NCP) within the FEP that defines the serial line connection to the remote sites specifies these two SNA PU's on a multipoint connection. Therefore, the router must also participate in this specification. The complete router configurations for Figure 22-2 are as follows:

R1 Configuration applied to Figure 22-2:

```
stun peer-name 172.16.1.1stun protocol-group 1 sdlc!interface loopback
0ip address 172.16.1.1 255.255.255.0!interface serial 0encapsulation
stunstun group 1sdlc virtual-multidropstun route address C1 tcp
172.16.2.1 local-ack prioritystun route address C2 tcp 172.16.3.1 local-
ack prioritystun route address FF tcp 172.16.2.1 stun route address FF
tcp 172.16.3.1 stun sdlc-role secondarysdlc address C1sdlc address
C2priority-group 1locaddr-priority 1!interface serial 1ip address
172.16.4.1 255.255.255.0!interface serial 2ip address 172.16.5.1
255.255.255.0!priority-list 1 protocol ip high tcp 1994priority-list 1
```

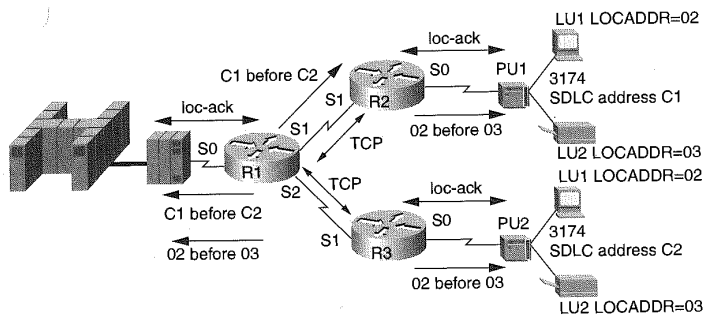


Figure 22-2 STUN virtual multidrop topology using local acknowledgement and prioritization.

```
protocol ip medium tcp 1990priority-list 1 protocol ip normal tcp
1991priority-list 1 protocol ip low tcp 1992priority-list 1 stun high
address 1 C1priority-list 1 stun normal address 1 C2!locaddr-priority-
list 1 02 highlocaddr-priority-list 1 03 normal
```

R2 Configuration applied to Figure 22-2:

```
stun peer-name 172.16.2.1stun protocol-group 1 sdlc!interface loopback
0ip address 172.16.2.1 255.255.255.0!interface serial 0encapsulation
stunstun group 1stun route address C1 tcp 172.16.1.1 local-ack priori-
tystun sdlc-role primarysdlc address C1priority-group 1locaddr-priority
1!interface serial 1ip address 172.16.4.2 255.255.255.0!priority-list 1
protocol ip high tcp 1994priority-list 1 protocol ip medium tcp 1990pri-
ority-list 1 protocol ip normal tcp 1991priority-list 1 protocol ip low
tcp 1992priority-list 1 stun high address 1 C1!locaddr-priority-list 1
02 highlocaddr-priority-list 1 03 normal
```

R3 Configuration applied to Figure 22-2:

```
stun peer-name 172.16.3.1stun protocol-group 1 sdlc!interface loopback
0ip address 172.16.3.1 255.255.255.0!interface serial 0encapsulation
stunstun group 1stun route address C2 tcp 172.16.1.1 local-ack priori-
tystun sdlc-role primarysdlc address C2priority-group 1locaddr-priority
1!interface serial 1ip address 172.16.5.2 255.255.255.0!priority-list 1
protocol ip high tcp 1994priority-list 1 protocol ip medium tcp 1990pri-
ority-list 1 protocol ip normal tcp 1991priority-list 1 protocol ip low
tcp 1992priority-list 1 stun normal address 1 C2!locaddr-priority-list 1
02 highlocaddr-priority-list 1 03 normal
```

Using TCP encapsulation allows you to provide prioritization and SDLC local acknowledgement of SDLC frames sent to a STUN peer. Encapsulation in TCP facilitates the transmission of SDLC frames across disparate media and is useful when transmitting the SDLC frames with STUN over a multiprotocol WAN backbone. Two forms of the stun route command can be used for encapsulating SDLC frames in TCP. These interface command forms are

stun route all tcp *ip-address*

stun route address *address-number* **tcp** *ip-address* **[local-ack] [priority]**

When using the **stun route all tcp** form of this command, it indicates to the router to forward all STUN traffic to the specified *ip-address* value in the command. The *ip-address* value is the IP address of the remote STUN peer connected to the serial interface over the WAN. Note here that it does not matter what the SNA PU station address is for forwarding to the STUN peer. All SDLC frames, regardless of the SNA PU station address, will be forwarded to the STUN peer specified by the *ip-address* value.

The second form of the TCP encapsulation interface command for STUN identifies the exact SNA PU station address with the *address-number* value and which STUN peer, specified by the *ip-address* value, connects to the SNA device involved in the STUN connection.

SDLC broadcasts are sent by the IBM host to end-station controllers. The hexadecimal value "FF" is the broadcast PU station address used by all SNA PU's. Enabling this capability of a host router to broadcast the hexadecimal "FF" to all remote STUN peers is through the use of the interface command:

sdlc virtual-multidrop

The **sdlc virtual-multidrop** interface command is specified on the serial link attaching to the host computer. This serial interface is often referred to as the SNA secondary link station. A secondary SNA link station receives and sends supervisory SDLC frames to an SNA PU Type 4 or PU Type 5 resource. A primary SNA link station sends and receives supervisory SDLC frames to the PU Type 1 or PU Type 2/2.1 SNA controller. The routers in Figure 22-2 must be given a primary or secondary link station role on the serial interfaces that connect to the SNA devices. The link station role is specified using the following interface commands:

stun sdlc-role primarystun sdlc-role secondary

The specification of either of these **stun sdlc-role** interface commands determines how the router participates in the SDLC polling process. Router R1 in Figure 22-2 participates as a secondary link station on the interface connecting to the FEP. Router R2 and R3 act as the primary link station to the SDLC-attached SNA controllers.

The polling and acknowledgement of SNA frame delivery uses valuable bandwidth over the WAN and is considered unproductive traffic. Cisco IOS can gain back this valuable bandwidth used by the polling and frame acknowledgement by implementing the **loc-ack** keyword on the **stun router address tcp** interface command. Local acknowledgement allows the routers to respond to reply back to the attached SNA devices as if it were the receiving SNA controller. The SNA supervisory control frames replied to by the router when using local acknowledgement are the receiver-ready (RR), receiver-not-ready (RNR), and reject frame (REJ). Local acknowledgement actually terminates the SNA session at the router serial interface ports versus at the SNA far-end devices, thereby eliminating this traffic from the WAN. When using local acknowledgement, the serial interfaces being connected on the routers must all use the **loc-ack** keyword.

The router is made aware of which SNA device is communicating using STUN by the use of the **sdlc address** command on the interface connecting the SNA device. The format of the command is

sdlc address *SNA-station-address*

The *SNA-station-address* variable value corresponds to the given station address of the attached SDLC SNA controller. In Figure 22-2, the PU1 controller has a station address of C1, and PU2 has a station address of C2.

The **priority** keyword on the **stun route address tcp** command indicates that the packets are to be prioritized based on a priority list. The priority list is identified using the **priority-group** interface command. The format of this command is

priority-group *number*

The *number* variable value identifies a global priority list defined on this router. The *number* variable of the **priority-group** interface command ranges from 1 to 10 and must match a specified priority list definition.

The priority list definitions enable STUN traffic to be prioritized on serial interfaces by specifying the SNA SDLC station address along with the TCP port number associated with the priority level. Prioritization on the serial link allows traffic destined for a specific PU to be prioritized over another PU, especially on a multidrop connection. The format of the global configuration commands for establishing prioritization on the serial links are

priority-list *list-number* **stun** *queue* **address** *group-number address-number*

priority-list *list-number* **protocol ip** *queue* **tcp** *tcp-port-number*

Using the **priority-list** *list-number* **stun** value specifies a number ranging from 1- to 0 identifying the priority list being defined. The *queue* parameter specifies the type of queue being defined for the priority definition. The possible values for *queue* are

- high
- medium
- normal
- low

The *group-number* parameter value identifies the stun group associated with this priority definition. Finally, the *address-number* is the SNA SDLC station address being assigned the priority.

The **priority-list stun** command must be defined prior to using the **priority-list protocol ip tcp** global command. The *list-number* and *queue* parameter values have identical meanings as the priority-list stun address command. The *tcp-port-number* parameter of the **priority-list stun** address command allows the assignment of a priority list to a TCP port defined for a queuing priority. In support of SDLC traffic, the *tcp-port-number* assignments are

- STUN TCP port 1994—High priority
- STUN TCP port 1990—Medium priority
- STUN TCP port 1991—Normal priority
- STUN TCP port 1992—Low priority

The default priority for all traffic on an interface is normal. Normal queuing is first-in-first-out. When using these two priority list commands together, the STUN traffic receives the highest service over all traffic out of the serial interface using the defined priority list.

The router configuration examples given for Figure 22-2 identify priority group number 1 on the serial interfaces supporting STUN. The priority list 1 specification lists the IP protocol in the order of STUN TCP port number to priority level and assigns PU1 station address C1 as having a higher priority for transmission than PU2 station address C2.

Further prioritization for delivering SNA data to end resources is provided by SNA LU prioritization. LU prioritization is specified using the **locaddr-priority-list** global router configuration commands. Just like the **priority-list** command, the **locaddr-priority-list** command is referenced on the STUN interface through the use of the **locaddr-priority** interface command identifying the associated **locaddr-priority-list**. Using these commands on a point-to-point or even a multidrop connection can receive a higher priority for transmission than other LUs. Normal SDLC line connection to a FEP utilizes SNA Class of Service (COS) assignment to prioritize traffic to the SNA PU. The Cisco IOS LU prioritization feature, though not fully providing the same functionality as a pure SNA connection, enables some form of control over connections that support both interactive and print traffic. The **locaddr-priority-list** command format is

locaddr-priority-list *list-number address-number queue-keyword*

The *list-number* parameter value identifies the **locaddr-priority-list** being defined. The address-number is the LU's address on the SNA PU to which the LU is owned. The LU address is obtained from the LOCADDR operand on the LU macro definition for the PU in the NCP configuration or

the VTAM major node describing the PU. The value is a one-byte hexadecimal value in the range of 01-FF. The *queue-keyword* is the type of priority this LU is to receive. The *queue-keyword* can be valued as

- high
- medium
- normal
- low

The locaddr-priority-list must be associated with the serial interface that attaches the end-to-end connection. This association is through the interface command **locaddr-priority** with the format:

locaddr-priority *list-number*

The *list-number* parameter is the number associated with the predefined **locaddr-priority-list** global command. Using this sets up the queuing policies for the LUs connected to this serial link.

The router configuration examples for Figure 22-2 identify locaddr-priority-list 1, which assigns a higher priority to the interactive terminal traffic over the print traffic.

STUN can also use Frame Relay as the encapsulation technique. In Figure 22-3, a virtual multidrop multipoint configuration is depicted using Frame Relay as the encapsulation method of the three SNA PUs. The Frame Relay connection uses DLCI 30 to link to the host site.

The two routers in Figure 22-3 have the following router configurations:

R1 configuration applied to Figure 22-3:

```
stun peer-name 172.16.1.1 stun protocol-group 1 sdlc!interface loopback
0ip address 172.16.1.1 255.255.255.0!interface serial 0encapsulation
stunstun group 1stun route address C1 interface s1 dlci 30 04 local-ack-
stun route address C2 interface s1 dlci 30 04 local-ackstun route address
```

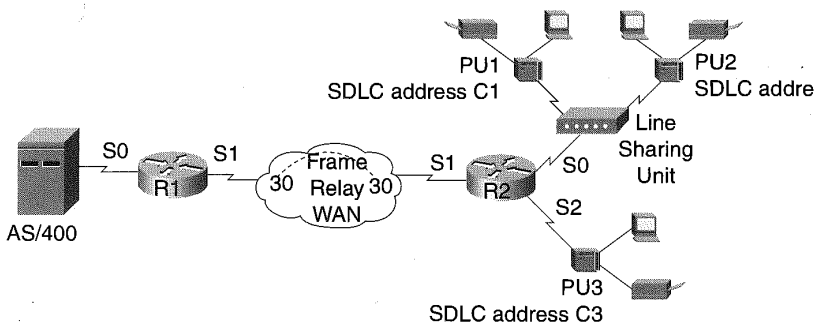


Figure 22-3 STUN using Frame Relay encapsulation with local acknowledgement.

```
C3 interface s1 dlci 30 04 local-ack stun sdlc-role secondarysdlc address
C1sdlc address C2sdlc address C3!interface serial 1ip address 172.16.4.1
255.255.255.0encapsulation frame-relayframe-relay lmi-type Annex Dframe-
relay map ip 172.16.4.2 30 broadcast
```

R2 configuration applied to Figure 22-3:

```
stun peer-name 172.16.2.1 stun protocol-group 1 sdlc!interface loopback
0ip address 172.16.2.1 255.255.255.0!interface serial 0encapsulation
stunstun group 1 stun route address C1 interface s1 dlci 30 04 local-ack
stun route address C2 interface s1 dlci 30 04 local-ack stun sdlc-role
primarysdlc address C1sdlc address C2sdlc address C3!interface serial
1ip address 172.16.4.2 255.255.255.0encapsulation frame-relayframe-relay
lmi-type Annex Dframe-relay map ip 172.16.4.1 30 broadcast!interface
serial 2encapsulation stunstun group 1 stun route address C3 interface
s1 dlci 30 04 local-ack stun sdlc-role primarysdlc address C3
```

STUN using direct Frame Relay encapsulation is made possible using the following interface command:

stun route address *sdlc-addr* **interface** *frame-relay-port* **dlci** *number localsap* **local-ack**

Frame Relay encapsulation with STUN must be used with local acknowledgement. The *sdlc-addr* value is the SNA PU station address found at the remote end of the Frame Relay connection. The interface connecting to the Frame Relay connection is specified by the *frame-relay-port* value, which identifies the serial port on the router.

Frame Relay has virtual connections on a single link. Each virtual connection, called permanent virtual circuit (PVC), is identified by a number. The number is the Data Link Connection Identifier (DLCI). Mapping the STUN connection to a DLCI is accomplished by specifying the *number* value after the dlci keyword. Each end-to-end connection will use a local Service Access Point (SAP) port for transmitting the encapsulated SDLC frame over the DLCI. The *localsap* value defines the SAP number in use for this SNA controller.

Finally, the **local-ack** keyword, which is required for Frame Relay encapsulation ensures that the far-end SNA devices will not timeout their SNA sessions due to delays in the Frame Relay network.

Source-Route Bridging Configuration

Source-Route Bridging (SRB) was established by IBM as the standard for delivering data over Token Ring networks. The Cisco implementation of SRB enables a Cisco router to function as a multiport Token Ring bridge.

The Cisco implementation also incorporates the concept of a virtual ring to which all physical rings can be bridged, enabling ring-to-ring connectivity without the rings being physically connected by a real ring. Figure 22-4 illustrates the use of a Cisco router as a multiport bridge utilizing a virtual ring to allow connectivity between all the rings.

In Figure 22-4, router R1 connects four Token Ring LANs. Ring number 1 connects the IBM 3745 FEP to the router, while ring numbers 2, 3, and 4 connect local end user rings. The configuration of router R1 to support this topology is as follows:

R1 configuration applied to Figure 22-4:

```
source-bridge ring-group 10!interface tokenring 0no ip addresssource-
bridge 1 1 10!interface tokenring 1no ip addresssource-bridge 2 1
10!interface tokenring 2no ip addresssource-bridge 3 1 10!interface
tokenring 3no ip addresssource-bridge 4 1 10
```

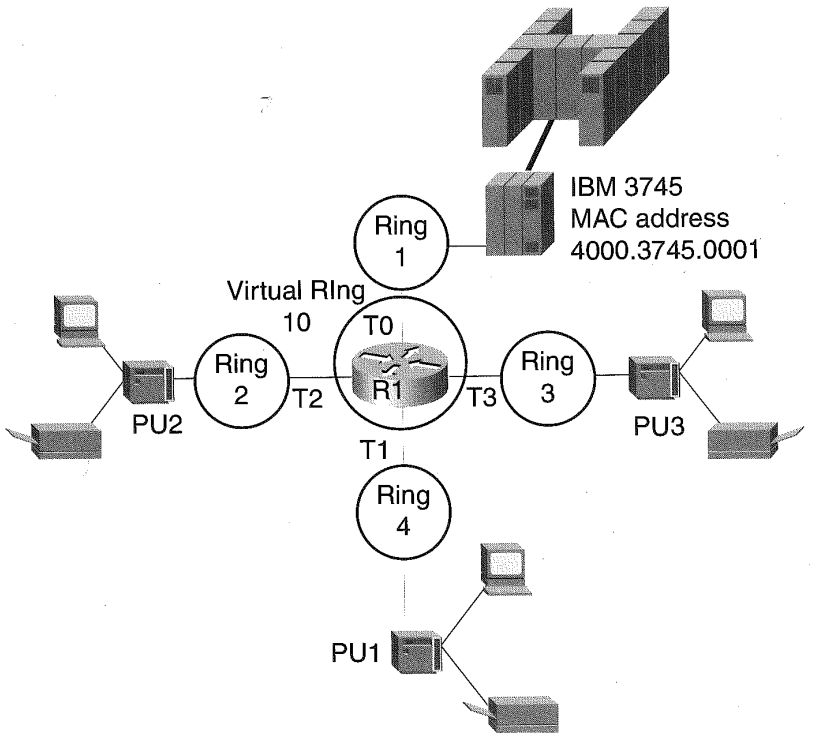


Figure 22-4
The virtual ring concept enabling the Cisco router to act as a multiport bridge.

Defining SRB on Cisco routers for multiring connectivity requires the specification of a virtual ring. The virtual ring, called a ring-group in Cisco IOS, is defined using the following global configuration command:

source-bridge ring-group *ring-group*

The *ring-group* variable is the unique virtual ring number you have chosen to link the bridged rings in your network. The *ring-group* value ranges from decimal 1 to 4,095 and must be unique in the network. In the example configuration, ring-group number 10 is assigned to the router. Each Token Ring interface on the router must have a definition of the bridge-ring number pair to map the routing information field (RIF) within the Token Ring frame. Specifying the mapping to the virtual ring under a Token Ring interface definition is accomplished using the source-bridge interface command. The format of the command is

source-bridge *local-ring bridge-number target-ring*

The *local ring* parameter is the ring number of the attached Token Ring segment connected to the router Token Ring interface port. This is a decimal number ranging from 1 to 4,095. In the example, each Token Ring interface is assigned a ring-number using the **source-bridge** *local-ring* variable value.

For example, interface tokenring0 defines its ring number as 1, while interface tokenring1 specifies its ring number as 2. The *target-ring* is the ring number assigned to the Token Ring segment attached to a router Token Ring interface port when using local SRB or the virtual ring number in support of multiport bridging. In our example, the virtual ring number 10 is specified because we want all the rings to be able to connect with one another as well as the Token Ring connecting the IBM 3745 FEP. The *target-ring* value ranges from 1 to 4,095.

The *bridge-number* is the number of the bridge that connects the *local-ring* to the *target ring*. The *bridge-number* ranges from 1 to 15. Our example uses the same bridge number for connecting all the Token Ring LANs to the virtual ring. Defining in this manner is allowed since the ring-bridge-ring sequence is unique due to the configuration in use. If another Token Ring interface on the router were connected to ring 1, however, then a different bridge number to the virtual ring would be required to establish a unique RIF.

Token Ring LANs connected over a WAN using Cisco routers as the conduit may also use SRB as the means for connectivity. This type of configuration is considered to be a remote SRB (RSRB) connection. RSRB requires the encapsulation of the Token Ring frame into a form acceptable to the

WAN link in use. RSRB configuration may use TCP, Fast Sequenced Transport (FST), or direct encapsulation for delivering the Token Ring frame from one ring to another over a WAN. Figure 22-5 illustrates this type of configuration.

The configuration used by the routers in Figure 22-5 is as follows:

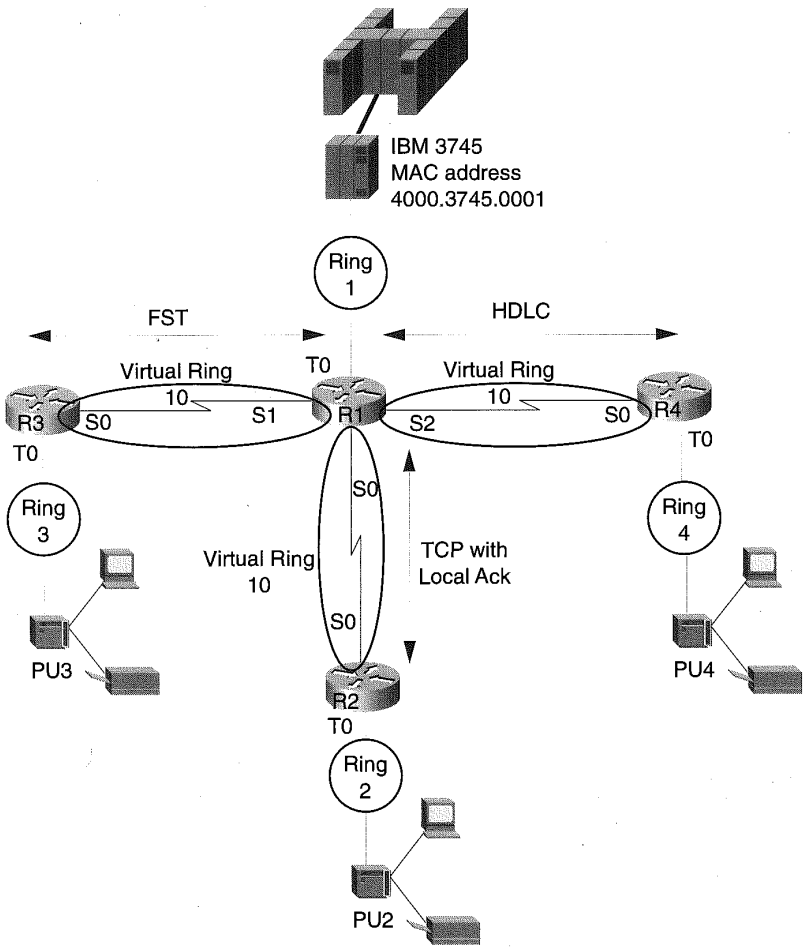


Figure 22-5 RSRB encapsulation configurations for delivering SRB traffic over a WAN.

R1 configuration applied to Figure 22-5:

```
source-bridge ring-group 10source-bridge remote-peer 10 tcp 172.16.1.1
source-bridge remote-peer 10 fst 172.16.3.1source-bridge remote-peer 10
tcp 172.16.2.1 local-ack
source-bridge remote-peer 10 fst 172.16.3.1
source-bridge remote-peer 10 interface serial 2!interface tokenring 0
no ip address
source-bridge 1 1 10!interface serial 0
ip address 172.16.252.1 255.255.255.0!!interface serial 1
ip address 172.16.253.1 255.255.255.0!!interface serial 2
ip address 172.16.254.1 255.255.255.0!interface loopback 0 ip address
172.16.1.1 255.255.255.0
!
router eigrp 10
 network 172.16.0.0
```

R2 configuration applied to Figure 22-5:

```
source-bridge ring-group 10source-bridge remote-peer 10 tcp
172.16.2.1source-bridge remote-peer 10 tcp 172.16.1.1 local-ack!inter-
face tokenring 0
no ip address
source-bridge 2 1 10!interface serial 0
ip address 172.16.252.2 255.255.255.0!interface loopback 0 ip address
172.16.2.1 255.255.255.0
!
router eigrp 10
 network 172.16.0.0
```

R3 configuration applied to Figure 22-5:

```
source-bridge ring-group 10source-bridge remote-peer 10 fst
172.16.3.1source-bridge remote-peer 10 fst 172.16.1.1!interface token-
ring 0
no ip address
source-bridge 3 1 10!interface serial 0
ip address 172.16.253.2 255.255.255.0!interface loopback 0 ip address
172.16.3.1 255.255.255.0
!
router eigrp 10
 network 172.16.0.0
```

R4 configuration applied to Figure 22-5:

```
source-bridge ring-group 10source-bridge remote-peer 10 interface serial
0!interface tokenring 0
no ip address
source-bridge 4 1 10!interface serial 0
ip address 172.16.254.2 255.255.255.0!interface loopback 0 ip address
172.16.4.1 255.255.255.0
!
router eigrp 10
 network 172.16.0.0
```

The key command for supporting SRB over the WAN is the **source-bridge remote-peer** global configuration command. Three types of encapsulation techniques can be used for RSRB. The first one discussed is TCP encapsulation. For each of the encapsulation types, a **source-bridge remote-peer** global configuration command is required to identify the RSRB peer on the local router and at least one other remote RSRB peer.

When transporting the bridged traffic over the WAN, we encapsulate the data in a routable shell, in this case TCP. The advantage of TCP encapsulation is the reliable characteristics of the transport mechanism. The benefits are as follows:

- possible packet reordering
- timers for retransmission
- acknowledgements

As a result, one must pay careful attention to the drain on the CPU. In some cases with TCP encapsulation, you may have two layers of protocols managing the life of the session. TCP will reorder packets and set timers while the underlying SNA devices perform the SNA reordering and timing.

There are no hard and fast rules for peering encapsulation. We can suggest that if your CPU is high, and your lines are close to being heavily utilized, FST may be a better choice. Here are the specifics for TCP encapsulation peer definitions:

source-bridge remote-peer *ring-group* **tcp** *ip-address* [**local-ack**]

The *ring-group* is the network-wide virtual ring number, and the *ip-address* value is the IP address of a router acting as a remote peer for the connection. The **local-ack** keyword is specified for having the router perform acknowledgement of SNA supervisory frames RR, RNR, and REJ. In the example, router R1 specifies TCP encapsulation to router R2 using local acknowledgement. The TCP encapsulation here is noted by the pairing of the IP address values in the source-bridge remote-peer tcp command in the router R1 and router R2 configurations.

A less taxing form of encapsulation for bridge peers is Fast Sequenced Transport (FST). FST uses the basic IP datagram as its mechanism with a sequence number. The router will interrogate the sequence number to see if it is greater than the last one received. If it is, it puts it out on the rings; if it is less, it discards the frame. FST relies on the underlying protocol, in this case Token Ring LLC2, to retransmit the packet. This has cost savings

on the CPU and works well in the long haul of WAN networks with multiple hops. The format for specifying FST encapsulation is

source-bridge remote-peer *ring-group* **fst** *ip-address*

The *ring-group* is the network-wide virtual ring number, or partial network, and the *ip-address* is the address of the router on the remote end of the link. Router R1 and R3 in the example configurations for Figure 22-5 illustrate the specification of FST for encapsulation SRB traffic over a WAN. Although FST causes less router CPU overhead, it does not have the capability to protect the WAN links from unproductive SNA supervisory frame transmissions because it doesn't support local acknowledgement.

source-bridge remote-peer *ring-group* **interface** *interface-name* [*mac-address*]

The *ring-group* is the network-wide virtual ring number specified on a **source-bridge ring-group** global command. The *mac-address* value is optional and is the MAC address of the remote interface directly connected to the router. The *mac-address* value is typically used when encapsulating a Token Ring frame over a LAN or FDDI ring. The *interface-name* value is any valid interface on the router that supports direct encapsulation. These are

- serial
- Ethernet
- FDDI
- ATM
- Token Ring

Direct encapsulation for Frame Relay is accomplished using the following command:

source-bridge remote-peer *ring-group* **frame-relay interface** *interface-name [mac-address] [dlci-number] [lf size]*

The *ring-group*, *interface-name,* and *mac-address* values are entered just as those specified for **the source-bridge ring-peer** interface command. The *dlci-number* value is the valid DLCI number assigned to the RSRB peer associated with the connection. The lf value is the largest frame that can be expected on the connection. The possible byte values are

- 516
- 1500
- 2052
- 4472
- 8144
- 11407
- 17800

The example given uses HDLC encapsulation over the serial line connection between router R1 and R4. Direct serial encapsulation requires a point-to-point connection.

In many corporate networks, there is a combination of Ethernet and Token Ring LANs that may require an SNA device attached using Ethernet to communicate with an SNA device attached to Token Ring. This is accomplished using transparent and source-route bridging together. The Ethernet IEEE 802.3 frame formats differ from the Token Ring IEEE 802.5 frame formats and must be translated for communication to occur between the two SNA devices. This scenario is depicted in Figure 22-6. In the figure, an Ethernet-attached SNA device communicates with an IBMN 3745 FEP. The Cisco IOS enables this communication by employing Source-Route/Translational Bridging (SR/TLB).

The router configuration used for Figure 22-6 follows:

R1 configuration applied to Figure 22-6:

```
source-bridge ring-group 10
source-bridge transparent 10 2 1 1
!
interface tokenring 0
source-bridge 1 1 10
!
interface tokenring 1
source-bridge 2 1 100
!
interface Ethernet 0
bridge-group 1
!
bridge 1 protocol IEEE
```

Two transparent bridging commands are required to bridge Ethernet frames over Cisco routers. The first is a global configuration command:

bridge *bridge-group* **protocol** [ieee | **dec**]

The *bridge-group* variable value ranges from 1 to 63. The **ieee** or **dec** keywords are used to specify the type of transparent bridging protocol being used for the identified bridge group. Our example uses the IEEE bridging protocol. Bridge-group 1 is defined using the **bridge protocol** global configuration command. The *bridge-group* is specified on any Ethernet interface that will be considered an extension of the same Ethernet LAN. The example specifies the **bridge-group** interface command on the Ethernet 0 interface of router R1 to identify bridge-group 1 is to be used for this interface.

The transparent bridge domain is viewed by the SRB domain as a ring using the Cisco IOS software virtual ring feature. A ring and bridge number are associated with the entire transparent bridge domain. From the transparent bridge domain, the SRB domain is viewed as another port on the transparent bridge domain.

When bridging from the SRB domain to the transparent domain, the RIF fields are cached by the router for use in the return frame. Bridging from transparent to SRB, the router first determines if the packet is a multicast, broadcast, or a unicast packet. When the received packet is a multicast or broadcast, the router forwards the packet as a spanning-tree explorer.

For unicast packets, the router first determines if an RIF is cached for

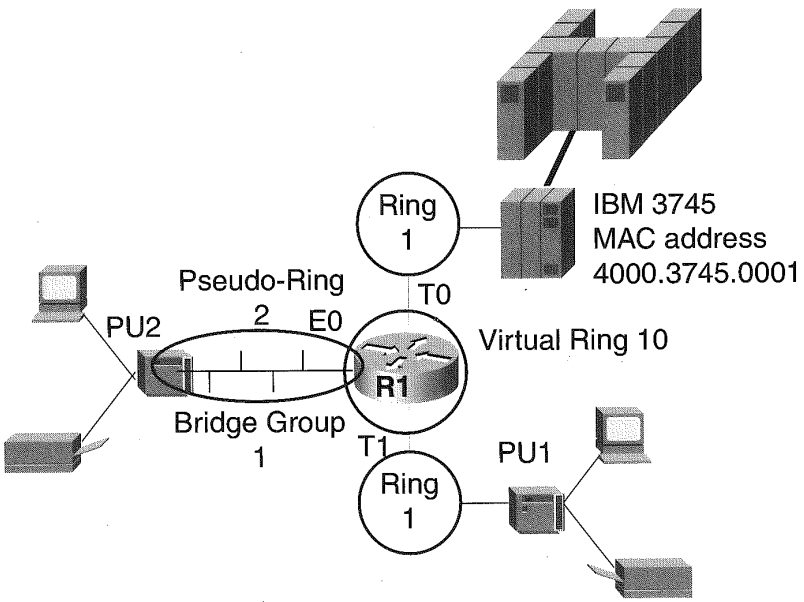


Figure 22-6 Example network configuration employing SR/TLB.

the destination MAC address in the packet. If a path is not found, the router sends the packet as a spanning-tree explorer.

SR/TLB is enabled by entering the following global command:

source-bridge transparent *ring-group pseudo-ring bridge-number bridge-group*

The *ring-group* variable is the virtual ring number assigned to the **source-bridge ring-group** global command used for connecting SRB networks by the router. This value is 10 in our example. The *pseudo-ring* is the virtual ring number assigned to the transparent bridging domain. The example specifies the pseudo-ring as number 2. The *bridge-number* variable is the virtual bridge connecting the transparent bridge domain *pseudo-ring* to the *ring-group* virtual SRB ring. In our example, bridge number 1 is specified.

Finally, the *bridge-group* variable is the transparent bridge group number that frames can be passed through to the SRB network. Note that all interfaces assigned to the transparent bridge domain using the interface command bridge-group have the potential for sending frames onto the SRB network. Usually, it is best to assign a unique bridge-group number, if possible, to the interface that requires connectivity to an SRB-attached station.

Data Link Switching Configuration Plus (DLSw+)

Data Link Switching Configuration Plus (DLSw+) is Cisco's implementation of the RFC 1795 Data Link Switching standard defined by the IETF. DLSw+ is actually a TCP application that connects to another DLSw+ TCP application and is executed by the DLSw+ application.

A TCP application is preferred over an IP transport mechanism provided by RSRB, even though router CPU overhead may be increased in some situations. The DLSw+ TCP segment is encapsulated for delivery to a partner or DLSw+ peer using direct, TCP, FST, or Frame Relay encapsulation methods. Figure 22-7 diagrams the use of the these encapsulation methods.

Figure 22-7 displays four connections deployed for establishing SRB traffic to the data center router R1. The first is direct HDLC connectivity from R1 to R2. Just as in RSRB, HDLC encapsulation must be on a point-to-point connection. TCP encapsulation is used from R1 to R3 over the WAN, and

FST is used from R1 to R4 over the same serial interface of R1. The remote router R5 connects to the data center router R1 using a Frame Relay connection. This connection employs Frame Relay definitions as well as a DLSw+ specification for the Frame Relay link.

The following router configurations apply to Figure 22-7:

R1 configuration applied to Figure 22-7:

```
source-bridge ring-group 1
dlsw local-peer peer-id 172.16.1.1
dlsw remote-peer 0 frame-relay interface serial 1 30
!
interface loopback 0
 ip address 172.16.1.1 255.255.255.0
!
interface serial 0
```

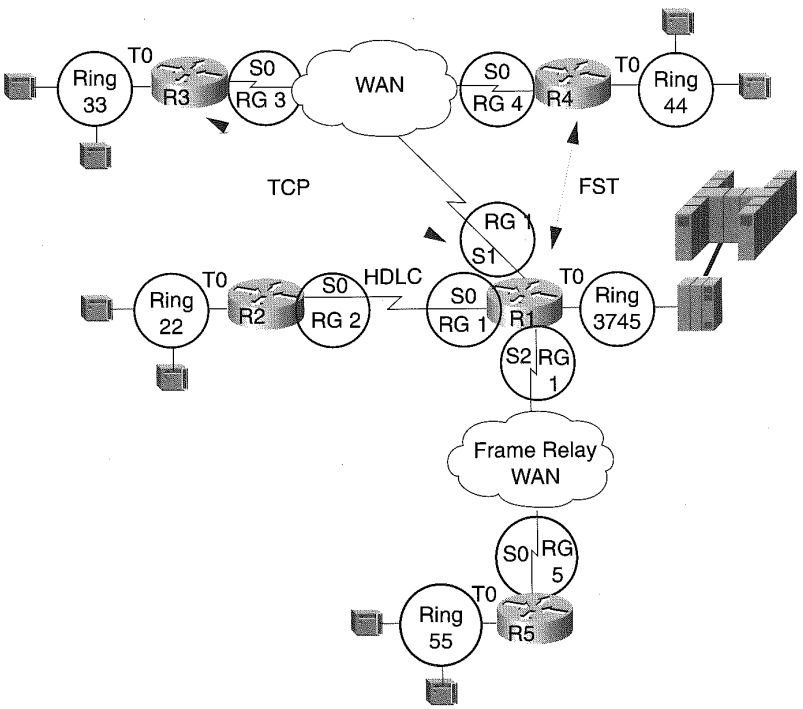


Figure 22-7 Cisco router DLSw+ network topology depicting the various encapsulation methods used by DLSw+.

```
description supports DIRECT HDLC ENCAPSULATION OF DLSw+ to R2
 ip address 172.16.253.1 255.255.255.0
!
interface serial 1
description TCP and FST ENCAPSUALITON OF DLSw+ to R3 and R4
 ip address 172.16.251.1 255.255.255.0
!
interface serial 1
 description Supports DIRECT FRAME RELAY ENCAPSULATION OF DLSw+ to R5
 ip address 172.16.254.1 255.255.255.0
 encapsulation frame-relay
 frame-relay lmi-type ansi
 frame-relay map llc2 30
!
interface tokenring 0
 ring-speed 16
 source-bridge 3745 1 1
```

R2 configuration applied to Figure 22-7:

```
source-bridge ring-group 2
dlsw local-peer peer-id 172.16.2.2
dlsw remote-peer 0 interface serial 0
!
interface serial 0
description DIRECT HDLC ENCAPSULATION OF DLSw+
 ip address 172.16.253.2 255.255.255.0
!
interface tokenring 0
 ring-speed 16
 source-bridge 22 1 2
!
router eigrp 10
 network 172.16.0.0
```

R3 configuration applied to Figure 22-7:

```
source-bridge ring-group 3dlsw local-peer peer-id 172.16.3.3
dlsw remote-peer 0 tcp 172.16.1.1
!
interface loopback 0
 ip address 172.16.3.3 255.255.255.0
!
interface serial 0
 description TCP ENCAPSULATION FOR DLSw+
 ip address 172.16.251.3 255.255.255.0
!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 33 1 3
 source-bridge spanning
!
router eigrp 10
 network 172.16.0.0
```

R4 configuration applied to Figure 22-7:

```
source-bridge ring-group 4dlsw local-peer peer-id 172.16.4.4
dlsw remote-peer 0 fst 172.16.1.1
!
interface loopback 0
 ip address 172.16.4.4 255.255.255.0
!
interface serial 0
 description FST ENCAPSULATION OF DLSw+
 ip address 172.16.251.4 255.255.255.0
!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 44 1 4
 source-bridge spanning!
router eigrp 10
 network 172.16.0.0
```

R5 configuration applied to Figure 22-7

```
source-bridge ring-group 5
dlsw local-peer peer-id 172.16.5.5
dlsw remote-peer 0 frame-relay interface serial 0 30
!
interface serial 0
description DIRECT FRAME RELAY ENCAPSULATION OF DLSw+
 ip address 172.16.254.5 255.255.255.0
 encapsulation frame-relay
 frame-relay lmi-type ansi
 frame-relay map llc2 30
!
interface loopback0
 ip address 172.16.5.5 255.255.255.0
!
interface tokenring 0
ring-speed 16
 source-bridge 55 1 5!
router eigrp 10
 network 172.16.0.0
```

DLSw+ implementation requires the use of a virtual ring. The format of the virtual ring specification is the same as that specified under RSRB configuration. The difference between the two uses is that DLSw+ does not require the same virtual ring definition on all routers participating in the connection. DLSw+ terminates the SRB RIF locally and caches the RIF accordingly with the DLSw+ circuit created. This allows designers to use different virtual rings at each location or even the same virtual ring at each location. In the configuration examples for Figure 22-7, a unique virtual ring is specified on each router to assist in identifying the origination of the frame. This is helpful in debugging connectivity and performance problems.

DLSw+ is enabled by using the **dlsw local-peer** global command. The **dlsw local-peer** global command identifies the parameter values used by the DLSw+ peer instance on this router. The format of the command is

dlsw local-peer [**peer-id** *ip-address*] [**group** *group*] [**border**] [**cost** *cost*] [**lf** *size*] [**keepalive** *seconds*] [**passive**] [**promiscuous**] [**init-pacing-window** *size*] [**max-pacing-window** *size*][**biu-segment**]

The **peer-id** keyword identifies the *ip-address* value used when DLSw+ is implementing FST or TCP. It is best to use the IP address of the loopback interface for the DLSw+ *ip-address* value since the loopback interface does not rely on a physical port and is therefore always active.

If this router is being defined in a peer group, the keyword **group** must be defined with a *group* value of the peer group used for this router. The value of *group* is a decimal range of 1 to 255. When participating in a peer group, it is possible for this router to act as the border peer. Border peer functionality is enabled using the **border** keyword of the **dlsw local-peer** command. The capabilities exchange of DLSw+ includes the assignment of a weighted cost for this router to successfully establish DLSw+ connections.

The weighted *cost* value is specified by the optional **cost** keyword. The *cost* value specified must be in the decimal range of 1 to 5. If defined, it is this value that is advertised to remote peers via the capabilities exchange. The *cost* value is useful when there are multiple peers to a destination. The value determines which router is a preferred peer and which is a capable peer. The *cost* keyword is applicable only in fault tolerance mode.

During capabilities exchange the DLSw+ peers will also determine the largest frame size accepted over a connection. This is determined by the exchange of the *size* value specified for the optional **lf** keyword with these possible values:

- **516**
- **1470**
- **1500**
- **2052**
- **4472**
- **8144**
- **11407**
- **11454**
- **17800**

The DLSw+ peers negotiate the largest frame size based on the lowest value defined for each peer. The optional keyword **keepalive** specifies the default remote peer keepalive interval in seconds. The default value for *seconds,* should the **keepalive** keyword not be coded, is 30 seconds. The valid range is 0 to 100 seconds. A value of 0 indicates that no keepalive messages are being exchanged between the peers.

The **passive** optional keyword indicates that this router will not be the initiator of peer connections to configured remote peers. It will only establish peer connections when requested by another peer. Specifying the optional keyword **promiscuous** enables this router to act as a dynamic peer for connecting to non-configured remote peers.

Using the **biu-segment** optional keyword causes each end station of the circuit to send the largest frame size it is capable of sending. This keyword is a performance and utilization enhancement that allows efficient use of the WAN links. If a frame received at a peer from a remote peer is larger than that capable of the end station to handle, DLSw+ will segment the frame prior to sending it to the end station. Implementing this function requires the definition of **biu-segment** on both DLSw+ peers.

The RFC 1795 defines an initial pacing window *size* variable with a valid range of 1 to 2,000. The optional keyword **init-pacing-window** *size* value allows you to specify this variable. The keyword **max-pacing-window** specifies the maximum size of the pacing window set forth by RFC 1795. The value is set by specifying the *size* variable following the keyword. The valid value ranges from 1 to 2,000.

