

Configuring IP

This chapter describes how to configure the Internet Protocol (IP). For a complete description of the commands in this chapter, refer to the “IP Commands” chapter of the *Network Protocols Command Reference, Part 1*. For information on configuring the various IP routing protocols, refer to the “Configuring IP Routing Protocols” chapter of this manual.

IP Configuration Task List

A number of tasks are associated with configuring IP. A basic and required task for configuring IP is to assign IP addresses to network interfaces. Doing so enables the interfaces and allows communication with hosts on those interfaces using IP. Associated with this task are decisions about subnetting and masking the IP addresses.

To configure IP, complete the tasks in the following sections:

- Assign IP Addresses to Network Interfaces
- Configure Address Resolution Methods
- Disable IP Routing
- Enable IP Bridging
- Enable Integrated Routing and Bridging
- Configure a Routing Process
- Configure Broadcast Packet Handling
- Configure IP Services
- Filter IP Packets
- Configure Network Address Translation (NAT)
- Configure the Hot Standby Router Protocol
- Configure Basic IP Security Options
- Configure Extended IP Security Options
- Configure the DNSIX Audit Trail Facility
- Configure IP Accounting
- Configure Performance Parameters
- Configure IP over WANs
- Monitor and Maintain the IP Network

Remember that not all the tasks in these sections are required. The tasks you must perform will depend on your network and your needs.

At the end of this chapter, the examples in the “IP Configuration Examples” section illustrate how you might configure your network using IP.

Assign IP Addresses to Network Interfaces

An IP address identifies a location to which IP datagrams can be sent. Some IP addresses are reserved for special uses and cannot be used for host, subnet, or network addresses. Table 1 lists ranges of IP addresses, and shows which addresses are reserved and which are available for use.

Table 1 Reserved and Available IP Addresses

Class	Address or Range	Status
A	0.0.0.0	Reserved
	1.0.0.0 through 126.0.0.0	Available
	127.0.0.0	Reserved
B	128.0.0.0	Reserved
	128.1.0.0 through 191.254.0.0	Available
	191.255.0.0	Reserved
C	192.0.0.0	Reserved
	192.0.1.0 through 223.255.254	Available
	223.255.255.0	Reserved
D	224.0.0.0 through 239.255.255.255	Multicast group addresses
E	240.0.0.0 through 255.255.255.254	Reserved
	255.255.255.255	Broadcast

The official description of IP addresses is found in RFC 1166, “Internet Numbers.”

To receive an assigned network number, contact your Internet service provider.

An interface can have one primary IP address. To assign a primary IP address and a network mask to a network interface, perform the following task in interface configuration mode:

Task	Command
Set a primary IP address for an interface.	ip address <i>ip-address mask</i>

A mask identifies the bits that denote the network number in an IP address. When you use the mask to subnet a network, the mask is then referred to as a *subnet mask*.

Note We only support network masks that use contiguous bits that are flush left against the network field.

The tasks required to enable additional, optional, IP addressing features are contained in the following sections:

- Assign Multiple IP Addresses to Network Interfaces
- Enable Use of Subnet Zero
- Enable Classless Routing Behavior
- Enable IP Processing on a Serial Interface

Assign Multiple IP Addresses to Network Interfaces

The software supports multiple IP addresses per interface. You can specify an unlimited number of secondary addresses. Secondary IP addresses can be used in a variety of situations. The following are the most common applications:

- There might not be enough host addresses for a particular network segment. For example, suppose your subnetting allows up to 254 hosts per logical subnet, but on one physical subnet you must have 300 host addresses. Using secondary IP addresses on the routers or access servers allows you to have two logical subnets using one physical subnet.
- Many older networks were built using Level 2 bridges, and were not subnetted. The judicious use of secondary addresses can aid in the transition to a subnetted, router-based network. Routers on an older, bridged segment can easily be made aware that many subnets are on that segment.
- Two subnets of a single network might otherwise be separated by another network. You can create a single network from subnets that are physically separated by another network by using a secondary address. In these instances, the first network is *extended*, or layered on top of the second network. Note that a subnet cannot appear on more than one active interface of the router at a time.