The **dlsw local-peer** global commands used in Figure 22-7 use the IP address of the loopback interface, with the exception of the HDLC example. The interface loopback definition is not warranted for this type of connection because there is only one way for the remote location at R2 to connect to router R1's data center. If a dial-backup connection were in use, then a loopback interface IP address as the HDLC peer-id value is most certainly warranted to guarantee DLSw+ peer connections when the backup line is invoked.

Each DLSw+ router identifies the DLSw+ peer using the **dlsw remote-peer** global command. This command identifies the DLSw+ peer IP address or the router interface being used for direct or Frame Relay encapsulation. The format of the dlsw remote-peer global command is:

dlsw remote-peer *list-number* **interface serial** *number*

[**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number*
| **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**cost** *cost*]

[**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*]
[**host-netbios-out** *host-list-name*] [**keepalive** *seconds*] [**lf** *size*]
[**linger** *minutes*] [**lsap-output-list** *list*] [**passive**] [**pass-thru**]

dlsw remote-peer *list-number* **frame-relay interface serial** *number dlci-number*

[**backup-peer** [*ip-address* | **frame-relay interface** *serial number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**cost** *cost*]
[**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*]
[**host-netbios-out** *host-list-name*] [**keepalive** *seconds*] [**lf** *size*]
[**linger** *minutes*] [**lsap-output-list** *list*] [**passive**] [**pass-thru**]

dlsw remote-peer *list-number* **fst** *ip-address*

[**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**cost** *cost*]
[**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*]
[**host-netbios-out** *host-list-name*] [**keepalive** *seconds*] [**lf** *size*]
[**linger** *minutes*]
[**lsap-output-list** *list*] [**passive**]

dlsw remote-peer *list-number* **tcp** *ip-address*
[**backup-peer** [*ip-address* | **frame-relay interface serial** *number dlci-number* | **interface** *name*]] [**bytes-netbios-out** *bytes-list-name*] [**cost** *cost*]

[**dest-mac** *mac-address*] [**dmac-output-list** *access-list-number*]
[**dynamic**] [**host-netbios-out** *host-list-name*] [**keepalive** *seconds*] [**lf** *size*]
[**linger** *minutes*] [**lsap-output-list** *list*] [**no-llc** *minutes*] [**passive**] [**priority**]
[**tcp-queue-max** *size*] [**timeout** *seconds*]

The **dlsw remote-peer** *list-number* parameter identifies the specific **dlsw-ring-list** used for connecting to the dlsw remote peer. The valid range for the list-number is 1 to 255. If a **dlsw ring-list** has not been specified, the default value 0 is used. A *list-number* value of 0 denotes that all ports or bridge groups with DLSw+ enabled forward explorer packets to establish connectivity to the remote peer. If a value other than 0 is specified, the value must match the number you specified on a **dlsw ring-list, dlsw port-list,** or **dlsw bgroup-list** command.

The *number* value following the **dlsw remote-peer** keywords **interface serial** identifies the physical serial port used for direct encapsulation of DLSw+ packets. The value is any valid port of slot/port combination used on the router that connects to the remote peer router. An example of using direct HDLC encapsulation to a remote DLSw+ router is

dlsw remote-peer 0 interface 0/1

The example specifies that all rings are available for connecting to the remote peer found at the remote side of the connection on serial interface 0 of the router.

The major distinction of defining remote DLSw+ peers between HDLC and Frame Relay is the inclusion of the **frame relay** keyword and the *dlci-number* parameter on the **dlsw remote-peer frame-relay interface serial** global command. The **frame relay interface serial** positional parameter *number* identifies the Frame Relay port, slot/port, or subinterface of a port used for direct connection to the remote peer. The *number* parameter must be followed by the positional *dlci-number* parameter, which identifies the Frame Relay DLCI of the remote router used for connecting to the remote peer. An example of using the **dlsw remote-peer** command for direct Frame Relay encapsulation is

dlsw remote-peer 0 frame relay interface serial 0 30

This **dlsw remote-peer frame relay** command specifies the use of Frame Relay serial interface 0, using DLCI 30 as the connection identifier to the remote peer.

There is a common optional parameter specific to direct encapsulation: the **pass-thru** parameter. Specifying the **pass-thru** parameter disables the local-acknowledgement feature of DLSw+ and therefore passes SNA RR and RNR polling from end-to-end.

DLSw+ uses fast-sequenced transport (FST) encapsulation when the **dlsw remote-peer** command is issued using the **fst** keyword. The *ip-address* parameter following the **fst** keyword specifies the IP address of the remote peer being defined for which this router can establish a peer connection. An example of enabling FST encapsulation with DLSw+ is as follows:

dlsw remote-peer 1 fst 10.1.1.1

The example identifies that the ports and bridge groups assigned to ring-list 1 are used for forwarding explorer packets when discovering a destination SNA device. The remote peer to receive these explorer packets is identified by IP address 10.2.1.1.

DLSw+ uses TCP encapsulation to a remote peer through the defining of the **dlsw remote-peer** command with the **tcp** keyword specified. The *ip-address* parameter following the **tcp** keyword specifies the IP address of the remote peer being defined for which this router can establish a peer connection. An example of enabling TCP encapsulation with DLSw+ is as follows:

```
dlsw remote-peer 0 tcp 10.1.1.1
```

The example indicates that all ports and bridge groups will forward explorer packets due to the 0 coded for the list-number positional parameter. The remote peer to receive these explorer packets is identified by IP address 10.1.1.1.

TCP encapsulation has five optional parameters that are specific to the **dlsw remote-peer tcp** global command. These are

- [**dynamic**]
- [**no-llc** *minutes*]
- [**priority**]
- [**tcp-queue-max** *size*]
- [**timeout** *seconds*]

Using the optional **dynamic** keyword allows DLSw+ to establish peer connections only when they are needed with the remote peer. Once there is no more DLSw+ data to be sent to the remote peer, the TCP connection is torn down. The **no-llc** keyword is used when you want the peer connection to the defined remote peer to disconnect after all LLC2 connections have been terminated. The *minutes* value for the **no-llc** keyword must be a valid number in the range of 1 to 300 minutes. The default time if the **no-llc** keyword is not coded is five minutes. The function of the **no-llc** keyword is implemented only when the **dynamic** keyword is specified.

Using the **priority** keyword enables TCP prioritization by port number. The TCP ports and their analogous priority are listed in Table 22-1. Traffic is assigned to the ports based on the priority group assignments. APPN over DLSw+ carries the APPN Class of Service (COS) characteristics through the routed network by mapping the APPN COS to the TCP TOS for SNA TOS.

The **tcp-queue-max** keyword is used to tune the performance of the TCP encapsulation for the defined peer. The *size* parameter indicates the maximum number of output TCP segments allowed to be queued for the

remote peer. The valid range is 10 to 200. The default is 100. If acknowledgement of a TCP segment is not received from the remote peer within 90 seconds, the router retransmits the segment. This default is modified by specifying a new retransmit time limit using the timeout keyword. The retransmit time limit for TCP can range anywhere from five to 1,200 seconds. The default is 90 seconds.

DLSw+ enables a network designer to reduce configuration and bandwidth overhead through the use of peer groups and border peers. The establishment of peer groups is accomplished by including the **group** keyword on the **dlsw local-peer** global command. Preserving connections and CPU overhead within the router in support of DLSw+ peer connections is further managed by using the **promiscuous** keyword on the **dlsw local-peer** global configuration command. Including the **promiscuous** keyword enables the DLSw+ process to establish connections for partners only when needed on a dynamic basis, versus having to maintain the DLSw+ peer connection even if there are no end-user sessions requiring connectivity between the two routers.

Figure 22-8 illustrates a peer group using two border peers in each group for redundancy. The router configurations following Figure 22-8 demonstrates the use of the **group, border,** and **promiscuous** keywords used to establish the peer groups in Figure 22-8.

The following router configurations apply to the routers in Figure 22-8:

R1 configuration applied to Figure 22-8:

```
source-bridge ring-group 1
dlsw local-peer peer-id 172.16.1.1 group 1 border promiscuousdlsw
remote-peer 0 tcp 172.16.5.1dlsw remote-peer 0 tcp 172.16.6.1dlsw
remote-peer 0 tcp 172.16.2.1!
interface loopback 0
 ip address 172.16.1.1 255.255.255.0
!
interface serial 0
 ip address 172.16.12.12 255.255.255.0
!
interface serial 1
 ip address 172.16.15.15 255.255.255.0
!
interface serial 2
 ip address 172.16.16.16 255.255.255.0
!
interface tokenring 0
 source-bridge 3745 1 1
!
router eigrp 10
 network 172.16.0.0
```

Table 22-1 Valid Port Numbers for DLSw+ TCP Connections

Priority	Port
High	2,065
Medium	1,981
Normal	1,982
Low	1,983

R2 configuration applied to Figure 22-8:

```
source-bridge ring-group 1
dlsw local-peer peer-id 172.16.2.1 group 2 border promiscuousdlsw
remote-peer 0 tcp 172.16.1.1dlsw remote-peer 0 tcp 172.16.3.1dlsw
remote-peer 0 tcp 172.16.4.1
!interface loopback 0
 ip address 172.16.2.1 255.255.255.0
!
interface serial 0
 ip address 172.16.12.21 255.255.255.0
!
interface serial 1
 ip address 172.16.24.24 255.255.255.0
!
interface serial 2
 ip address 172.16.22.23 255.255.255.0
!
interface tokenring 0
 source-bridge 2 1 1
!
router eigrp 10
 network 172.16.0.0
```

R3 configuration applied to Figure 22-8:

```
source-bridge ring-group 1
dlsw local-peer peer-id 172.16.3.1 group 2 promiscuousdlsw remote-peer 0
tcp 172.16.2.1dlsw remote-peer 0 tcp 172.16.4.1
!interface loopback 0
 ip address 172.16.3.1 255.255.255.0!
interface serial 0
 ip address 172.16.34.34 255.255.255.0
!
interface serial 2
 ip address 172.16.32.23 255.255.255.0
!
interface tokenring 0 source-bridge 3 1 1
!
router eigrp 10
 network 172.16.0.0
```

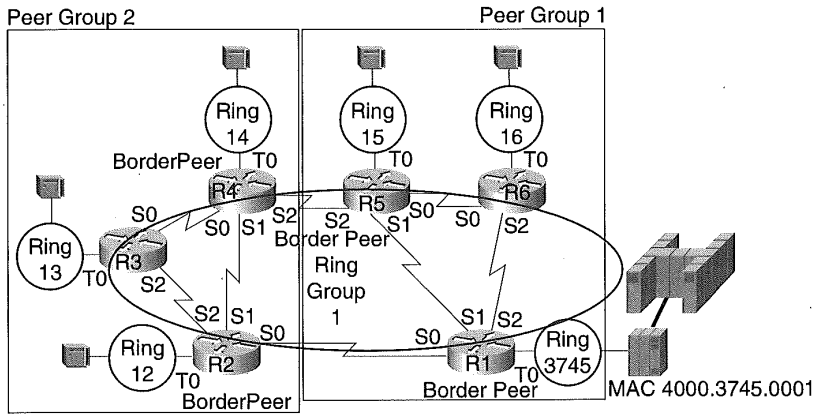


Figure 22-8
DLSw+ peer group configuration using dual border peers for redundancy.

R4 configuration applied to Figure 22-8:

```
source-bridge ring-group 1
dlsw local-peer peer-id 172.16.4.1 group 2 border promiscuousdlsw
remote-peer 0 tcp 172.16.3.1dlsw remote-peer 0 tcp 172.16.2.1dlsw
remote-peer 0 tcp 172.16.5.1
!interface loopback 0
 ip address 172.16.4.1 255.255.255.0
!
interface serial 0
 ip address 172.16.34.43 255.255.255.0
!
interface serial 1
 ip address 172.16.24.42 255.255.255.0
!
interface serial 2
 ip address 172.16.45.45 255.255.255.0
!
interface tokenring 0
 source-bridge 4 1 1
!
router eigrp 10
 network 172.16.0.0
```

R5 configuration applied to Figure 22-8:

```
source-bridge ring-group 1
dlsw local-peer peer-id 172.16.5.1 group 1 border promiscuousdlsw
remote-peer 0 tcp 172.16.4.1dlsw remote-peer 0 tcp 172.16.6.1dlsw
remote-peer 0 tcp 172.16.1.1
!interface loopback 0
 ip address 172.16.5.1 255.255.255.0
!
interface serial 0
 ip address 172.16.56.56 255.255.255.0
```

```
!
interface serial 1
 ip address 172.16.15.51 255.255.255.0
!
interface serial 2
 ip address 172.16.45.54 255.255.255.0
!
interface tokenring 0
 source-bridge 5 1 1
!
router eigrp 10
 network 172.16.0.0
```

R6 configuration applied to Figure 22-8:

```
source-bridge ring-group 1
dlsw local-peer peer-id 172.16.6.1 group 1 promiscuousdlsw remote-peer 0
tcp 172.16.1.1dlsw remote-peer 0 tcp 172.16.5.1!
interface loopback 0
 ip address 172.16.6.1 255.255.255.0
!
interface serial 0
 ip address 172.16.56.65 255.255.255.0
!
interface serial 2
 ip address 172.16.16.61 255.255.255.0
!
interface tokenring 0
 source-bridge 6 1 1
!
router eigrp 10
 network 172.16.0.0
```

The configuration examples for Figure 22-8 do not necessarily require point-to-point connections between the routers acting as border peers. The network topology is shown in this manner to also indicate the importance of physical diversity in connecting the border peers as well as demonstrating the grouping of peers.

Note that in the router configuration examples router R3 and R6 do not have dlsw remote-peer global commands specifying the DLSw+ peer IP addresses for routers in the other peer groups. If connectivity to resources attached to routers other than those defined within its peer group are required, the peer connection is set up dynamically because of the use of the promiscuous keyword on the dlsw local-peer global commands for each router.

The target DLSw+ peer router has the capability to use load balancing and in effect provide high availability to the target resource using the **dlsw duplicate-path-bias** global command. The format of this command is

dlsw duplicate-path-bias [**load-balance**]

The **dlsw duplicate-path-bias** global command without the optional **load-balance** keyword will favor session establishment for the SNA resources over the preferred path to the duplicate MAC address. The preferred path is the path over which the first explorer response is received or the peer connection with the least cost value. Using the **load-balance** keyword enables DLSw+ to establish SNA sessions with the mainframe in a round-robin manner between the two paths.

Figure 22-9 diagrams a DLSw+ network designed to support duplicate paths and load balancing. The two IBM 3745 FEPs in Figure 22-9 use the same Token Ring MAC Address. Establishment of the duplicate path and load balancing feature of DLSw+ is accomplished by specifying the dlsw duplicate-path-bias global command on only the data center router R1.

The following configurations apply to Figure 22-9:

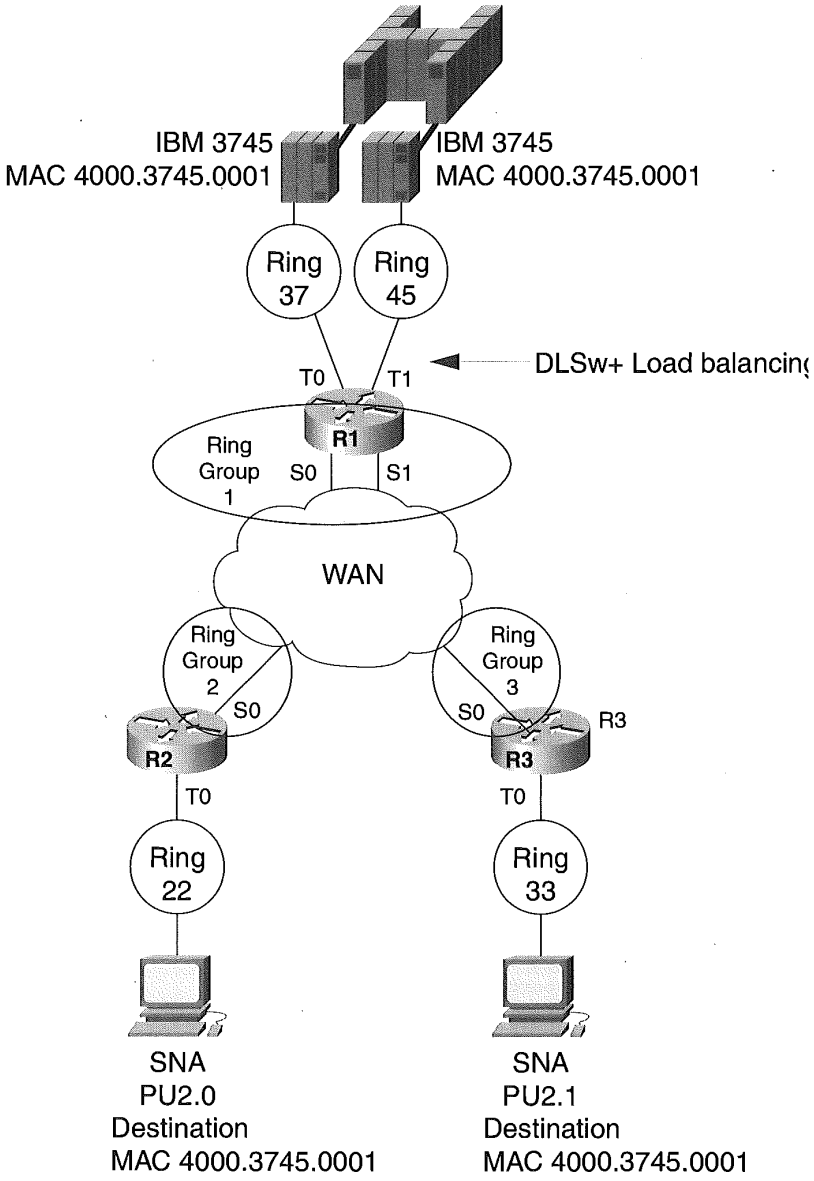
R1 configuration applied to Figure 22-9:

```
source-bridge ring-group 1 !
dlsw local-peer peer-id 172.16.1.1
dlsw remote-peer 0 tcp 172.16.2.1
dlsw remote-peer 0 tcp 172.16.3.1dlsw duplicate-path-bias load-balance
!
interface loopback 0
 ip address 172.16.1.1 255.255.255.0
!
interface serial 0
 ip address 172.16.22.1 255.255.255.0
!
interface serial 1
 ip address 172.16.32.1 255.255.255.0
!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 37 1 1
 source-bridge spanning
!
interface tokenring 1
 no ip address
 ring-speed 16
 source-bridge 45 1 1
 source-bridge spanning
!
router eigrp 10
 network 172.16.0.0
```

R2 configuration applied to Figure 22-9:

```
source-bridge ring-group 2 !
dlsw local-peer peer-id 172.16.2.1
dlsw remote-peer 0 tcp 172.16.1.1
!
interface loopback 0
 ip address 172.16.2.1 255.255.255.0
```

Figure 22-9
Duplicate path support and load balancing using DLSw+.



```
!
interface serial 0
 ip address 172.16.22.2 255.255.255.0
!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 22 1 2
 source-bridge spanning!
router eigrp 10
 network 172.16.0.0
```

R3 configuration applied to Figure 22-9:

```
source-bridge ring-group 3!
dlsw local-peer peer-id 172.16.3.1
dlsw remote-peer 0 tcp 172.16.1.1
!
interface loopback 0
 ip address 172.16.3.1 255.255.255.0
!
interface serial 0
 ip address 172.16.32.2 255.255.255.0
!
interface tokenring 0
 no ip address
 ring-speed 16
 source-bridge 33 1 3
 source-bridge spanning
!
router eigrp 10
 network 172.16.0.0
```

DLSw+ has also built into its functionality the capability to dynamically reroute peer connections to a predefined backup peer using the **backup-peer** keyword definition in the **dlsw remote-peer** global command on the remote routers. The example depicted in Figure 22-10 illustrates backup peering over a Frame Relay network to an Ethernet-attached AS/400. Router R2, R4, and R5 are the routers that are specific to establishing the backup DLSw+ peer connections. The configuration for R4 and R5 specifies the Frame Relay interface and DLCI number used to connect to the backup router R2. Router R2 must also employ translating the Token Ring frame format to Ethernet enabling transport to the AS/400.

The following configurations apply to Figure 22-10:

R2 configuration

```
source-bridge ring-group 10
dlsw local-peer peer-id 172.16.2.1 promiscuous
dlsw remote-peer 0 frame-relay interface serial 0 31dlsw remote-peer 0
frame-relay interface serial 0 34 dlsw bridge-group 1
!
interface loopback 0
```

```
 ip address 172.16.2.1 255.255.255.0
!
interface serial 0
 ip address 172.16.16.1 255.255.255.0
 encapsulation frame-relay
 frame-relay lmi-type ansi
 frame-relay map llc2 30
 frame-relay map llc2 31
!
interface Ethernet 0
 no ip address bridge-group 1!
bridge 1 protocol IEEE
!
router eigrp 10
 network 172.16.0.0
```

R4 configuration applied to Figure 22-10:

```
source-bridge ring-group 10
dlsw local-peer peer-id 172.16.4.1
dlsw remote-peer 0 frame-relay interface serial 0 30dlsw remote-peer 0
frame-relay interface serial 0 31 backup-peer frame-relay interface ser-
ial 0 30
!
interface loopback 0
 ip address 172.16.4.1 255.255.255.0
!
interface serial 0
 ip address 172.16.16.4 255.255.255.0
 encapsulation frame-relay
 frame-relay lmi-type ansi
 frame-relay map llc2 30
 frame-relay map llc2 31
!
interface tokenring 0
 no ip address ring-speed 16 source-bridge 4 1 10
!
router eigrp 10
 network 172.16.0.0
```

R5 configuration applied to Figure 22-10:

```
source-bridge ring-group 10
dlsw local-peer peer-id 172.16.5.1
dlsw remote-peer 0 frame-relay interface serial 0 35dlsw remote-peer 0
frame-relay interface serial 0 34 backup-peer frame-relay interface ser-
ial 0 35
!
interface loopback 0
 ip address 172.16.5.1 255.255.255.0
!
interface serial 0
 ip address 172.16.16.5 255.255.255.0
 encapsulation frame-relay
 frame-relay lmi-type ansi
 frame-relay map llc2 34
 frame-relay map llc2 35
!
interface tokenring 0
 no ip addressring-speed 16source-bridge 5 1 10
!
```

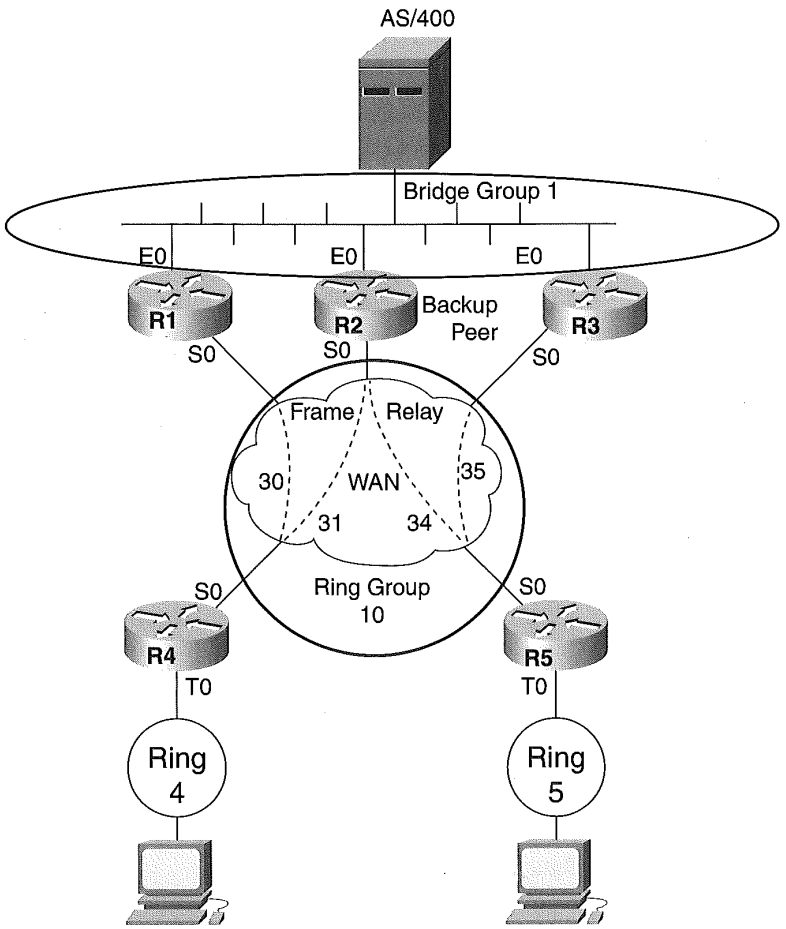


Figure 22-10
DLSw+ backup peer configuration and SR/TLB functionality.

```
router eigrp 10
 network 172.16.0.0
```

The **dlsw bridge-group** global command on router R2 enables automatic SR/TLB to occur without specifying any type of pseudo-ring, as is found with RSRB. However, the bridge definitions for the Ethernet and the bridging protocol being used must still be specified. The configurations indicate the backup connections for router R4 and R5. R4 will use the Frame Relay DLCI 31, should connectivity to Frame Relay DLCI 30 become unavailable, while R5 will use DLCI 34 if connectivity to DLCI 35 is disrupted.

SDLC to LLC2 (SDLLC)

The investment made by corporations in support of their SNA networks leads to expansive WAN SDLC connections to SNA controllers. Although this technology is not "state of the art," it is paramount in many organizations because it enables the delivery of mission-critical business processes built on SNA applications residing on mainframes. The Cisco IOS software protects this investment while at the same time enabling the merger of the two networks over the same WAN by providing SDLC frame to Token Ring LLC2 frame translations.

Two basic configurations are used in SDLLC. The first configuration connects serial SDLC lines that attach SNA controllers to a Token Ring connected to an SNA FEP using Cisco routers. The second configuration reverses this topology and connects a serial SDLC-attached SNA FEP to Token Ring-attached SNA controllers using Cisco routers. This second configuration is referred to as reverse SDLLC.

Figure 22-11 diagrams a forward SDLLC topology. The SNA controller named PU1 in Figure 22-11 connects to router R1 at the data center using an SDLC serial line. Router R1 connects the IBM 3745 FEP using Token Ring. SDLLC requires the use of a virtual ring to enable the mapping and translation of the SDLC to a Token Ring frame. The Cisco IOS configuration for router R1 uses local SRB for the connections. Router R2 uses the same topology, but it uses local RSRB to enable the translation. Finally, router R3 uses DLSw+ as the means for enabling the translation between the SLDC and Token Ring frames. The use of the three routers in Figure 22-11 does not mean that these three different methods cannot operate in the same router. The diagram is depicted in this manner for clarity of the router configurations.

The router configurations for Figure 22-11 follow:

R1 configuration applied to Figure 22-11:

```
source-bridge ring-group 1
!interface tokenring 0 source-bridge 3745 1 1 source-bridge
spanning!interface serial 0 encapsulation sdlc-primary sdlc address c1
sdllc traddr 4000.3174.1000 1174 1 1 sdllc partner 4000.3745.0001 c1
sdllc xid c1 01731741
R2 Configuration applied to Figure 22.11:source-bridge ring-group
2source-bridge remote-peer 2 fst 10.8.1.1source-bridge remote-peer 2 fst
10.8.2.2!interface tokenring 0
 ip address 10.8.1.1 255.255.255.0
 source-bridge 3745 1 2!interface serial 0
 encapsulation sdlc-primary ip address 10.8.2.2 255.255.255.0
```

```
sdlc address c1
sdllc traddr 4000.3174.1000 2174 1 2
sdllc partner 4000.3745.0001 c1
sdllc xid c1 01731742
```

R3 configuration applied to Figure 22-11:

```
source-bridge ring-group 3dlsw local-peer!interface tokenring 0 ring-speed 16
 source-bridge 3745 1 3 source-bridge spanning!
interface serial 0 encapsulation sdlc sdlc role primary sdlc vmac
4000.3174.1000 sdlc address c1 sdlc sdlc-largest-frame C1 1033 sdllc partner
4000.3745.0001 c1 sdllc xid c1 01731743 sdlc dlsw C1
```

SDLLC uses the virtual ring concept to perform the conversion when employing SRB, RSRB, or DLSw+. The virtual ring used for SDLLC is defined using the **source-bridge ring-group** command, identifying a ring number that can be used by all the interfaces on the router. The virtual ring number used for each of the routers in Figure 22-11 matches the router number for clarity.

In the figure, local SRB is in use. The connections between the Token Ring interface and the virtual ring are specified by the source-bridge interface command on Token Ring 0 of R1. Router R2 uses local RSRB and hence requires the use of the source-bridge remote-peer global configuration command to identify the local IP address used for RSRB and the remote IP address used for RSRB.

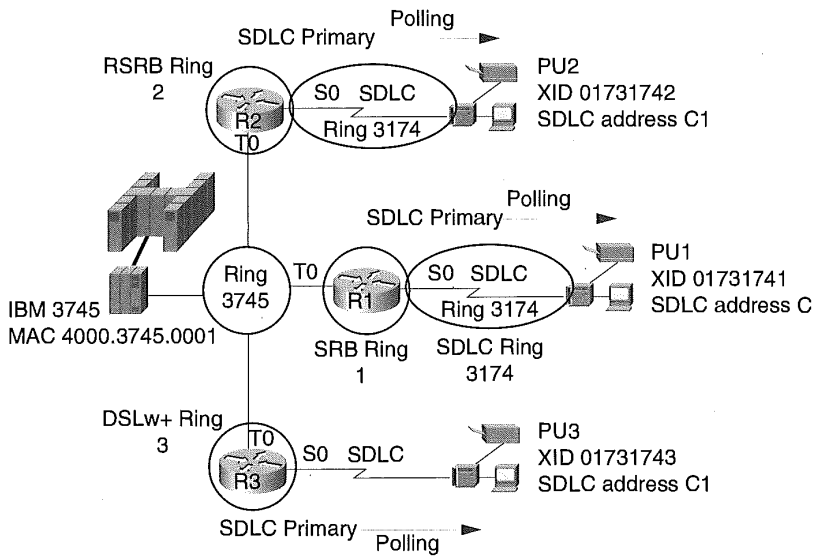


Figure 22-11 SDLLC connectivity using local SRB, RSRB, and DLSw+.

In this example, the IP addresses on the same router are being used. This is a great example of when FST is the preferred encapsulation technique providing less overhead on the CPU than TCP encapsulation. Router R3 is employing local DLSw+, whereby the DLSw+ process is made aware that the translation and delivery of SRB frames is local. In both router R2 and R3, the Token Ring interfaces use the **source-bridge** command to map the tokenring 0 interface to the virtual ring identified on each router by the **source-bridge ring-group global** configuration command.

Instructing the three routers to use SDLLC conversion is accomplished in part by specifying SDLC as the encapsulation technique for the serial interface connecting the SNA controller to the Cisco router. The SDLC encapsulation configuration interface command has three formats:

- **encapsulation sdlc**
- **encapsulation sdlc-primary**
- **encapsulation sdlc-secondary**

SNA SDLC devices can act as either a primary or secondary link station. Primary link stations poll the secondary link stations for data. If the secondary link station has data, it will then send it to the polling primary link station. SNA PU 2.0 resources are defined as secondary link stations; therefore, the SDLLC router connecting to the SNA PU 2.0 node over SDLC defines its encapsulation mode as primary using the **encapsulation sdlc-primary** interface command. The IBM 3745, a PU 4 device, acts as a primary link station when communicating to the attached Cisco router.

In this case, the serial interface connecting to the IBM 3745 is defined as a secondary node using the **encapsulation sdlc-secondary** command. This is used in reverse SDLLC topology, which is discussed below. The **encapsulation sdlc** command is used when RSRB, DLSw+, or FRAS is used as the underlying transport. The **sdlc-primary** and **sdlc-secondary** encapsulation methods are used on routers R1 and R2 to mange the polling through the router. In Figure 22-11, the **encapsulation sdlc** interface command is used on the serial interface connecting to the SDLC device on router R3 because the router is performing local conversion between the SDLC and Token Ring networks, allowing the FEP to issue the polling through the Cisco router without consuming WAN bandwidth. The role of the interface in R3 is determined by the sdlc role interface configuration command.

Making changes to the encapsulation technique on an interface resets the interface, causing connectivity to drop. Changing the role of the interface between primary and secondary in configuration mode using the sdlc role command does not cause the interface to reset, but it may disrupt the SNA sessions. The format of the sdlc role interface configuration command is

sdlc role {none | primary | secondary | prim-xid-poll}

The primary or secondary role of the router is specified by including either the **primary** or **secondary** keyword on the **sdlc role** interface command. If the router is attached to a PU 2.0 device, it must be defined as a primary node. If the router is SDLC-attached to a PU 4 (IBM 3745 FEP), it must be defined as a secondary node. The **none** keyword is used to enable dynamic determination of the role when the attached SNA device uses a negotiable link connection. Typically, this is found when using SNA Type 2.1 nodes as primary nodes. If the attached SNA PU 2.1 device is set to negotiate and prefers a primary role, the router will assume the secondary role. If the attached SNA PU is a PU 2.1 secondary device, **the prim-xid-poll** keyword must be specified.

Cisco IOS is aware of the SNA controllers connected to the serial line by the specification of the SDLC station address using the **sdlc address** interface configuration command. The format of this command is

sdlc address *hexbyte*

The *hexbyte* variable is any valid SNA/SDLC station address in hexadecimal ranging from 1 to FE. If the SDLC interface supports a multidrop configuration, then a **sdlc address** command is required for defining each SDLC station address of the attached SDLC devices. If the SDLC interface connects to a FEP, the **sdlc address** value must match the NCP major node PU ADDR definition.

Both SRB and RSRB require a mapping definition that provides a virtual MAC address and a virtual ring number that identify the SDLC SNA controller attached to the serial interface. This enables the router to build an LLC2 frame for delivery to the LAN-attached FEP. The interface configuration command that enables this mapping is the **sdllc traddr** command and it has the following format:

sdllc traddr *xxxx.xxxx.xx00 lr bn tr*

The first parameter (*xxxx.xxxx.xx00*) of the **sdllc traddr** command is the virtual MAC address used to represent the SDLC-attached PU device. The value used must have the last two hexdigits as 00. These two digits are replaced by the router with the value defined from the **sdlc address** interface command. Together these values create a unique MAC address for mapping the SDLC-attached PU to the LLC2 protocol used for connecting to the IBM 3745 FEP.

The *lr* variable is the SDLLC virtual ring number to which the virtual MAC address attaches. The *tr* variable is the SDLLC target ring and is typically the virtual ring defined by the **source-bridge ring-group** global command. In direct connection topology, the target ring is the Token Ring number used that attaches the IBM 3745 FEP. The *bn* variable is the virtual bridge used to connect the SDLLC virtual ring to the target ring. In our examples for router R1 and R2, the target ring is the *ring-group* number defined on the **source-bridge ring-group** global configuration commands. In the examples for router R1 and R2, the same MAC address is being used. This does not violate Token Ring architecture because the RIF built by the router uses the unique virtual ring numbers of each router and therefore allows the IBM FEP to direct the frames to the appropriate router.

DLSw+ does not employ the **sdllc traddr** command. Instead, DLSw+ requires only a virtual MAC address that is to be used for the SDLC-attached device. This is specified using the **sdlc vmac** interface configuration command:

sdlc vmac *mac-address*

The **sdlc vmac** *mac-address* command value is the 12-digit hexadecimal value used as the MAC address of the SDLC-attached device. The last two digits must be 00. These digits are replaced by the SDLC station address to provide a unique MAC address for this interface. The virtual MAC address is presented to the LAN-attached FEP and should match the PATH parameter value used on the PU definition in VTAM's switch major node. Router R3 using local DLSw+ specifies a virtual MAC address of 4000.3174.10000, which is resolved to 4000.3174.10C1 by using the value assigned by the previous sdlc address command. Again, this becomes the same MAC address as that used by router R1 and R2. However, because the DLSw+ virtual ring is different than that defined on router R1 and R2, the FEP will be able to deliver the frame to router R3.

DLSw+ also requires that the SDLC device be specified to use DLSw+. This is accomplished using the sdlc dlsw interface configuration command. The format of this command is

sdlc dlsw *sdlc-address*

The *sdlc-address* variable is a previously defined sdlc address value used on this serial interface. Multiple *sdlc-address* values can be specified to map multiple SDLC devices to DLSw+ with the single command.

The specification of the partner SDLC device attached to the Token Ring is required to enable the connection to the IBM FEP. The partner device is specified using the **sdllc partner** command:

sdllc partner *mac-address sdlc-address*

The *mac-address* variable is the LAN MAC address of the IBM FEP or the Cisco CIP internal LAN adapter. In most installations, this is a Token Ring MAC address, but it can be an Ethernet MAC address. The *sdlc-address* variable must match an **sdlc address** value previously defined on this interface definition. There must be one **sdllc partner** command for each attached SDLC end station on the SDLC serial interface.

A final interface configuration command necessary to map the SDLC-attached device to LLC2 is the **sdllc xid** command. This command enables VTAM on the IBM host to receive the correct SNA XID request to the SNA device attached to the router. The VTAM XID request will interpret the four-byte XID response from the SNA PU as the PU IDBLK and IDNUM parameter values for the PU definition in VTAM's switch major node. The XID for SDLLC is defined using the sdllc xid interface command. The format of the command is

sdllc xid *address idblkidnum*

The *address* variable is the SDLC station address defined on a previous **sdlc address** interface command. The *idblkidnum* variable is the four-byte hexadecimal XID value. The first three hex-digits (*idblk*) must match the IDBLK value defined in VTAM. Likewise, the remaining five hex-digits (*idnum*) must match the IDNUM parameter of the VTAM PU switch major node. The *idblkidnum* variable value must be unique within the controlling VTAM definitions. Using a duplicate *idblkidnum* value will cause uncertain errors.

SNA PUs set their information frame size in their configuration. The SDLC-attached router can segment received information frames down to

the size accepted by the SDLC device. To set the largest information frame, use the following interface command:

sdlc sdlc-largest-frame *address size*

The *address* variable is a value previously defined on an **sdlc address** command used on this serial interface. The *size* variable defaults to 265. Typically, use 265 or 521 for older IBM 3274 type controllers. The newer IBM 3174 type controllers and SNA PU 2.0 gateways can use larger frame sizes of **521, 1033**, and **2057**. Router R3 in the sample configuration sets the largest frame size that can be delivered to the controller at 1033. A separate **sdlc sdlc-largest-frame** command is required for each SDLC device defined on this line if the default value is not acceptable.

Reverse SDLLC uses all the same SDLLC commands discussed above except that they are applied to the serial interface connecting the IBM FEP. Figure 22-12 diagrams this configuration.

The following router configuration applied to Figure 22-12:

R1 configuration applied to Figure 22-12:

```
source-bridge ring-group 1dlsw local-peer!interface tokenring 0 ring-speed 16
 source-bridge 3174 1 1 source-bridge spanning!
interface serial 0
 encapsulation sdlc sdlc role secondary
 sdlc address c1
 sdlc sdlc-largest-frame c1 1033 sdllc vmac 4000.3745.0000
 sdllc xid c1 01731741 sdllc partner 4000.3174.0001 c1 sdlc dlsw c1
```

In this configuration for Figure 22-12, the router serial interface has its SDLC link station role defined as secondary because the router is connecting the FEP, which is always primary when connecting downstream SNA PU T2.0 devices.

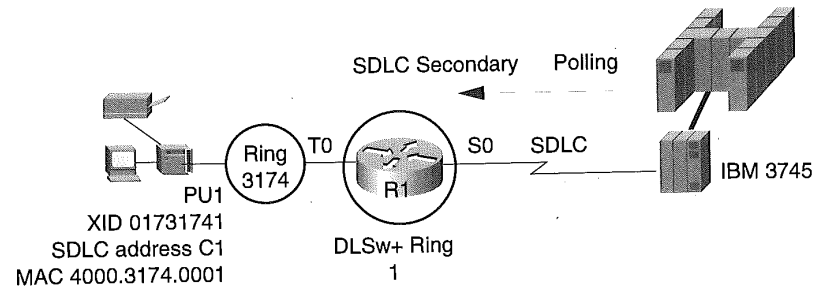


Figure 22-12 Reverse SDLLC example using local DLSw+.

In configuration scenarios where the remote SNA controllers are connected over a WAN, RSRB and DLSw+, along with FAM relay, can be used for delivering the SNA SDLC-attached data to the Token Ring-attached FEP. Figure 22-13 illustrates the use of RSRB and DLSw+ for connecting remote SDLC resources over a WAN using Cisco routers.

The following router configurations apply to Figure 22-13:

R1 configuration applied to Figure 22-13:

```
source-bridge ring-group 1
source-bridge remote-peer 1 tcp 10.1.1.1
source-bridge remote-peer 1 tcp 10.1.2.2 local-ack
!
interface tokenring 0
 source-bridge 3745 1 1
!
interface loopback 0
 ip address 10.1.1.1 255.255.255.0
!
interface serial 0
 ip address 10.254.1.1 255.255.255.0
```

R2 configuration:

```
source-bridge ring-group 1
source-bridge remote-peer 1 tcp 10.1.1.1 local-ack
source-bridge remote-peer 1 tcp 10.1.2.2source-bridge sdllc local-ack
!
interface loopback 0
 ip address 10.1.2.2 255.255.255.0
!
interface serial 0
 ip address 10.254.1.2 255.255.255.0
!
interface serial 1
 encapsulation sdlc-primary
 sdlc address c1
 sdllc traddr 4000.3174.1000 3174 1 1
 sdllc partner 4000.3745.0001 c1
 sdllc xid c1 01731741
```

R3 configuration applied to Figure 22-13:

```
source-bridge ring-group 2
dlsw local-peer peer-id 10.2.1.1
dlsw remote-peer 0 tcp 10.2.2.2 !
interface tokenring 0
 source-bridge 3745 1 2
!
interface loopback 0
 ip address 10.2.1.1 255.255.255.0
!
interface serial 0
 ip address 10.252.1.1 255.255.255.0
```

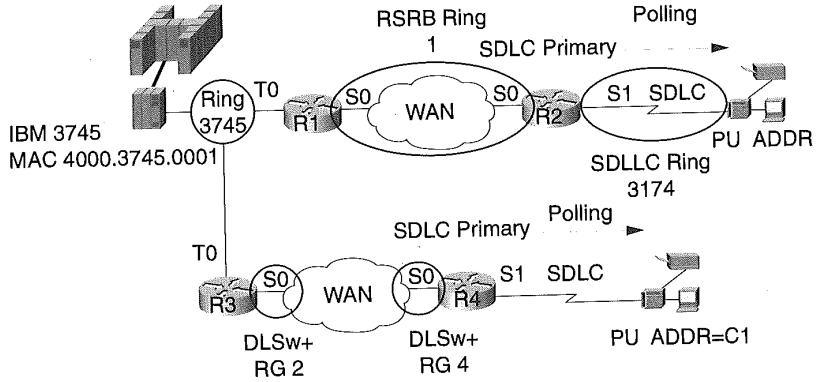


Figure 22-13 RSRB and DLSw+ for connecting SDLC devices over a WAN to a Token Ring-attached IBM FEP.

R4 configuration applied to Figure 22-13:

```
source-bridge ring-group 4
dlsw local-peer peer-id 10.2.2.2
dlsw remote-peer 0 tcp 10.2.1.1
!
interface loopback 0
 ip address 10.2.2.2 255.255.255.0
!
interface serial 0
 ip address 10.252.1.2 255.255.255.0
!
interface serial 1
 encapsulation sdlc sdlc role primary
 sdlc address c1
 sdlc vmac 4000.3174.1000
 sdllc partner 4000.3745.0001 c1
 sdllc xid c1 01731742 sdlc dlsw c1
```

In the router examples for Figure 22-13, local acknowledgement is being used. Specifying local acknowledgement for SDLLC while using RSRB as the transport mechanism requires the specification of the **source-bridge sdllc local-ack** global configuration command on the router connecting to the downstream SDLC-attached device. DLSw+ used for the R3 to R4 connection provides local acknowledgement automatically since it is part of the DLSw+ specification and does not require specific coding.

Frame Relay BNN and BAN Connectivity

Cisco routers support RFC 1490 FRF.3 for encapsulating SNA over Frame Relay. The standard identifies two methods of delivery: Boundary Network Node (BNN) and Boundary Access Node (BAN). The Cisco IOS software feature that enables these two methods is called Frame Relay Access Support (FRAS). Cisco IOS has an enhanced version of this standard called FRAS Host. This feature is used when the connection to the mainframe is a channel-attached Cisco router using a CIP instead of the IBM 3745 FEP. These features provide the conduit for communicating with Token Ring, Ethernet, or SDLC-attached SNA devices using Cisco routers over a Frame Relay network.

The BNN function can be defined as a static mapping or dynamic mapping of network addresses. Figure 22-14 diagrams the BNN static and dynamic network configuration used for examining FRAS for BNN. Router

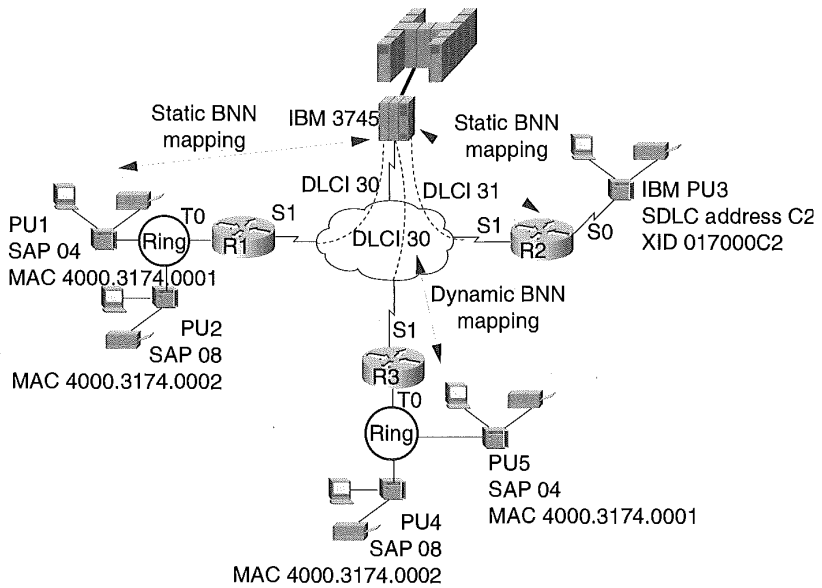


Figure 22-14 Static and dynamic BNN configuration for SNA over Frame Relay.

R1 configuration uses static BNN connectivity for Token Ring-attached IBM controllers over the Frame Relay network. Router R2 configuration illustrates the use of SDLC-connected SNA devices over Frame Relay using static BNN definitions. Router R3 illustrates the use of dynamic BNN for Token Ring-attached SNA devices.

The following router configurations apply to Figure 22-14:

R1 configuration applied to Figure 22-14:

```
interface serial 1
no ip address
encapsulation frame-relay IETF
frame-relay lmi-type ansi
frame-relay map llc2 30
!
interface tokenring 0
no ip address
ring-speed 16
fras map llc 4000.3174.0001 4 4 serial 1 frame-relay 30 4 4
fras map llc 4000.3174.0002 8 8 serial 1 frame-relay 30 8 8
```

R2 configuration applied to Figure 22-14:

```
interface serial 0
no ip address
encapsulation sdlc
no keepalive
clockrate 56000
sdlc address C2
sdlc xid C2 017000C2
sdlc role primary
fras map sdlc C2 serial 1 frame-relay 31 4 4
!
interface serial 1
no ip address
encapsulation frame-relay ietf
frame-relay lmi-type ansi
frame-relay map llc2 31
```

R3 configuration Applied to Figure 22-14:

```
interface Serial 1
no ip address
encapsulation frame-relay IETF
frame-relay lmi-type ansi
frame-relay map llc2 30
!
interface TokenRing 0
no ip address
ring-speed 16
fras map llc 4 Serial 1 frame-relay dlci 30 4
```

Frame Relay is not a reliable transport and is therefore without the capability to resequence, acknowledge, or provide flow control for time-sensitive SNA frames. The RFC 1490 FRF.3 SNA support, however, overcomes these requirements for delivering SNA using LLC2 as part of the encapsulation method. The Cisco router IOS software is made aware of this encapsulation technique with the following interface subcommand:

frame-relay-map llc2 *dlci-number*

The *dlci-number* parameter identifies that RFC 1490 FRF.3 encapsulation techniques are used for transporting SNA data on the specified DLCI to the Cisco IOS software. The DLCI number used is the DLCI number defined for the PVC by the Frame Relay switch connecting the router to the Frame Relay network. In the example configurations, DLCI 30 and 31 are used in router R1 and R2 respectively for connecting to the IBM FEP, while router R3 uses DLCI 30 also.

Support for LAN-attached SNA devices is specified using the **fras map llc** LAN interface command. The **fras map llc** command is specified on the LAN interface connecting the LAN-attached SNA devices. The format of the command is

fras map llc *mac-address lan-lsap lan-rsap* **serial** *port* **frame-relay** *dlci fr-lsap fr-rsap* [**pfid2** | **afid2** | **fid4**]

The **fras map llc** command sets up the MAC/SAP address pair mapping between the downstream SNA device and the upstream SAP address of the IBM SNA host computer. The *mac-address* parameter value is the Medium Access Control (MAC) 48-bit dotted-triple decimal address of the LAN-attached SNA device. The next parameter *lan-lsap* identifies the service access point (SAP) address used by the downstream SNA device being mapped. SNA devices use SAP addresses in multiples of four, starting with the value 04. The value is a hexadecimal number.

The *lan-rsap* parameter defines the destination SAP address of the IBM SNA host from the perspective of the downstream SNA device. Because the **fras map llc** command maps the downstream SNA device to the DLCI, which is mapped to the destination SAP, the *lan-rsap* value does not necessarily have to match the true destination SAP address. However, for clarity and ease of configuration, it is best practice to have the local and remote SAP addresses match.

The value for the *lan-rsap* parameter follows the same definition rules as those described for the *lan-lsap* parameter. The *port* parameter following

the **serial** keyword of the **fras map llc** command identifies which serial port on the router connects to the Frame Relay network. The *dlci* parameter of the **frame-relay** keyword identifies the DLCI number used for this mapping over the Frame Relay network. The *fr-lsap* parameter is the local SAP address used by the router for identifying the DLCI in the LLC2 frames. The *fr-rsap* parameter defines the actual SAP address of the SNA host found on the other end of the Frame Relay connection. In the router R1 example, the downstream SNA device PU1 connects to the IBM 3745 using SAP 04 over DLCI number 30.

The three remaining optional parameters of the **fras map llc** interface command reduce router processor overhead by indicating the only type of SNA format identifier (FID) to expect from the downstream SNA device. Entering a value of **pfid2** at the end of the command identifies traditional SNA PU Type 2 device connectivity. A value of **afid2** indicates that APPN Node Type 2.1 FID2 transmission headers are in use and a value of **fid4** indicates that a SNA subarea connection is being used over this Frame Relay configuration.

Static BNN support for serial SDLC-attached devices requires the use of the **fras map sdlc** command on the serial interface connecting the SDLC-attached SNA device. The format of the **fras map sdlc** command is

fras map sdlc *sdlc-address* **serial** *port* **frame-relay** *dlci fr-lsap fr-rsap* [**pfid2** | **afid2** | **fid4**]

The key in mapping the serial-attached SDLC SNA device is the SNA SDLC station address in the command. The *sdlc-address* parameter defines the actual SNA SDLC station address defined on the SNA device. PU3 in the example configuration shown in Figure 22-14 uses C2 as the SDLC station address and it is this value that is used in the subsequent **fras map sdlc** interface command. This value must be unique for all serial-attached SDLC SNA-attached devices using the same DLCI on this router. The remaining parameter values on the **fras map sdlc** command are entered by the same rules found on the **fras map llc** command.

Router R3 is defined in the router configuration to use dynamic BNN connectivity. Dynamic BNN negates the requirement to code the MAC address and the destination SAP address for the LLC2 frame for LAN-attached SNA devices because dynamic BNN learns the mapping of MAC/SAP pairs through analyzing the frames as they enter the router LAN interface port. The format for using dynamic BNN **fras map llc** is

fras map llc *lan-lsap* **serial** *interface* **frame-relay dlci** *dlci fr-rsap*

Dynamic BNN's only *lan-lsap* parameter is defined to identify the SAP address used for communicating to the downstream SNA device. Also note that in this command the *fr-lsap* parameter used in the static BNN definition is not required. However, the DLCI number and the destination SAP address on the opposite Frame Relay connections are required. These values are defined by the positional parameters *dlci* and *fr-rsap* respectively. All the parameters in the dynamic BNN **fras map llc** command are defined using the same rules as found on the static BNN **fras map llc** command. The router R3 configuration specifies that SAP 4 is to be used for the LAN-attached SNA device as well as the upstream connection to the SNA host computer on Frame Relay DLCI 30.

FRAS BAN connectivity is different from that of FRAS BNN connectivity in that BAN includes the MAC address within the frame. Figure 22-15 illustrates the use of BAN with Token Ring and SDLC-attached SNA devices.

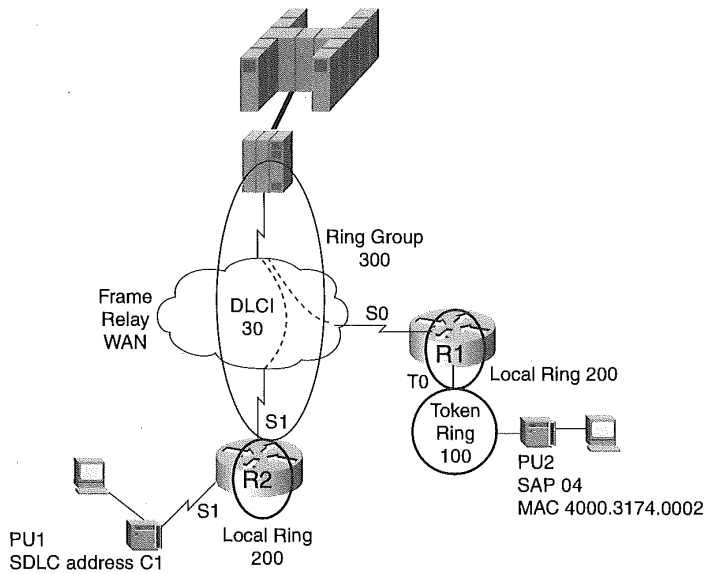


Figure 22-15 Using FRAS BAN for Token Ring and SDLC-attached SNA devices.

The following router configurations apply to Figure 22-15:

R1 configurations for Figure 22-15:

```
source-bridge ring-group 300
!
interface serial1
 encapsulation frame-relay ietf
 frame-relay lmi-type ansi
 frame-relay map llc2 30 fras ban 200 1 300 4000.3745.0001 dlci 30
!
interface tokenring 0
 source-bridge 100 1 300
```

R2 configuration applied to Figure 22-15:

```
source-bridge ring-group 300
!
interface serial 0
encapsulation frame-relay ietf
 frame-relay lmi-type ansi
 frame-relay map llc2 30
 fras ban 200 1 300 4000.3745.0001 dlci 30
!
interface Serial 1
no ip address
 encapsulation sdlc
 clockrate 19200
 sdlc role primary
 sdlc vmac 4000.0000.C300
 sdlc address C3
 sdllc xid C3 017000C3
sdllc partner 4000.3745.0001 C3 fras ban frame-relay Serial 0
4000.3745.0001 dlci 30
```

Boundary Access Node (BAN) support for Cisco FRAS requires the coding of the **source-bridge ring-group** global command prior to defining the **fras ban** interface command. Using BAN requires the definitions of the Token Ring logical connectivity along with a logical MAC address representing the DLCI used for the transport of the SNA frames. The format of the **fras ban** interface configuration command is

fras ban *local-ring bridge-number ring-group ban-dlci-mac* dlci *dlci#1[dlci#2-dlci#5]* [bni *mac-addr*]

This format of the **frs ban** interface command is applied to the serial interface connecting the Frame Relay network. The *local-ring* parameter is used to define the actual ring number assigned to the LAN interface being used for connecting the downstream LAN-attached SNA devices. The *local-ring* value ranges from 1 through 4,095 in decimal format.

The *bridge-number* parameter is the number of the logical bridge connecting the physical LAN Token Ring to the logical virtual ring defined by the *ring-group* parameter. The *ring-group* parameter must match the virtual ring number defined on the **source-bridge ring-group** global command used for this connection. The valid range for values used on the *local-ring* parameter are 1 through 15 in decimal. The *ring-group* value is a valid decimal number from 1 to 4,095, matching **the source-bridge ring-group** value.

The *ban-dlci-mac* parameter represents the BAN PVC MAC address used in the frame sent to the receiving BAN device. This value is coded in accordance with the 48-bit triple-dotted decimal address as noted in the discussion on defining static BNN.

The *dlci* parameter following the **dlci** keyword specifies one or more DLCI numbers used for the BAN connection. Each DLCI number must be unique and range between 16 and 1,007. Defining multiple DLCI numbers, if available, up to a maximum of five allows the router to perform load balancing over the DLCIs for connectivity to the destination SNA device.

The final optional parameter **bni** is used to define the MAC address used by the IBM SNA host for connecting to the network. The Boundary Node Identifier (BNI) value uses the 48-bit triple-dotted decimal representation of the MAC address. Using the BNI value allows the IBM 3745 NCP or AS/400 to be configured without the need for the BAN DLCI MAC address. With this in mind, it is good practice to have the *ban-dlci-mac* value equal to the *bni-mac-addr* values.

In the example connecting the Token Ring for router R1 to the FEP shown in Figure 22-15, the Frame Relay connection is defined as being connected to ring number 200. We have defined bridge 1 as the connection from ring 200 to the ring group's ring 300. The DLCI MAC address is defined to equal the actual IBM 3745 MAC address and the connection uses Frame Relay DLCI number 30 to make the connection.

SDLC FRAS BAN connectivity is accomplished by specifying the fras ban command on the SDLC serial interface connecting the SNA devices. The format for specifying SDLC-connected SNA devices over Frame Relay using BAN is

fras ban frame-relay *port fr-mac* **dlci** *dlc#*

The *port* variable is the serial interface of the Frame Relay connection used in the mapping. The *fr-mac* variable is the MAC address used as the destination MAC and the *dlci#* variable value is the DLCI number onto

which this SDLC frame is mapped over the specified frame relay port. In the example for router R2 in Figure 22.15, the SDLC interface maps the SDLC frame information to the serial interface 0 for the Frame Relay connection. The DLCI used in the mapping for delivering the SNA data is DLCI 30.

The FRAS host connectivity feature enables the Cisco router to be LAN-attached to an AS/400 or channel-attached to an IBM mainframe serving the functions of the FEP. In Figure 22-16, a Cisco router is using a CIP to channel-connect to the IBM mainframe. Routers R2 and R3 connect using a Frame Relay connection. Router R2 uses dynamic BNN to connect the Token Ring-attached SNA devices, while Router R3 uses BAN to connect them. The channel-attached router uses a virtual Token Ring interface definition to specify the Frame Relay connection to the mainframe.

The following router configurations apply to Figure 22-16:

R1 configuration applied to Figure 22-16:

```
source-bridge ring-group 300
!
Interface Serial 1
 encapsulation frame-relay IETF
 frame-relay map llc2 30
 frame-relay map llc2 31
!
interface Channel 6/0
 no ip address csna E210 02
!
interface Channel 6/2
 no ip address
 lan TokenRing 0
 source-bridge 6 1 300
 adapter 0 4000.7513.4200
!
interface Virtual-TokenRing0
 source-bridge 100 1 300
 source-bridge spanning
  fras-host bnn Serial 1 fr-lsap 04 vmac 4000.7513.0000 hmac
4000.7513.4200
 fras-host ban Serial 1 hmac 4000.7513.4200
```

R2 configuration applied to Figure 22-16:

```
interface Serial 1
 encapsulation frame-relay IETF
 frame-relay map llc2 30
!
interface TokenRing0
 fras map llc 4 Serial 1 frame-relay 30 4
```

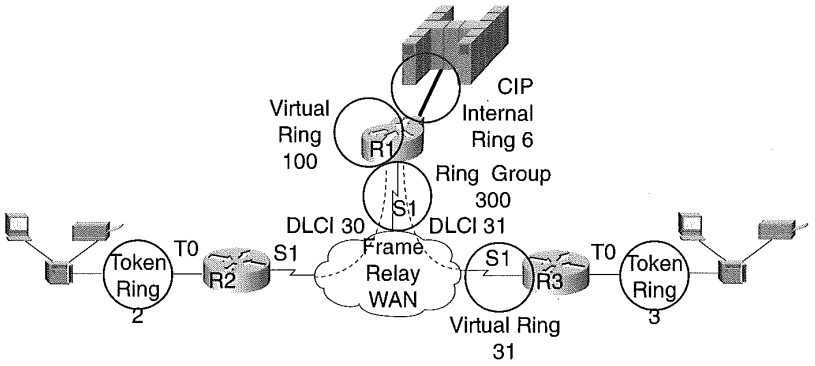


Figure 22-16
FRAS Host connectivity to an IBM mainframe.

R3 configuration applied to Figure 22-16:

```
source-bridge ring-group 300
!
interface Serial 1
 encapsulation frame-relay IETF
 frame-relay map llc2 31
 fras ban 31 1 300 4000.7513.4200 dlci 31
!
interface TokenRing0
 source-bridge 3 1 300
```

The FRAS host feature enables Cisco routers to be the Frame Relay connection point to the channel-attached IBM SNA host. Both BNN and BAN techniques are available for use by the Cisco host-attached router. The Cisco router attached to the host computer can use either LLC2 passthru or LLC2 local termination functions to enable LLC2 transport.

The LLC2 passthru functions of the FRAS host feature are most useful when combined with the Cisco router attached to the IBM SNA mainframe using a Channel Interface Processor (CIP) board for connecting to the mainframe. LLC2 passthru requires the use of the following interface command:

interface virtual-tokenring *number*

The **interface virtual-tokenring** command along with the **fras-host bnn** or **fras-host ban** interface commands enables the use of LLC2 passthru mode. This virtual ring is used as the connection point for transporting LLC2 traffic over the Frame Relay connection between the SNA host and the downstream SNA device. The *number* parameter of the **interface virtual-tokenring** command identifies the virtual port associated with the **fras-host** definitions. Under the virtual ring definition, the **source-bridge** command is also defined to associate the virtual ring number of the virtual-tokenring to the global *ring-group* number.

In the router R1 configuration example, ring 100 is being identified with the virtual-tokenring interface and is being bridged to ring group 300. This configuration enables LLC2 passthru mode between the mainframe and the end SNA devices.

The LLC2 local termination function of the FRAS host feature is possible using the following configuration commands:

dlsw local-peer

fras-host dlsw-local-ack

The first command **dlsw local-peer** enables the router to use data link switching as it indicates local delivery of LLC2-type traffic through the virtual-tokenring interface defined on the router. The **dlsw local-peer** command is a global command and is not specific to any particular interface. Couple the **dlsw local-peer** command with the **fras-host dlsw local-ack** defined on the virtual-tokenring interface and the LLC2 local termination function is enabled for the router.

LLC2 local termination is most appropriate for connectivity to a LAN-attached SNA host. The SNA host can be directly attached to the LAN or it can be channel-attached to an IBM 3745 FEP. TO employ LLC2 local termination in the example, the **dlsw local-peer** command is added to the global configuration for router R1 and the **fras-host dlsw local-ack** command is added to the virtual tokenring0 interface configuration.

To configure a router to use the FRAS Host BNN feature, the following command must be issued on the virtual-tokenring interface:

fras-host bnn *(sub)interface* **fr-lsap** *sap* **vmac** *virt-mac* **hmac** *hmac* [**hsap** *hsap*]

The **fras-host bnn** command enables the service on the interface defined by the value set for the positional parameter *(sub)interface*. The *(sub)interface* references the Frame Relay serial interface port number. The *sap* parameter following the **fr-lsap** keyword defines the service access point address used by the LLC2 as the destination SAP address for inbound BNN frames on the Frame Relay connection. A virtual MAC address is used along with the DLCI number to create a unique MAC address for the BNN connection. The *virt-mac* value is a full 48-bit dotted-triple decimal value (six full bytes) whose last two bytes must be zero. These last two bytes are replaced with the DLCI number associated with the BNN connection.

The *hmac* value following the **hmac** keyword is the actual MAC address defined on the Cisco router CIP internal adapter or the MAC address of the AS/400 or IBM 3745 LAN interfaces. The final optional parameter *hsap* indicates the host SAP address used for connecting to the SNA host. Not coding the *hsap* value and keyword defaults this SAP address to the one used for the **fr-lsap** keyword. In the example for router R1, the serial interface 1 is used to map the Frame Relay LSAP 04 to a virtual MAC address of 4000.7513.0000 with a destination MAC address of 4000.3745.4200. The *hmac* value here maps to the value given as the adapter address on the CIP internal-tokenring definition.

The BAN configuration for the FRAS host function is simpler in form. The format for the **fras-host ban** command is

fras-host ban *(sub)interface* **hmac** *hmac* [**bni** *bni-mac*]

Again the command identifies the interface used for connecting the virtual-tokenring to the Frame Relay network using the *(sub)interface* positional parameter. The *hmac* value is the MAC address of the destination MAC assigned to the adapter of the CIP internal-tokenring interface. The *bni-mac* value is the Boundary Node Identifier (BNI) MAC address of the Frame Relay connection to the host computer. The router R1 configuration for Figure 22-16 identifies BAN support for the Frame Relay serial interface. The MAC address of the SNA host computer is 4000.3745.4200 and matches the adapter address of the internal-tokenring interface definition for the CIP.

APPN Support on Cisco Routers

The implementation of APPN with Cisco IOS software requires an acknowledgement of the definitions prior to committing them to the APPN subsystem. Specifying this commitment is accomplished by issuing the **complete** configuration command under every APPN definition.

The update of an APPN definition or the deletion of an APPN definition is not performed until the **complete** command is entered. When modifying a definition, you must perform the following tasks in order:

1. Enter the definition as it currently exists.
2. Enter the **no complete** command.
3. Change any of the parameters of the definition desired.
4. When completed with the changes, enter the **complete** command to commit the new definitions to the APPN subsystem.

At this point, the definition will take effect by the APPN subsystem applying the changes to the configuration.

Four elements are needed to enable APPN on Cisco routers:

- APPN routing
- A defined control point
- A specified APPN port
- Linkstation

The Cisco IOS software becomes aware of APPN functionality by entering the **appn routing** global command. The router on startup will execute the APPN subsystem once reading the **appn routing** command. The APPN subsystem can also be started and stopped manually by entering the privileged execute commands **appn start** and **appn stop**. Specifying APPN routing automatically enables ISR functionality. HPR functions must be defined on the APPN control point definitions in order for the router to use HPR as the preferred APPN routing protocol.

APPN control point definitions identify the process of control for the APPN network node defined on this router. An APPN control point must be defined to use APPN. The **appn control-point** command defines the fully qualified control point name for the node. There is only one control point definition per router. The format of the **appn control-point** command is

appn control-point *netid.cpname*

The *netid.cpname* identifies which network this node is a member of and the network-wide unique control point name assigned to this router. The *netid* variable must match the *netid* variable of any other network node connecting to this router. If the *netid* does not match, CP-CP sessions will not be established. Both the *netid* and *cpname* variables are one to eight alphanumeric characters in length, separated by a period. The variable name cannot begin with a numeric value. In the following example, a control point name of NODE1 is defined as part of the network NETX.

```
appn control-point NETX.NODE1

complete
```

The fully qualified control point name is not case-sensitive. It is in upper case here for display purposes only. The **complete** command is shown in the example to illustrate that the **appn control-point** definition is not committed unless the complete command is entered.

HPR can now be defined as the routing protocol in use for APPN instead of the default ISR. Specifying HPR is simply done by entering the following commands under configuration mode:

appn control-point NETX.NODE1

no complete

appn control-point NETX.NODE1

hpr

complete

The APPN port definitions associate the APPN subsystem to an interface on the router. Any interface used for establishing connectivity to an APPN network requires a port definition. The **appn port** command associates a router interface to APPN for the control point. The format of the **appn port** command is

appn port *portname interface*

The *portname* variable is an assigned label for the port being defined. The *portname* parameter is usually eight alphanumeric characters long and identifies the type of connection, destination of connection, and the interface used for APPN connectivity. The interface types available to APPN ports are

- ATM
- Frame Relay
- SDLC
- PPP
- SMDS
- X25
- ISDN (via PPP)

The discussion in this section focuses on the most widely used interface types, which are ATM, Frame Relay, and SDLC for APPN over Cisco routers.

Remote SRB (RSRB) can also be used for transporting APPN over various media types. When employing RSRB as the APPN, the transport mechanism requires the **rsrb** keyword on the **appn port** command. The format of the **appn port rsrb** command is

appn port *portname* rsrb

The *portname* variable is defined as previous discussed.

RSRB requires a virtual MAC address and an associated virtual ring and bridge must be defined for connectivity to the ring group used by the RSRB connections. The format of the command defining the virtual RSRB station and ring is

rsrb-virtual-station *mac-address local-ring bridge-number target-ring*

The *mac-address parameter* value of the **rsrb-virtual-station** command is a triple-dotted hexadecimal 12-digit representation of the MAC address of the virtual station used by the APPN port. The *local-ring* parameter is the virtual ring number of the APPN station associated with the port. The *bridge-number* identifies the virtual bridge connecting the virtual token ring station to the ring group identified by the *target-ring* parameter. The *target-ring* parameter value must match a *ring-group* virtual ring number defined on a **source-bridge ring-group** global command used by this router. For example, if the following command were entered under the port configuration mode

```
source-bridge ring-group 1
source-bridge remote-peer 1 tcp 10.1.1.1
appn control-point node1
complete
appn port rsrbring rsrb
rsrb-virtual-station 4000.cdef.0001 2 1 1
complete
```

the virtual RSRB ring 2 bridges to ring group 1 over bridge 1. The APPN traffic will be encapsulated in TCP segments and sent to the peer at 10.1.1.1.

DLSw+ can also be used for transporting APPN over various media types. APPN transport over DLSw+ requires the **vdlc** keyword at the end of the **appn port** command. The **vdlc** keyword indicates the use of the Cisco Virtual Data Link Control (VDLC) internal protocol for connecting DLSw+ to an APPN port. The format of the appn port for implementing DLSw+ is

appn port *portname* **vdlc**

Assigning the port to VDLC for DLSw+ again requires the mapping of the port to the **source-bridge ring-group** used by DLSw+. The format of the command that associates the port the ring-group for DLSw+ use is

vdlc *ring-group* [**vmac** *vdlc-mac-address*]

The *ring-group* variable must match a ring group used by DLSw+ on this router. The optional **vmac** keyword and its associated variable *vdlc-mac-address* assigns a virtual MAC address to identify the APPN port being defined. For example, if the configuration included

```
source-bridge ring-group 1
dlsw local-peer peer-id 10.1.1.1
dlsw remote-peer 0 tcp 10.1.2.1
appn control-point node1
complete
appn port DLSW vdlc
vdlc 1 vmac 4000.abcd.0001
complete
```

a virtual MAC address of 4000.abcd.0001 is used by the port named DLSW for connecting to ring group 1.

If the APPN connection is over an SDLC interface, then the **appn port** command requires the assignment of a secondary SDLC station address to the port. The format of the command is

sdlc-sec-addr *address*

The *address* variable of the **sdlc-sec-addr** command is a hexadecimal value ranging from 00 to FE. The default value used is 00. As an example, the following statements define a connection through an SDLC interface on port 2 of the Cisco router:

```
interface serial 2
 encapsulation sdlc
 sdlc address C1
appn control-point node1
 complete
appn port SDLC serial 2
 sdlc-sec-addr C1
 complete
```

Each port definition has a minimum of one link station defined for use on the port. Cisco APPN NN allows for a dynamic link station definition based on the XID3 exchange between the NN and the EN connecting or other NNs. The **appn port** command supports dynamic link station definitions for PU2.1 devices using the **service-any** command entered under the port configuration mode. The **service-any** command is the default for the port and allows the NN to attempt to build a dynamic link station definition for

the incoming connection request. Specifying the **no service-any** command under port configuration mode disables dynamic link-station definitions, thereby requiring a predefined link station for using the port.

For dynamic link station definitions concerning PU 2.0 devices, the **null-xid-poll** command must be entered under the port configuration definition. The router issues a null XID over the port in an attempt to solicit an XID0 or XID3 response. Specifying the **null-xid-poll** command works for both PU 2.0 and PU 2.1 devices. The **null-xid-poll** command indicates to the router that it expects the partner PU 2.0 to issue an XID0 response upon which the NN function builds a link station. PU 2.1 XID3 negotiation and exchange continues to function as normal. The default is XID3 polling.

Configuring the **null-xid-poll** command for two Cisco APPN NN routers connecting over a port results in a failed connection. Each router will expect the other to send an XID0 or XID3 response. Similar issues arise if **null-xid-poll** is defined on a port connecting to an IBM 3745 FEP configured for XID polling. The **null-xid-poll** command should be used on ports connecting PU 2.0 devices that do not support XID3 polls.

APPN link stations represent a connection or the capability to have a connection to another APPN node. Cisco IOS enables complete dynamic link station definitions if the partner node initiates the connection. Cisco APPN network nodes will initiate connections if a link station definition exists, identifying a partner node for the link. The most frequently used link station commands are discussed here.

The **appn link-station** global command may also define link characteristics for the APPN connection, regardless of which node is initiated. The format of the **appn link-station** global command is

appn link-station *linkname*

The *linkname* variable is a one- to eight-alphanumeric character string identifying the link established between two PU 2.1 or PU 2.0 devices over the associated port. The name cannot begin with a number. An APPN port is associated with a link station using the **port** command under **appn link-station** configuration mode. The format of the port command is

port *port-name*

The **port** command under link station configuration mode maps the access to the predefined link station over a specific previously defined APPN port. The *port-name* variable must match the name assigned to an **appn port** definition. For example, connection to another NN over a serial line is defined as

```
appn port node1 serial 1
complete
appn link-station tonode2
port node1
complete
```

In the example interface, serial 1 on the router has an HDLC line. The port name identified as node1 maps to the serial interface on the router that connects the two locations. The specified link station defines the link station name as tonode2 and maps the link station to the port named node1.

The initiation of a link requires the destination address of the link station being connected on the associated port. The address assigned is also used from mapping the incoming connection to the link station matching the address. The media types discussed for our purposes are

- ATM
- Frame Relay
- Token Ring
- Ethernet
- SDLC

The format of the ATM link station destination address is

atm-dest-address *pvcid*

The **atm-dest-address** command under **appn link-station** configuration mode identifies which interface provides the ATM PVC information being mapped for this link station partner. The *pvcid* variable must match an ATM interface PVC previously defined. The *pvcid* value is the virtual circuit descriptor value of the previously defined interface command **atm pvc,** found under the ATM interface. For example,

```
interface ATM 1/0
 atm pvc 250 1 2 aal5nlpid
 map-group atm-appn
!
appn control-point netx.node1
 complete
!
appn port ATM ATM 1/0
 complete
!
appn link-station ATMtoSNA
 port ATM
 atm-dest-address 250
 complete
```

The virtual circuit descriptor of the PVC used for the ATM connection is 250 in the above example.

For Frame Relay interface support when defining the link station, the following command defines the associated destination address:

fr-dest-address *dlci [sap]*

The *dlci* variable is the DLCI number used by the partner router attached to the Frame Relay network. The value used for the *dlci* variable should match a previously defined Frame Relay definition. The *sap* variable is optional and defines the service access point (SAP) address used for identifying the link station being defined when multiple link stations are connected over the same Frame Relay DLCI. The default *sap* value is hexadecimal 04 but can be coded in the range of 04 to EC as long as the number is divisible by four. The following example illustrates the use of the **fr-dest-address** command under the **appn link-station** command:

```
interface serial 0
 encapsulation frame-relay IETF
 frame-relay map llc2 16
!
appn control-point netx.node1
 complete
!
appn port framerly serial 0
 complete
!
appn link-station node1
 port framerly
 fr-dest-address 16
 complete
```

In this example, the APPN link station node1 is reached using the destination address 16, which maps the DLCI of the **frame-relay map** command found under router interface serial 0.

For Token Ring and Ethernet link station definitions, the **lan-dest-address** command under **appn link-station** configuration mode defines the MAC address and optionally the SAP address of the LAN-attached destination partner. The format of the command is

lan-dest-address *mac-addr [sap]*

The *mac-addr* variable defines the 12-digit triple-dotted hexadecimal value of the MAC address assigned to the destination partner LAN interface. The optional *sap* variable defaults to 04 and allows for multiplexing connections over the link station using SAP addresses. The *sap* value must be within the hexadecimal range of 04 to EC and be divisible by four. The

following example defines the destination partner attached to Token Ring interface 0 that has a MAC address of 4000.0000.0001 and uses the default SAP address of 04.

```
interface tokenring 0
!
appn control-point netx.node2
 complete
!
appn port tr0 tokenring 0
 complete
!
appn link-station node2
 port tr0
 lan-dest-address 4000.0000.0001
 complete
```

SDLC-attached nodes over serial interfaces use the SDLC station address for establishing connectivity. The value of the destination partner is defined using the **sdlc-dest-address** command under **appn link-station** configuration mode. The format for defining the destination SDLC station address is

sdlc-dest-address *address*

The value of the *address* variable is a two-digit hexadecimal number in the range of 00 to FE. The value used here must match the SDLC address value defined on the serial interface of the partner node. The following illustrates the use of the **sdlc-dest-address** command:

Configuration for router R1:

```
interface serial 0
 encapsulation sdlc
 sdlc address c2
!
appn control-point netx.r1
 complete
!
appn port sdlc serial 0
 sdlc-sec-addr c2
 complete
!
appn link-station r1
 port sdlc
 sdlc-dest-address c1
 complete
```

Configuration for router R2:

```
interface serial 1
 encapsulation sdlc
 sdlc address c1
!
```

```
appn control-point netx.r2
 complete
!
appn port sdlc serial 1
 sdlc-sec-addr c2
 complete
```

In the above example, the SDLC station address defined on interface serial 1 of the R2 router matched the SDLC destination station address defined on the R1 router **sdlc-dest-address appn link-station** command.

SNA Support Using a Channel-Attached Cisco Router

The Cisco Channel Interface Processor (CIP) and the Channel Port Adapter (CPA) enable direct channel connection to the IBM mainframe. The CIP is available on Cisco 7000 and 7500 router platforms, while the CPA is available only on the Cisco 7200 router platform. For purposes of this text, we will refer only to the CIP for connecting to the mainframe, yet all features, functions, and commands used for connecting resources to the mainframe through a Cisco router using the CPA are supported and specified as discussed for the CIP. Direct connection of a channel-attached Cisco router to the mainframe enables a Cisco router to perform functions previously only available to IBM 3745 FEPs and IBM 3172 interconnect controllers. It also enables direct IP connectivity to an IP stack executing on the mainframe.

In support of connecting SNA to the mainframe, the CIP or CPA must be assigned and connected to a mainframe I/O channel. All SNA services are available using Cisco SNA (CSNA) connectivity. The CSNA services are enabled using the following channel interface command:

csna *path device* [**maxpiu** *value*] [**time-delay** *value*] [**length-delay** *value*]

The **csna** *path* parameter is subdivided into three arguments:

- logical path
- control unit address
- channel logical address

The logical path's two hexadecimal digits constitute the address of the physical connection from the mainframe point of view: one hexadecimal

digit for the control unit address and one for the device address. PCA channel connections require the path value to be 0100. The ESCON logical channel argument of the csna *path* parameter must match the ESCON PATH input port value, as defined in the mainframe IOCP generation. When using ESCON directors, this is the port connecting to the mainframe. Without them (direct ESCON attachment to the mainframe), the logical channel argument is 01.

The IOCP/HCD macro definition CHPID points to the logical partition by a named value. The IOCP/HCD RESOURCE macro defining the named partition specifies the LPAR number of the partition. If the channel connection is among the different LPARs on the mainframe, the specification of the LPAR number attaching to the csna definition through a ESCON director is required. If an ESCON director is not used, the path value is 01.

The control-unit-address argument of the csna *path* parameter is a single hexadecimal digit that must match the IOCP/HCD CUADD parameter of the CNTLUNIT macro. If the CUADD parameter is not coded in the IOCP/HCD CNTLUNIT macro for this channel, the default value of 0 is used for the control-unit-address argument of the csna path parameter.

The **csna** *device* parameter represents the position of the CIP interface as a device attached to the channel. The value here is from the UNITADD parameter of the CTLUNIT macro in the IOCP/HCD definitions. If given the IOCP IODEVICE address of EB2 and the UNITADD parameter of the corresponding CTLUNIT macro specified as EB0, the UNITADD IOCP parameter begins at 00. The **csna** interface command would be defined as

```
csna E220 02
```

where the ESCON port address connecting to the mainframe is E2. The LPAR using this connection is LPAR number 2 and the CUADD value defaults to 0. The CIP is connected as the third device on the channel (counting from 0) since the device parameter is 02.

Suppose the IODEVICE address is 97A. The IODEVICE ADDRESS parameter specifies a beginning IO address of 920 for 64 addresses (IODEVICE ADDRESS = 0920,64). The corresponding UNITADD parameter begins at 20 for 64 devices (UNITADD = 20,64). The **csna** *device* parameter is the difference between 7A and 20, which equals 5A. The corresponding **csna** command would be coded as

```
csna E220 5A
```

The optional **maxpiu** keyword value denotes the largest packet size in bytes that can be placed on the channel being defined. This can be thought of as equivalent to SNA MAXDATA and IP MTU size. The value defaults to 20470 if not coded and ranges from 4,096 to 65,535 bytes.

The optional **time-delay** keyword value is in milliseconds and specifies the delay used prior to transmitting a received packet on the interface. The default is 10 milliseconds with a range of 0 to 100 milliseconds.

The optional **length-delay** keyword value is the number of bytes to buffer before placing the data on the interface for transmission. The default is 20470 and the valid range is 0 to 65,535.

SNA communications over the CIP require an internal virtual LAN definition. This internal virtual LAN is defined as a LAN under the virtual x/2 subinterface of the CIP. In the following example, the CIP is installed in a 7513 router in slot 4 and the channel is connected using port 0 of the CIP. The virtual interface of the CIP is port 2 of the same slot 4. The virtual LAN tokenring-interface is defined under the 4/2 definition:

```
source-bridge ring-group 4
!
interface Channel 4/0
no ip address
csna E220 02
!
interface Channel4/2
 no ip address
 no keepalive
 LAN Tokenring 0
 source-bridge 6 1 4
 adapter 0 4000.0000.0001
```

Using the table, the CIP virtual interface channel 4/2 is used for defining the internal virtual LAN. CSNA does not require an IP address for connectivity to the mainframe and therefore it is not defined. This is denoted by the command no ip address. Since this is a virtual interface and is considered up and active only when the physical interface Channel 4/0 is up, there is no need for keepalive messages. We denote this using the no keepalive command.

The internal virtual LAN interface is defined using the **LAN** command, which can specify FDDI, Ethernet, and Tokenring. Tokenring is currently the only supported internal LAN at this time. The value 0 following the LAN Tokenring parameter indicates to the CIP that this is virtual LAN interface 0. The number of LAN interfaces can range from 0 to 31. The virtual Channel 4/2 interface can have multiple virtual LAN interfaces assigned to it.

The **source-bridge** statement following the LAN Tokenring statement defines the connection between this LAN virtual ring and the WAN virtual ring. The WAN virtual ring is the value defined on the **source-bridge ring-group** global command, in this case ring-group 4. The **source-bridge** on LAN Tokenring0 defines connectivity between the internal LAN virtual ring segment (ring 6) and the WAN virtual ring segment (4) through bridge 1.

The last statement required for CSNA connectivity is the **adapter** statement. It identifies the relative adapter number (RAN) and the MAC address assigned to the RAN for use on the internal virtual LAN segment. The RAN value ranges from 0 to 17 and must match the VTAM XCA major node parameter ADAPNO. The CIP allows multiple virtual adapters defined to the internal LAN virtual Token Ring segment.

The internal LAN virtual MAC address as defined in the example is 4000.0000.0001 on adapter 0. This MAC address will be the destination MAC address for devices connecting to the mainframe using SNA.

In Figure 22-17, PU1 is Token Ring-connected to a Cisco router at a remote location. Router R1 is the CIP router and is connected to the WAN. Router R2 utilizes DLSw+ for transporting the SNA data between the mainframe and PU1.