Note If any router on a network segment uses a secondary address, all other routers on that same segment must also use a secondary address from the same network or subnet.

To assign multiple IP addresses to network interfaces, perform the following task in interface configuration mode:

Task	Command
Assign multiple IP addresses to network interfaces.	ip address <i>ip-address mask secondary</i>

Note IP routing protocols sometimes treat secondary addresses differently when sending routing updates. See the description of IP split horizon in the “Configuring IP Routing Protocols” chapter for details.

See the “Creating a Network from Separated Subnets Example” section at the end of this chapter for an example of creating a network from separated subnets.

Enable Use of Subnet Zero

Subnetting with a subnet address of zero is illegal and strongly discouraged (as stated in RFC 791) because of the confusion that can arise between a network and a subnet that have the same addresses. For example, if network 131.108.0.0 is subnetted as 255.255.255.0, subnet zero would be written as 131.108.0.0—which is identical to the network address.

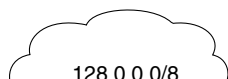
You can use the all zeros and all ones subnet (131.108.255.0), even though it is discouraged. Configuring interfaces for the all ones subnet is explicitly allowed. However, if you need the entire subnet space for your IP address, perform the following task in global configuration mode to enable subnet zero:

Task	Command
Enable the use of subnet zero for interface addresses and routing updates.	<code>ip subnet-zero</code>

Enable Classless Routing Behavior

At times, a router might receive packets destined for a subnet of a network that has no network default route. Figure 2 shows a router in network 128.20.0.0 connected to subnets 128.20.1.0, 128.20.2.0, and 128.20.3.0. Suppose the host sends a packet to 120.20.4.1. By default, if the router receives a packet destined for a subnet it does not recognize, and there is no network default route, the router discards the packet.

Figure 2 No IP Classless Routing



In Figure 3, classless routing is enabled in the router. Therefore, when the host sends a packet to 120.20.4.1, instead of discarding the packet, the router forwards the packet to the best supernet route.

Figure 3 IP Classless Routing

To have the Cisco IOS software forward packets destined for unrecognized subnets to the best supernet route possible, perform the following task in global configuration mode:

Task	Command
Enable classless routing behavior.	ip classless

Enable IP Processing on a Serial Interface

You might want to enable IP processing on a serial or tunnel interface without assigning an explicit IP address to the interface. Whenever the unnumbered interface generates a packet (for example, for a routing update), it uses the address of the interface you specified as the source address of the IP packet. It also uses the specified interface address in determining which routing processes are sending updates over the unnumbered interface. Restrictions are as follows:

- Serial interfaces using HDLC, PPP, LAPB, and Frame Relay encapsulations, as well as SLIP and tunnel interfaces, can be unnumbered. Serial interfaces using Frame Relay encapsulation can also be unnumbered, but the interface must be a point-to-point subinterface. It is not possible to use the unnumbered interface feature with X.25 or SMDS encapsulations.
- You cannot use the **ping EXEC** command to determine whether the interface is up, because the interface has no IP address. The Simple Network Management Protocol (SNMP) can be used to remotely monitor interface status.
- You cannot netboot a runnable image over an unnumbered serial interface.
- You cannot support IP security options on an unnumbered interface.

If you are configuring Intermediate System-to-Intermediate System (IS-IS) across a serial line, you should configure the serial interfaces as unnumbered. This allows you to conform with RFC 1195, which states that IP addresses are not required on each interface.

Note Using an unnumbered serial line between different major networks requires special care. If, at each end of the link, there are different major networks assigned to the interfaces you specified as unnumbered, any routing protocols running across the serial line should be configured to not advertise subnet information.

To enable IP processing on an unnumbered serial interface, perform the following task in interface configuration mode:

Task	Command
Enable IP processing on a serial or tunnel interface without assigning an explicit IP address to the interface.	ip unnumbered <i>type number</i>

The interface you specify must be the name of another interface in the router that has an IP address, not another unnumbered interface.