The following router configurations are applied to Figure 22-17:

R1 Configuration for Figure 22-17:

```
source-bridge ring-group 4dlsw local-peer peer-id 10.1.1.1dlsw remote-
peer 0 tcp 10.1.2.1!interface loopback 0 ip address 10.1.1.1
255.255.255.0!interface serial 0 ip address 10.254.1.1
255.255.255.0!interface Channel 4/0 no ip address csna E500 00!inter-
face Channel 4/2 no ip address lan tokenring 0 source-bridge 390 1 4
adapter 0 4000.0000.0001
!
router eigrp 10
 network 10.0.0.0
```

R2 configuration applied to Figure 22-17:

```
source-bridge ring-group 4dlsw local-peer peer-id 10.1.2.1dlsw remote-
peer 0 tcp 10.1.1.1!interface tokenring 0
source-bridge 2 1 4!interface loopback 0 ip address 10.1.2..1
255.255.255.0!interface serial 0
ip address 10.254.1.2 255.255.255.0!
router eigrp 10
 network 10.0.0.0
```

In the sample configuration above, the CIP router R1 is channel-connected to the mainframe using the ESCON port E5. The unit device being represented by the channel interface on the router is the first device on the

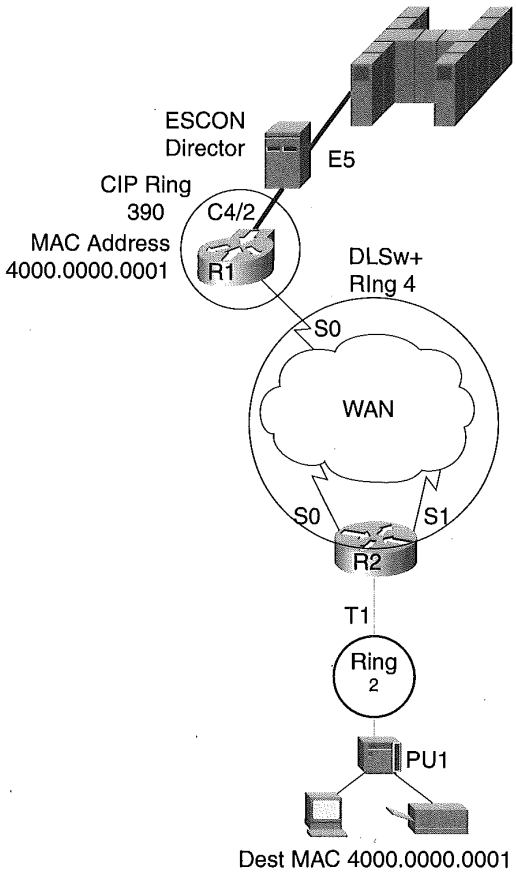


Figure 22-17
Remote Token Ring SNA connectivity using DLSw+ for transporting the SNA data between the mainframe and PU1.

defined string denoted by the 00 device number value. The PU1 contacts the mainframe by sending an explorer frame for the MAC address 4000.0000.0001, which is the MAC address assigned to the internal token-ring0 adapter 0 of the CIP.

The Cisco CIP/CPA interfaces to channels connecting to a mainframe can also take advantage of a high-performance channel protocol called Multi-Path Channel (MPC). This protocol uses two channels to communicate CAN to the mainframe from the CIP/CPA. One channel is used for reading data from the mainframe and the second is for writing data to the mainframe. This is in contrast to CSNA where a single channel is used for both reading and writing. The MPC subchannel address pairs can be defined on separate physical CIP interface ports of the same CIP or on the same physical port. MPC is specific to SNA/APPN connectivity to the mainframe. Specifying Cisco MPC (CMPC) is done by including the cmpc command under the physical channel definition of the CIP/CPA. The format of the cmpc command is

cmpc *path device tg-name* {**read** | **write**}

The **cmpc** *path* and *device* parameter are defined exactly like that described for **csna**. The **cmpc** *tg-name* variable is a name associated with the subchannel being defined. A **read** and **write** subchannel must be defined to use CMPC. The *tg-name* value ties the two cmpc definitions together to form the CMPC transmission group. An example of coding the cmpc command is as follows:

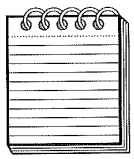
```
interface channel 4/0
!
cmpc 97A 5A R1CIP read
cmpc 97B 5B R1CIP write
```

The CMPC transmission group is defined by using the **tg** command under the virtual channel interface x/2 definition. The format of the tg command is

tg *name* **llc** *type adapter-number lsap* [**rmac** *rmac*] [**rsap** *rsap*]

The *name* parameter is the transmission group name used for a previously defined **cmpc** statement. This *name* ties the subchannel pair to the LLC2 driver for the internal LAN. The **llc** keyword denotes connectivity to the LLC stack on the CIP. The *type* parameter specifies the type of internal LAN defined for use by the transmission group. The only value allowed at this time is **token-adapter**.

The *adapter-number* parameter identifies which internal virtual LAN adapter definition is used by the transmission group. The *lsap* parameter defines the local SAP address used for communicating to the host. The SAP address value used here must be unique within the router and host, along with any IEEE 802.2 clients using the specified adapter. The default is 04. It may be wise, however, to specify the high end of the allowable range, FC, and move down for additional CMPC connections to avoid any unknown conflicts. The value is a multiple of four, ranging from 04 to FC. The optional keyword **rmac** and its associated variable *rmac* is a MAC address assigned for use by the cmpc driver for LLC2 connectivity. The optional **rsap** keyword and variable *rsap* defaults to 04 and is the remote SAP address used by the driver for communications.



NOTE *The **no tg** command must be entered first prior to changing any of the parameters. The full **tg** definition must then be entered since the **no tg** command deletes the entry from the configuration.*

For example, in Figure 22-18, a CMPC configuration is established from a Token Ring-attached APPN EN to a VTAM NN through a Cisco channel-attached router. The router is a direct ESCON connection on ESCON port EB.

The following configuration is applied to Figure 22-18:

R1 configuration applied to Figure 22-18:

```
source-bridge ring-group 2
!interface tokenring 0 source-bridge 10 1 2! interface Channel6/1
 no ip address
 no keepalive
 cmpc EB10 50 VTAM1 READ
 cmpc EB10 51 VTAM1 WRITE
!
interface Channel6/2
 no ip address
 no keepalive
 lan TokenRing 0
 source-bridge 62 2 2
 adapter 0 4000.0000.0001
 lan TokenRing 1
 tg VTAM1 llc token-adapter 0
```

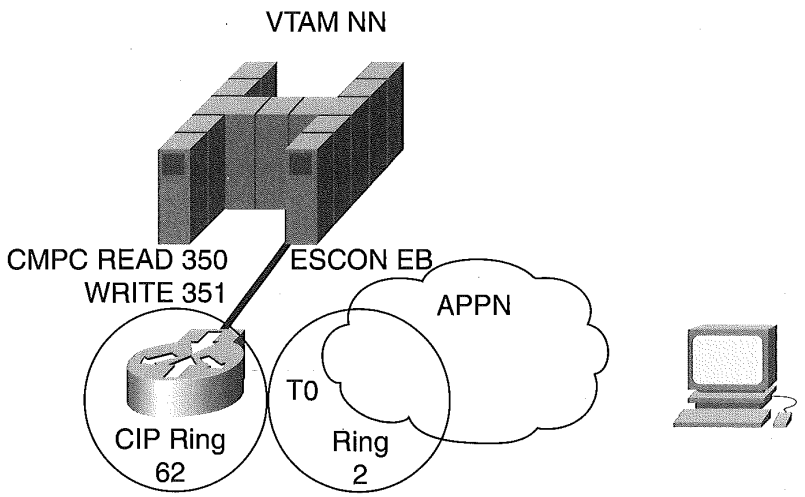


Figure 22-18 CMPC sample configuration for connecting an APPN EN to a VTAM NN.

In the router R1 configuration for Figure 22-18, the CMPC TG is assigned to a second internal tokenring-interface that connects adapter 0 of LAN tokening0 to the MPC channels connecting to the mainframe.

SNA connectivity to the mainframe from the CIP/CPA interface is also possible by having the router act as a TN3270 server. Using TN3270 server functionality enables the network designer to focus, as much as possible, SNA protocol on the data center location only. This is because the TN3270 protocol uses TCP/IP for access to the server. The protocol itself allows non-SNA to be connected as SNA logical units (i.e., 3,270 display terminals) without the overhead of SNA. The TN3270 server functionality can be combined with APPN DLUS/DLUR service to provide backup for the LUs defined to the TN3270 server. Figure 22-19 illustrates such a configuration.

The following router configurations apply to Figure 22-19:

R1 configuration applied to Figure 22-19:

```
source-bridge ring-group 1
dlsw local-peer
!
microcode CIP flash slot0:cip32-1.bin
microcode reload !
interface Channel1/0
 no ip address
 csna ED00 00
 csna EC00 00
!
```

```
interface Channel1/2
 ip address 10.1.1.1 255.255.255.0

 no keepalive
 lan Tokenring 0
   source-bridge 2 1 1
     adapter 0 4000.7513.3270
 tn3270-server
   pu PTN32701 05D7701C 10.1.1.2 token-adapt 1 1C rsap 04
   pu PTN32702 05D77020 10.1.1.3 token-adapt 1 20 rsap 08
   dlur NETA.DLUR3270 NETA.DLUS3270
    lsap token-adapter 0
   link VTAM1 rmac 40000.7513.9220 rsap 04
   link VTAM2 rmac 40000.7513.9220 rsap 08
   vrn VRNODE
   pu DLURPU 05D09220 10.1.1.2
   dlus-backup NETA.VTAM2
!
router eigrp 10
   network 10.0.0.0\
```

R2 configuration applied to Figure 22-19:

```
interface serial 0ip address 10.253.1.3 255.255.255.0!interface Ethernet
0ip address 10.30.1.1 255.255.255.0
!
router eigrp 10
 network 10.0.0.0\
```

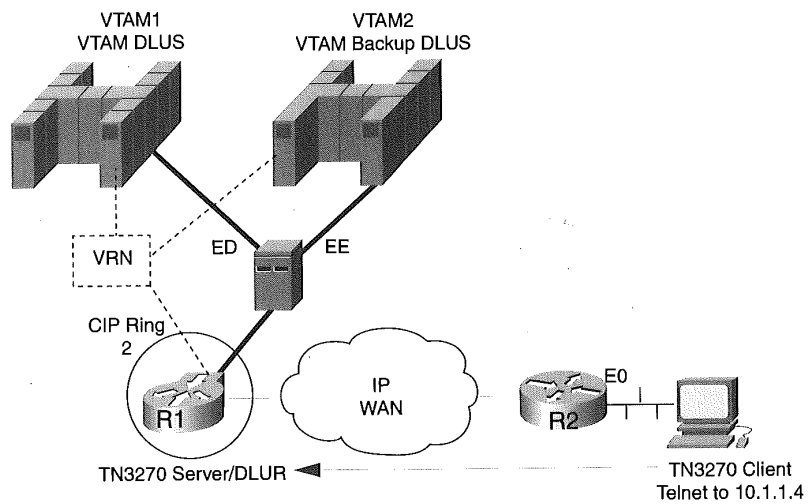


Figure 22-19 TN3270 server and DLUS/DLUR backup connectivity.

TN3270 server support requires IP to be active on the virtual channel interface definition. The statement **max-llc2-sessions** specifies the maximum number of active SNA LLC2 sessions at any given time through this virtual Channel 4/2 interface. The TN3270 server uses a second virtual adapter on the virtual LAN. Relative adapter 1 and MAC address 4000.3270.7006 define the addresses for the TN3270 server adapter:

```
source-bridge ring-group 4
interface channel 4/0
interface Channel 4/2
 ip address 192.168.6.1 255.255.255.0
 no keepalive
 max-llc2-sessions 2000
 LAN Tokenring 0
 source-bridge 6 1 4
 adapter 0 4000.0000.7006
 adapter 1 4000.3270.7006
 tn3270-server
 maximum-lus 4000
 pu PUEB2001 017FABC0 192.168.6.2 token-adapter 1 04 luseed LUTNS###
```

Entering the **tn3270-server** command enables the TN3270 server functions. The TN3270 server can handle a maximum of 30,000 LU sessions. The **maximum-lus** statement limits the number of LU's supported by the TN3270 server. In the example, the maximum number of LU's is 4,000. The previous **max-llc2-sessions** value thereby limits the number of active LU's to only 2,000 at any given time.

The core of the TN3270 server functions is the definition of the SNA PU. The **pu** statement defines the variables required to establish PU and LU sessions with the VTAM on the mainframe. The format of the statement is as follows:

PU *puname idblkidnum ip-address adapter-type ran lsap* **luseed** *lunamestem*

The *puname* on the TN3270 PU statement for the router matches the name of the VTAM-switched PU name. The *idblkidnum* value must match the IDBLK and IDNUM values specified on the VTAM PU definition statement defined for the TN3270 PU representation. The *ip-address* is the IP address used by the TN3270 client for connecting to the TN3270 server PU. The *adapter-type* value must match the virtual LAN type defined for the Channel 4/2 interface. The *ran* value identifies the adapter that is used for connectivity to the mainframe.

The *lsap* variable of the TN3270 PU statement identifies the service access point (SAP) used for communication over this virtual interface for this specific PU. If a second PU were defined under the TN3270 server

using the same virtual adapter, then a different SAP value would be assigned to the second PU definition. Usually, the SAP value for SNA begins with 04 and increments by four. Thus, a second PU using the same rmac macaddress would use a lsap value of 08.

The **luseed** keyword indicates that the VTAM DDDLU feature will be used to define LU names and their associated characteristics dynamically. The names are based on the value given to the *lunamestem* variable. The ### positions are replaced by VTAM during definition time with the decimal value of the LU's local address (locaddr) value as assigned by VTAM in the switched major node. Use of two ## indicates that hexadecimal value count is used.

The TN3270 server function can also be employed by using the DLUR/DLUS features of Cisco IOS and VTAM. The DLUR function of the TN3270 feature allows the TN3270 PU to appear as an APPN end node. The commands applicable to DLUR are as follows:

dlur *fq-cpname fq-dlusname*

dlus-backup *dlusname2*

preferred-nnserver *name*

lsap *type adapter-number* [*lsap*]

link *name* [**rmac** *rmac*] [**rsap** *rsap*]

vrn *vrn-name*

pu *pu-name idblk-idnum ip-address*

The **dlur** command follows the usage as discussed for APPN connections. The *fq-cpname* is a fully qualified control point name (net id) and the LU name used for the session switching. The *fq-dlusname* is the name of the control point providing the DLUS services.

The **dlus-backup** command identifies the fully qualified CP name of the VTAM DLUS providing backup. Only one dlus-backup can be used on a CIP.

The **preferred-nnserver** *name* specifies the name of the APPN network node server to which this DLUR definition belongs. The name is the CP name of an adjoining NN. This is an optional parameter and is not required for SNA switching to take place.

The **lsap** *type adapter-number* [*lsap*] is the local SAP address definition for the DLUR end node. The *type* parameter identifies the type of internal

LAN adapter in use for the DLUR. The only valid value at this time is **token-adapter**. The *adapter-number* parameter identifies which adapter on the internal LAN is being used for the DLUR connection. The optional *lsap* variable is the local SAP address used by the DLUR, ranging from 04 to FC in multiplies of four. The value selected must be unique for all LL2 connections traversing the adapter. The default value is C0.

The **link** *name* [**rmac** *rmac*] [**rsap** *rsap*] command defines an APPN link to the host for the end node DLUR. The *name* parameter is the eight-character alphanumeric string identifying the link and must be unique for the DLUR being defined. The *rmac* value is optional and defines the remote MAC address used for connecting the end node. The *rsap* value is the SAP address used for communicating to the DLUS over the link. The default here is 04 and ranges from 04 to FC in multiples of four.

The **vrn** *vrn-name* identifies the name of the virtual routing node for connecting the DLUR to the DLUS and possible backup DLUS. The DLUS and backup DLUS must have the same VRN name specified in the VTAM-switched major node representing the DLUR or the switch will fail.

The **pu** *pu-name idblk-idnum ip-address* command defines the TN3270 DLUR PU used for connecting TN3270 clients. The *pu-name* parameter is a unique PU name assigned to the PU command. For operational and documentation purposes, the name should match the VTAM PU-switched major node name defined for supporting the TN3270 PU definition. The *idblk-idnum* parameter must match the unique IDBLK and IDNUM parameters of the PU definition statement in VTAM that represents this TN3270 connection. The *ip-address* is the IP address used by the TN3270 client for connecting to the TN3270 server.

IP Connectivity Using a Channel-Attached Cisco Router

IP connectivity to a mainframe on Cisco routers using the CIP or CPA interface employs the Common Link Access to Workstation (CLAW) channel protocol. CLAW uses a dedicated pair of even or odd channel unit addresses for sending and receiving data traffic to the mainframe. CLAW supports 256 concurrent addressable devices and 128 concurrent connections per interface adapter. The number of IP connections through the router to the mainframe is unlimited. The limit for these connections is dependent on the capabilities of the TCP/IP application residing on the mainframe.

The CLAW statement is made up of several different parameters. The use of these is described in the appendix. This section details their specific use with guidelines for naming conventions at MSKCC.

The CLAW statement has the following specification:

claw *path device-address ip-address host-name device-name host-app device-app* [**broadcast**]

The *path* parameter is subdivided into three arguments:

- the channel path
- the channel logical address
- the control unit logical address

The *path* and device-address variables are defined by the same criteria as discussed for CSNA connectivity in the section, "SNA Support Using a Channel-Attached Cisco Router."

The CLAW protocol uses an even/odd pair of I/O addresses for communicating with TCP/IP on the mainframe. Therefore, as an example, if the I/O address of the CIP2 interface on the mainframe is EB0, then addresses EB0 and EB1 are in use for communications between the mainframe and the CIP2 router.

Table 22-1 identifies the Cisco IOS/390 for MVS parameters equivalent to the associated Cisco IOS CLAW definition parameters. The values associated with these parameters must be identical on the CIP/CPA CLAW statement and Cisco IOS/390 configuration file in order for the CIP/CPA to connect and communicate with TCP/IP on the mainframe.

In the following router configuration example, the CIP connects to an ESCON director, which shares the channel among four LPARs on the mainframe. Figure 22-20 illustrates this logical configuration.

The following router configuration applies to Figure 22-20 and is concerned only with the CIP channel definition for IP access:

Table 22-1 Cisco IOS CLAW Parameters That Must Match Cisco IOS/390 Parameters

CLAW	IOS/390
ip-address	Network host
Host-name	Hostname
Device-name	Wsname

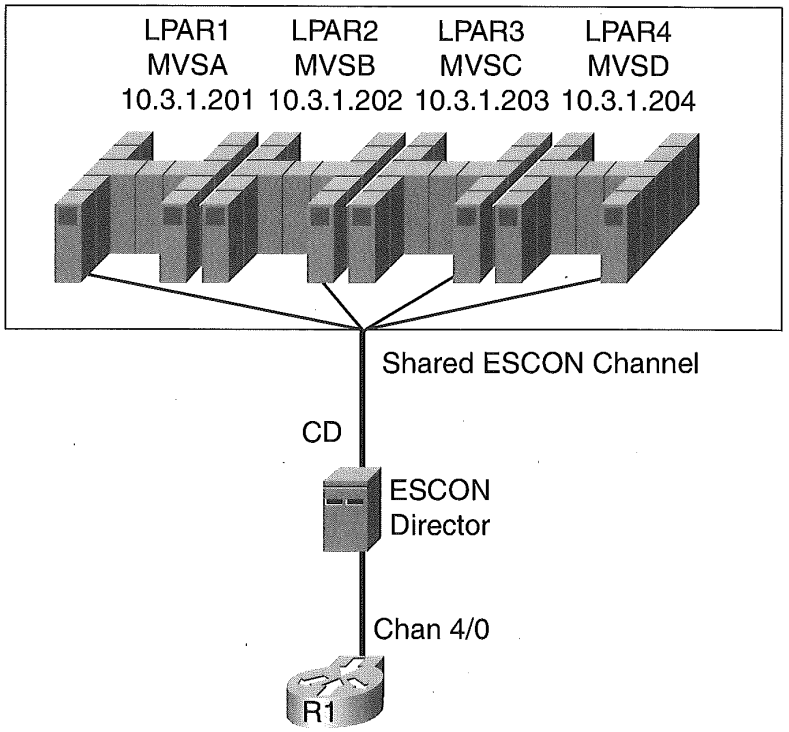


Figure 22-20
CIP connectivity for IP communication to four LPARs on a mainframe computer.

```
interface Channel4/0
 ip address 10.3.1.3 255.255.255.0
 claw CD10 00 10.3.1.201 MVSA R1 TCPIP TCPIP
 claw CD20 00 10.3.1.202 MVSB R1 TCPIP TCPIP
 claw CD30 00 10.3.1.203 MVSC R1 TCPIP TCPIP
 claw CD40 00 10.3.1.204 MVSD R1 TCPIP TCPIP
!
router rip
 network 10.0.0.0
```

In the sample configuration, the ESCON port connecting to the mainframe from the ESCON director is address CD. Each LPAR on the mainframe is identified by LPAR numbers 1 through 4. The IOCP defaults the CUADD value to 0. This makes up the CD10 value on the first CLAW statement in the example. The other three CLAW statements follow the same pattern. The IP address assigned to each connection is used to access the mainframe via IP. For example, if FTP were to be executed to LPAR MVSC,

then the IP address 10.3.1.203 would be used to open the FTP connection. The host names supplied in the example are from the host name parameter value specified on the mainframe TCP/IP stack.

The R1 value in the configuration above is the device-name variable of the CLAW statement and matches the Wsname parameter value on the mainframe. The last two values in the sample configuration of the CLAW statement are specified as TCP/IP and match the application name on the mainframe.

The IP stack on the mainframe must process every IP packet that is presented to it whether or not the packet is destined for the mainframe. A feature available to reduce this unproductive overhead is called offload. Defining offload requires the same variables and value criteria as the CLAW statement. The only difference is the use of the offload keyword. In the following example, the offload feature has been added to the previous router configuration example used for Figure 22-20:

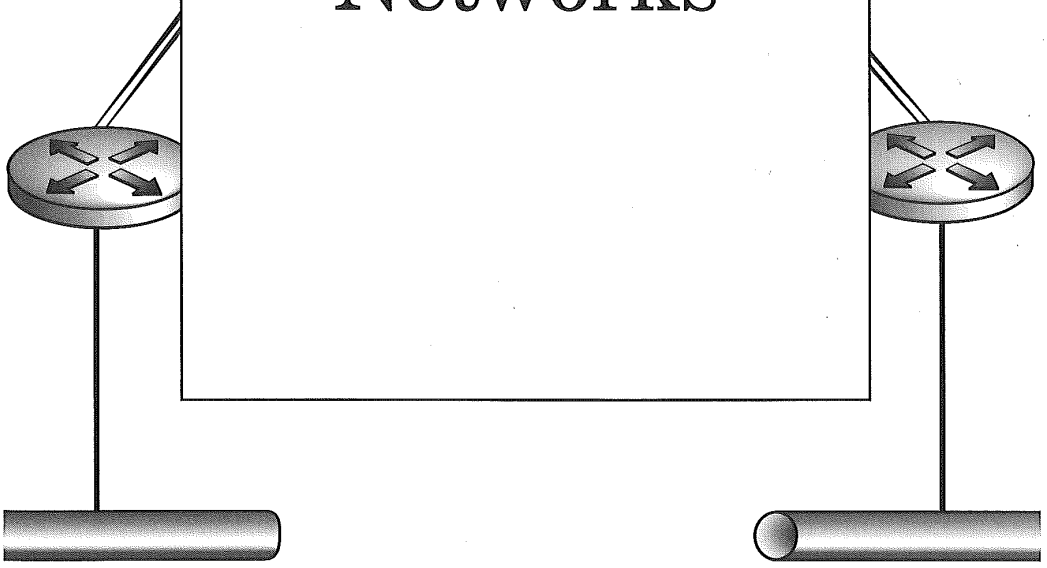
interface Channel4/0

```
 ip address 10.3.1.3 255.255.255.0
 claw CD10 00 10.3.1.201 MVSA R1 TCPIP TCPIP
 claw CD20 00 10.3.1.202 MVSB R1 TCPIP TCPIP
 claw CD30 00 10.3.1.203 MVSC R1 TCPIP TCPIP
 claw CD40 00 10.3.1.204 MVSD R1 TCPIP TCPIP
 offload CD10 00 10.3.1.201 MVSA R1 TCPIP TCPIP
 offload CD20 00 10.3.1.202 MVSB R1 TCPIP TCPIP
 offload CD30 00 10.3.1.203 MVSC R1 TCPIP TCPIP
 offload CD40 00 10.3.1.204 MVSD R1 TCPIP TCPIP
!
router rip
 network 10.0.0.0
```

Using the offload feature with the checksum feature of the IOS/390 IP stack also enables the router to perform checksum processing, further reducing the overhead on the mainframe.

CHAPTER 23

Defining Novell Networks



Novell networks became the dominant LAN protocol in the early 1990s. Although it could now be classified as a legacy system, as could IP, Novell is given special credence due to its deployment and continued dominance in corporate networks. Internet Packet Exchange (IPX) is actually based on Xerox Network Services (XNS) protocol, but the concept of IPX is akin to IP. Each physical wire on the network is assigned a network number. The hosts on the network, however, are not assigned a Novell networking address. Instead the hosts are addressed by layer 2 of the OSI Reference Model. It is the services of the Novell IPX resources, file and print servers, that utilize the layer 3 network services of IPX.

IPX Processing

Unlike IP routing, IPX processing is not enabled by default in Cisco's IOS. You must manually configure IPX before you can route IPX traffic. In some small designs, you may not find it necessary to route IPX at all. You may, for example, wish to simply bridge two IPX segments together over a WAN connection. In this chapter, we will consider both routing and bridging configurations for IPX.

Configuring for IPX Routing

IPX processing is started using the following global configuration command:

ipx routing

This command assigns a node address for the IPX routing process. The results of this action are displayed when using the show running-config command. The following listing is an example of this display:

```
router# show running-config
Building configuration...

Current configuration:
!
version 11.3
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
service udp-small-servers
```

```
service tcp-small-servers
!
hostname router
!
enable password cisco
!
ipx routing 0010.7b3a.a8b3
!
 —More—
```

Now that IPX processing is initiated, the router is configured to recognize and forward IPX traffic between different IPX networks. This is accomplished by defining unique IPX addresses to each router interface on which IPX is a required network protocol. The IPX address is made up of an IPX network number and node number combination. Cisco IOS assumes the interface's "Burned In Address" (BIA) for the node number when IPX is specified for Ethernet, Token Ring, and FDDI interfaces. For other interfaces, like serial, HSSI, or ATM, Cisco IOS uses the BIA of the lowest available IEEE type interface on the router. If you have a router with only non-IEEE interfaces, Cisco IOS generates a 48-bit quantity to use as the node number for IPX interfaces. Consider this example:

```
router#show interface ethernet0
Ethernet0 is administratively down, line protocol is down
 Hardware is Lance, address is 0010.7b3a.a8b3 (bia 0010.7b3a.a8b3)
(additional output omitted)
router#config term
router(config)#int e0
router(config-if)#ipx network 10
router(config-if)#end
router#show ipx interface e0
Ethernet0 is up, line protocol is up
 IPX address is 10.0010.7b3a.a8b3, NOVELL-ETHER [up]
(additional output omitted)
```

Examining the output from the **show interface** command, we see that the BIA of Ethernet 0 is the hexadecimal MAC address ***0010.7b3a.a8b3.*** Likewise, the output of the **show ipx interface** command reveals that Cisco IOS has created an address for our Ethernet 0 interface. The interface uses the 32-bit network number ***10*** (really 0x00000010—Cisco IOS suppresses leading zeros when displaying network numbers) and the 48-bit node number ***0010.7b3a.a8b3*** to create an 80-bit IPX address.

In the output above, you will also notice the term NOVELL-ETHER. This refers to the encapsulation chosen for IPX network number ***10***. The term

describes the OSI Layer 2 protocol that encapsulates the IPX packet before its transmission on the physical network. Novell uses a different term when describing encapsulation formats; the documentation for your NetWare server calls encapsulation "frame type." You have several choices for encapsulation on each interface of your Cisco router. Each Cisco encapsulation option corresponds to a specific Novell frame type. Table 23-1 lists the various frame types.

In order for two directly connected IPX devices to communicate, they must both have addresses on the same IPX network number and they must use the same encapsulation. In Figure 23-1, routers R1 and R2 meet these criteria and are able to communicate. Routers R1 and R3 share the same network number on the link between them, but the administrator has chosen different encapsulations for each router. This misconfiguration will prevent communication between the pair of routers.

Table 23-1
Cisco Encapsulation Type Compared to Novell Frame Type

Cisco Encapsulation	Novell Frame Type
arpa	Ethernet_II
sap	Ethernet_802.2
novell-ether	Ethernet_802.3
snap	Ethernet_SNAP

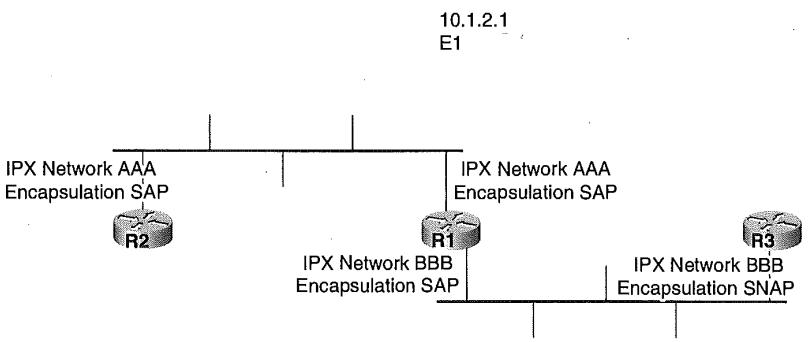


Figure 23-1
IPX encapsulation misconfiguration.

Routing IPX

We have a choice of three dynamic routing protocols for use with IPX. You can also choose to implement static routes in your IPX environment. In this section, we examine the use of three IPX routing protocols:

- RIP
- Netware Link Services Protocol (NLSP)
- Cisco's EIGRP for IPX

We will conclude with a discussion of the use of static routes in an IPX environment.

IPX RIP

When you complete the statements in, "IPX Processing," you also configure IPX RIP routing by default. When you start IPX processing with the **ipx routing** command, Cisco IOS automatically inserts the global configuration command:
ipx router rip

When you configure IPX network numbers on each of your router's interfaces, IPX also adds commands like
router(config-ipx-router)# network 10
router(config-ipx-router)# network 20
into the IPX RIP routing section of your running configuration file. As is the case with almost all default commands in IOS, you will not see these commands in the output of the **show running-config** command. You can make the commands visible by inserting a non-default command like
router(config-router)# no network 30

Observe the result of this configuration using the **show running-config** statement.

Such a configuration will be useful to us when we consider using EIGRP or NLSP to save bandwidth on WAN links. We will choose to disable RIP routing on the WAN interfaces while still running it on the LAN interfaces. The ***no network 30*** command in IPX RIP routing configuration mode has the effect of stopping IPX RIP broadcasts on the matching interface. In the next two sections, we will see how to replace IPX RIP routing with NLSP and EIGRP on the WAN interface.

You may wonder why we don't dispense with IPX RIP altogether. After all, RIP is relatively bandwidth-intensive and creates a fair amount of

processor overhead. Wouldn't it be better to use EIGRP or NLSP to perform all IPX routing? It turns out that this is not a practical approach to improving network performance as devices other than Cisco routers depend on IPX RIP broadcasts for their proper operation. For instance, the process by which a client PC locates and accesses an authentication server requires that a server or Cisco router send it a RIP route.

Consider the network shown in Figure 23-2. We begin our configuration example by enabling IPX processing on each of these routers. We will address each of the interfaces shown in the figure and then examine the IPX RIP routing table that results. Let's begin with router R1:

```
R1 (config)#ipx routing
R1 (config)#interface e0
R1 (config-if)#ipx network 11010
R1 (config-if)#no shutdown
R1(config-if)#interface s1
R1(config-if)#ipx network 11020
R1(config-if)#clock rate 64000
R1(config-if)#no shutdown
```

Since we are configuring back-to-back serial connections in the example, it is necessary to supply a clock signal on the DCE-connected interface, **serial 1**. It is unlikely that you will use such a configuration except in a lab environment.

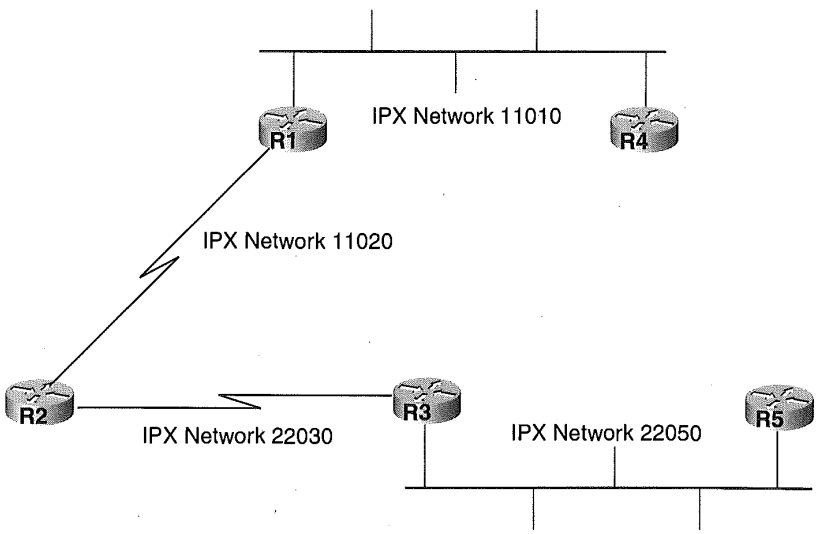


Figure 23-2
IPX internetwork.

Now we continue our configuration with the remaining routers:

```
R2(config)#ipx routing
R2(config-if)#interface s0
R2(config-if)#ipx network 11020
R2(config-if)#no shutdown
R2(config-if)#interface s1
R2(config-if)#ipx network 22030
R2(config-if)#clock rate 64000
R2(config-if)#no shutdown

R3(config)#ipx routing
R3(config)#interface e0
R3(config-if)#ipx network 22050
R3(config-if)#no shutdown
R3(config-if)#interface s0
R3(config-if)#ipx network 22030
R3(config-if)#no shutdown

R4(config)#ipx routing
R4(config)#interface e0
R4(config-if)#ipx network 11010
R4(config-if)#no shutdown

R5(config)#ipx routing
R5(config)#interface e0
R5(config-if)#ipx network 22050
R5(config-if)#no shutdown
```

We can view the result of our effort by examining the IPX routing table on R5. Use the ***show ipx routes*** command to view the IPX routing table:

```
R5#show ipx route
Codes: C - Connected primary network, c - Connected secondary network
 S - Static, F - Floating static, L - Local (internal), W - IPXWAN
 R - RIP, E - EIGRP, N - NLSP, X - External, A - Aggregate
 s - Seconds, u - Uses, U - Per-user static

5 Total IPX routes. Up to 1 parallel paths and 16 hops allowed.

No default route known.

C  22050 (NOVELL-ETHER), Et0
R  11010 [08/02] via 22050.0010.7b3a.d4b3, 16s, Et0
```

```
R  11020 [02/01] via 22050.0010.7b3a.d4b3, 16s, Et0
R  11030 [08/02] via 22050.0010.7b3a.d4b3, 16s, Et0
R  22040 [02/01] via 22050.0010.7b3a.d4b3, 16s, Et0
R5#
```

NLSP Routing

NLSP is the preferred IPX routing protocol for NetWare 4.x servers. It is an IPX implementation of the OSI Intermediate System to Intermediate System (IS-IS) routing protocol. NLSP can be classified as a link-state protocol. That is, NLSP routers establish a neighbor relationship with adjacent devices and then create advertisements of the state of their directly connected networks or links.

NLSP routers use the advertisements to build a database that represents the topology of the IPX internetwork. In order to distribute topology information to distant parts of the internetwork, each NLSP router also forwards each new link state message received from one neighbor to all other neighbors. This process, called flooding, ensures that all routers converge quickly on a new routing solution whenever the internetwork topology changes.

NLSP sends an advertisement, called a Link State Packet (LSP), when the state of a directly connected network changes, such as when a link fails. The router collects LSP's into a link state database. When new LSP's are inserted into the database, a calculation is performed on the entire database, resulting in a table of the best routes to each destination network.

To ensure that routing tables are consistent on all devices in the internetwork, it is essential that neighboring routers have identical databases. NLSP routers synchronize databases with their neighbors using Complete Sequence Number Packets (CSNP) and Partial Sequence Number Packets (PSNP). Every 120 minutes by default, NLSP routers send an advertisement that identifies the LSP's in their link state database. When a router compares the LSP list received from a neighbor with its own table and finds one missing request, it updates its own table.

NLSP creates a hierarchy of routing information within the internetwork. In NLSP, as in its predecessor IS-IS, certain routers share detailed information about destination networks with each other, while other routes only share summarized information. Devices that share detailed information are said to belong to the same NLSP "area."

NLSP routing areas are identified by their area address ranges. Each range is given by an area address and mask. The area address is a 32-bit

quantity usually written in hexadecimal digits. The mask, also written in hex characters, identifies the digits that will be common to all networks in the area. A "one" bit in a mask bit position indicates that the position will be common to the networks in the area. A "zero" bit in a mask bit position indicates that the position will be used to differentiate networks that fall within the range (recall that all IPX networks require a unique 32-bit network number). NLSP areas can accommodate from one to three area address ranges.

Three different levels of routing functions are defined for NLSP. In NLSP, routing functions are determined by the neighbor relationship formed between two routers. Any pair of routers can form an NLSP neighbor relationship by exchanging small multicast or broadcast messages called "hello" packets. The messages advertise the area to which a router belongs. Two routers form a Level 1 relationship when they both advertise membership in the same area. A Level 1 router will flood detailed link state information to its Level 1 neighbors. Level 1 routers also belong to the same routing area. Figure 23-3 illustrates Level 1 routers in an area with two address/mask combinations.

A Level 2 neighbor relationship forms between routers that belong to different areas. This relationship is diagrammed in Figure 23-4.

A Level 2 router can also have Level 1 neighbors in its own area. Level 2 routers don't flood LSP's to their Level 2 neighbors. Instead, they send an advertisement of one or more address ranges in the Level 1 area to which they belong. This allows the advertisements of many physical networks to be summarized into only a few area advertisements. Since the size of an internetwork is often limited by the number of networks that show up in each device's routing tables, summarization allows you to design much larger IPX internetworks. In Figure 23-4, we add Level 2 routing to our internetwork.

Level 3 routers further reduce the quantity of routing information maintained throughout the internetwork by allowing you to manually restrict and summarize the advertisements sent between routing domains. NLSP Level 3 routing differs from Level 1 and Level 2 in that it is the IPX implementation of the OSI Inter-Domain Routing Protocol (IDRP). In IOS releases prior to 11.1, Cisco supported only Level 1 routing.

Four tasks are required to configure NLSP routing on your Cisco IOS router:

1. First, you must create an internal network number for the NLSP routing process.
2. Next, you start an NLSP routing process.

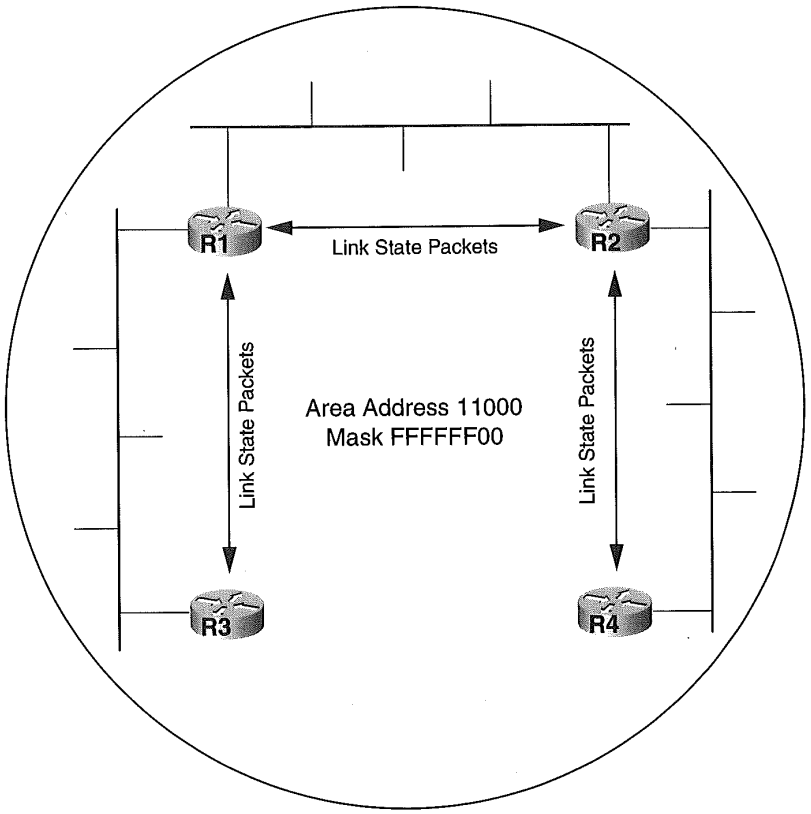


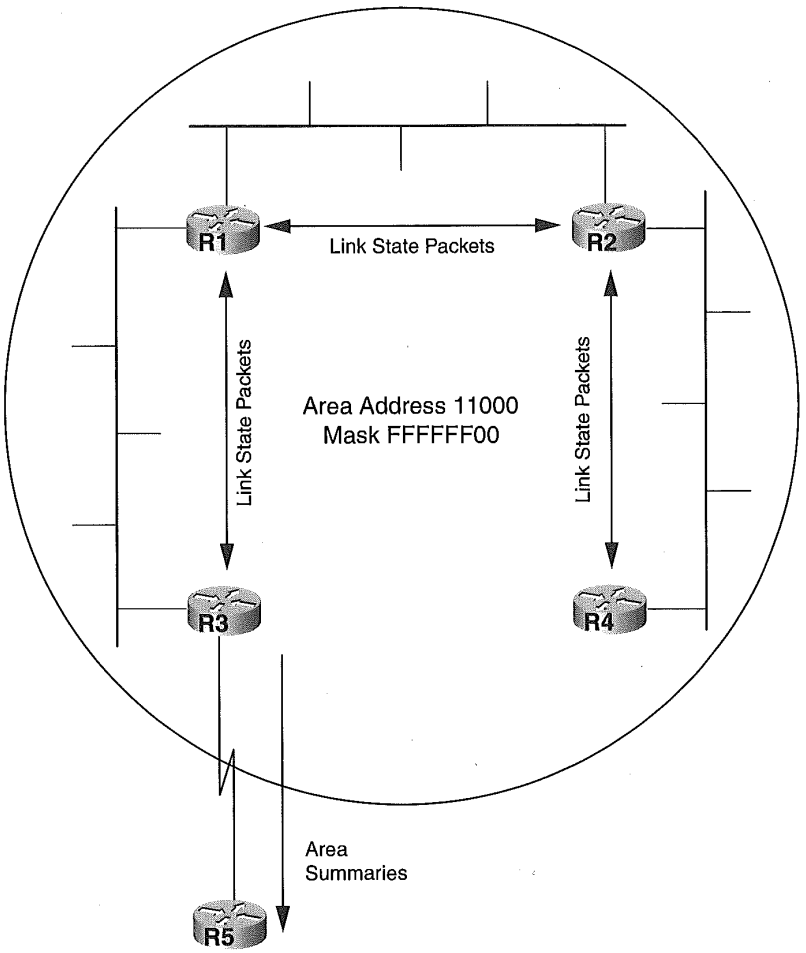
Figure 23-3
NLSP Level 1 routing area.

3. Once the process is running, you will specify one or more area addresses for the process.
4. The final step is to specify which physical or virtual interfaces will participate in the NLSP process.

The internal network number represents the routing process to other NLSP routers in the internetwork. It is essential that you choose unique internal network numbers for each NLSP router. If a Cisco IOS router configured for NLSP detects another IPX device with the same internal network number, it will in some circumstances disable its interfaces to prevent corruption of other NLSP databases in the area.

Use the following commands to disable IPX RIP and replace it with NLSP routing on routers 1, 2, and 4, as shown in Figure 23-5:

Figure 23-4
NLSP Level 2 routing.




```
R1(config)#no ipx router rip
R1(config)#ipx internal-network 11001
R1(config)#ipx router nlsp
R1(config-ipx-router)#area-address 11000 FFFFFF00
R1(config-ipx-router)#interface e0
R1(config-if)#ipx nlsp enable
R1(config-if)#interface s0
R1(config-if)#ipx nlsp enable
```

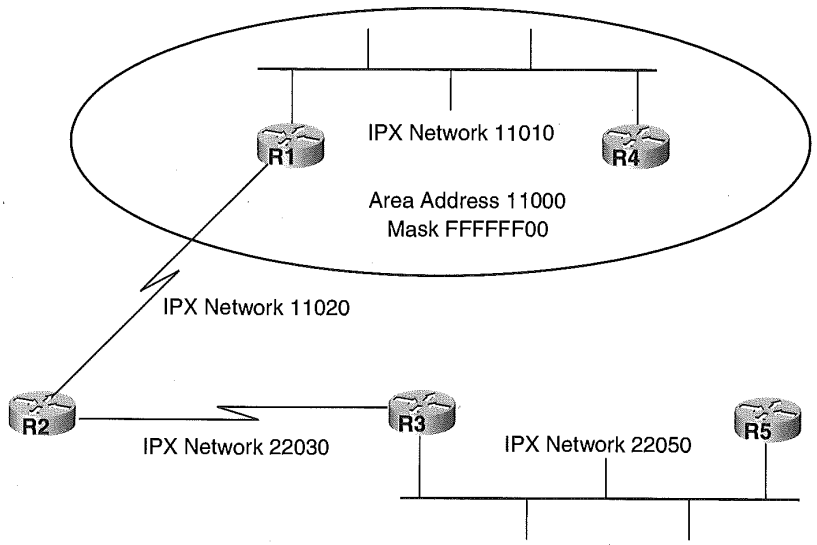


Figure 23-5
NLSP Level 1 routing example.

```
R1(config-if)#interface s1
R1(config-if)#ipx nlsp enable

R2(config)#no ipx router rip
R2(config)#ipx internal-network 11002
R2(config)#ipx router nlsp
R2(config-ipx-router)#area-address 11000 FFFFFF00
R2(config-ipx-router)#interface s0
R2(config-if)#ipx nlsp enable
R4(config-if)#interface s1
R4(config-if)#ipx nlsp enable

R4(config)#no ipx router rip
R4(config)#ipx internal-network 11004
R4(config)#ipx router nlsp
R4(config-ipx-router)#area-address 11000 FFFFFF00
R4(config-ipx-router)#interface e0
R4(config-if)#ipx nlsp enable
```

To verify that our NLSP configuration is working, we can examine the NLSP neighbor list and the ipx routing table. First, let's check the neighbors of R1:

```
R1#
R1#show ipx nlsp neighbor
NLSP Level-1 Neighbors: Tag Identifier = notag

System Id  Interface State Holdtime Priority Cir Adj Circuit Id
R4  Et0  Up  22  64  mc mc R4.01
R2  Se1  Up  55  0   — — 01
R1#
```

As we expected, we see that the R1 has formed relationships with R2 and R4. Now let's see what R4 has learned through NLSP:

```
R4#
R4#show ipx route
Codes: C - Connected primary network, c - Connected secondary network
 S - Static, F - Floating static, L - Local (internal), W - IPXWAN
 R - RIP, E - EIGRP, N - NLSP, X - External, A - Aggregate
 s - Seconds, u - Uses, U - Per-user static

7 Total IPX routes. Up to 1 parallel paths and 16 hops allowed.

No default route known.

L  11004 is the internal network
C  11010 (NOVELL-ETHER), Et0
N  11001 [20][02/01] via 11010.0010.7b3a.a8b3, 758s, Et0
N  11002 [47][03/02] via 11010.0010.7b3a.a8b3, 759s, Et0
N  11020 [20][01/01] via 11010.0010.7b3a.a8b3, 759s, Et0
N  22040 [47][02/02] via 11010.0010.7b3a.a8b3, 753s, Et0
NX 22050 [20][08/02][07/01] via 11010.0010.7b3a.a8b3, 759s, Et0
R4#
```

Several items of note show up in the table. First, notice that we see not only the physical networks we configured on the router interfaces, but the internal network numbers as well. We can conclude that our Level 1 routers are functioning properly.

In addition, for the route to IPX network number 20050 in the last line, the type code ***NX*** appears. This indicates that the route was learned from a source other than an NLSP-level 1 router. In fact, this route was injected into NLSP from the RIP routing process. We will examine the circumstances under which routing information from one protocol is passed into another in the "IPX Route Redistribution" section.

Configuring the NLSP Designated Router[ri]When two or more NLSP routers are connected by a multipoint-capable network such as Ethernet, the potential exists for a large number of relationships to form. Considering the NLSP description to this point, each of n routers connected to the network would form n-1 relationships. The total number of relationships in this arrangement would be given by the formula n*(n-1)/2. The relationships would increase exponentially with the number of routers on the network. Obviously, this growth in relationships would consume resources and ultimately limit the number of routers permitted in an NLSP area.

Fortunately, NLSP removes this scalability limitation by defining a special role for one of the routers. A router is "designated" to form a relationship with all other Level 1 routers on the multi-access network. The designated router has extra processing requirements to form and maintain these relationships. It makes sense to select a router with sufficient memory and CPU resources as your designated router.

Designated router selection is based on router priority. Each NLSP router has a default priority of 44. A higher priority router will become the designated router on its multi-access network. To manually configure a router as the designated router, use the ***ipx nlsp priority*** interface configuration command. To select R1 in Figure 23-4 as the designated router on its ethernet 0 interface, use the following configuration:

```
R1(config)#
R1(config)#interface e0
R1(config-if)#ipx nlsp priority 100
R1(config-if)#
```

You can verify your NLSP configuration using the ***show ipx interface*** command. First look at R4:

```
R4#show ipx interface e0
Ethernet0 is up, line protocol is up
(output omitted)
IPX address is 11010.0010.7b3a.d213, NOVELL-ETHER [up]IPX NLSP is
running on primary network 11010
RIP compatibility mode is AUTO (ON), last traffic rcvd 16 sec ago
SAP compatibility mode is AUTO (ON), last traffic rcvd 16 sec ago
Level 1 Hello interval 20 sec
Level 1 Designated Router Hello interval 10 sec
Level 1 CSNP interval 30 sec
Level 1 LSP retransmit interval 5 sec, LSP (pacing) interval 55 mSec
Level 1 adjacency count is 1
Level 1 circuit ID is R1.02
```

```
 Level 1 Designated Router is R1
R4#
```

On the last line of the show command output, you see that ***R1*** is the designated router. On the other side of the Ethernet connection, R1 agrees with R4 on the designated router election.

```
R1#show ipx interface e0
Ethernet0 is up, line protocol is up
 IPX address is 11010.0010.7b3a.a8b3, NOVELL-ETHER [up]
 (output omitted)
IPX NLSP is running on primary network 11010
 RIP compatibility mode is AUTO (OFF)
 SAP compatibility mode is AUTO (OFF)
 Level 1 Hello interval 20 sec
 Level 1 Designated Router Hello interval 10 sec
 Level 1 CSNP interval 30 sec
 Level 1 LSP retransmit interval 5 sec, LSP (pacing) interval 55 mSec
 Level 1 adjacency count is 0
 Level 1 circuit ID is R1.02
 Level 1 Designated Router is R1
R1#
```

Configuring NLSP Route Aggregation[ri]In order to implement NLSP Level 2 routing in a Cisco environment, you will configure the NLSP Route Aggregation feature. A Level 2 router will run multiple copies of the NLSP process, one for each Level 1 area to which it belongs. We will use the term "tag" for the NLSP parameter that differentiates each copy of the NLSP process.

An aggregate route contains a 32-bit area address and an eight-bit field that indicates the number of bits common to all the networks in the area range. A router receiving an aggregate route assumes that all network numbers that could fall into the area range are in fact available in the area. For example, if we summarize the routes 22040 and 22050 into an aggregate route, as displayed in Figure 23-6, R4 should see only a single advertised route, 22000, and an indication that the high-order 24-bits are common to all routes in the summary.

If this seems confusing, recall that IPX suppresses leading zeros in network numbers. Thus, network 22040 is represented internally as 0x00022040 and 22050 as 0x00022050. Now consider the left-most 24 bits (six hex digits or 0x000220) as analogous to an IP network number. IPX networks 0x00022040 and 0x00022050 look like two subnets of 0x00022000.

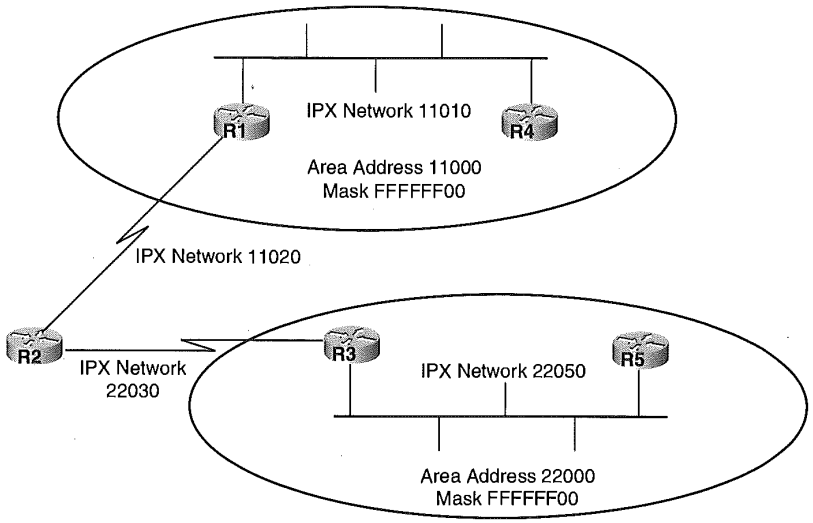


Figure 23-6
NLSP Level 2 routing example.

Now how might we configure our network to implement the design in Figure 23-6? The syntax below starts an NLSP routing process for area 22000 on R2, R3, and R5:

```
R2(config)#ipx router nlsp Area2
R2(config-ipx-router)#area-address 22000 FFFFFF00
R2(config-ipx-router)#interface s1
R2(config-if)#no ipx nlsp enable
R2(config-if)#ipx nlsp Area2 enable

R3(config)#ipx router nlsp Area2
R3(config-ipx-router)#area-address 22000 FFFFFF00
R3(config-ipx-router)#interface s0
R3(config-if)#no ipx nlsp enable
R3(config-if)#ipx nlsp Area2 enable

R5(config)#ipx routing
R5(config)#ipx router nlsp Area2
R5(config-ipx-router)#area-address 22000 FFFFFF00
R5(config-ipx-router)#interface e0
R5(config-if)#ipx network 22050
R5(config-if)#ipx nlsp Area2 enable
```

Examining the routing tables on R4 and R5 shows interarea routing without aggregate routes.

```
R4#show ipx route
Codes: C - Connected primary network, c - Connected secondary network
 S - Static, F - Floating static, L - Local (internal), W - IPXWAN
 R - RIP, E - EIGRP, N - NLSP, X - External, A - Aggregate
 s - Seconds, u - Uses, U - Per-user static

10 Total IPX routes. Up to 1 parallel paths and 16 hops allowed.

No default route known.

L  10004 is the internal network
C  11010 (NOVELL-ETHER), Et0
N  11001 [20][02/01] via 11010.0010.7b3a.a8b3, 236s, Et0
N  11002 [47][03/02] via 11010.0010.7b3a.a8b3, 236s, Et0
NX 11020 [47][03/03][01/01] via 11010.0010.7b3a.a8b3, 229s, Et0
NX 11030 [47][03/03][01/01] via 11010.0010.7b3a.a8b3, 236s, Et0
N  22003 [47][03/02]  via 11010.0010.7b3a.a8b3, 236s, Et0
NX 22005 [47][04/03][02/01] via 11010.0010.7b3a.a8b3, 229s, Et0
N  22040 [47][02/02]  via 11010.0010.7b3a.a8b3, 229s, Et0
N  22050 [47][02/02]  via 11010.0010.7b3a.a8b3, 229s, Et0
R4#

R5#show ipx route
Codes: C - Connected primary network, c - Connected secondary network
 S - Static, F - Floating static, L - Local (internal), W - IPXWAN
 R - RIP, E - EIGRP, N - NLSP, X - External, A - Aggregate
 s - Seconds, u - Uses, U - Per-user static

10 Total IPX routes. Up to 1 parallel paths and 16 hops allowed.

No default route known.

L  22005 is the internal network
C  22050 (NOVELL-ETHER), Et0
NX 10004 [20][04/03][03/02] via 22050.0010.7b3a.d4b3, 179s, Et0
NX 11001 [20][03/02][02/01] via 22050.0010.7b3a.d4b3, 180s, Et0
N  11002 [47][03/02]  via 22050.0010.7b3a.d4b3, 180s, Et0
NX 11010 [20][02/02][01/01] via 22050.0010.7b3a.d4b3, 180s, Et0
```

```
N  11020 [20][01/01]  via 22050.0010.7b3a.d4b3, 180s, Et0
N  11030 [47][02/02]  via 22050.0010.7b3a.d4b3, 180s, Et0
N  22003 [20][02/01]  via 22050.0010.7b3a.d4b3, 180s, Et0
R  22040 [02/01]   via 22050.0010.7b3a.d4b3, 187s, Et0
R5#
```

As a final step in implementing Level 2 routing, we must configure the aggregation of individual IPX network numbers. Since R2 is the Level 2 router that interfaces between our two areas, we use this syntax to configure route aggregation:

```
R2(config)#ipx router nlsp
R2(config-ipx-router)#route-aggregation
R2(config-ipx-router)#ipx router nlsp Area2
R2(config-ipx-router)#route-aggregation
```

IPX EIGRP Routing

Cisco's Enhanced Internetwork Gateway Routing Protocol (EIGRP) combines some of the best features of link state protocols with the configuration simplicity of distance-vector protocols. EIGRP has the unique capability to route traffic for three different protocol suites:

- IP
- IPX
- AppleTalk

It isn't necessary to configure all of the protocols, only those of use in your network. In this section, we will examine configuring EIGRP for Novell IPX.

Start EIGRP for IPX[ri]One of the advantages of using EIGRP for IPX is that configuration is trivially simple when compared to NLSP. To enable EIGRP, use the following syntax:

```
router(config)#ipx router eigrp 200
```

This command runs an instance of the EIGRP program on your router. The parameter ***200*** in this command is called the EIGRP Autonomous System (AS) number. We will see momentarily that the AS number is globally significant and must be configured the same on all routers that you intend to share routing information.

Once the routing process is started, you must specify the router interfaces that you want to participate in EIGRP. Unlike NLSP, we will select

EIGRP interfaces while in routing protocol configuration mode, rather than interface configuration mode. Use the ***network*** command as in the following:

```
router(config-ipx-router)#network 11020
router(config-ipx-router)#network 22040
```

If all the interfaces on your router will process EIGRP, you can simplify the configuration using the keyword ***all***:

```
router(config-ipx-router)#network all
```

As soon as you select an interface to participate in EIGRP, a process called "Neighbor Discover" begins. Your router sends a multicast packet on LAN interfaces or a broadcast packet on WAN interfaces to locate other EIGRP routers. These messages are called "hello" packets. EIGRP compiles the results of the discovery process in a data structure called the neighbors table.

When a neighbor is discovered and it belongs to the same EIGRP AS number, our router sends information about the locally configured networks and any networks learned through EIGRP routing. In this sense, EIGRP acts like a distance-vector routing protocol. These advertisements are sent in the form of EIGRP update messages.

When an EIGRP router receives an update message, the information is added to its topology table. The router processes this information to find the best path to each destination network in the topology table. EIGRP uses a metric called Distance to define the best path to each destination. The calculation of the EIGRP metric can include up to five different parameters. By default, only the lowest link bandwidth between your router, the destination network, and the delay encountered by packets passing along the path is taken into account in calculating Distance.

We can examine these details once we configure a sample network. In Figure 23-7, we will configure IPX throughout the network and then configure EIGRP on all the serial links, while leaving RIP active on the Ethernet links. First, starting with R1, we use these commands:

```
R1(config)#ipx routing
R1(config)#interface e0
R1(config-if)#ipx nework 110
R1(config-if)#no shutdown
R1(config-if)#interface s0
R1(config-if)#ipx network 120
R1(config-if)#no shutdown
R1(config-if)#interface s1
R1(config-if)#ipx network 130
R1(config-if)#clock rate 64000
R1(config-if)#no shutdown
```

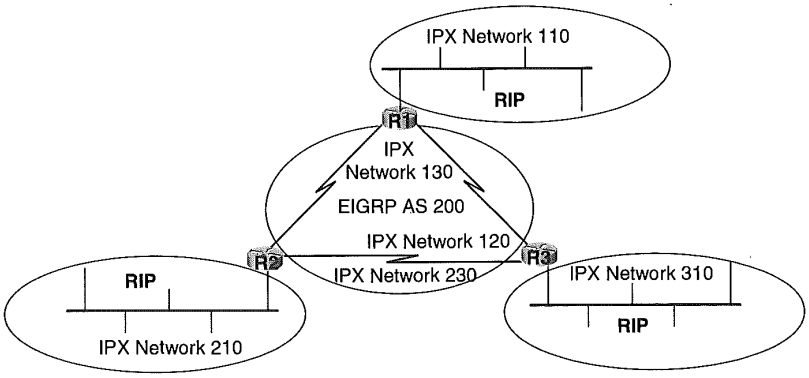


Figure 23-7
IPX EIGRP routing example.

Recall that the ***clock rate*** statement simply provides the capability to operate two routers in a back-to-back configuration in a lab environment. Continuing with the remaining routers, we use these commands:

```
R2(config)#ipx routing
R2(congif)#interface e0
R2(config-if)#ipx network 210
R2(config-if)#no shutdown
R2(config-if)#interface s0
R2(config-if)#ipx network 130
R2(config-if)#no shutdown
R2(config-if)#interface s1
R2(config-if)#ipx network 230
R2(config-if)#clock rate 64000
R2(config-if)#no shutdown

R3(config)#ipx routing
R3(config)#interface e0
R3(config-if)#ipx network 310
R3(config-if)#no shutdown
R3(config-if)#interface s0
R3(config-if)#ipx network 120
R3(config-if)#no shutdown
R3(config-if)#interface s1
R3(config-if)#ipx network 230
R3(config-if)#clock rate 64000
R3(config-if)#no shutdown
```

With IPX configured on the router, we turn our attention to the routing protocol:

```
R1(config)#ipx router eigrp 200
R1(config-ipx-router)#network 120
R1(config-ipx-router)#network 130
R1(config-ipx-router)#ipx router rip
R1(config-ipx-router)#no network 120
R1(config-ipx-router)#no network 130

R2(config)#ipx router eigrp 200
R2(config-ipx-router)#network 130
R2(config-ipx-router)#network 230
R2(config-ipx-router)#ipx router rip
R2(config-ipx-router)#no network 130
R2(config-ipx-router)#no network 230

R3(config)#ipx router eigrp 200
R3(config-ipx-router)#network 120
R3(config-ipx-router)#network 230
R3(config-ipx-router)#ipx router rip
R3(config-ipx-router)#no network 120
R3(config-ipx-router)#no network 230
```

When this configuration is entered on our small network, the following IPX routing table looks like this on R1:

```
R1#
R1#show ipx route
Codes: C - Connected primary network, c - Connected secondary network
  S - Static, F - Floating static, L - Local (internal), W - IPXWAN
  R - RIP, E - EIGRP, N - NLSP, X - External, A - Aggregate
  s - Seconds, u - Uses, U - Per-user static

6 Total IPX routes. Up to 1 parallel paths and 16 hops allowed.

No default route known.

C  110 (NOVELL-ETHER), Et0
C  120 (HDLC),  Se0
C  130 (HDLC),  Se1
E  210 [2195456/1] via  130.0010.7b3a.d4ae, age 00:01:32,
     1u, Se1
E  230 [2681856/0] via  130.0010.7b3a.d4ae, age 00:01:40,
```

```
     1u, Se1
E  310 [2195456/1] via  120.0010.7b3a.d4b3, age 00:01:29,
     1u, Se0
R1#
```

The IPX routing table on R3 looks like this:

```
R3#show ipx route
Codes: C - Connected primary network, c - Connected secondary network
 S - Static, F - Floating static, L - Local (internal), W - IPXWAN
 R - RIP, E - EIGRP, N - NLSP, X - External, A - Aggregate
 s - Seconds, u - Uses, U - Per-user static

6 Total IPX routes. Up to 1 parallel paths and 16 hops allowed.

No default route known.

C  120 (HDLC),   Se1
C  230 (HDLC),   Se0
C  310 (NOVELL-ETHER), Et0
E  110 [2195456/1] via  120.0010.7b3a.a8b3, age 00:01:20,
     1u, Se1
E  130 [2681856/0] via  230.0010.7b3a.d4ae, age 00:01:21,
     1u, Se0
E  210 [2195456/1] via  230.0010.7b3a.d4ae, age 00:01:21,
     1u, Se0
R3#
```

With the routing protocol configured, we can examine some of the commands and techniques you will find most useful for troubleshooting your own IPX networks. First, let's look at the EIGRP neighbors table. Use the ***show ipx eigrp neighbors*** command:

```
R1#sh ipx eigrp neighbors

IPX EIGRP Neighbors for process 200
H Address     Interface Hold Uptime SRTT RTO Q Seq
        (sec)  (ms)  Cnt Num
1 120.0010.7b3a.d4b3  Se0   11 00:05:46 39 234 0 8
0 130.0010.7b3a.d4ae  Se1   14 00:07:21 33 200 0 10
R1#
```

In this display we are looking for the IPX address of our neighboring routers. If a router is missing from this display, we should examine the configuration of both devices to ensure that the same EIGRP AS number is used on both sides of the link. You can also examine details about the

EIGRP neighbors discovered on a particular using the ***show ipx eigrp interface*** command. Here's an example:

```
R3#sh ipx eigrp int s0

IPX EIGRP Interfaces for process 200

   Xmit Queue Mean Pacing Time Multicast Pending
Interface Peers Un/Reliable SRTT Un/Reliable Flow Timer Routes
Se0  1 0/0  257 0/15  1275  0
R3#
```

We see that R3 has discovered R1 on its serial 0 interface.

Besides the ease of configuration, another strong reason for implementing EIGRP is the speed with which it converges after a change in the internetwork topology. Consider, for example, the complete loss of a link, as shown in Figure 23-8.

We can examine the convergence event using IOS debug services. Starting in Privileged Exec mode on R3, issue these debug commands:

```
R3#debug ipx eigrp
IPX EIGRP debugging is on
R3#debug ipx eigrp 200 110
IPX Target enabled on AS 200 for 110
R3#debug ipx eigrp neighbor 200 120.0010.7b3a.a8b3
IPX Neighbor target enabled on AS 200 for 120.0010.7b3a.a8b3
R3#
```

Now it is time to test worst-case convergence in our IPX EIGRP network. We will disable the Ethernet 0 interface on R1 using the ***shutdown*** interface command to simulate a link failure. Here is the debug output on R3:

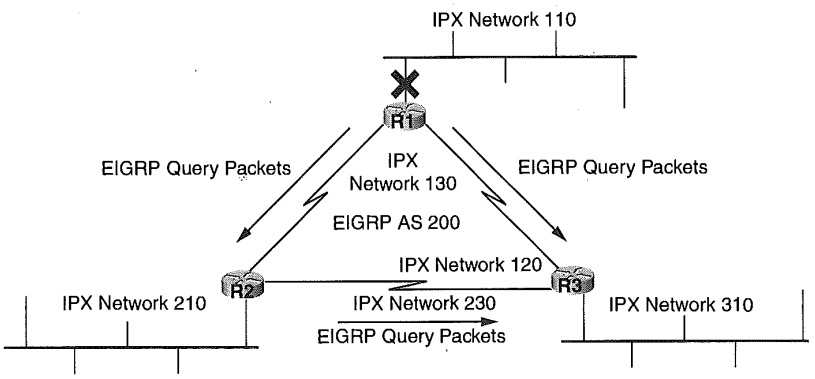


Figure 23-8 EIGRP convergence—loss of a route.

```
00:55:36: IPXEIGRP: 110 via 120.0010.7b3a.a8b3 metric 4294967295
00:55:36: IPXEIGRP: 110 via 230.0010.7b3a.d4ae metric 4294967295
00:55:36: IPXEIGRP: 110 via 120.0010.7b3a.a8b3 metric 4294967295
00:55:36: IPXEIGRP: 110 via 230.0010.7b3a.d4ae metric 4294967295
00:55:36: IPXEIGRP: Received update from 120.0010.7b3a.a8b3 for net 110
00:55:36: IPXEIGRP: worse [0/0/4294967295] route for 110 from
120.0010.7b3a.a8b3
,
  old [65535/255/2195456] route/paths
00:55:36: IPXEIGRP: Received update from 230.0010.7b3a.d4ae for net 110
00:55:36: IPXEIGRP: worse [0/0/4294967295] route for 110 from
230.0010.7b3a.d4ae
,
  old [65535/255/2195456] route/paths
00:55:36: IPXEIGRP: External 110 metric 4294967295 hop 16 delay 44
R3#
```

In the first debug output line, notice that R1 (IPX address ***120.0010.7b3a.a8b3***) responds to the link failure by sending an EIGRP update message, setting the metric for network ***110*** to the maximum value of ***4294967295***. The same update message is quickly relayed by R2 in the second line (IPX address ***230.0010.7b3a.d4ae***) and the exchange is repeated.

The change in our topology table caused by the loss of R1's Ethernet 0 interface results in a process called the Diffusing Update Algorithm (DUAL). EIGRP routers perform one or two DUAL calculations when the topology changes. If a primary route is lost but an alternate path is available, DUAL performs a "local computation" to determine whether the alternate route can be guaranteed to be loop-free. If the guarantee can't be inferred from the topology table, the router must perform a second "diffusing computation." During this calculation, our router sends an EIGRP query message to the routers listed in the neighbor table asking each one to provide a guaranteed loop-free path to the destination in question. Each neighboring router performs a local computation and, if necessary, a diffusing computation as well. Once the neighbors have made their determinations, they send an EIGRP reply message with information regarding guaranteed loop-free routes.

In our example, we know that there is no alternate path so we are not surprised that both neighbors R2 and R1 send us a reply with Distance still set to the maximum. Notice the time-stamps on the debug message, which is proof of the fast convergence claim. All these events transpired in less than one second. EIGRP is designed to give subsecond convergence in many

network designs. In real networks, the actual convergence time is largely governed by the different methods for determining that a given link has failed.

Now let's see what happens when the link returns. Issue the ***no shutdown*** command on the R1 Ethernet 0 interface. If we were using a distance-vector protocol, we might see several minutes loss of connectivity after the link is restored due to hold-down timers. In the debug output below, we notice subsecond convergence once again.

Now let's look at the case where we do find an alternate path to the destination network. Instead of disabling the Ethernet link on R1, we will fail-out the primary path between network 110 and R3, the R1 serial 0 interface:

```
08:25:43: IPXEIGRP: 110 via 120.0010.7b3a.a8b3 metric 2195456
08:25:43: IPXEIGRP: Received update from 120.0010.7b3a.a8b3 for net 110
08:25:43: IPXEIGRP: create route to 110 via 120.0010.7b3a.a8b3, metric
2195456
08:25:43: IPXEIGRP: External 110 metric 2195456 hop 1 delay 7
08:25:43: IPXEIGRP: External 110 metric 2195456 hop 1 delay 7
08:25:43: IPXEIGRP: 110 via 230.0010.7b3a.d4ae metric 4294967295
08:25:43: IPXEIGRP: Received update from 120.0010.7b3a.a8b3 for net 110
08:25:43: IPXEIGRP: Received update from 230.0010.7b3a.d4ae for net 110
08:25:43: IPXEIGRP: worse [0/0/4294967295] route for 110 from
230.0010.7b3a.d4ae
,
  old [0/0/2195456] route/paths
08:25:43: IPXEIGRP: External 110 metric 2195456 hop 1 delay 7
08:25:43: IPXEIGRP: 110 via 230.0010.7b3a.d4ae metric 2707456
08:25:43: IPXEIGRP: Received update from 120.0010.7b3a.a8b3 for net 110
08:25:43: IPXEIGRP: Received update from 230.0010.7b3a.d4ae for net 110
08:25:43: IPXEIGRP: worse [0/0/2707456] route for 110 from
230.0010.7b3a.d4ae,
  old [0/0/2195456] route/paths
R3#
```

Novell NetWare servers before version 4.x software were obligated to advertise all services available on a NetWare internetwork. The protocol used for this process was called the Service Advertisement Protocol (SAP). In a large Novell internetwork, it was common to see many thousands of service advertisements. The SAP protocol required that every SAP-capable device send its entire SAP table as a broadcast onto all networks to which it was connected. The structure of the SAP message allowed seven services

to be advertised in each 576-byte IPX packet. Because of this limitation, relatively low-bandwidth links like WAN interfaces could be saturated with repetitive broadcast traffic. This usually resulted in services that disappeared or reappeared from SAP tables at irregular intervals.

Today several alternatives to SAP advertisements are available to NetWare administrators. You may choose to use the popular NetWare Directory Service (NDS) or the more recent Service Location Protocol (SLP), for instance. If you must support older 3.x versions of NetWare in your internetwork, you need a way to cope with SAP. Fortunately, EIGRP allows you to stop the periodic broadcast of SAP tables on interfaces with no NetWare servers. EIGRP maintains the SAP table but only sends SAP updates when the table changes. Since traditional SAP devices depend on periodic SAP broadcasts to refresh their SAP table entries, an EIGRP incremental SAP advertisement is only useful on interfaces where our router speaks only to other Cisco routers.

In Figure 23-9, we have a Novell NetWare version 3.2 server connected to the R1 Ethernet0 interface. We can see the advertisement by examining router SAP tables with the ***show ipx servers*** command:

```
R3#show ipx servers
Codes: S - Static, P - Periodic, E - EIGRP, N - NLSP, H - Holddown, + =
Detail
U - Per-user static
1 Total IPX Servers

Table ordering is based on routing and server info
```

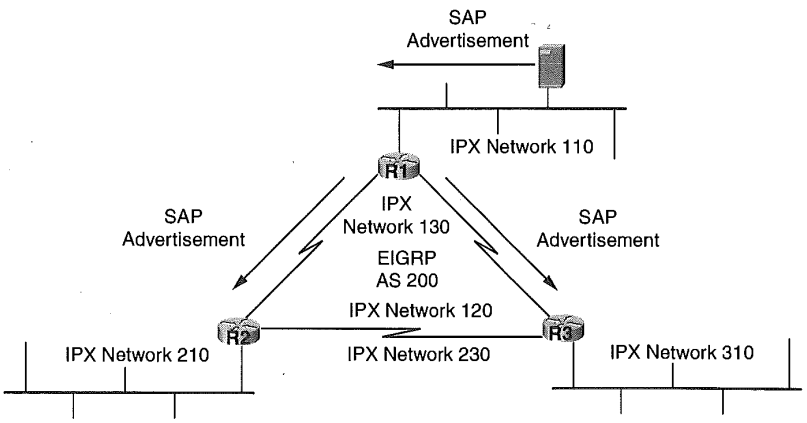


Figure 23-9 IPX SAP advertisements.

```
 Type Name      Net  Address Port  Route Hops Itf
E  4 FileServer_1   110.0000.0c00.1234:0251 2195456/01 2 Se1
R3#
```

Now we can configure EIGRP on the WAN interfaces to reduce the bandwidth used to advertise our lone server. Use the ***ipx sap-incremental eigrp*** command on each router's serial interface:

```
R1(config)#interface s0
R1(config-if)#ipx sap-incremental eigrp 200
R1(config-if)#interface s1
R1(config-if)#ipx sap-incremental eigrp 200

R2(config)#interface s0
R2(config-if)#ipx sap-incremental eigrp 200
R2(config-if)#interface s1
R2(config-if)#ipx sap-incremental eigrp 200

R3(config)#interface s0
R3(config-if)#ipx sap-incremental eigrp 200
R3(config-if)#interface s1
R3(config-if)#ipx sap-incremental eigrp 200
```

IPX Static Routes

Static routing is not as easy to implement in an IPX environment as it is with other protocols such as IP. The reason for this is that several services provided by Novell NetWare servers and third-party products require the action of the IPX RIP routing protocol for their functions. One reason for using static routes is to overcome weaknesses of the IPX RIP routing protocol. If you must use IPX RIP, rather than NLSP or EIGRP, perhaps to support another vendor's routers in your IPX internetwork, you can configure static routes and redistribute them into the IPX RIP routing protocol. Redistribution of IPX routing information is the subject of our next section.

The command to create a static IPX RIP route to the network ***BADC0FFEE*** is

```
R1(config)#ipx route BADC0FFE 110.0000.0c00.1234 6 1
```

In this command, ***110.0000.0c00.1234*** represents the next-hop gateway for the route, the parameter ***6*** is the ipx delay or "tick" count associated with the route to ***BADC0FFEE,*** and ***1*** is the hop count. You can examine the result of this configuration using the ***show ipx route*** command:

```
R1#
```

```
00:39:51: %SYS-5-CONFIG_I: Configured from console by console
R1#show ipx route
Codes: C - Connected primary network, c - Connected secondary network
 S - Static, F - Floating static, L - Local (internal), W - IPXWAN
 R - RIP, E - EIGRP, N - NLSP, X - External, A - Aggregate
 s - Seconds, u - Uses, U - Per-user static

7 Total IPX routes. Up to 1 parallel paths and 16 hops allowed.

No default route known.

C  110 (NOVELL-ETHER), Et0
C  120 (HDLC),  Se0
C  130 (HDLC),  Se1
E  210 [2195456/1] via  130.0010.7b3a.d4ae, age 00:31:54,
     1u, Se1
E  230 [2681856/0] via  130.0010.7b3a.d4ae, age 00:31:54,
     1u, Se1
E  310 [2195456/1] via  120.0010.7b3a.d4b3, age 00:31:54,
     1u, Se0
S BADC0FFE[06/01] via  110.0000.0c00.1234,  Et0
R1#
```

The IPX default route is a special case of a static route. The default route in general represents the path to a device with more complete routing information than our own router. You configure an IPX default route using the ***ipx route*** statement, as in

```
R1(config)#ipx route default 110.0000.0c00.1234 6 1
```

IPX Route Redistribution

Route redistribution refers to the process by which one routing protocol receives information from another protocol on the same router. In general, IPX routing protocols redistribute information by default. For instance, the two static routes configured on the previous section were automatically redistributed into IPX EIGRP for us. Examining R3's IPX routing table, we see that the default route configured on R1 is now the default route on R3 and the route to network ***BADC0FFEE*** has likewise been redistributed into IPX EIGRP:

```
R3#show ipx route
```

```
Codes: C - Connected primary network, c - Connected secondary network
 S - Static, F - Floating static, L - Local (internal), W - IPXWAN
 R - RIP, E - EIGRP, N - NLSP, X - External, A - Aggregate
 s - Seconds, u - Uses, U - Per-user static

8 Total IPX routes. Up to 1 parallel paths and 16 hops allowed.

Current default route is:

E FFFFFFFE [267008000/2] via  120.0010.7b3a.a8b3, age 00:04:37,
     1u, Se1

C  120 (HDLC),  Se1
C  230 (HDLC),  Se0
C  310 (NOVELL-ETHER), Et0
E  110 [2195456/1] via  120.0010.7b3a.a8b3, age 00:45:53,
     7u, Se1
E  130 [2681856/0] via  230.0010.7b3a.d4ae, age 00:45:54,
     1u, Se0
E  210 [2195456/1] via  230.0010.7b3a.d4ae, age 00:45:54,
     1u, Se0
E BADC0FFE [267008000/2] via  120.0010.7b3a.a8b3, age 00:19:57,
     1u, Se1
R3#
```

Virtually the only time you will have to manually configure redistribution in an IPX environment is when you wish to have EIGRP and NLSP share routing information. The command to accomplish this configuration would be

```
router(config)#ipx router eigrp 1
router(config-ipx-router)#redistribute nlsp
router(config-ipx-router)#ipx router nlsp
router(config-ipx-router)#redistribute eigrp 1
```

To reduce the possibility of introducing a routing information loop through redistribution, you will want to ensure that only one device in your IPX internetwork is configured to perform this sort of redistribution.

Bridging IPX

In many simple Novell environments, you may not need to introduce the complexities we have discussed involving the routing of IPX traffic. For

instance, if your organization has offices in two buildings with only one NetWare 3.11 server, you may consider bridging IPX traffic across a WAN interface. The router will not need to become involved in the IPX SAP process, nor will it send periodic routing updates. In this section, we will discuss the configuration necessary to configure Cisco routers to bridge IPX traffic.

Transparent Bridging

The term "transparent bridging" refers to the process of receiving a link-layer frame of data on one interface and retransmitting it unchanged on another interface of the same type. Except for the buffering delay, the bridging device is invisible or "transparent" to the hosts that are communicating through it. To configure transparent bridging, we must first start an instance of the Spanning-Tree Protocol (STP) to prevent topological loops in our bridged network. Use this syntax to start STP:

```
router(config)#bridge-group 1 protocol ieee
```

Configuring transparent bridging for IPX requires that we identify interfaces that participate in the STP and ensure that the interfaces are not configured to route IPX. Use this syntax to accomplish these tasks:

```
router(config)#interface e 0
router(config-if)#no ipx network
router(config-if)#bridge-group 1
router(config-if)#interface e 1
router(config-if)#no ipx network
router(config-if)#bridge-group 1
```

Encapsulated Bridging

Encapsulated bridging is used when unlike interfaces are configured as part of the same bridge group. When a link-layer frame of data arrives on an interface of one type, such as an Ethernet interface, it is encapsulated in the link-layer protocol appropriate to the outgoing interface, perhaps a serial interface. When the frame is received by another bridge, the serial link header must be removed before the frame is transmitted onto another Ethernet interface in the bridge group. Figure 23-10 illustrates a common encapsulated bridging configuration.