The interface you specify also must be enabled (listed as “up” in the **show interfaces** command display).

See the “Serial Interfaces Configuration Example” section at the end of this chapter for an example of how to configure serial interfaces.

Configure Address Resolution Methods

Our IP implementation allows you to control interface-specific handling of IP addresses by facilitating address resolution, name services, and other functions. The following sections describe how to configure address resolution methods:

- Establish Address Resolution
- Map Host Names to IP Addresses
- Configure HP Probe Proxy Name Requests
- Configure the Next Hop Resolution Protocol

Establish Address Resolution

A device in the IP can have both a local address (which uniquely identifies the device on its local segment or LAN) and a network address (which identifies the network to which the device belongs). The local address is more properly known as a *data link* address because it is contained in the data link layer (Layer 2 of the OSI model) part of the packet header and is read by data link devices (bridges and all device interfaces, for example). The more technically inclined will refer to local addresses as *MAC addresses*, because the Media Access Control (MAC) sublayer within the data link layer processes addresses for the layer.

To communicate with a device on Ethernet, for example, the Cisco IOS software first must determine the 48-bit MAC or local data link address of that device. The process of determining the local data link address from an IP address is called *address resolution*. The process of determining the IP address from a local data link address is called *reverse address resolution*. The software uses three forms of address resolution: Address Resolution Protocol (ARP), proxy ARP, and Probe (similar to ARP). The software also uses the Reverse Address Resolution Protocol (RARP). The ARP, proxy ARP, and RARP protocols are defined in RFCs 826, 1027, and 903, respectively. Probe is a protocol developed by the Hewlett-Packard Company (HP) for use on IEEE-802.3 networks.

The Address Resolution Protocol (ARP) is used to associate IP addresses with media or MAC addresses. Taking an IP address as input, ARP determines the associated media address. Once a media or MAC address is determined, the IP address/media address association is stored in an ARP cache for rapid retrieval. Then the IP datagram is encapsulated in a link-layer frame and sent over the network. Encapsulation of IP datagrams and ARP requests and replies on IEEE 802 networks other than Ethernet is specified by the Subnetwork Access Protocol (SNAP).

The Reverse Address Resolution Protocol (RARP) works the same way as ARP, except that the RARP Request packet requests an IP address instead of a local data link address. Use of RARP requires a RARP server on the same network segment as the router interface. RARP often is used by diskless nodes that do not know their IP addresses when they boot. The Cisco IOS software attempts to use RARP if it does not know the IP address of an interface at startup. Also, our routers are able to act as RARP servers by responding to RARP requests that they are able to answer. See the “Loading Images and Configuration Files” chapter in the *Configuration Fundamentals Configuration Guide* to learn how to configure a router as a RARP server.

Perform the following tasks to set address resolution:

- Define a Static ARP Cache
- Set ARP Encapsulations
- Disable Proxy ARP
- Configure Local-Area Mobility

The procedures for performing these tasks are described in the following sections.

Define a Static ARP Cache

ARP and other address resolution protocols provide a dynamic mapping between IP addresses and media addresses. Because most hosts support dynamic address resolution, you generally do not need to specify static ARP cache entries. If you must define them, you can do so globally. Doing this task installs a permanent entry in the ARP cache. The Cisco IOS software uses this entry to translate 32-bit IP addresses into 48-bit hardware addresses.

Optionally, you can specify that the software respond to ARP requests as if it was the owner of the specified IP address. You also have the option of specifying an ARP entry timeout period when you define ARP entries.

The following two tables list the tasks to provide static mapping between IP addresses and media address.

Perform either of the following tasks in global configuration mode:

Task	Command
Globally associate an IP address with a media (hardware) address in the ARP cache.	arp <i>ip-address hardware-address type</i>
Specify that the software respond to ARP requests as if it was the owner of the specified IP address.	arp <i>ip-address hardware-address type alias</i>

Perform the following task in interface configuration mode:

Task	Command
Set the length of time an ARP cache entry will stay in the cache.	arp timeout <i>seconds</i>

To display the type of ARP being used on a particular interface and also display the ARP timeout value, use the **show interfaces EXEC** command. Use the **show arp EXEC** command to examine the contents of the ARP cache. Use the **show ip arp EXEC** command to show IP entries. To remove all nonstatic entries from the ARP cache, use the privileged EXEC command **clear arp-cache**.