You do not need to specify that encapsulated bridging should be used. It is the default behavior when unlike interface types belong to the same STP. Use these commands to configure the network of Figure 23-10 for encapsulated bridging:

```
R1(config)#interface e 0
R1(config-if)#no ipx network
R1(config-if)#bridge-group 1
R1(config-if)#interface s 0
R1(config-if)#no ipx network
R1(config-if)#bridge-group 1

R2(config)#interface e 0
R2(config-if)#no ipx network
R2(config-if)#bridge-group 1
R2(config-if)#interface s 1
R2(config-if)#clock rate 64000
R2(config-if)#no ipx network
R2(config-if)#bridge-group 1
```

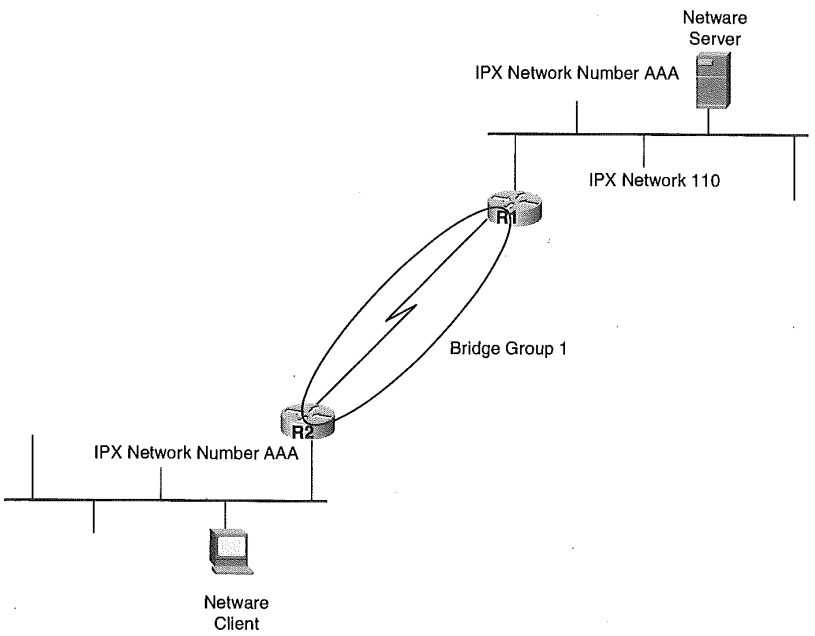
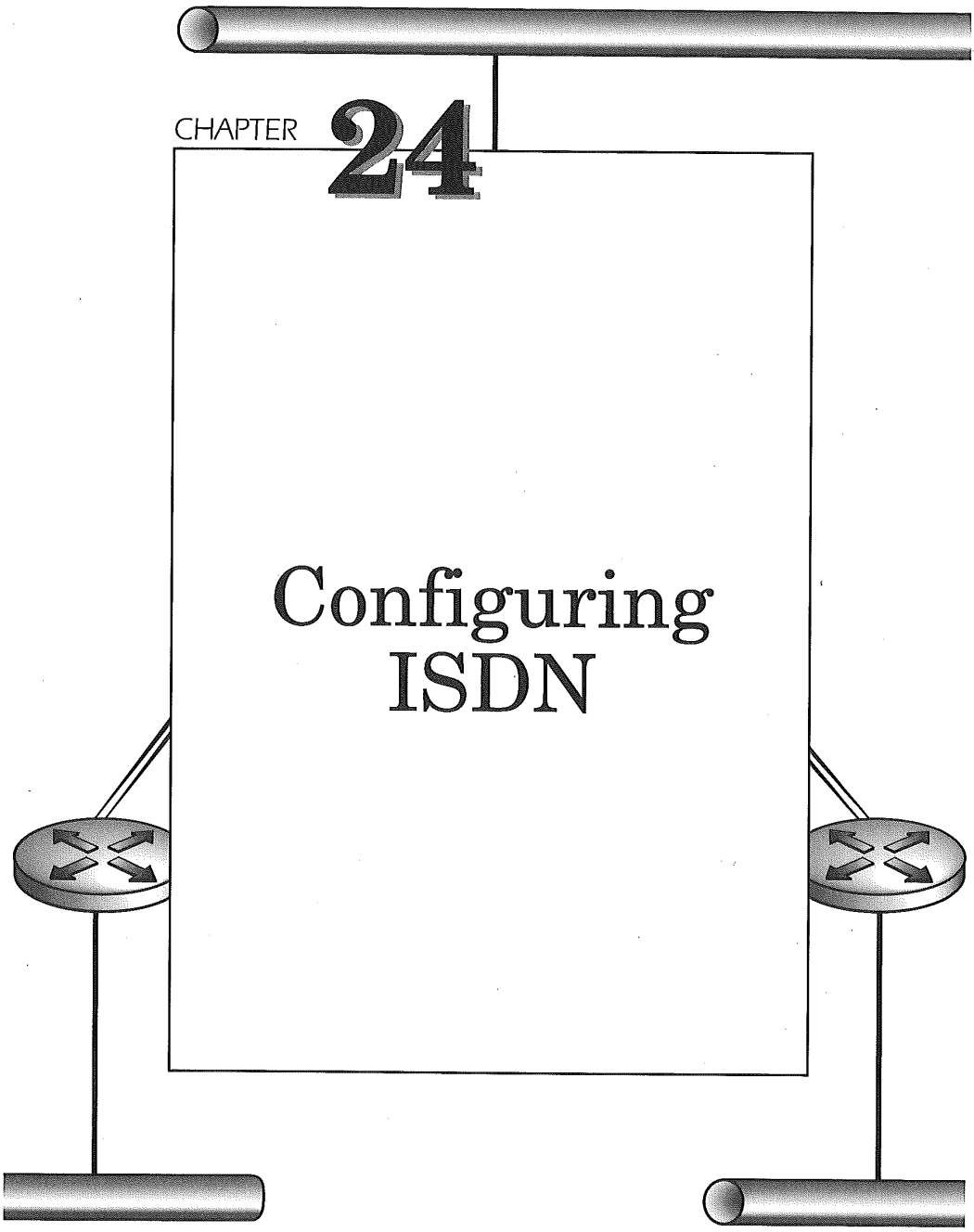


Figure 23-10
IPX encapsulated bridging.

CHAPTER 24

Configuring ISDN



Technologies for remote connectivity were once based on analog systems, yet today it has progressed towards digital operations. Digital technology has become the choice for high-speed, cost-effective telecommunications. The Integrated Services Digital Network (ISDN) is a leading digital technology used for providing temporary remote access to corporate networks.

Situations may exist in which a location needs to communicate with another site for only small intervals. A site may need, for instance, to transfer a large file at a specific time every week or use a dialup connection as backup to the dedicated link. Another use for temporary remote connectivity is telecommuting or remote technical support to enterprise networks. In situations like these, ISDN dial connectivity provides a solution.

ISDN eliminates any analog signal conversion in the transmission process. The "call" is digital from end to end. ISDN has two different types of services:

- Basic Rate Interface (BRI) service
- Primary Rate Interface (PRI) service

BRI service provides two ISDN Bearer (B) channels and one ISDN Data (D) channel. The B channels are used for data, voice, video, or fax transmission. The D channel provides out-of-band signaling that sets up and tears down the calls. PRI service provides 23 B channels and one D channel. The transmission of each B channel is either 56 Kbps or 64 Kbps. The transmission rate is dependent on the type of ISDN switch used at the central office of the ISDN provider. Cisco IOS and selected interface processors support both BRI and PRI services.

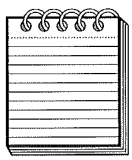
Configuring BRI Service

BRI is typically used by the remote location, requiring access to the central location of an enterprise or corporate network. Establishing the BRI configuration on Cisco routers is done in two basic steps. The first step is to define the type of ISDN switch connecting the router to the ISDN provider central office. The ISDN switch types supported by Cisco IOS are found in Table 24.1. Specifying the type of ISDN switch used for the connection is done by entering the `isdn switch-type` global configuration command. The format of this command is

```
isdn switch-type bri-switch-type
```

The configuration is for an ISDN circuit, so the `isdn` keyword is used. To specify the exact switch type, the keyword `switch-type` is followed by the *bri-switch-type* variable value. The *bri-switch-type* values available are found in Table 24.1 below.

here may be some instances where a router may have multiple ISDN switch types connected. For instance, in a situation where the temporary access does not warrant a dedicated line however, the information received form the location is vital to business processes, two ISDN providers connect to the same router. The ISDN providers are using a different ISDN switch type. In this case, the `isdn switch-type` interface command can be used on the ISDN interface configuration overriding the global configuration command. The global `isdn switch-type` configuration command must be set prior to configuring the `isdn switch-type` interface command. Allowing the `isdn switch-type` interface command enables the router to connect to multiple switch types concurrently.



NOTE: *The global* `isdn switch-type` *configuration command takes effect only during the very first initialization of ISDN on the router or when the router is reloaded. Subsequent changes to the global* `isdn switch-type` *after the initial configuration will not modify the switch type without reloading the router. If the router cannot be reloaded and the switch type must be altered to enable ISDN connectivity the* `isdn switch-type` *interface command should be used.*

Associated with defining the ISDN switch type is determining the occurrence of Layer 2 ISDN terminal endpoint identifier (TEI) negotiation. The terminal endpoint identifier negotiation is the process of negotiating an identifier that is obtained from the ISDN network. This can be done on power up of the router or when the first call is initiated. The default is when the router is powered up. TEI negotiation is done in instances when the switches release layer 1 or 2 when there are no calls present. The command for designating TEI negotiation is:

```
isdn tei (first-call | powerup)
```

The keyword `isdn` is used to designate the ISDN command set. The `tei` keyword indicates to the IOS that terminal endpoint negotiation is being specified rather than defaulting to `powerup`. Setting the keyword first-call s used to say that terminal endpoint identifier will be negotiated either at power up of the router or on the first call.

The ISDN provider will provide Service Profile Identifiers (SPIDs) for the two B channels on your ISDN circuit. These SPIDs are numbers that

Table 24–1 BRI ISDN switch types and there associated keywords.

bri-switch-type variable value	Switch Names and Their Locations
basic-5ess	AT&T basic rate switch (North America)
basic-dms100	Northern Telecom DMS100 basic rate switch (North America)
basic-ni	National ISDN switches (North America)
basic-ts103	TS013 switch (Australia)
basic-1tr6	1TR6 ISDN switch (Germany)
basic-net3	NET3 ISDN, (Norway and New Zealand)
ntt	NTT ISDN switch (Japan)
vn3	VN3 and VN4 ISDN switches (France)

usually resemble telephone numbers but are not at all related to a phone number. Two SPIDs are provided. Each SPID represents a B channel on the ISDN circuit. Configuring the SPID numbers in the Cisco router configuration is dependent on the type of switch used by the ISDN provider. A DMS-100 or a National ISDN-1 (NI-1) switch requires the router to have the SPIDs defined. If the ISDN provider is using an AT&T 5ess switch the router can learn them dynamically and hence are not required.

NOTE: *Even though the AT&T 5ess switch does not require the SPID to be defined in the router it may be prudent to do as a means of documentation for troubleshooting purposes.*

The SPID numbers are configured under the interface definition for the ISDN port. The format of the command is:

```
isdn spid1 spid-number [ldn]
isdn spid2 spid-number [ldn]
```

The `isdn spid1` and `isdn spid2` interface commands identify the SPID number being defined. The `isdn spid1` interface command defines the SPID number assigned to B channel 1. The `isdn spid2` interface command defines the SPID number assigned to B channel 2. The *spid-number* value must be 10 digits long. A value specified for the optional variable named *ldn* is used as a local directory number. This may be used if the router is going to answer

calls to a second directory number. With the SPIDs in place calls can now be received and initiated by the router so protocols need to be configured.

Point to Point connections are desired in a dial-up environment. By default the encapsulation of a ISDN B channel is HDLC. Since routing protocols and network layer protocols are the choice PPP encapsulation is the preferred method. To enable PPP encapsulation on a BRI use the following interface configuration command is issued:

```
encapsulation ppp
```

The `encapsulation` keyword signifies that the interface will do some type of encapsulation of other protocol packets. The `ppp` keyword indicates that packets will be encapsulated in Point to Point Protocol over the ISDN connection.

Now that there desired encapsulation is configured the protocols that will be encapsulated are configured. For a router to start an ISDN call it must see interesting packets that will trigger a call. The command that defines interesting packets and who to call is as follows:

```
dialer map protocol next-hop-address name hostname [spc] [speed 56
| 64] [broadcast]
            dial-string[:isdn-subaddress]
```

The `dialer map` is what causes the router to do an actual ISDN call. The value of the *protocol* variable identifies the type4 of networking protocol being transported over the connection. The possible values are: `appletalk`, `bridge`, `clns`, `decnet`, `ip`, `ipx`, `novell`, `snapshot`, `vines`, and `xns`. The *next-hop-address* variable value is the protocol address of the partner ISDN device being contacted. Use of the bridge protocol value negates the specification of the next-hop-address variable value. The `name` keyword and its associated hostname variable value identifies and is used as authentication of incoming calls. For ISDN this value is the calling line identification number of the remote ISDN device. The `spc` keyword is useful only in Germany where for connections supporting the semipermanent connection feature between the customer premise equipment and the central office switch. The `speed` keyword specifies either `56` Kbps or `64` Kbps transmission rates for each B channel of the ISDN circuit. The default value is 64. The broadcast keyword is used if any of the supporting network protocols use broadcasts for communications between resources. Finally, the *dial-string* variable specifies the calling number used to connect to the other ISDN device. The optional *subaddress* variable is used if the router dials into a multipoint ISDN device.

To control who can dial into an interface a BRI is assigned a dialer-group which associates it to a dialer-list. The dialer-group interface command syntax is as follows:

```
dialer-group group-number
```

The `dialer-group` keyword is used to identify the dialer-list definitions that are assigned to the interface. The *group-number* variable ranges from 1 to 10 and must match a `dialer-list` *group-number*. Using the `dial-group` interface command allows the router administrator to apply the rules of a single dialer-list to any ISDN interface.

The `dialer-list` global configuration command is configured as follows:

```
dialer-list dialer-group list access-list-number
```

The `dialer-group` is associated to an access list so only certain protocols or users can communicate over the BRIs in the specific `dialer-group`. The `dialer-list` keyword is used to configure the dialer list followed by the `dialer-group` being associated to the list. Then the dialer-list is also associated to an access list that is used to enable specific communication for the BRI being configured.

If additional security is wanted then a screening process can be added for further verification. On the BRI interface configuration mode the following command can be entered to enable screening of incoming calls:

```
isdn caller number
```

This designates that the `isdn caller` having a *number* that matches the one given on the interface configuration is allowed to connect to the router. If the number does not match then the router will not accept the call and disconnect.

These are the basic commands for configuring a BRI. The following section reviews the basic configuration requirements for PRI service.

Configuring Primary Rate Interface (PRI) Service

The PRI service also requires the definition of the ISDN switch type. This was discussed fully under BRI and does not need to be explained again. However, the variable values possible for the PRI interfaces on the router are limited to the ISDN switch types listed in Table 24.2 below.

Table 24–2 PRI ISDN switch types and there associated keywords.

pri-switch-type variable value	Switch Names and Their Locations
pri-4ess	AT&T primary rate switch (United States)
pri-5ess	AT&T primary rate switch (United States)
pri-dms100	Northern Telecom DMS100 primary rate switch (North America)
pri-ntt	NTT ISDN PRI switch (Japan)
pri-net5	European ISDN PRI switch
none	No switch defined

The format for the global and interface configuration commands specifying the PRI ISDN switch is:

```
isdn switch-type pri-switch-type
```

The *pri-switch-type* variable value must be one of the values listed in Table 24.2 above.

The PRI ISDN service uses a full T1/E1 and as such requires the Cisco router to be able to distinguish the various channels available on the circuit. The Cisco 7x00 series utilizes the MIP interface to provide the functionality of discerning the different channels on the circuit. The instance of the MIP is configured by configuring the global `controller` interface command with the following format:

```
controller {t1 | e1} slot/port | number
```

The `controller` keyword identifies the use of a MIP card to support the ISDN PRI service. The `t1` keyword indicates that the service is using the T1 specifications. Specifying the `e1` keyword indicates the PRI service is using the E1 (European) specifications. The *slot[/port]* variable value identifies the MIP position in the Cisco 7x00 series router. The number variable value identifies the MIP position in the Cisco 4x00 series routers.

Under the controller interface configuration there are two subsequent options that are usually defined to enable the MIP to properly send and receive data on the T1/E1 connection supporting the ISDN. The first of these is the framing type. The specification of the framing type is accomplished using the framing interface configuration command with the following formats:

```
framing {sf | esf}
framing {crc4 | no-crc4} [australia]
```

The `framing` keyword under the `controller` definition identifies the type of framing being used on the line. The `sf` and `esf` values identify the use of a T1 line. Specifying `sf` indicates the line is using super frame encoding for the data. The `esf` instructs the MIP to apply extended super frame encoding for the data transmitted on the T1 circuit. The second form of the framing interface configuration command is used for E1 circuits. The use of the `australia` optional value instructs the MIP to use the E1 framing type employed in Australia.

The second option coded on under the controller interface configuration identifies the type of line-coding used on the circuit. The line-code type is specified using the linecode interface configuration command with the following format:

```
linecode {ami | b8zs | hdb3}
```

The values for the `linecode` interface command identify the type of encoding performed on the line. The alternate mark inversion (`ami`) is the default for T1 lines but is available for use on the T1 or E1 lines. The binary 8 zero substitution (`b8zs`) type is valid only on T1 lines. High-density bipolar 3 (`hdb3`) is the default for E1 lines and valid only on E1 lines. The `linecode` interface configuration command is used when the T1 or E1 is fractional, using each channel as an individual line, versus using the entire bandwidth. Check with the ISDN PRI service provider on which value to code.

A final optional controller interface configuration subcommand identifies which channels (timeslots) of the T1/E1 circuit will be in use. This is specified using the `pri-group` interface command in the following format:

```
pri-group [timeslots range]
```

The range variable value for T1 circuits is anywhere from 1-24. E1 circuits may range from 1-31. The range defined establishes which channels of the T1 are used for ISDN connections.

The controller definitions must be associated with an interface configuration. The `interface serial` global configuration commands following identify how the T1/E1 are channelized in support of ISDN. The format of the `interface serial` global configuration command is:

```
interface serial slot[/port]:timeslot
interface serial number:timeslot)
```

The `interface serial` *slot*[/*port*] command is the format used on Cisco 7x00 series routers. The `interface serial` *number* format is used on the

Cisco 4x00 series routers. The *:timeslot* value identifies which channel is acting as the D channel for the PRI service. For T1 circuits it must always be `:23` and E1 it must always be `:15`. (NOTE: the ':' in the timeslot value is required).

Dial-On-Demand routing (DDR) with ISDN

Now that the basics for configuring BRI and PRI ISDN connections have been discussed we move to implementing ISDN using Cisco IOS Dial-on-Demand Routing (DDR) feature. Figure 24.1 illustrates a typical ISDN solution for telecommuters and remote technical troubleshooting found in many of corporations. Router R1, is a Cisco 7500 series router providing PRI service for various remote locations. Router R2 is a telecommuter and Router R3 is a remote technical support personnel. The two remote locations are using Cisco 2500 series routers.

The following router configurations apply to Figure 24.1:

Router R1 applied to Figure 24.1:

```
hostname R1-isdn
!
username R2-isdn password 7 68280FDE321
username R3-isdn password 7 879BA478928
isdn switch-type primary-dms100
!
interface ethernet 0
 ip address 10.10.4.1 255.255.255.0
!
controller t1 1/0
 framing esf
 linecode b8zs
 pri-group timeslots 1-24
!
interface serial 1/0:23
 ip address 10.10.9.1 255.255.255.0
 encapsulation ppp
 dialer idle-timeout 300
 dialer map ip 10.10.9.2 name R2-isdn speed 56 2125554040
 dialer map ip 10.10.9.3 name R3-isdn 5411
 dialer-group 1
!
router igrp 10
 network 10.0.0.0
 redistribute static
!
! route to R3-isdn
ip route 10.10.13.0 255.255.255.0 10.10.9.3
```

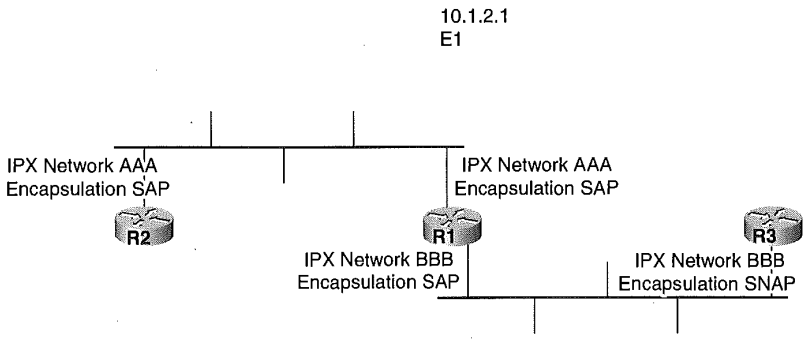


Figure 24–1
DDR using ISDN to a central site router.

```
! route to R2-isdn
ip route 10.10.14.0 255.255.255.0 10.10.9.2
!
access-list 101 deny igrp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!NTP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 123
!SNMP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 161
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

The host name of the central site router at the corporate data center is R1 as denoted by the hostname command in the router configuration of router R1. The PRI router must have a list of usernames defining what remote resources can access the router through the ISDN dial up connection. The username values must be the hostnames of the remote routers connecting through the ISDN PRI service. The MIP controller card is installed in slot 1 and port 0 of the MIP is being used for this T1 connection.

This is specified on the controller configuration command. The controller is to use extended super frame and b8zs line-coding for sending and receiving data on the T1 circuit. The entire T1 is being used and this is denoted by the pri-group timeslot range of 1-24. This enables the single T1 line to support up to 23 B channel connections at 64 Kbps each because the R1 router is connected to a DMS-100 ISDN switch. If it were connected to a 5ess switch it would only support 56 Kbps on each B channel. The Point-to-Point Protocol (PPP) encapsulation method is used for the ISDN transport of data. The two dial-map commands of the interface serial 1/0 configuration identify the ISDN connections to router R2 and R3. The R2 outer connects to a 5ess switch and hence we must specify the 56 Kbps speed on the dialer-map definition to router R2. Router R3 in the example connects to the same DMS-100 used by R1. In this case, because they are located on the same switch only the last four digits of the dial-string are necessary in the dialer-map command representing router R2. The PRI is associated with dialer-group 1 which identifies dialer-list 1 which points to the 101 access list to determine interesting or uninteresting packets used by DDR in determining connectivity to the remote locations. The access list 101 uses an extended access list to deny (mark as uninteresting) IGRP updates, Network Time Protocol (NTP) and SNMP packets. All other IP packets however are of interest and will case DDR to initiate and ISDN connection to the remote routers R2 and R3 if a connection does not already exist. DDR is made ware of the interesting IP packets by use of static routes configured and distributed to the IGRP routing process within router R1.

R3 router configuration applied to Figure 24.1:

```
hostname R3-isdn
!
username R1-isdn password 7 98765FEA398
isdn switch-type basic-dms100
!
interface ethernet 0
 ip address 10.10.13.1 255.255.255.0
 no mop enabled
!
interface bri 0
 ip address 10.10.9.3 255.255.255.0
 encapsulation ppp
 no ip route-cache
 isdn spid1 212555541101 5551111
 isdn spid2 212555541102 5551112
 dialer idle-timeout 300
 dialer map ip 10.10.9.1 name R1-isdn 1111
 dialer map ip 10.10.9.1 name R1-isdn 1112
 dialer-group 1
!
ip route 10.10.0.0 255.255.0.0 10.10.9.1
```

```
!
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 177
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

Router R3 connects to the DMS-100 and therefore requires the definitions of the SPIDs for the BRI service connection. The username specified is the hostname of router R1. Two numbers are provided on router R3 to contact router R1. If the first `dialer-map` *next-hop-address* connection fail the second number is used to connect to router R1. The values used in the SPID definitions identify 212 as the area code, 555 is the local exchange and 5411 is the station ID. The last two digits on both SPIDs represents the terminal identifier assigned by the ISDN service provider.

Router R2 configuration applied to Figure 24.1:

```
hostname R2-isdn
!
username R1-isdn password 7 ABE41688
isdn switch-type basic-5ess
!
interface ethernet 0
 ip address 10.10.14.1 255.255.255.0
 no mop enabled
!
interface bri 0
 ip address 10.10.9.2 255.255.255.0
 encapsulation ppp
 no ip route-cache
 bandwidth 56
 dialer map ip 10.10.9.1 name R1-isdn speed 56 12125554442
 dialer-group 1
!
ip route 10.10.0.0 255.255.0.0 10.10.9.1
!
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 177
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

Router R2 connects its ISDN BRI line to a different central office and thus uses a different type of ISDN switch. In this case R2 is connected to an AT&T 5ESS ISDN switch. Because R2 connects to a 5ESS it does not have to have the SPIDs defined in the router configuration. The speed 56 value coded on the dialer-map command sets the ISDN BRI line speed to 56 Kbps.

Using Caller ID

The Cisco IOS can use the Calling Line Identification (CLID) number received from an ISDN switch and use this number to verify the connecting router. The connection between router R1 and R3 can make use of this because they are connected to the same switch. A slight modification to router R3 and R1 will enable the use of the CLID feature. The following configurations apply the CLID feature to router R1 and R3.

Router R1 configuration using the CLID feature:

```
hostname R1-isdn
!
username R2-isdn password 7 68280FDE321
username R3-isdn password 7 879BA478928
isdn switch-type primary-dms100
!
interface Ethernet 0
 ip address 10.10.4.1 255.255.255.0
!
controller t1 1/0
 framing esf
 linecode b8zs
 pri-group timeslots 1-24
!
interface serial 1/0:23
 ip address 10.10.9.1 255.255.255.0
 dialer idle-timeout 300
 dialer map ip 10.10.9.2 name R2-isdn speed 56 2125554040
 dialer map ip 10.10.9.3 name 5555411 1111
 dialer-group 1
!
router igrp 10
 network 10.0.0.0
 redistribute static
!
! route to R3-isdn
ip route 10.10.13.0 255.255.255.0 10.10.9.3
! route to R2-isdn
ip route 10.10.14.0 255.255.255.0 10.10.9.2
!
access-list 101 deny igrp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!NTP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 123
!SNMP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 161
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

The name keyword on the dialer-map for R3 specifies the actual calling line identification number returned by the switch.

```
Router R3 configuration using the CLID feature:

hostname R3-isdn
!
username R1-isdn password 7 98765FEA398
isdn switch-type basic-dms100
!
interface Ethernet 0
 ip address 10.10.13.1 255.255.255.0
 no mop enabled
!
interface bri 0
 ip address 10.10.9.3 255.255.255.0
 no ip route-cache
 isdn spid1 212555541101 5551111
 isdn spid2 212555541102 5551112
 dialer idle-timeout 300
 dialer map ip 10.10.9.1 name 5551111 1111
 dialer map ip 10.10.9.1 name 5551112 1112
 dialer-group 1
!
ip route 10.10.0.0 255.255.0.0 10.10.9.1
!
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 177
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

Just as in the central site router R1, the router R3 uses the name keyword and value to identify the actual number being returned form the switch.

ISDN Callback

The callback feature enables the billing for the ISDN BRI connection to be charged to the central site by having the central site call back the remote location after an initial connection. This assists in managing the billing aspect of ISDN BRI telecommuting and remote technical support.

The following router configurations apply the callback feature to routers R2 and R1.

Router R2 callback configuration:

```
hostname R2-isdn
!
```

```
username R1-isdn password 7 ABE41688
isdn switch-type basic-5ess
!
interface ethernet 0
 ip address 10.10.14.1 255.255.255.0
 no mop enabled
!
interface bri 0
 ip address 10.10.9.2 255.255.255.0
 encapsulation ppp
 no ip route-cache
 bandwidth 56
 dialer map ip 10.10.9.1 name R1-isdn speed 56 12125554442
 dialer-group 1
 ppp callback request
 dialer hold-queue 100 timeout 20
!
ip route 10.10.0.0 255.255.0.0 10.10.9.1
!
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 177
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

The ppp callback request command indicates to the router that when it contacts the PRI router R1 it will request the R1 call back. The dial-hold-queue command indicates that up to 100 packets of data will be held in the outbound queue while waiting for router R1 to return the call. Any queued packets are dropped after the timeout value of 20 seconds is reached.

Router R1 configuration in support of the callback feature to router R2:

```
hostname R1-isdn
!
username R2-isdn password 7 68280FDE321
username R3-isdn password 7 879BA478928
isdn switch-type primary-dms100
!
map-class dialer class1
 dialer callback-server username
!
interface Ethernet 0
 ip address 10.10.4.1 255.255.255.0
!
interface serial 1/0:23
 ip address 10.10.9.1 255.255.255.0
 dialer idle-timeout 300
 dialer map ip 10.10.9.2 name R2-isdn speed 56 class class1 1
 dialer map ip 10.10.9.3 name 5555411 1111
 dialer-group 1
 ppp callback accept
 dialer callback-secure
 dialer enable-timeout 1
 dialer hold-queue
!
```

```
router igrp 10
 network 10.10.0.0
 redistribute static
!
! route to R3-isdn
ip route 10.10.13.0 255.255.255.0 10.10.9.3
! route to R2-isdn
ip route 10.10.14.0 255.255.255.0 10.10.9.2
!
access-list 101 deny igrp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!NTP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 123
!SNMP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 161
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

The map-class global configuration command enables the IOS to establish a Quality of service (QoS) for the connection to router R2 by associating a static map. The dialer keyword of the map-class command identifies the dialer-map being used. The class1 value defined on the command is user-defined and maps a class to the dialer-map. The dialer-map interface command now includes a class keyword ant the name of the class to which the dialer-map is now associated. The name keyword on the dialer-map is used to provide the dial-string to router R1 for the callback. Added security is built into this configuration through the use of the dialer callback-secure interface configuration command. Using this command ensures that the R1 router will only call back R2 if it has a dial-map command with a defined class for router R2 defined. The dialer enable-timeout interface command determines the amount of time in seconds to wait prior to disconnecting the initial call and beginning the callback.

SNAPSHOT Routing with ISDN

Connection time on ISDN networks can become costly especially if bandwidth is being utilized for routing protocols versus the delivery of end user data. A means for reducing routing protocol consumption of valuable ISDN bandwidth is through the use of Snapshot routing. Using snapshot routing bandwidth is preserved for an extended period of time for the transmission of end user data without the over head of routing protocols. Snapshot is effective only in connections that use distance-vector routing protocols. The

snapshot routing feature is available starting with IOS Release10.2+ and only with the full Enterprise IOS Release 11.0(3) or higher.

Snapshot routing enables distance-vector routing protocols (i.e., IP RIP, IP IGRP, IPX RIP/SPX, AppleTalk RTMP and Banyan VINES RTP) to continue to function dynamically for a specified period of time (active period). After the active period a quiet period (up to 65 days) may be configured to suppress the normal routing updates over the ISDN connection. In essence, the routing tables of the connecting routers using snapshot routing have "picture" of the routing tables at a specific moment in time. Future routing updates are held until the quiet time has expired and the routers enter into another active period. The configured active period time must be long enough to provide for multiple routing updates to ensure a complete routing table be built in the routers participating in the snapshot routing protocol. Table 24.3 lists the default routing update intervals for the aforementioned distance-vector routing protocols.

The importance of understanding these routing update intervals is because many ISDN connections are based on interesting packets. For example, if IP is defined as an interesting packet for ISDN connectivity, then every 90 seconds the IGRP process within the router will initiate an ISDN call to send the IGRP routing updates. This will occur even if the end user location does not require connectivity. This becomes cost inefficient. Snapshot routing quiet period enables the router administrator to place controls on the routing update process.

Snapshot routing is recommended only in hub and spoke or temporary point-to-point topologies. Snapshot routing is based on a client-server model. The client (spoke or remote router) sets the quiet period time and the server (hub or central router) is defined to accept incoming snapshot connections from one or more snapshot client routers.

A router is enabled as the client for snapshot routing when the `snapshot client` interface configuration command is specified. The format of the `snapshot client` interface configuration command is:

```
snapshot client active-time quiet-time [suppress-statechange-
updates] [dialer]
```

The `snapshot client` keyword specified on an interface initiates the snapshot routing process on the router. Specifying the `snapshot client` keyword indicates to the IOS that this router will act as a client connecting to a snapshot server. The *active-time* variable value may range from 5 to 100 and defines the number of minutes the client will exchange routing updates with the snapshot server. The *active-time* variable does not have a default. Typically the value chosen results in the routers exchanging routing tables

Table 24–3 Default routing update intervals for consideration in defining snapshot routing active period.

Routing Protocol	Default routing update Interval
IP RIP	30 seconds
IP IGRP	90 seconds
IPX RIP	60 seconds
IPX SAP	60 seconds
AppleTalk RTMP	10 seconds
Banyan VINES RTP	30 seconds

anywhere from 6-10 times during the active period. The value specified as the *active-time* variable in the **snapshot client** interface command must match the *active-time* variable value in the **snapshot server** command. Table 24.4 lists the three time periods used in snapshot routing protocol and their minimum and maximum times to assist in determining the appropriate *active-time* and *quiet-time* values. The *quiet-time* variable value ranges from 8 to 100000 measured in minutes. The high end results in 65 days for the quiet period. The *quiet-time* variable value does not default. A rule of thumb is to use the *active-time* variable value plus 3 for the *quiet-time* variable value. The **optional suppress-statechange-updates** keyword stops the router from exchanging routing updates whenever the line protocol state changes from down to up or from dialer spoofing to fully up. Using this keyword protects the bandwidth from being overrun with routing update exchanges. The optional **dialer** keyword is useful for directing the client router to contact the server router even though user data is not present to activate the connection.

A router is enabled as the server for snapshot routing when the **snapshot server** interface configuration command is specified. The format of the **snapshot server** interface configuration command is:

snapshot server *active-time* **[dialer]**

The **snapshot server** interface command initializes the server function of the snapshot routing protocol process in the router. The *active-time* variable value specifies the amount of time routing update exchange will occur between the server and any client. The range is 5 to 100 measured in minutes and follows the same guidelines as given for the *active-time* variable value on the **snapshot client** interface command. The optional **dialer** keyword enables the client router to contact the server even though user data is absent.

Table 24–4 Default routing update intervals for consideration in defining snapshot routing active period.

Period	Configurable	Minimum Length	Maximum Length
Active	Yes	5 minutes	100 minutes
Quiet	Yes	5 minutes	65 days
Retry	No	8 minutes	8 minutes

Snapshot routing used in conjunction with dial-on-demand routing (DDR) from a client requires the `dialer map snapshot` interface command is required. The format of this command is:

```
dialer map snapshot sequence-number dial-string
```

A `dialer map snapshot` interface command is required for each snapshot server the client calls during the active period. The *sequence-number* variable value ranges from 1 to 254 providing a unique identifier for the dialer map. The dial-string variable value is the telephone number of the snapshot server to contact during the active period over the ISDN connection.

Using the previous routing configuration example, router R1 (hub) is defined as the server and routers R2 and R3 are defined as the clients (spokes). The following configuration may be applied to the router R1 to enable snapshot routing:

```
hostname R1-isdn
!
username R2-isdn password 7 68280FDE321
username R3-isdn password 7 879BA478928
isdn switch-type primary-dms100
!
interface ethernet 0
 ip address 10.10.4.1 255.255.255.0
!
controller t1 1/0
 framing esf
 linecode b8zs
 pri-group timeslots 1-24
!
interface serial 1/0:23
 ip address 10.10.9.1 255.255.255.0
 encapsulation ppp
 dialer idle-timeout 300
 dialer map ip 10.10.9.2 name R2-isdn speed 56 2125554040
 dialer map ip 10.10.9.3 name R3-isdn 5411
 dialer-group 1
 snapshot server 5
 ppp authentication chap
```

```
!
router igrp 10
 network 10.0.0.0
!
access-list 101 deny igrp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!NTP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 123
!SNMP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 161
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

Router R1 no longer needs the static routes to router R2 and R3. The "5" following the snapshot server interface command indicates the active period is five minutes in length. The following configurations apply to router R2 and R3 in Figure 24.1:

Router R2 configuration using snapshot routing:

```
hostname R2-isdn
!
username R1-isdn password 7 ABE41688
isdn switch-type basic-5ess
!
interface ethernet 0
 ip address 10.10.14.1 255.255.255.0
 no mop enabled
!
interface bri 0
 ip address 10.10.9.2 255.255.255.0
 encapsulation ppp
 no ip route-cache
 bandwidth 56
 dialer map snapshot 1 name R1-isdn 12125554442
 dialer map ip 10.10.9.1 name R1-isdn speed 56 12125554442
 dialer-group 1
 snapshot client 5 43200 suppress-statechange-updates dialer
!
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 177
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

```
Router R3 configuration using snapshot routing:

hostname R3-isdn
!
username R1-isdn password 7 98765FEA398
isdn switch-type basic-dms100
```

```
!
interface ethernet 0
 ip address 10.10.13.1 255.255.255.0
 no mop enabled
!
interface bri 0
 ip address 10.10.9.3 255.255.255.0
 encapsulation ppp
 no ip route-cache
 isdn spid1 212555541101 5551111
 isdn spid2 212555541102 5551112
 dialer idle-timeout 300
 dialer map snapshot 1 name R1-isdn 1111
 dialer map snapshot 1 name R1-isdn 1112
 dialer map ip 10.10.9.1 name R1-isdn 1111
 dialer map ip 10.10.9.1 name R1-isdn 1112
 dialer-group 1
 snapshot client 5 43200 suppress-statechange-updates dialer
!
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 177
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101
```

Because snapshot routing is employed the static routes for R2 and R3 to reach R1 are no longer required. The dialer map snapshot command on the BRI0 interface of R2 and R3 creates a map using sequence number 1 to connect to router R1 for routing update exchange. The name keyword value is used during ppp authentication to verify the name of the connecting router hostname. The active period on the snapshot client command must match the snapshot server active period time in router R1. The snapshot client command also specifies the quiet period (in this case 43,200 seconds or 12 hours) will pass before a refresh of the routing table. The dialer keyword is used in this case to enable the client routers (R2 and R3) to connect to the server router even when end user traffic is absent on the ISDN link to update the routing tables. The suppress-statechange-updates keyword is coded to stop the router from exchanging routing updates while end user connections are being used. The coding of this keyword requires the dialer keyword be configured enabling routing updates to flow.

Using ISDN for dial backup

Many topologies involving ISDN utilize the speed of ISDN to provide dial backup to high speed T1, fractional T1 or 56-Kbps dedicated lines. Figure 24.2 illustrates such a configuration.

The dial backup feature is enabled using the **backup interface** interface configuration command on the serial interface definition for which ISDN dial backup is desired. The format of the **backup interface** interface configuration command is :

backup interface *interface-name*

The **backup interface** command identifies the dial up interface on the router that is to be used if data terminal ready (DTR) drops on the serial interface to which this **backup interface** command is defined. The *interface-name* variable value specifies the interface number, slot/port pair or slot/adapter/port grouping used as the dial backup connection for the serial interface. Routers will only support serial dial up and ISDN as the backup interfaces.

Once in dial backup mode it is important to mange and control the length of the dial backup connection and when to drop the dial backup connection once the original serial interface no longer drops DTR. This determination is accomplished using the backup delay interface configuration command. The format of the command is:

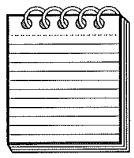
backup delay {*enable-delay* | **never**} {*disable-delay* | **never**}

Using the backup delay interface configuration command on the serial interface being backed up allows the router administrator to adjust when to drop the dial backup connection. The enable-delay variable value is measured in seconds and determines how long the serial interface DTR signal must be lost (down) prior to initiating dial backup. The disable-delay variable value indicates the number of seconds that must pass after the serial interface DTR is high (up) before terminating the dial backup connection. The values chosen for these two variables is important to consider especially in the case of "flapping" serial interfaces. Specifying never as the first positional parameter indicates that the dial backup will never occur automatically forcing manual intervention. Specifying never as the second positional parameter forces manual intervention to terminate the dial backup connection. Both of the delay variables default to 0 seconds making it prudent to provide some type of delay for both activation and deactivation. It is advisable to make the activation (enable-delay) a minimum of 0 second and the deactivation delay a minimum of six minutes to ensure a stable dedicated connection once again.

The following router configuration applies to Figure 24.2:

Router R1 configuration applied to Figure 24.2:

```
hostname R1
!
```



NOTE: *Issuing the shutdown EXEC configuration command on the serial interface will not cause the backup process to initiate since this is a controlled shut down. The backup process only takes place when the DTR drops from electrical signal failure.*

```
username R2 password 7 68280FDE321
isdn switch-type primary-dms100
!
interface ethernet 0
 ip address 10.10.4.1 255.255.255.0
!
controller t1 1/0
 framing esf
 linecode b8zs
 pri-group timeslots 1-24
!
interface serial 1/0:23
 ip address 10.10.9.1 255.255.255.0
 encapsulation ppp
 dialer idle-timeout 300
 dialer map ip 10.10.9.2 name R2-isdn speed 56 2125554040
 dialer-group 1
!
interface serial 2/0
 ip address 10.10.8.1 255.255.255.0
!
router igrp 10
 network 10.0.0.0
 redistribute static
! route to R2
ip route 10.10.14.0 255.255.255.0 10.10.9.2
!
access-list 101 deny igrp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!NTP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 123
!SNMP
access-list 101 deny udp 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255 eq 161
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
!
dialer-list 1 list 101

The following router configuration applies to router R2 in Figure
24.2:

Router R2 configuration:

hostname R2
!
username R1 password 7 ABE41688
isdn switch-type basic-5ess
!
```

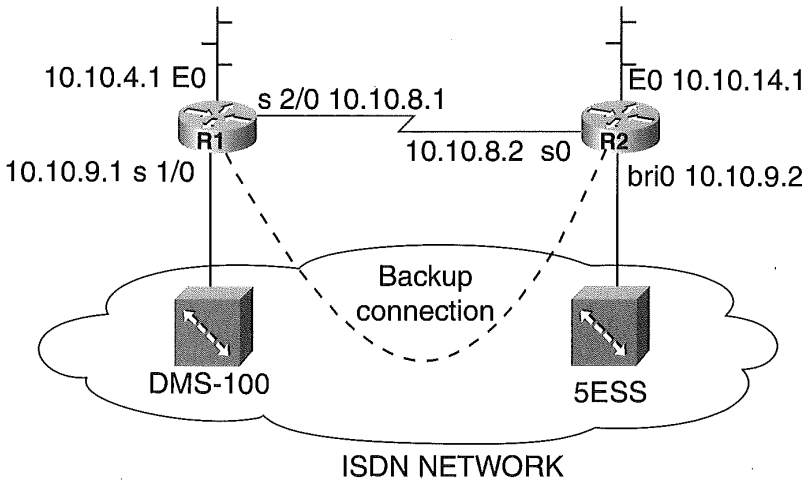


Figure 24–2
Dial backup of a dedicated line using ISDN

```
interface ethernet 0
 ip address 10.10.14.1 255.255.255.0
!
interface serial 0
 ip address 10.10.8.2 255.255.255.0
 backup interface bri0
 backup delay 1 360
!
interface bri 0
 ip address 10.10.9.2 255.255.255.0
 encapsulation ppp
 no ip route-cache
 bandwidth 56
 dialer map ip 10.10.9.1 name R1 speed 56 12125554442
 dialer-group 1
!
router igrp 10
 network 10.0.0.0
!
dialer-list 1 list 101
```

Router R2 connects to router R1 using a dedicated serial line. Failure of the dedicated serial line causes router R2 to initiate dial backup to router R1. The dial backup connection on router R2 is terminated only if six minutes passes while DTR is high on serial 0 of router R2. If DTR should drop within the disable-delay value of 360 seconds and then become high again the disable timer starts again.

Appendix A

Cisco IOS Platform Cross Reference and Naming Standard

Cisco Platform	*platform* Variable in IOS Name
5200	as5200
Cisco Advantage 1003	ca1003
Cisco Advantage 1005	ca1005
CiscoPro 1003, 1004	cpa1003
CiscoPro 1005	cpa1005
CiscoPro 2500	cpa25
CiscoPro 3620	cpa3620
CiscoPro 3640	cpa3640
CiscoPro 4500	cpa45
Communication server	*cs
ASM/MSM	cs3
cs500	cs500
1003,4	c1000
1005	c1005
1600	c1600
25xx, 3xxx, 5100, AP (11.2 and later only)	c2500
c2600	c25FX Fixed Frad platform
c2800	2600 Quake platform
c2900	Catalyst 2820
c3620	2910, 2950
c3640	3620
c3800	3640
c4000	3800
c4500	4000 (11.2 and later only)
c5rsm	4500, 4700
c5atm	Catalyst 5K RSP platform
c5200	Catalyst 5K ATM platform

Cisco Platform	*platform* Variable in IOS Name
c5300	5200 (11.2 and later only)
c5800	AS5300
c6400s	Nitro
c6400r	6400 NSP
c7000	6400 NRP
c7200	7000, 7010 (11.2 and later only)
igs	7200
gs3	IGS, 25xx, 3xxx, 5100, AP
gs7	Gateway server (AGS, AGS+)
gsr Gigabit Switch Router (12000)	Gateway server (7000, 7010)
LightSteam 1010	ls1010
Ardent multiservice Cisco 3810 platform	mc3810
Partner's platform n	p
Protocol translator	*pt
75xx, RSP7000	rsp
Universal Broadband Router 7200	ubr7200
Universal Broadband Router 900	ubr900
4000	xx
2500 (Media-specific image that supports only Ethernet, Token Ring, and X.25)	igsetx

feature Variable in IOS Name	Feature Description
a	APPN
a2	ATM
b	AppleTalk
boot	used for boot images

feature Variable in IOS Name	Feature Description
c	Comm-server/Remote Access Server (RAS) subset (SNMP, IP, Bridging, IPX, Atalk, DECnet, FR, HDLC, PPP, X.25, ARAP, tn3270, PT, XRemote, LAT) (non-CiscoPro)
c	CommServer lite (CiscoPro)
c2	Comm-server/Remote Access Server (RAS) subset (SNMP, IP, Bridging, IPX, Atalk, Decnet, FR, HDLC, PPP, X.25, ARAP, tn3270, PT, XRemote, LAT) (CiscoPro)
d	Desktop subset (SNMP, IP, Bridging, WAN, Remote Node, Terminal Services, IPX, Atalk, ARAP) (11.2 Decnet)
d2	Reduced Desktop subset (SNMP, IP, IPX, ATALK, ARAP)
diag	IOS-based diagnostics images
e	IPeXchange (no longer used in 11.3 and later), StarPipes DB2 Access enables Cisco IOS to act as a "gateway" to all IBM DB2 products for downstream clients/servers in 11.3T
eboot	Ethernet boot image for mc3810 platform
f	FRAD subset (SNMP, FR, PPP, SDLLC, STUN)
f2	Modified FRAD subset, EIGRP, Pcbus, LAN Mgr. removed, OSPF added
g	ISDN subset (SNMP, IP, Bridging, ISDN, PPP, IPX, Atalk)
g2	Gatekeeper proxy, voice and video
h	For Malibu (2910), 8021D, switch functions, IP host
hdiag	Diagnostics image for Malibu (2910)
i	IP subset (SNMP, IP, Bridging, WAN, Remote Node, Terminal Services)
i2	Subset similar to IP subset for system controller image (3600)
i3	Reduced IP subset with BGP/MIB, EGP/MIB, NHRP, DIRRESP removed
j	Enterprise subset (formerly bpx, includes protocol translation), not used until 10.3
k	Kitchen sink (enterprise for high-end, same as bx, not used after 10.3)
k2	High-end enterprise with CIP2 ucode (not used after 10.3)
k1	Baseline privacy key encryption (on 11.3 and up)
k2	Triple DES (on 11.3 and up)
k3	Reserved for future encryption capabilities (on 11.3 and up)
k4	Reserved for future encryption capabilities (on 11.3 and up)
k5	Reserved for future encryption capabilities (on 11.3 and up)

feature Variable in IOS Name	Feature Description
k6	Reserved for future encryption capabilities (on 11.3 and up)
k7	Reserved for future encryption capabilities (on 11.3 and up)
k8	Reserved for future encryption capabilities (on 11.3 and up)
k9	Reserved for future encryption capabilities (on 11.3 and up)
l	IPeXchange IPX, static routing, gateway
m	RMON (11.1 only) for 11.2, Catalyst 2820-kernel, parser, ATM signaling, Lane Client, bridging
n	IPX
o	Firewall (formerly IPeXchange Net Management)
p	Service provider (IP RIP/IGRP/EIGRP/OSPF/BGP, CLNS ISIS/IGRP)
p2	Service provider with CIP2 ucode
p3	as5200 service provider
p4	5800 (Nitro) service provider
q	Async
q2	IPeXchange Async
r	IBM base option (SRB, SDLLC, STUN, DLSW, QLLC), used with i, in, d
r2	IBM variant for 1600 images
r3	IBM variant for Ardent images (3810)
r4	Reduced IBM subset with BSC/MIB, BSTUN/MIB, ASPP/MIB, RSRB/MIB removed.
s	Source route switch (SNMP, IP, Bridging, SRB) (10.2 and following)
s	(11.2 only) Additions to the basic subset: c1000: (OSPF, PIM, SMRP, NLSP, ATIP, ATAURP, FRSVC, RSVP, NAT) c1005: (X.25, full WAN, OSPF, PIM, NLSP, SMRP, ATIP, ATAURP, FRSVC, RSVP, NAT) c1600: (OSPF, IPMULTICAST, NHRP, NTP, NAT, RSVP, FRAME_RELAY_SVC); AT "s" images also have (SMRP,ATIP,AURP); IPX "s" images also have (NLSP,NHRP) c2500: (NAT, RMON, IBM, MMP, VPDN/L2F) c2600: (NAT, IBM, MMP, VPDN/L2F, VOIP and ATM) c3620: (NAT, IBM, MMP, VPDN/L2F) in 11.3T added VOIP

feature Variable in IOS Name	Feature Description
	c3640: (NAT, IBM, MMP, VPDN/L2F) in 11.3T added VOIP
	c4000: (NAT, IBM, MMP, VPDN/L2F)
	c4500: (NAT, ISL, LANE, IBM, MMP, VPDN/L2F)
	c5200: (PT, v.120, managed modems, RMON, MMP, VPDN/L2F)
	c5300: (MMP, VPDN, NAT, Modem Management, RMON, IBM)
	c5rsm: (NAT, LANE and VLANS)
	c7000: (ISL, LANE, IBM, MMP, VPDN/L2F)
	c7200: (NAT, ISL, IBM, MMP, VPDN/L2F)
	rsp: (NAT, ISL, LANE, IBM, MMP, VPDN/L2F)
t	(11.2) AIP with modified Ucode to connect to Teralink 1000 Data
u	IP with VLAN RIP (Network Layer 3 Switching Software, rsrb, srt, srb, sr/tlb)
v	VIP and dual RSP (HSA) support
v2	Voice V2D
w	Reserved for WBU (remaining characters are specific to WBU)
	I: IISP
	l: LANE & PVC
	p: PNNI
	v: PVC traffic shaping
w2	Reserved for CiscoAdvantage ED train (remaining characters are specific to CiscoAdvantage)
	a: IPX, static routing, gateway
	b: Net management
	c: FR/X.25
	y: Async
w3	Reserved for Distributed Director
	x: X.25 in 11.1 and earlier releases, FR/X.25 in 11.2 (IPeXchange), H.323 Gatekeeper/Proxy in 11.3 releases for 2500, 3620, 3640
y	Reduced IP (SNMP, IP RIP/IGRP/EIGRP, Bridging, ISDN, PPP) (C1003/4), reduced IP (SNMP, IP RIP/IGRP/EIGRP, Bridging, WAN, X.25) (C1005) (11.2 includes X.25) (c1005)

feature Variable in IOS Name	Feature Description
y	IP variant (no Kerberos, Radius, NTP, OSPF, PIM, SMRP, NHRP...) (c1600)
y2	IP variant (SNMP, IP RIP/IGRP/EIGRP, WAN, X.25, OSPF, PIM) (C1005)
y2	IP Plus variant (no Kerberos, Radius, NTP, ...) (c1600)
y3	IP/X.31 y4, reduced IP variant (Cable, Mibs, DHCP, EZHTTP)
z	Managed modems
40	40-bit encryption
56	56-bit encryption
56i	56-bit encryption with IPSEC

Appendix B

NetFlow Version Formats

Version 1 Header Format

Bytes	Content	Description
0-1	version	NetFlow export format version number.
2-3	count	Number of flows exported in this packet (1-24).
4-7	SysUptime	Current time in milliseconds since router booted.
8-11	unix_secs	Current seconds since 0000 UTC 1970.
12-16	unix_nsecs	Residual nanoseconds since 0000 UTC 1970.

Version 1 Flow Record Format

Bytes	Content	Description
0-3	srcaddr	Source IP address.
4-7	dstaddr	Destination IP address.
8-11	nexthop	Next-hop router's IP address.
12-13	input	Input interface's SNMP index.
14-15	output	Output interface's SNMP index.
16-19	dPkts	Packets in the flow.
20-23	dOctets	Total number of Layer 3 bytes in the flow's packets.
24-27	First	SysUptime at start of flow.
28-31	Last	SysUptime at the time the last packet of flow was received.
32-33	srcport	TCP/UDP source port number or equivalent as defined in RFC 1340.
34-35	dstport	TCP/UDP destination port number or equivalent as defined in RFC 1340
36-37	pad1	Unused (zero) byte.
38	prot	IP protocol (for example, 6 = TCP, 17 = UDP), as defined in RFC 1340.
39	tos	IP type of service.

Bytes	Content	Description
40	flags	Cumulative OR of TCP flags.
41-43	pad2 and pad3	Unused (zero) bytes.
44-48	reserved	Unused (zero) bytes.

Version 5 Header Format

Bytes	Content	Description
0-1	version	NetFlow export format version number.
2-3	count	Number of flows exported in this packet (1-30).
4-7	SysUptime	Current time in milliseconds since router booted.
8-11	unix_secs	Current seconds since 0000 UTC 1970.
12-15	unix_nsecs	Residual nanoseconds since 0000 UTC 1970.
16-19	flow_sequence	Sequence counter of total flows seen.
20-24	reserved	Unused (zero) bytes.

Version 5 Flow Record Format

Bytes	Content	Description
0-3	srcaddr	Source IP address.
4-7	dstaddr	Destination IP address.
8-11	nexthop	Next hop router's IP address.
12-13	input	Input interface's SNMP index.
14-15	output	Output interface's SNMP index.
16-19	dPkts	Packets in the flow.
20-23	dOctets	Total number of Layer 3 bytes in the flow's packets.
24-27	First	SysUptime at start of flow.
28-31	Last	SysUptime at the time the last packet of flow was received.
32-33	srcport	TCP/UDP source port number or equivalent.

Bytes	Content	Description
34-35	dstport	TCP/UDP destination port number or equivalent.
36	pad1	Unused (zero) bytes.
37	tcp_flags	Cumulative OR of TCP flags.
38	prot	IP protocol (for example, 6 = TCP, 17 = UDP).
39	tos	IP type of service.
40-41	src_as	AS of the source, either origin or peer.
42-43	dst_as	AS of the destination, either origin or peer.
44	src_mask	Source address prefix mask bits.
45	dst_mask	Destination address prefix mask bits.
46-47	pad2	Unused (zero) bytes.

Appendix C

Access List Number Ranges for Each Protocol and List Type

Protocols with Access Lists Specified by Numbers

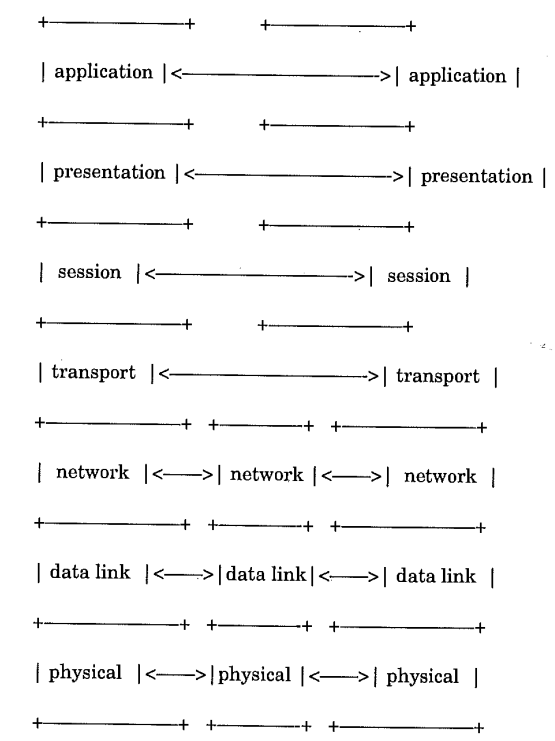
Protocol	Range
IP	1 to 99
Extended IP	100 to 199
Ethernet type code	200 to 299
Ethernet address	700 to 799
Transparent bridging (protocol type)	200 to 299
Transparent bridging (vendor code)	700 to 799
Extended transparent bridging	1100 to 1199
DECnet and extended DECnet	300 to 399
XNS	400 to 499
Extended XNS	500 to 599
AppleTalk	600 to 699
Source-route bridging (protocol type)	200 to 299
Source-route bridging (vendor code)	700 to 799
IPX	800 to 899
Extended IPX	900 to 999
IPX SAP	1000 to 1099
Standard VINES	1 to 100
Extended VINES	101 to 200
Simple VINES	201 to 300

Appendix D

OSI Reference Model

Developed by the International Standards Organization (ISO) to help standardize international protocols being used, OSI stands for Open Systems Interconnection. The main idea is to have independent standards for the different layers so that a change in a layer would not cause changes in other layers. In the layered approach, it is possible to use different network hardware without changing the existing application programs, for example.

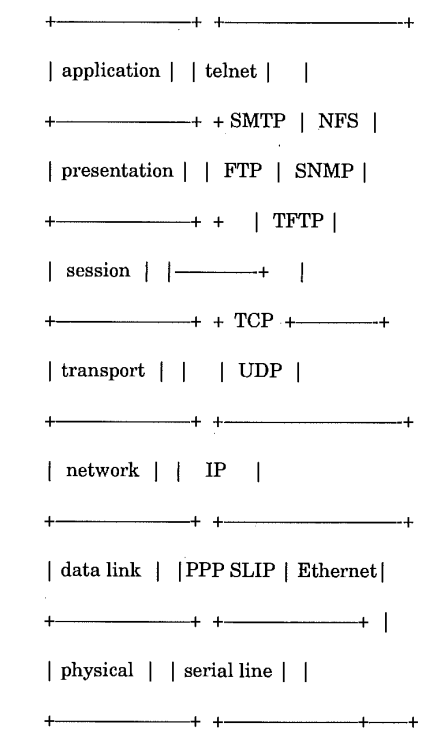
ISO/OSI Network Architecture Reference Model



Each layer communicates with its peer, using services that the layer below provides. As can be seen from the following picture, the transport layer and layers above it are end-to-end layers and they do not know anything about the network below them.

1. The physical layer defines electrical signaling on the transmission channel and how bits are converted into electrical current, light pulses, or any other physical form. *Serial_line* is an example of the physical layer. A network device functioning at this layer only is called a *repeater*.
2. The data link layer defines how the network layer packets are transmitted as bits. Examples of data link layer protocols are *PPP* (Point to Point Protocol) and *Ethernet* framing protocols. *Bridges* work at the data link layer only.
3. The network layer defines how information from the transport layer is sent over networks and how different hosts are addressed. An example of a network layer protocol is the *Internet Protocol (IP)*. A device that takes care of the network level functions is called a *router* or sometimes a *gateway* in the Internet terminology.
4. The transport layer takes care of data transfer, ensuring the integrity of data if desired by the upper layers. *TCP* and *UDP* are operating at this layer.
5. The session layer establishes and terminates connections and arranges sessions to logical parts. *TCP* and *RPC* provide some functions at this layer.
6. The presentation layer takes care of data type conversion. An example of a protocol residing at this layer is *XDR* (External Data Representation), which is used by *RPC* applications to provide interoperability between heterogeneous computer systems.
7. The application layer defines the protocols to be used between the application programs. Examples of protocols at this layer are protocols for electronic mail (such as *SMTP*), file transfer (such as *FTP*), and remote login.

OSI Model and TCP/IP Protocols



The IP Suite was not designed according the OSI reference model, but it is possible to arrange the functionality of the Internet protocols to fit into the OSI model.

Glossary

AAL—ATM Adaptation Layer. A collection of standardized protocols that provide services to higher layers by adapting user traffic to a cell format. The AAL is divided into the Convergence Sublayer (CS) and the Segmentation and Reassembly (SAR) sublayer.

AAL1—AAL Type 1. Protocol standard used for the transport of Constant Bit Rate (CBR) traffic (audio and video) and for emulating TDM-based circuits (such as DS1 and E1).

AAL2—AAL Type 2. Protocol standard for supporting the time-dependent Variable Bit Rate (VBR-RT) of connection-oriented traffic (packetized video and audio).

AAL3/4—AAL Type 3 and 4. Protocol standard for supporting both connectionless and connection-oriented Variable Bit Rate (VBR) traffic. Used also to support SMDS.

AAL5—AAL Type 5. Protocol standard for supporting the transport of Lightweight Variable Bit Rate (VBR) traffic and signaling messages. Also used to support Frame Relay services.

ABR—Available Bit Rate. One of the two best-effort service types (the other one is UBR) in which the network makes no absolute guarantee of cell delivery, but it guarantees a minimum bit rate for user transmission. An effort is also made to keep cell loss as low as possible.

access control byte—The byte following the start delimiter of a token or frame that is used to control access to the Token Ring network.

access priority—The maximum priority that a token can have for the adapter to use it for transmission.

access rate—The bit per second (bps) rate at which a user can transmit over the network's lines.

access unit—A unit that allows multiple attaching devices access to a Token Ring network at a central point. Sometimes these devices may also be referred to as a concentrator.

ACK—Acknowledgement. A message that acknowledges the reception of a transmitted packet. ACKs can be separate packets or piggybacked on reverse traffic packets.

acknowledgment—In data communications, the transmission of characters from the receiving device, indicating that data sent has been received correctly.

ACR ???

active—(1) Operational. (2) Pertaining to a file, page, or program that is in main storage or memory, as opposed to a file, page, or program that must be retrieved from auxiliary storage. (3) Pertaining to a node or device that is connected or is available for connection to another node or device.

active monitor—A function in a single adapter on a Token Ring network that initiates the transmission of tokens and provides token error recovery facilities. Any active adapter on the ring has the capability to provide the active monitor function if the current active monitor fails.

active program—Any program that is loaded into memory and ready to be executed.

active session - The session in which a user is currently interacting with the computer.

adapter address—Twelve hexadecimal digits that identify a LAN adapter.

Adapter Card ???

adaptive routing—A method of routing packets of data or data messages in which the system's intelligence selects the best path. This path might change with traffic patterns or link failures.

adaptive session-level pacing—A form of session-level pacing in which session components exchange pacing windows that may vary in size during the course of a session. This allows transmission within a network to adapt dynamically to variations in availability and demand of buffers on a session-by-session basis. Session-level pacing occurs within independent stages along the session path according to local congestion at the intermediate nodes.

address—(1) A character or group of characters that identify a data source or destination or a network node. (2) The destination of a message sent through a communications system. In computer network terms, it is a set of numbers that uniquely identify a workstation on a LAN.

adjacent—In a network, pertaining to devices, nodes, or domains that are directly connected by a data link or that share common control.

adjacent link station—A link station directly connected to a given node by a link connection over which network traffic can be carried.

adjusted ring length (ARL)—In a multiple-wiring-closet ring, the sum of all wiring closet-to-wiring closet cables in the main ring path less the length of the shortest of those cables.

Advanced peer-to-peer networking (APPN)—An extension to SNA featuring (a) greater distributed network control that avoids critical hierarchical dependencies, thereby isolating the effects of single points of failure; (b) dynamic exchange of network topology information to foster ease of connections and reconfiguration, adaptive route selection, and simplified network definition; and (c) automated resource registration and directory lookup. APPN extends the LU 6.2 peer orientation for end-user services to network control. It also uses LU 6.2 protocols on its own control point sessions that provide the network control.

Advanced program-to-program communication (APPC)—(1) The general facility characterizing the LU 6.2 architecture and its various implementations in products. (2) Sometimes used to refer to the LU 6.2 architecture and its product implementation as a whole, or an LU 6.2 product feature in particular, such as an APPC application program interface.

AIR—Additive Cell Rate. The cell rate a source can transmit after increasing its rate by the RIF.

AIS—Alarm Indication Signal. One of the OAM function types used for fault management (*see also* CC and RDI).

alert—A message sent to a management services focal point in a network to identify a problem or an impending problem.

algorithm—A prescribed finite set of well-defined rules or processes for the solution of a problem in a finite number of steps. In normal English, it is the mathematical formula for an operation, such as computing the check digits on packets of data that travel via packet-switched networks.

all-routes broadcast frame—A frame that has bits in the routine information field set to indicate the frame is to be sent to all LAN segments in the network. The destination address is not examined and plays no role in bridge routing.

all-stations broadcast frame—A frame whose destination address bits are set to all ones. All stations on any LAN segment that the frame appears on will copy it. It is independent of all-routes broadcasting.

Allowed (or available) cell rate—The available bandwidth in cells.

ANSI—American National Standards Institute. A U.S. technology standards organization.

API—Application Programming Interface. A set of functions used by an application program as a means of providing access to a system's capabilities (operating and communications).

application—A program or set of programs that perform a task.

application program interface (API)—The formally defined programming language interface that allows a programmer to write to the interface.

application transaction program—A program written for or by a user to process the user's application; in an SNA network, an end user of a type 6.2 logical unit.

applications layer—The seventh and highest layer of Systems Network Architecture (SNA) and Open Systems Interconnection (OSI). It supplies functions to applications or nodes, allowing them to communicate with other applications or nodes.

APPN end node—A type 2.1 end node that provides full SNA end-user services and supports sessions between its local control point (CP) and the CP in an adjacent network node to dynamically register its resources with the adjacent CP (its network node server), to send and receive directory search requests, and to obtain management services. It can also attach to a subarea network as a peripheral node.

APPN intermediate routing—The capability of an APPN network node to accept traffic from one adjacent node and pass it on to another with an awareness of session affinities in controlling traffic flow and outage notifications.

APPN intermediate routing network—The portion of an APPN network consisting of the network nodes and their connections.

APPN network—A type 2.1 network having at least one APPN node.

APPN network node—A type 2.1 (T2.1) node that, besides offering full SNA end-user services, provides intermediate routing services within a T2.1 network and network services to its local LU's and attached T2.1 end nodes in its domain. It can also attach to a subarea network as a peripheral node.

APPN node—An APPN network node or an APPN end node.

ARP—Address Resolution Protocol. A TCP/IP protocol used for resolving local network addresses by mapping a physical address (such as a MAC address) to an IP address.

asynchronous—Asynchronous transmission is an approach for acquiring synchronization on a per-byte basis. Start and stop bits are used as delimiters. Asynchronous transfer is an efficient approach for transmitting information where time slots are used on a demand basis (such as in ATDM and ATM) rather than on a periodical one (such as in TDM and STM).

asynchronous transmission—A method of data transmission that allows characters to be sent at irregular intervals by preceding each character with a start bit and following it with a stop bit. No clocking signal is provided. This is in contrast to synchronous transmission.

ATDM—Asynchronous Time-Division Multiplexing. A asynchronous and intelligent TDM in which time slots are allocated to the users on demand (dynamically).

ATM—Asynchronous Transfer Mode. A broadband switching, multiplexing, connection-oriented, high-performance and cost-effective integrated technology for supporting B-ISDN services (multimedia). Since no clock control is necessary, it is called asynchronous (see also STM). Information is transmitted at very high rates (up to hundreds of Mbps) in fixed-size format packets called cells. Traffic streams are distinguished and supported according to different QoS classes.

ATM CSU/DSU—ATM Channel/Data Service Unit. A device that converts information bits (transmitted over the telephony network) or frame-based information into (or from) a stream of ATM cells (*see also* CSU, DSU, and DXI).

ATM forum—Originally founded by a group of vendors and telecommunication companies, this formal standards body is comprised of various committees responsible for making recommendations and producing implementation specifications.

ATM layer—The second layer of the ATM protocol stack model that constructs and processes the ATM cells. Its functions also include a Usage Parameter Control (UPC) and support of QoS classes.

ATM-SAP—ATM-Service Access Point. The physical interface at the boundary between the AAL and the ATM layer (*see also* SAP and PHY-SAP).

AToMMIB—ATM MIB. IETF-defined Management Information Base (MIB) for managing VP/VC links and ATM PVC-supported services and interfaces.

attenuation—A decrease in magnitude of current, voltage, or electrical or optical power of a signal in transmission between points. It can expressed in decibels or nepers.

average cell rate—The mean number of cells that the source can inject into a network over a given virtual connection (VC).

average cell transfer delay—The arithmetic average of a number of cell transfer delays (CTD). See also Mean Cell Transfer Delay.

B-ICI—Broadband Inter-Carrier Interface. An interface that supports service connections (such as CRS, CES, SMDS, and FR) across public ATM networks and/or carriers.

B-ISDN—Broadband Integrated Services Digital Network. An ITU-T protocol platform to support the integrated, high-speed transmission of data, audio, and video in a seamless fashion. ATM emerged as a suitable transport standard.

backup server—A program or device that copies files so at least two up-to-date copies always exist.

backbone—A LAN, a WAN, or a combination of both dedicated to providing connectivity between subnetworks in an enterprise-wide network. Subnetworks are connected to the backbone via bridges and/or routers and the backbone serves as a communications highway for LAN-to-LAN traffic.

backbone LAN segment—In a multisegment LAN configuration, a centrally located LAN segment that other LAN segments are connected to by means of bridges or routers.

backup path—In an IBM Token Ring Network, an alternative path for signal flow through access units and their main ring path cabling.

balun—Balanced/unbalanced. An impedance matching transformer. Baluns are small passive devices that convert the impedance of coaxial cable so that its signal can run on twisted-pair wiring. They are used often so that IBM 3270-type terminals, which traditionally require coaxial cable connection to their host computer, can run on twisted-pair.

bandwidth—The range of electrical frequencies a device can handle.

BASize—Buffer Allocation Size. A one-byte field in the CPCS-PDU header to indicate to the receiving end the buffer space that needs to be reserved for reassembling the CPCS-PDU.

beacon—A Token Ring frame sent by an adapter indicating that it has detected a serious ring problem, such as a broken cable or a multistation access unit. An adapter sending these frames is said to be beaconing.

BEC—Backward Error Correction. An error-correction scheme where the sender retransmits any data to be found in error, based on the feedback from the receiver.

best effort—A QoS class in which no specific traffic parameters and no absolute guarantees are provided. Best effort includes UBR and ABR (see also service types).

bisynchronous transmission—Also called BISYNC. A data character-oriented communications protocol developed by IBM for synchronous transmission of binary-coded data between two devices.

bit—Abbreviation for binary digit. The smallest unit of information (data) and the basic unit in data communications. A bit can have a value of 0 or 1.

bit rate—The number of bits of data transmitted over a communications line each second.

BNC—A bayonet-locking connector for slim coaxial cables.

BOM—Beginning of Message. A PDU that constitutes the beginning of a message.

bps—Bits per second. A measurement of data transmission speeds.

BRI—Basic Rate Interface. An ISDN service specification that provides two 64-kbps data B-channels and one 16-kbps control D-channel, all sharing the same physical medium.

bridge—(1) An interface connecting two similar LANs. (2) A device that connects two LANs. It performs its functions at the data link control (DLC) layer.

bridge ID—The bridge label combined with the adapter address of the adapter connecting the bridge to the LAN segment with the lowest LAN segment number.

bridge label—A two-byte hexadecimal number that the user can assign to each bridge.

bridge number—The bridge identifier that the user specifies in the bridge program configuration file. The bridge number distinguishes between parallel bridges.

broadcast—The simultaneous transmission of data to more than one destination.

broadcast message—A message from one station sent to all other users. On a Token Ring LAN, the destination address is unspecified; thus, all devices receive the message.

brouter—In local area networking, a device that combines the dynamic routing capabilities of an internetwork router with the capability of a bridge to interconnect LANs.

BSC—Binary Synchronous Communication. A set of IBM operating procedures for synchronous transmission used in teleprocessing networks.

BT—Burst Tolerance. Proportional to the MBS, burst tolerance is used as a measure (leaky bucket parameter) for conformance checking of the SCR.

buffer—In data transmission, a buffer is a temporary storage location for information being sent or received. Usually located between two different devices that have different capabilities or speeds for handling the data.

burstiness—A source traffic characteristic that is defined as the ratio of the peak cell rate (PCR) to the average cell rate. It is a measure of the intercell spacing (*see also* MBS).

bus—A network configuration in which nodes are interconnected through a bidirectional transmission medium.

BUS—Broadcast and Unknown Server. A server that forwards multicast, broadcast, and unknown-destination address traffic to the attached LECs.

BW—Bandwidth. Transmission capacity of a communications medium.

byte—A binary character operated upon as a unit and usually shorter than a computer word. It is eight consecutive bits representing a character.

cable loss—The amount of radio frequency signal attenuation caused by a cable.

cable riser—Cable running vertically in a multi-story building to serve the upper floors.

CAC—Connection Admission Control. An ATM function that determines whether a virtual circuit (VC) connection request should be accepted or rejected.

campus—A networking environment in which users (voice, video, and data) are spread out over a broad geographic area, as in a university, hospital, or medical center. There may be several LANs on a campus and they are connected with bridges and/or routers communicating over telephone or fiber-optic cable.

carrier—A wave or pulse train that can be varied by a signal bearing information to be transmitted over a communication system.

CAT-3—Category 3 Unshielded Twisted. A type of UTP commonly used with ATM interfaces for cell transmission at low speeds, 25-50 Mbps, and at distances up to 100 meters.

CAT-5—Category 5 Unshielded Twisted Pair. A type of UTP commonly used with ATM interfaces for higher-speed cell transmission (more than 50 Mbps).

CBR—Constant (or Continuous) Bit Rate. One of the five ATM classes of service that support the transmission of a continuous bit-stream of information where traffic, such as voice and video, needs to meet certain QoS requirements (*see also* QoS Classes).

CC—Continuity Cell. A cell used periodically to check whether a connection is idle or has failed (at the cross-connect nodes) in order to guarantee a continuation in the flow of the information cells. Continuity checking is one of the OAM function types for fault management (*see also* AIS and RDI).

CCITT—Consultative Committee on International Telegraphy and Telephony. A standards and specifications body whose published recommendations cover a wide spectrum of areas that include definition of terms, basic principles and characteristics, protocol design, description of models, and other specifications. Currently known as ITU-T.

CCR—Current Cell Rate. A field in the RM cell header that indicates the current complying cell rate a user can transmit over a virtual connection (VC).

CDV—Cell Delay Variation. A QoS parameter that measures the difference between a single cell's transfer delay (CTD) and the expected transfer delay. It gives a measure of how closely cells are spaced in a Virtual Circuit (VC). CDV can be introduced by ATM multiplexers (MUXs) or switches.

CDVT—Cell Delay Variation Tolerance. Used in CBR traffic, it specifies the acceptable tolerance of the CDV (jitter).

cell—Basic ATM transmission unit. It is a 53-byte packet, comprised of a five-byte header and a 48-byte payload. User traffic is segmented into cells at the source and reassembled at the destination.

cell header—The five-byte ATM cell header contains control information regarding the destination path and flow control. More specifically, it contains the following fields: GFC, VPI, VCI, PT, CLP, and HEC.

cell layer—Same as ATM Layer.

CER—Cell Error Rate. A QoS parameter that measures the fraction of transmitted cells that are erroneous (they have errors when they arrive at the destination).

CES—Circuit Emulation Service. An ATM-provided class of service, where TDM-type, constant-bit-rate (CBR) circuits are emulated by the AAL1.

CI—Congestion Indication. A bit in the RM cell to indicate congestion (it is set by the destination if the last cell received was marked).

CIF—Cell Information Field. The payload (48 bytes) of an ATM cell.

CIR—Committed Information Rate. A term used in Frame Relay, which defines the information rate the network is committed to provide the user with, under any network conditions.

circuit emulation—A virtual-circuit (VC) service offered to end users in which the characteristics of an actual, digital bitstream line (such as video traffic) are emulated (such as a 2-Mbps or 45-Mbps signal).

class of service (COS)—A designation of the transport network characteristics, such as route security, transmission priority, and bandwidth, needed for a particular session. The class of service is derived from a mode name specified by the initiator of a session.

Classical IP—IETF-defined protocols for developing IP over ATM networks (IP support for the QoS classes, ARP over SVC and PVC networks) so that common applications (FTP, Telnet, SMTP, SNMP) can be supported in an ATM environment. The main issues in the transport of IP over ATM are the packet encapsulation and the address resolution.

CLP—Cell Loss Priority. A one-bit field in the ATM cell header that corresponds to the loss priority of a cell. Lower priority (CLP = 1) cells can be discarded under congestion situations.

CLR—Cell Loss Ratio. A QoS parameter that gives the ratio of the lost cells to the total number of transmitted cells.

CMIP—Common Management Information Protocol. An ITU-T-defined management interface standard that can support administration, maintenance, and operation information functions (*see also* OAM&P).

CMIP—Common Management Information Protocol. A protocol formally adapted by the International Standards Organization used for exchanging network management information over OSI. Typically, this information is exchanged between two management stations. It can be used to exchange information

between an application and a management station. Although designed for OSI networks, it is transport-independent. Theoretically, it can run across a variety of transports, including IBM's SNA.

CMOT—CMIP over TCP/IP. The use of CMIP over a TCP/IP-based transport.

CMR—Cell Misinsertion Rate. A performance measure that is defined as the number of misinserted cells (those that arrive from the wrong source) per (virtual) connection second.

CO—Central Office. Premises of a carrier service provider where customer lines (telephone lines) are multiplexed and switched to other COs.

coaxial cable—A cable composed of an insulated central conducting wire wrapped in another cylindrical conducting wire. The whole thing is usually wrapped in another insulating layer and an outer protective layer. A coaxial cable has great capacity to carry great quantities of information. It is typically used to carry high-speed data for cable TV.

COM—Continuation of Message. A PDU that is part of a message.

communication—The transmission and reception of data.

communication adapter—A circuit card with associated software that enables a processor, controller, or other device to be connected to a network.

communication network management (CNM)—The process of designing, installing, operating, and managing the distribution of information and control among users of communication systems.

communications line—The physical link (such as wire or a telephone circuit) that connects one or more workstations to a communications control unit or that connects one control unit to another.

Communications Manager—A component of OS/2 Extended Edition that lets a workstation connect to a host computer and use the host resources as well as the resources of other personal computers to which the workstation is attached either directly or through a host.

communications port—(1) An access point for data entry or exit to or from a communication device such as a terminal. (2) On a personal computer or workstation, a synchronous or asynchronous serial port to which a modem can be attached.

composite end node—To a type 2.1 node, a group of nodes that appears to be a single end node.

concurrent—Pertaining to the occurrence of two or more activities within a given interval of time.

CONFIG.SYS—A file that contains configuration options for an OS/2 program or DOS program installed on a workstation or personal computer. It defines the devices, system parameters, and resource options of a workstation or personal computer.

congestion control—A resource and traffic management mechanism to avoid and/or prevent excessive situations (buffer overflow or insufficient bandwidth) that can cause the network to collapse. Various congestion control methods exist (*see also* flow control).

connection-oriented network—A communications service in which an initial connection between the end points (source and destination) has to be set up. Examples are ATM and Frame Relay (*see also* virtual circuit VC).

connection-oriented—See Connection-oriented Network.

connectionless network—Communications service in which packets are transferred from source to destination without the need of a pre-established connection. Examples are IP and SMDS (*see also* datagram).

connectionless service—A networking node in which individual data packets in a LAN traveling from one point to another are directed from one intermediate node to the next until they reach their ultimate destination. The receipt of a transmission is typically acknowledged from the ultimate destination to the point of origin.

control point (CP)—A component of a node that manages resources of that node and optionally provides services to other nodes in the network.

control vector—One of a general class of RU substructures that has variable length, is carried within some enclosing structure, and has a one-byte key used as an identifier.

controller—A unit that controls input/output operations for one or more devices.

conversation—A logical connection between two transaction programs using an IBM LU 6.2 session.

corporate network - A network of networks that connects most or all of a corporation's LANs. Connections between networks and LANs are made with bridges and routers. Also called an internetwork, a WAN, or an enterprise network.

COS—Class of Service. See QoS Classes.

CPCS—Common Part Convergence Sublayer. Part of the AAL convergence sublayer (CS). It must always be present in the AAL implementation. Its task is to pass primitives to the other AAL sublayers (SAR or SSCS). It supports the functions of the standardized Common Part AALs: AAL1, AAL3/4, and AAL5.

CPE—Customer Premises Equipment. Computer and communications equipment (hardware and software) used by a carrier's customer and located at the customer's site (*see also* DTE).

CPI—Common Part Indicator. A one-byte field in the header of the CPCS-PDU in AAL3/4 that indicates the number of bits of which the BASize field consists.

CRC—Cyclic Redundancy Check. A bit-errors detection technique that employs a mathematical algorithm that, based on the transmitted bits, calculates a value attached to the information bits in the same packet. The receiver using the same algorithm recalculates that value and compares it to the one received. If the two values do not agree, the transmitted packet is then considered to be in error.

CRC (Cyclic Redundancy Check)—A process used to check the integrity of a block of data.

CRM—Cell Rate Margin. A measure of the residual useful bandwidth for a given QoS class, after taking into account the SCR.

CRS—Cell Relay Service. A bearer service offered to the end users by an ATM network that delivers (transports and routes) ATM cells.

CS—Convergence Sublayer. The upper half of the AAL, which is divided into two sublayers, the Common Part (CPCS) and the Service Specific (SSCS). It is service-dependent and its functions include manipulation of cell delay variation (CDV), source clock frequency recovery, and forward error correction (FEC). Although each AAL has its own functions, in general the CS describes the services and functions needed for conversion between ATM and non-ATM protocols (*see also* SAR).

CS-PDU—Convergence Sublayer Protocol Data Unit. The PDU used at the CS for passing information between the higher layers and the SAR where they are converted into cells.

CSF—Cell Switch Fabric. See Switch Fabric.

CSR—Cell Missequenced Ratio. A performance measure that is defined as the number of missequenced cells (those that arrive in the wrong order) per (virtual) connection second.

CSU—Channel Service Unit. Equipment at the user end that provides an interface between the user and the communications network. CSU can be combined with DSU in the same device (*see also* DCE).

CTD—Cell Transfer Delay. A QoS parameter that measures the average time for a cell to be transferred from its source to its destination over a virtual connection (VC). It is the sum of any coding, decoding, segmentation, reassembly, processing, and queueing delays.

data circuit-terminating equipment (DCE)—The equipment installed at the user's premises that provides all the functions required to establish, maintain, and terminate a connection for data transmission and the signal conversion and coding between the data terminal equipment device and the line.

data communication—The transmission and reception of data.

data link—A physical link, like a wire, that connects one or more devices or communication controllers.

data link control (DLC)—(1) The physical means of connecting one location to another for the purpose of transmitting and receiving data. (2) In SNA, the second layer of the seven-layer architecture. (3) In OSI, the second layer of the seven-layer architecture.

datagram—A packet transport mode in which packets are routed independently and may follow different paths. Thus, there is no guarantee of sequence delivery (*see also* VC).

datastream—(1) All data transmitted through a data channel in a single read or write operation. (2) A continuous stream of data elements being transmitted, or intended for transmission, in character or binary-digit form, using a defined format.

DCE—Data Circuit-terminating Equipment or Data Communications Equipment. A device at the user end, typically a modem or other communications device, which acts as an access point to the transmission medium.

dependent logical unit (DLU)—An LU controlled by an SNA host system.

destination—In a network, any point or location, such as a node, a station, or a terminal, to which data is sent.

destination address—The part of a message that indicates for whom the message is intended. It's synonymous with an address on an envelope. IBM Token Ring Network addresses are 48 bits in length.

device—(1) An input/output unit such as a terminal, display, or printer. (2) In computers, it can be used for direct access storage (such as the hard disk).

differential Manchester encoding—A transmission encoding scheme in which each bit is encoded as a two-segment signal with a signal transition (polarity change) at either the bit time or half-bit time. Transition at a bit time represents a 0. No transition at a bit time indicates a 1.

diskless workstation—A workstation without a hard disk or diskette drive.

distributed processing—A process within a network of computers in which information is initiated in local computers and the resultant data is sent to a central computer for further processing with the data from other local systems. A LAN is an example of distributed processing.

domain—In IBM's SNA, a host-based systems services control point (SSCP), the physical units (PUs), logical units (LUs), links, link stations, and all the affiliated resources that the host (SSCP) controls.

downloading—The act of receiving data from one computer into another.

downstream physical unit (DSPU)—A controller or a workstation downstream from a gateway that is attached to a host.

DQDB—Distributed Queue Dual Bus. The IEEE 802.6 standard that is a metropoitan area network (MAN) protocol based on 53-byte packets that support connectionless and connection-oriented, isochronous integrated services. It is implemented as two unidirectional buses configured in a physical ring topology.

DS-0—Digital Signal 0. The physical interface for digital transmission at the rate of 64 Kbps.

DS-1—Digital Signal 1. The physical interface for digital transmission at the rate of 1.544 Mbps. Also known as a T-1 standard, it can simultaneously support 24 DS-0 circuits.

DS-2—Digital Signal 2. The physical interface for digital transmission at the rate of 6.312 Mbps.

DS-3—Digital Signal 3. The physical interface for digital transmission at the rate of 44.736 Mbps.

DSU—Data Service Unit. Equipment at the user end that acts as a telephony-based interface between low-rate (56 kbps) services and higher-rate circuits.

DTE—Date Terminal Equipment. The host computer (PC or workstation) to provide the end-user with access to a communications network. The DTE is connected to a DCE that performs the signaling operation (*see also* CPE).

DXI—Data Exchange Interface. A frame-based ATM interface between a DTE (such as a router or a local switch) and a DCE. DXI interfaces to the ATM UNI and has been chosen by the ATM Forum as an affordable solution for providing ATM capabilities over a WAN.

E-1—European Digital Signal 1. The European standard for digital physical interfaces at 2.048 Mbps.

E-3—European Digital Signal 3. The European standard for digital physical interfaces at 34.368 Mbps. It can simultaneously support 16 E-1 circuits.

E-4—European Digital Signal 4. The European standard for digital physical interfaces at 139.264 Mbps.

E.164—An ITU-T-defined eight-byte address format. In ATM, it is typically used in public networks and is provided by the telecommunication carriers, while 20-byte NSAP format addresses are used within private networks.

early token release—This is a method of token passing that allows for two tokens to exist on the network simultaneously. It is used on 16-Mbps Token Ring networks.

EFCI—Explicit Forward Congestion Indication. A one-bit field in the PTI that contains information whether congestion at an intermediate node has been experienced. The EFCI bit is set when a buffer threshold has been exceeded.

ELAN—Emulated LAN. See LAN Emulation.

end user—The ultimate source or destination of application data flowing through a network. An end user can be an application program or a human operator.

end-node domain—An end-node control point, its attached links, and its local LUs.

ENR—Enterprise Network Roundtable. An ATM Forum-associated group of ATM users to provide feedback on ATM-related issues and also present the users with completed interoperable capabilities and functionality.

ER—Explicit Rate. A field in the RM cell header specifying the cell rate a user should use for transmission over a virtual connection (VC) as dictated by the RM (*see also* CCR).

error rate—In data transmission, the ratio of the number of incorrect elements transmitted to the total number of elements transmitted.

ETSI—European Telecommunications Standards Institute. The corresponding body of ANSI in Europe involved in providing and adapting standards for the European telecommunications.

FDDI—Fiber Distributed Data Interface. An ANSI-defined standard for implementing a high-speed (100 Mbps) LAN over fiber.

FDM—Frequency-Division Multiplexing. A technique that allows for the channel bandwidth of a circuit to be subdivided into many little channels (one per traffic stream).

FEC—Forward Error Correction. An error correction technique in which there are no retransmissions and therefore the receiver is responsible for correcting any errors in the packets.

file—A set of related records treated as a unit.

file server—A device that serves as a central location for commonly used files by everyone on a LAN.

flow control—A method used in networking for congestion avoidance and traffic regulation. There are three techniques: window-based control, where a sliding window is used to determine how many cells can be transmitted during a predefined period; rate-based control, where the rate that the source transmits at is monitored and controlled; and credit-based control, where a source can transmit a cell if there is a credit available. CAC is also part of the flow control.

forum—Same as ATM Forum.

FR—Same as Frame Relay.

frame—A group of bits sent serially (one after another). It is a logical transmission unit.

frame check sequence—In a Token Ring LAN, a 32-bit field that follows the data field in every Token Ring frame.

Frame Relay—A packet-switching technology to provide a very reliable packet delivery over virtual circuits (VC). Some of the concepts used in Frame Relay have been incorporated in ATM networks.

FRM—Fast Resource Management. A form of network management for allocating resources (buffers or bandwidth) dynamically.

front-end processor—A processor that offloads line control, message handling, code conversion, error control, and routing of data from the host computer. IBM's 3725 and 3745 are examples of front-end processors. Also known as a communication controller.

FTP—File Transfer Protocol. A protocol used for transferring files between different machines across a network.

functional address—In IBM network adapters, this is a special kind of group address in which the address is bit-significant, each "on" bit representing a function performed by the station (active monitor, ring-error monitor, LAN error monitor, or configuration report server).

gateway—A functional unit that connects two different computer network architectures.

gateway function—(1) The capability of a subarea node to provide protocol support to connect two or more subarea networks. (2) The component that provides this capability.

Gbps—Gigabits per second. Transmission speed or rate of a hundred million bits per second.

GCRA—Generic Cell Rate Algorithm. A reference model proposed by the ATM Forum for defining cell-rate conformance in terms of certain traffic parameters. It is usually referred as the Leaky Bucket algorithm (*see also* traffic shaping).

GFC—Generic Flow Control. A four-bit field in the ATM cell header for supporting multiplexing functions. Its default value is '0000' when the GFC protocol is not enforced. The GFC mechanism is intended to support simple flow control in ATM connections.

group address—In a LAN, a locally administered address assigned to two or more adapters to allow the adapters to copy the same frame.

group SAP—A single address assigned to a group of service access points (SAPs).

half-session—A session-layer component consisting of the combination of data flow control and transmission control components comprising one end of a session.

hard error—An error condition on a network that requires the source of the error to be removed or that the network be reconfigured before the network can resume reliable operation.

header—The portion of a message that contains control information for the message. Usually found at the beginning of a frame.

HEC— Header Error Check or Header Error Control. A one-byte field in the cell header used for the header error correction and detection. Due to the information contained in the header, HEC is quite significant.

hertz (Hz)—A unit of frequency equal to one cycle per second.

hexadecimal—A numbering base in which four bits are used to represent each digit. The digits can have one of 16 values: 0, 1, 2...9, A, B, C, D, E, F.

hierarchical network—A multisegment network configuration providing only one path through intermediate segments between source segments and destination segments.

HOL—Head-of-Line. The head position of a buffer (inside a switch). A blocking phenomenon is associated with the HOL that refers to the fact that cells in the queue have to wait for the HOL cell to depart first.

hop count—The number of ring segments spanned to establish a session between two workstations. In IBM Token Ring networks, the maximum is seven.

hop—In Token Ring networking, the connection between ring segments. The connection is usually made using bridges.

host node—(1) A node at which a host computer is situated. (2) In SNA, a subarea node that contains an SSCP.

host system—(1) A data processing system that is used to prepare programs and the operating environments for use on another computer or controller. (2) The data processing system to which a network is connected and with which the system can communicate.

HSSI—High-Speed Serial Interface. An interface between CSU/DSU and DXI.

ICR—Initial Cell Rate. The rate that a source is allowed to start up at following an idle period. It is established at connection setup and is between the MCR and the PCR.

IEEE—Institute of Electrical and Electronic Engineers. A standards and specification organization with extensive activities in the areas of computers and electronics.

IEEE 802—IEEE committee on LANs.

IEEE 802.1—IEEE standard for overall architecture of LANs and internetworking.

IEEE 802.2—IEEE data link control layer standard used with IEEE 802.3, 802.4, and 802.5.

IEEE 802.3—IEEE carrier-sense multiple access with collision detection (CSMA/CD). A physical layer standard specifying a LAN with a CSMA/CD access method on a bus topology. Ethernet and Starlan both follow subsets of the 802.3 standard. Typically, they transmit at 10 Mbps.

IEEE 802.4—IEEE physical layer standard specifying a LAN with a token-passing access method on a bus topology. Used with Manufacturing Automation Protocol (MAP) LANs. Typical transmission speed is 10 Mbps.

IEEE 802.5—IEEE physical layer standard specifying a LAN with a token-passing access method on a ring topology. Used by IBM's Token Ring network. Typical transmission rates are four Mbps and 16 Mbps.

IETF—The Internet Engineering Task Force that was initially responsible for developing specifications required for the interoperable implementation of IP. One of the issues IETF has been focusing on is the implementation of Classical IP over ATM.

IISP—Interim Interswitch Signaling Protocol. A protocol that uses UNI-based signaling for switch-to-switch communication (*see also* NNI).

ILMI—Interim Local Management Interface. An ATM forum-defined Network Management System (NMS) based on SNMP that can provide configuration, performance, and fault management information concerning VC connections available at its UNI (public and private). It operates over AAL3/4 and AAL5 and will be eventually replaced once it becomes standardized by ITU-T.

impedance—The combined effect of resistance, inductance, and capacitance on a signal at a particular frequency.

independent logical unit (ILU)—In SNA, a logical unit that does not require assistance from an SSCP to establish a LU-LU session.

Institute of Electrical and Electronic Engineers (IEEE)—A publishing and standards-making body responsible for many standards used in LANs.

International Standards Organization (ISO)—An international standards-making body for creating internationally accepted standards. One such standard is Open Systems Interconnection (OSI).

IP—Internet Protocol. A networking protocol for providing a connectionless (datagram) service to the higher transport protocol. It is responsible for discovering and maintaining topology information and for routing packets across homogeneous or heterogeneous networks. Combined with TCP, it is commonly known as the TCP/IP platform.

IPX—A protocol similar to IP that was developed by Novell.

ISDN—Integrated Services Digital Network. An early, CCITT-adopted protocol reference model intended for providing a ubiquitous, end-to-end, interactive, digital service for data, audio, and video.

isochronous—Refers to the fact that a time slot can be divided into equal-size mini-slots allocated to different channels for the synchronous transmission of information (used in DQDB).

ITU-T—International Telecommunications Union-Telecommunications Standards Sector. A formal international standards, specifications, and recommendations body, formerly known as CCITT. ITU-T is part of the International Telecommunications Union (ITU) founded in 1948 and sponsored by the UN to promote telephone and telegraphy issues.

IXC—Inter-Exchange Carrier. A public switching network carrier that provides connectivity across and between LATAs.

jitter—The Cell Delay Variation (CDV).

jitter—Undesirable variation in the arrival time of a transmitted digital signal.

JPEG—Joint Photographic Experts Group. A standard developed for encoding, transmitting, and decoding still images.

Kbps—Kilobits per second. Transmission speed or rate of 1,000 bits per second.

LAN—Local Area Network. A network that interconnects PCs, terminals, workstations, servers, printers, and other peripherals at a high speed over short distances (usually within the same floor or building). Various LAN standards have been developed, with Ethernet as the most widely used.

LAN adapter—A circuit board installed in workstations that connects the workstation with the LAN media.

LAN Emulation—A technique that specifies the interfaces and protocols needed for providing LAN-supported functionality and connectivity in an ATM environment so that legacy protocols can be interoperable with the ATM protocols, interfaces, and devices.

LAN Emulation Service—An ATM Forum-appointed technical workgroup to address LAN Emulation.

LANE—Same as LAN Emulation.

LATA—Local Access and Transport Area. Geographically defined telecommunication areas, within which a local carrier can provide communications services (*see also* LEC and IXC).

layer—In networking architectures, a collection of network processing functions that together comprise a set of rules and standards for successful data communication.

LE—Same as LAN Emulation.

LE-ARP—LAN Emulation ARP. The ARP used in LAN Emulation for binding a requested ATM address to the MAC address.

leaky bucket—A flow control algorithm in which cells are monitored to check whether they comply with the connection parameters. Non-conforming cells are either tagged (as violators) or dropped from the network. The analogy is taken from a bucket (memory buffer) that loses its contents (cells) due to a leak (*see also* GCRA, traffic contract, and UPC).

leased line—A dedicated communications link usually owned by a telecommunications provider that charges for the use of the line.

LEC—(1) LAN Emulation Client. Typically located in an ATM end system (ATM host or LAN switch), its task is to maintain address resolution tables and forward data traffic. It is uniquely associated with an ATM address. (2) Local Exchange Carrier. An intra-LATA communication services provider.

LECS—LAN Emulation Configuration Server. Its main function is to provide configuration information to a LEC (such as the ELAN it belongs to or its LES).

LENNI—LAN Emulation Network Node Interface. Same as LNNI.

LES—LAN Emulation Server. A server that provides support for the LAN emulation address resolution protocol (LE-ARP). The LECs register their own ATM and MAC addresses with the LES, which is uniquely identified by an ATM address.

LI—Length Indicator. A six-bit field in the AAL3/4 SAR-PDU trailer that indicates the number of bytes in the SAR-PDU that contain CPCS information.

LLC—Logical Link Control. The upper half of the Data Link Layer in LANs which performs error control, broadcasting, multiplexing, and flow control functions (*see also* MAC).

LMI—Local Management Interface. An ITU-T-defined interface that provides an ATM end-system user with network management information (*see also* ILMI).

LNNI—LAN Emulation Network Node Interface. Specifies the NNI operation between the LANE servers (LES, LECS, and BUS).

lobe—A term used to describe the connection from a workstation to a Token Ring concentrator, such as a multistation access unit.

local area network (LAN)—A network of two or more computing units connected to share resources over moderate-sized geographic areas such as an office, a building, or a campus.

locally administered address (LAA)—An adapter address that the user can assign to override the universally administered address (UAA).

logical connection—In a network, devices that can communicate or work with one another because they share the same protocol.

logical link—The conceptual joining of two nodes for direct communications. Several logical links may be able to utilize the same physical hardware.

logical link control (LLC)—A protocol developed by the IEEE 802 committee, common to all of its LAN standards, for data link-level transmission control; the upper sublayer of the IEEE Layer 2 (OSI) protocol that complements the media access control protocol; IEEE standard 802.2.

logical link control protocol (LLC protocol)—In a LAN, the protocol that governs the exchange of frames between data stations independently of how the transmission medium is shared.

logical unit (LU)—An access port for users to gain access to the services of a network.

LU-LU session—In SNA, a session between two logical units in an SNA network. It provides communication between two end users or between an end user and an LU services component.

LUNI—LAN Emulation User Network Interface. Specifies the UNI between a LEC and the network providing the LAN Emulation.

MAC—Medium Access Control. A set of protocols that are (the lower) part of the Data Link Layer and consist of the basis of the IEEE LAN specifications. Generally, MAC determines the way devices can transmit in a broadcast network (*see also* LLC).

MAC frame—Frames used to carry information to maintain the ring protocol and for the exchange of management information.

MAC protocol—The LAN protocol sublayer of the data link control (DLC) protocol that includes functions for adapter address recognition, copying of message units from the physical network, and message unit format recognition, error detection, and routing within the processor.

MAN—Metropolitan Area Network. A term to describe a network that provides regional connectivity within a metropolitan area (such as a city). MANs are classified to be between LANs and WANs.

Mbps—Megabits per second. The transmission speed or rate of one million bits per second.

MBS—Maximum Burst Size. A traffic parameter that specifies the maximum number of cells that can be transmitted at the peak rate (PCR).

MCDV—Maximum Cell Delay Variation. As the name suggests, it is the maximum CDV over a given QoS class.

MCLR—Maximum Cell Loss Ratio. As the name suggests, it is the maximum CTD over a given QoS class, defined for CBR and VBR traffic and for cells with a CLP of 0.

MCR—Minimum Cell Rate. A parameter that gives the minimum rate that cells can be transmitted by over a VC.

MCTD—Maximum Cell Transfer Delay. As the name suggests, it is the maximum CTD over a given QoS class.

mean cell transfer delay—The average of the processing, queueing, and propagation delays.

medium access control (MAC)—A media-specific access control protocol within IEEE 802 specifications. The physical address of a station is often called the MAC address.

mesh network—A multisegment network configuration providing more than one path through intermediate LAN segments between source and destination LAN segments.

MIB—Management Information Base. A data structure that defines objects for referencing variables such as integers and strings. In general, it contains information regarding network's management and performance, such as traffic parameters (*see also* ILMI and AToMMIB).

MID—Multiplex Identification. A 10-bit field in the AAL3/4 SAR-PDU header for identifying the different CPCS-PDUs multiplexed over the same VCC.

MIN—Multistage Interconnection Network. A switch fabric built from switching elements organized in a series and/or in parallel for providing physical connections between the inputs and the outputs of a switch.

mips—Million instructions per second. A measure of computer speed.

modem (modulate/demodulator)—A device that converts digital data from a computer to an analog signal that can be transmitted on a telecommunication line and converts the received analog signal to digital data for the computer.

MPEG—Motion Picture Experts Group. A video technology standard that specifies the digital encoding, transmission, and decoding protocols and is capable of presenting VCR-quality motion video.

MPOA—Multiprotocol Over ATM. A set of standards to support (distributed) routing protocols other than IP. Developed on top of LANE and NHRP, it supports switches, route servers, and hosts all attached to an ATM network.

MR—Mean Rate. Same as Average Cell Rate.

multimedia—A way of presenting a combination of different forms of information such as text, data, images, video, audio, and graphics (such as a videoconference).

multiple-domain network—In SNA, a network with more than one SSCP. In APPN, a network with more than one network node.

multiprotocol encapsulation—Multiprotocol Encapsulation over ATM provides for higher protocols, such as IP, to perform bridging and routing functions over an ATM network.

MUX—Multiplexer. A networking local device in which multiple streams of information are combined so they can share a common physical medium.

N-ISDN—Narrowband Integrated Services Digital Network. Predecessor to the B-ISDN, N-ISDN encompasses the original standards for the ISDN.

native applications—Applications that have been developed for non-ATM environment communications platforms (such as LAN applications).

NAU—In SNA, network addressable unit. In APPN, network accessible unit.

NDIS—Network Driver Interface Specification. Generic name for a device driver for a NIC, which is independent of any hardware or software implementation.

NetBIOS (Network Basic Input/Output System)—Provides an interface to allow programs to operate the Token Ring adapter in a personal computer or workstation.

network—A group of nodes and the links interconnecting them.

network addressable unit (NAU) - The origin or destination of data transmitted by the path control layer. Synonymous with network accessible unit.

network management—The conceptual control element of a station that interfaces with all of the architectural layers of that station and is responsible for the resetting and setting of control parameters, obtaining reports of error conditions, and determining if the station should be connected to or disconnected from the network.

network management vector transport (NMVT)—One of the SNA formats used for the transmission of communications and systems management data.

NHRP—Next-Hop Resolution Protocol. A protocol proposed to be used for ATM address resolution based on Classical IP. In particular, if an address request cannot be served by a node, it is forwarded to the next server-node on the path to the destination until finally the ATM-IP address mapping can be accomplished.

NIC—Network Interface Card or Controller. The hardware communications interface (circuit board) required for the DTE (workstation or PC) to access the network (same as Adapter Card).

NMS—Network Management System. Set of OAM&P functions for setting the required hardware and software parameters used in managing a network.

NNI—Network Node Interface (or Network-to-Network Interface). ITU-T-specified standard interface between nodes within the same network. The ATM Forum distinguishes between two standards, one for private networks called P-NNI and one for public networks known as public NNI.

node—A device connected into a network.

node type—The classification of a network device based on the protocols it supports and the network addressable unit it can contain.

noise—A disturbance that affects a signal and that can distort the information carried by the signal.

non-broadcast frame—A frame containing a specific destination address and that may contain routing information specifying which bridges are to forward it. A bridge will forward a non-broadcast frame only if that bridge is included in the frame's routing information.

NPC—Network Parameter Control. Traffic management mechanism (performed at the NNI) exercised by a network for traffic received by another network.

NSAP—Network Services Access Point. In the OSI environment, it is the SAP between the network and the transport layers, which identifies a DTE by a unique address.

OAM—Operations and Maintenance. A set of administrative and supervisory actions regarding network performance monitoring, failure detection, and system protection. Special-type cells are used to carry OAM-related information.

OAM&P—Operations, Administration, Maintenance, and Provisioning. A set of network management functions and services that interact to provide the necessary network management tools and control.

OC-n—Optical Carrier-n. ITU-T-specified physical interface for transmission over optical fiber at n times 51.84 Mbps (OC-3 is at 155.52 Mbps, OC-12 at 622.08 Mbps, and OC-48 at 2.488 Mbps).

octet—Eight bits or one byte.

Open Systems Interconnection (OSI)—The only internationally accepted framework for communication between two systems made by different vendors. It is a seven-layer architecture developed by ISO.

operating system—A software program that manages the basic operating of a computer system.

OSI—Open Systems Interconnection. The OSI Reference Model introduced by the International Organization for Standardization (ISO) consists of seven layers, each specifying the protocols and functions required for two nodes to communicate using the underlying network infrastructure (physical medium, switches, routers, bridges, multiplexers, and intermediate nodes).

OSIRM—Open Systems Interconnection Reference Model. See OSI.

P-NNI—Private Network Node Interface. The NNI used in private networks.

P-UNI—Private User Network Interface. The UNI used between a user and a private network.

pacing—In data communications, a technique by which receiving the receiving device controls the rate of transmission of a sending device to prevent overrun.

parallel bridge—One of the two or more bridges that connects the same two LAN segments in a network.

payload—Part of the ATM cell, it contains the actual information to be carried. It occupies 48 bytes (*see also* PTI).

PBX—Private Branch Exchange. A circuit switch that relays telephones, terminals, or other equipment and provides access to the public telephone system.

PC—Priority Control. A congestion control function that uses the CLP bit to perform priority queueing and scheduling actions.

PCR—Peak Cell Rate. A traffic parameter that characterizes the source and gives the maximum rate at which cells can be transmitted. It is calculated as the reciprocal of the minimum inter-cell interval (time between two cells) over a given VC. The field in the RM cell header indicates the maximum acceptable ER.

PDH—Plesiochronous Digital Hierarchy. A hierarchy that refers to the DS-0, DS-1, DS-2, and DS-3 interfaces for digital transmission. Originally developed to efficiently carry digitized voice over twisted-pair.

PDU—Protocol Data Unit. Term originally used in the OSI model, also known as message, to describe the primitive passed across different layers and contains header, data, and trailer information.

peak duration—A source traffic characteristic that gives the duration of a transmission at the peak cell rate (PCR). It is equivalent to the burst length (in cells).

permanent virtual circuit—A virtual connection established by the network management between an origin and a destination that can be left up permanently (used in X.25 and FR protocols).

PHY—Physical Layer. The bottom layer of the ATM protocol reference model, it is subdivided into two sublayers, the Transmission Convergence (TC) and the Physical Medium (PM). It provides the ATM cells transmission over the physical interfaces that interconnect the ATM devices.

PHY-SAP—Physical Layer Service Access Point. The physical interface at the boundary between the PHY and the ATM layers (*see also* SAP and ATM-SAP).

physical unit (PU)—The component that manages and monitors the resources of a node.

PL—Physical Layer. See PHY.

PLCP—Physical Layer Convergence Protocol. A protocol that specifies a TC mapping of ATM cells to DS-3 frames.

PM—Physical Medium. One of the two PHY sublayers that provides the bit timing and performs the actual transmission of the bits over the physical medium.

PMD—Physical Medium Dependent. Same as PM.

port—A physical connection to the link hardware. Also referred to as an adapter.

PRI—Primary Rate Interface. An ISDN specification that provides 23 64-kbps B-channels and one 64-kbps D-channel intended for use over a single DS1 or a E-1 line.

print server—A computer or program providing LAN users with access to a centralized printer.

private network—A communications network comprised of dedicated circuits between DTEs and other devices (multiplexers, switches, and routers) where bandwidth is dedicated and network management is much simpler (*see also* PVN and Public Network).

protocol—The set of rules governing the operation of functional units of a communication system that must be followed if communication is to be achieved.

PT—Payload Type. See PTI.

PTI—Payload Type Identifier. A three-bit cell header field for encoding information regarding the AAL and EFCI.

public network—A communications network where users have shared access to the network resources. Network services are usually provided by common carriers, such as telephone companies (*see also* Private Network).

PVC—Permanent (or Provisioned) Virtual Connection. A virtual connection (VPC/VCC) provisioned for indefinite use in an ATM network, established by the network management system (NMS) (*see also* SVC).

Q.2110—ITU-T recommendation for specifying the UNI SSCOP.

Q.2130—ITU-T recommendation for specifying the UNI SSCF.

Q.2931—ITU-T recommendation derived from both Q.931 and Q.933 to provide SVC specifications and standards.

Q.931—ITU-T recommendation for specifying the UNI signaling protocol in N-ISDN.

Q.933—ITU-T recommendation for specifying the UNI signaling protocol in Frame Relay.

Q.93B—Currently called Q.2931.

QoS—Quality of Service. A term that refers to the set of ATM performance parameters that characterize the traffic over a given VC. These parameters include the CLR, CER, CMR, CDV, CTD, and the average cell transfer delay.

QoS classes—Quality of Service Classes. Five service classes are defined by the ATM Forum in terms of the QoS parameters. Class 0 refers to best-effort service. Class 1 specifies the parameters for circuit emulation, CBR (uncompressed) video, and for VPN. AAL1 supports this kind of connection-oriented service. Class 2 specifies the parameters for VBR audio and video. AAL2 supports this delay-dependent, connection-oriented class. Class 3 specifies the parameters for connection-oriented data transfer. AAL3/4 and mostly AAL5 supports this delay-independent class of service.

RBOC—Regional Bell Operating Company. Local service telephone companies that resulted from the break-up of AT&T.

RDF—Rate Decrease Factor. A factor by which a source should decrease its transmission rate if there is congestion (*see also* RIF).

RDI—Remote Defect Indication. One of the OAM function types used for fault management (*see also* AIS and CC).

repeater—A device inserted at intervals along a circuit to boost and amplify a signal being transmitted.

RFC—Request for Comment. Draft documents that contain proposed standards and specifications. RFCs can then be approved or just archived as historical recommendations.

RIF—Rate Increase Factor. A factor by which a source can increase its transmission rate if the RM cell indicates no congestion. This can result in a Additive Cell Rate (ACR) (*see also* RDF).

ring error monitor (REM)—A function that compiles error statistics reported by adapters on a network, analyzes the statistics to determine probable error

cause, sends reports to network manager programs, and updates network status conditions. It assists in fault isolation and correction.

ring in (RI)—The receive or input receptacle on an access unit or repeater.

ring out (RO)—The transmit or output receptacle on an access unit or repeater.

RM—Resource Management. The management of critical network resources such as bandwidth and buffers at the node level. A value of six is reserved in the PTI to indicate an RM cell.

route—An ordered sequence between origin and destination stations that represent a path in a network between the stations.

router—An intelligent device that connects two LAN segments that use similar or different architectures at the network layer.

routing—A network management function responsible for forwarding the packets from the source to their destination. Numerous algorithms exist that satisfy various network topologies and requirements.

RSVP—ReSerVation Protocol. A protocol developed for supporting different QoS classes in IP applications (such as in videoconferencing and multimedia).

RTT—Round-Trip Time. The round-trip time between a source and a device, such as a switch, and it is usually measured in number of cells (which depends on the buffering capabilities of the device). It is used as a window in flow control.

SAAL—Signaling AAL Service-specific parts of the AAL protocol responsible for signaling. Its specifications, being developed by ITU-T, were adopted from N-ISDN.

SAP—(1) Service Access Point. The physical interface between the layers in the OSI model through which lower layers provide services to the higher layers passing over the Protocol Data Units (PDUs). (2) Subnetwork Attachment Point. The unique address maintained by a subnetwork for each of the DTEs attached to it.

SAR—Segmentation and Reassembly. The lower half of the AAL. It inserts the data from the information frames into the cell. It adds any necessary header or trailer bits to the data and passes the 48-octet to the ATM layer. Each AAL type has its own SAR format. At the destination, the cell payload is extracted and converted to the appropriate PDU (*see also* CS).

SAR-PDU—Segmentation and Reassembly Protocol Data Unit. The 48-octet PDU that the SAR sublayer exchanges with the ATM layer. It comprises of the SAR-PDU payload and any control information that the SAR sublayer might add.

SCR—Sustainable Cell Rate. A traffic parameter that characterizes a bursty source and specifies the maximum average rate at which cells can be sent over a given VC. It can be defined as the ratio of the MBS to the minimum burst interarrival time.

SDH—Synchronous Digital Hierarchy. A hierarchy that designates signal interfaces for very high-speed digital transmissions over optical fiber links (*see also* SONET).

SEAL—Simple Efficient Adaptation Layer. The original name and recommendation for AAL5.

segment—(1) In an IBM Token Ring network, a portion of a LAN that consists of cables, components, or lobes up to a bridge. (2) An entire ring without bridges.

segment number—The identifier that uniquely distinguishes a LAN segment in a multisegment LAN.

server—A computer providing a service to LAN users. Services may be a shared file.

service access point (SAP)—The point of access to services provided by the layers of a LAN architecture.

service types—There are four service types: CBR, VBR, UBR, and ABR. CBR and VBR are guaranteed services, while UBR and ABR are described as best-effort services.

session—A connection between two stations that allows them to communicate.

SIG—SMDS Interest Group. An industry forum active in producing specifications in the area of SMDS. It has also joined some of the ATM Forum activities.

single-route broadcast—The forwarding of specially designated broadcast frames only by bridges that have single-route broadcast enabled. If the network is configured correctly, a single-route broadcast frame will have exactly one copy delivered to every LAN segment in the network.

SIR—Sustained Information Rate. A flow control mechanism used in SMDS.

SMDS—Switched Multimegabit Digital Service. A connectionless MAN service, based on 53-byte packets, that target the interconnection of different LANs into a switched public network.

SMTP—Simple Mail Transfer Protocol. The protocol standard developed to support e-mail services.

SN—Sequence Number. Part of the header of the SAR-PDU (two bits in AAL1, four bits in AAL3/4), it is used as a sequence counter for detecting lost, out-of-sequence, or misinserted SAR-PDUs.

SNA—Systems Network Architecture. A host-based network architecture introduced by IBM, where logical channels are created between end-points.

SNMP—Simple Network Management Protocol. An IETF-defined standard for handling management information. It is normally found as an application on top of the user datagram protocol (UDP).

SNP—Sequence Number Protection. A four-bit field in the header of the AAL1 SAR-PDU that contains the CRC and the parity bit fields.

soft error—An intermittent error on a network that causes data to be transmitted more than once to be received.

SONET—Synchronous Optical Network. An ANSI-defined standard for high-speed and high-quality digital optical transmission. It has been recognized as the North American standard for SDH.

source routing—A method used by a bridge for moving data between LAN segments. The routing information is embedded in the Token Ring.

source routing transparent (SRT) bridge—A combination bridge utilizing IBM's source-routing mechanism along with a transparent routing mechanism.

SPANS—Simple Protocol for ATM Network Signaling. A protocol supported by FORE Systems switches that provides a SVC tunneling capability over a PVC network.

SPVC—Switched or Semi-Permanent Virtual Connection. A PVC-type connection where SVCs are used for call setup and (automatic) rerouting. It is also called smart PVC.

SS7—Signaling System Number 7. A common channel-signaling standard developed by CCITT. It was designed to provide the internal control and network intelligence needed in ISDNs.

SSCF—Service-Specific Coordination Function. Part of the SSCS portion of the SAAL. Among other functions, it provides a clear interface for relaying user data and providing independence from the underlying sublayers (*see also* SSCOP).

SSCOP—Service-Specific Connection-Oriented Protocol. Part of the SSCS portion of the SAAL. SSCOP is an end-to-end protocol that provides error detection and correction by retransmission and status reporting between the sender and the receiver, while it guarantees delivery integrity (*see also* SSCF).

SSCS—Service-Specific Convergence Sublayer. One of the two components of the Convergence Sublayer (CS) of the AAL that is particular to the traffic service class to be converted. It is developed to support certain user applications such as LAN Emulation, transport of high-quality video, and database management.

SSM—Single Segment Message. A message that constitutes a single PDU.

ST—Segment Type. A two-bit field in the SAR-PDU header that indicates whether the SAR-PDU is a BOM, COM, EOM, or SSM.

station—An input or output device that uses telecommunications facilities.

STDM—Statistical Time-Division Multiplexing. Same as ATDM.

STM—Synchronous Transfer Mode. A packet-switching approach where time is divided in time slots assigned to single channels during which users can transmit periodically. Basically, time slots denote allocated (fixed) parts of the total available bandwidth (*see also* TDM).

STM-1—Synchronous Transport Module-1. An ITU-T-defined SDH physical interface for digital transmission in ATM at the rate of 155.52 Mbps.

STM-n—Synchronous Transport Module-n. An ITU-T-defined SDH physical interface for digital transmission in ATM at n times the basic STM-1 rate. There is a direct equivalence between the STM-n and the SONET STS-3n transmission rates.

STP—Shielded Twisted Pair. Two insulated copper wires twisted together and wrapped by a protective jacket shield (*see also* UTP).

STS-1—Synchronous Transport Signal-1. SONET signal standard for optical transmission at 51.84 Mbps (*see also* OC-1).

STS-n—Synchronous Transport Signal-n. SONET signal format for transmission at n times the basic STS-1 signal (STS-3 is at 155.52 Mbps).

subarea—A portion of an SNA network consisting of the subarea node and any attached resources to that node.

subarea address—A value defined to identify the subarea node and placed in the subarea address field of the network address.

subarea network—The interconnection of subareas.

subarea node—A node that used subarea addressing for routing.

SVC—Switched Virtual Connection. A connection that is set up and taken down dynamically through signaling (*see also* PVC).

switch fabric—The central functional block of the ATM switch, which is responsible for buffering and routing the incoming cells to the appropriate output ports.

switch ATM—An ATM device responsible for switching the cells. Various switch architectures exist that can be classified according to different aspects (buffering, switch matrix, interconnection design, division multiplexing).

switched line—A telecommunications line in which the connection is established by dialing.

switched virtual circuit—A connection in which control signaling is used to establish and tear it down dynamically. Examples are the telephone system, ISDN, and X.25.

symbolic name—A name that can be used instead of an adapter or bridge address to identify an adapter location.

synchronous data link control (SDLC) A bit-oriented synchronous communications protocol developed by IBM.

synchronous time-division multiplexing— A TDM scheme where the interleaved time slots are preassigned to the users.

system services control point (SSCP)—A function within IBM's VTAM that controls and manages an SNA network and its resources.

systems network architecture (SNA)—IBM's seven-layer networking architecture.

T1—A TDM digital channel carrier that operates at a rate of 1.544 Mbps. Known also as a repeater system, it is often referred as DS-1.

T3—A TDM digital channel carrier that operates at 44.736 Mbps. It can multiplex 28 T1 signals and it is often used to refer to DS-3.

TAXI—Transparent Asynchronous Transmitter/Receiver Interface. An interface that provides connectivity over multi-mode fiber links at a speed of 100 Mbps.

TC—Transmission Convergence. One of the two PHY sublayers that is responsible for adapting the ATM cells into a stream of bits to be carried over the physical medium (*see also* PM).

TCP—Transmission Control Protocol. A standardized transport protocol developed for interconnecting IP-based networks. Operating on top of IP (combined known as TCP/IP), it is responsible for multiplexing sessions, error recovery, end-to-end reliable delivery, and flow control.

TCP/IP—A protocol platform, known also as the Internet protocol suite, that combines both TCP and IP. Widely used applications such as Telnet, FTP, and SMTP interface to TCP/IP.

TCS—Transmission Convergence Sublayer. Same as TC.

TDJ—Transfer Delay Jitter. See CDV.

TDM—Time-Division Multiplexing. A technique for splitting the total bandwidth (link capacity) into several channels to allow bit streams to be combined (multiplexed). The bandwidth allocation is done by dividing the time axis into fixed-length slots and a particular channel can then transmit only during a specific time slot.

telephone twisted pair (TTP)—One or more twisted pairs of copper wire in the unshielded voice-grade cable commonly used to connect a telephone to its wall jack. It is also known as unshielded twisted pair (UTP).

Telnet—An asynchronous virtual terminal protocol that allows for remote access.

TM—Traffic Management. A means for providing connection admission (CAC), congestion, and flow control (such as UPC and traffic shaping).

token—A sequence of bits passed from one device to another on the Token Ring network that signifies permission to transmit over the network. It consists of a starting delimiter, an access control field, and an end delimiter.

token passing—In a Token Ring network, the process by which a node captures a token, inserts a message, addresses the token, adds control information, transmits the frame, and then generates another token after the original token has made a complete circuit.

Token Ring—A network with a ring topology that passes tokens from one attaching device to another.

Token Ring interface coupler (TIC)—The hardware interface for connecting front-end processors and controllers to a Token Ring network.

Token Ring network—A network that uses a ring topology in which tokens are passed in a sequence from one node to another.

topology—The physical or logical arrangement of nodes in a computer network.

traffic contract—An agreement between the user and the network management agent regarding the expected QoS provided by the network and the user's compliance with the predetermined traffic parameters (such as PCR, MBS, burstiness, and average cell rate).

traffic descriptors—A set of parameters that characterize the source traffic. These are the PCR, MBS, CDV, and SCR.

traffic shaping—A method for regulating non-complying traffic that violates the traffic parameters, such as PCR, CDV, and MBS, as specified by the traffic contract (*see also* GCRA).

Transmission Control Protocol/Internet Protocol (TCP/IP) - A set of protocols that allows cooperating computers to share resources across a heterogeneous network.

transmission group (TG)—A single link or a group of links between adjacent nodes logically grouped together. In SNA, these nodes are adjacent subarea nodes. In APPN, it is a single link.

transparent routing—A method used by a bridge for moving data between two networks through learning the station addressees on each network.

twisted pair—A transmission medium that consists of two insulated conductors twisted together to reduce noise.

type 2.0 (T2.0) node—A node that attaches to a subarea network as a peripheral node and provides full end-user services but no intermediate routing services.

type 2.1 (T2.1) node—An SNA node that can be configured as an end point or intermediate routing node in a T2.1 network, or as a peripheral node attached to a subarea network. It can act as an end node, network node, or an intermediate node in an APPN network.

type 4 node—An SNA subarea node that provides routing and data link control functions for a type 5 node. Type 5 nodes control type 4 nodes.

type 5 node—An SNA subarea node that contains an SSCP and controls type 4 and type 2 SNA node types.

UBR—Unspecified Bit Rate. One of the best-effort service types (the other one is ABR). Realistically, no traffic parameters are specified by the source, so no actual quality commitment is made by the network management.

UDP—User Datagram Protocol. A connectionless transport protocol without any guarantee of packet sequence or delivery. It functions directly on top of IP.

UME—UNI Management Entity. Software at the UNIs for providing the ILMI functions.

UNI—User-Network Interface. Defined as a set of protocols and traffic characteristics (such as a cell structure), the interface between the CPE (user) and the ATM network (ATM switch). The ATM Forum specifications refer to two standards being developed, one between a user and a public ATM network called public UNI, and one between a user and a private ATM network called P-UNI.

UNI 2.0—An ATM Forum UNI specification for the physical (PHY) and ATM layers, the ILMI, OAM (traffic control), and PVC support.

UNI 3.0—An upgrade of UNI 2.0 with traffic control for PCR and the operation over current transmission systems as some of the additional features.

UNI 3.1—A corrected version of UNI 3.0, this specification also includes SSCOP standards.

UNI 4.0—This UNI specification refers to signaling issues in ABR and VP and QoS negotiation.

universally administered address (UAA)—The address permanently encoded in an adapter at the rime of manufacture. All universally administered addresses are unique.

unnumbered acknowledgment (UA)—A data link control command used in establishing a link and in answering receipt of logical link control frames.

unshielded twisted pair (UTP)—See telephone twisted pair.

UPC—Usage Parameter Control. A form of traffic control that checks and enforces a user's conformance with the traffic contract and the QoS parameters. Commonly known as traffic policing, it is performed at the UNI level.

UTOPIA—Universal Test and Operation Physical Interface. An interface to provide connectivity at the PHY level among ATM entities.

UTP—Unshielded Twisted Pair. A twisted-pair (copper) wire without any protective sheathing, used for short-distance wiring (such as building). Two categories are specified by the ATM Forum for cell transmission: 3 (CAT-3) and 5 (CAT-5).

VBR-NRT—Variable Bit Rate-Non-Real Time. One of the service types for transmitting traffic where timing information is not critical and that is characterized by the average and peak cell rates. It is well-suited for long data packet transfers.

VBR-RT—Variable Bit Rate-Real Time. One of the service types for transmitting traffic that depends on timing information and control. It is characterized by the average and peak cell rates. VBR-RT is suitable for carrying traffic such as packetized (compressed) video and audio.

VC—Virtual Channel. A term to describe the unidirectional flow of ATM cells between connecting (switching or end-user) points that share a common identifier number (VCI).

VCC—Virtual Channel Connection. Defined as a concatenation of virtual channel links.

VCI—Virtual Channel Identifier. A 16-bit value in the ATM cell header that provides a unique identifier for the VC that carries that particular cell.

VF—Variance Factor. It is the CRM normalized by the variance of the total cell rate over a given circuit.

virtual channel—See VC.

virtual circuit—A connection setup across the network between a source and a destination where a fixed route is chosen for the entire session and bandwidth is dynamically allocated (*see also* datagram).

virtual connection—A connection established between end-users (source and destination) where packets are forwarded along the same path and bandwidth is not permanently allocated until it is used.

Virtual Telecommunications Access Method (VTAM)—A set of programs that control communication between nodes and application programs in SNA.

VLAN—Virtual LAN. A networking environment where users on physically independent LANs are interconnected in such a way that it appears as if they are on the same LAN workgroup (*see also* LANE).

VOD—Video on Demand. A technology that enables the customer to remotely select and play a video transmitted over communications links.

VP—Virtual Path. A term to describe a set of VCs grouped together between crosspoints (switches, in other words).

VPC—Virtual Path Connection. Defined as a concatenation of VP links.

VPCI/VCI—Virtual Path Connection Identifier/Virtual Channel Identifier. A combination of two numbers, one for identifying the VP and one for VCI.

VPI—Virtual Path Identifier. An eight-bit value in the cell header that identifies the VP and accordingly the virtual channel to which the cell belongs.

VPN—Virtual Private Network. Network resources provided to users on demand by public carriers such that the users view this partition of the network as a private network. The advantage of the VPNs over the dedicated private networks is that the former allows a dynamic allocation of network resources.

WAN—Wide Area Network. A network that covers long-haul areas and usually utilizes public telephone circuits. The LAN segments are bridged or routed using communication lines, increasing the geographic size of the LAN.

WATM—Wireless ATM. An emerging technology for interfacing wireless and ATM networks.

workstation—A terminal or computer attached to a network.

X.25—One of the first CCITT-standardized public (data) packet-switching network protocols. Originally designed to operate over unreliable communications links, it supports both VC and datagram services.

INDEX

Note: Boldface numbers indicate illustrations.

CISCO / NETWORKING $70.00 U.S.A.

A complete configuration guide to Cisco routers!

With the Cisco Usenet Group handling more than 600 messages a day on Cisco router configuration, there's clear evidence that network administrators and engineers are hungry for complete and practical fast-access information on the world's most popular routers.

The *Cisco Router Handbook* from today's leader in Cisco publishing gives you all the practical guidance you'll ever need on Cisco router configuration. This hands-on A-to-Z compendium defines, analyzes, and describes the most common router configurations you're likely to encounter. This single self-contained resource contains:

- Detailed coverage of all routing protocols that can run on a Cisco network, including RIP, RIPv2, IGRP, EIGRP, and OSPF
- In-depth analysis of SNA Transport over Cisco routers
- Practical insight into connectivity to multiple network infrastructures
- A discussion of security issues, including how to create, apply, monitor, and maintain access lists.

Offering one-stop, field-tested solutions you won't find anywhere else, the *Cisco Router Handbook* provides both the nuts-and-bolts instruction basic technicians count on and the higher-level guidance that discriminating network professionals demand.

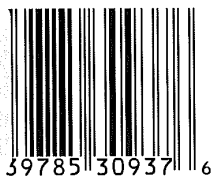
A VOLUME IN THE CISCO TECHNICAL EXPERT SERIES

About the Author

GEORGE C. SACKETT is Managing Director of NetworX Corporation, a Cisco professional service provider that develops corporate data networks for the telecommunications, entertainment, medical, financial, and transportation industries. The author of other McGraw-Hill titles on Internetworking, he has more than 17 years of technical and managerial experience with corporate data networks.

ISBN 0-07-058098-7

Visit us on the World Wide Web at **www.computing.mcgraw-hill.com**



Cover Design: 12E Design
Cover Photo: Chip Simons/Photonica

McGraw-Hill
A Division of The McGraw-Hill Companies