Set ARP Encapsulations

By default, standard Ethernet-style ARP encapsulation (represented by the **arpa** keyword) is enabled on the IP interface. You can change this encapsulation method to SNAP or HP Probe, as required by your network, to control the interface-specific handling of IP address resolution into 48-bit Ethernet hardware addresses.

When you set HP Probe encapsulation, the Cisco IOS software uses the Probe protocol whenever it attempts to resolve an IEEE-802.3 or Ethernet local data link address. The subset of Probe that performs address resolution is called Virtual Address Request and Reply. Using Probe, the router can communicate transparently with Hewlett-Packard IEEE-802.3 hosts that use this type of data encapsulation. You must explicitly configure all interfaces for Probe that will use Probe.

To specify the ARP encapsulation type, perform the following task in interface configuration mode:

Task	Command
Specify one of three ARP encapsulation methods for a specified interface.	arp {arpa probe snap}

Disable Proxy ARP

The Cisco IOS software uses proxy ARP (as defined in RFC 1027) to help hosts with no knowledge of routing determine the media addresses of hosts on other networks or subnets. For example, if the router receives an ARP request for a host that is not on the same interface as the ARP request sender, and if the router has all of its routes to that host through other interfaces, then it generates a proxy ARP reply packet giving its own local data link address. The host that sent the ARP request then sends its packets to the router, which forwards them to the intended host. Proxy ARP is enabled by default.

To disable proxy ARP, perform the following task in interface configuration mode (as necessary) for your network:

Task	Command
Disable proxy ARP on the interface.	no ip proxy-arp

Configure Local-Area Mobility

Local-area mobility provides the ability to relocate IP hosts within a limited area without reassigning host IP addresses and without changes to the host software. Local-area mobility is supported on Ethernet, Token Ring, and FDDI interfaces only.

To create a mobility area with only one router, perform the following tasks:

Task	Command
Step 1 Enable bridging.	bridge group protocol {dec ieee}¹
Step 2 Enter interface configuration mode.	interface type number
Step 3 Enable local-area mobility.	ip mobile arp [timers keepalive hold-time] [access-group access-list-number name]
Step 4 Configure bridging on the interface.	bridge-group group¹

1. This command is documented in the “Transparent Bridging Commands” chapter of the *Bridging and IBM Networking Command Reference*.

To create larger mobility areas, you must first redistribute the mobile routes into your IGP. The IGP must support host routes. You can use Enhanced IGRP, OSPF, or IS-IS; you can also use RIP in some cases, but this is not recommended. To redistribute the mobile routes into your existing IGP configuration, perform the following tasks:

Task	Command
Step 1 Enter router configuration mode.	router { eigrp <i>autonomous-system</i> isis [<i>tag</i>] ospf <i>process-id</i> }
Step 2 Set default metric values.	default-metric <i>number</i> or default-metric <i>bandwidth delay reliability loading mtu</i>
Step 3 Redistribute the mobile routes.	redistribute mobile

If your IGP supports summarization, you should also restrict the mobile area so that it falls completely inside an IGP summarization area. This lets hosts roam within the mobile area without affecting routing outside the area.

The mobile area must consist of a contiguous set of subnets.

Hosts that roam within a mobile area should rely on a configured default router for their routing.

Map Host Names to IP Addresses

Each unique IP address can have a host name associated with it. The Cisco IOS software maintains a cache of host name-to-address mappings for use by the EXEC **connect**, **telnet**, **ping**, and related Telnet support operations. This cache speeds the process of converting names to addresses.

IP defines a naming scheme that allows a device to be identified by its location in the IP. This is a hierarchical naming scheme that provides for *domains*. Domain names are pieced together with periods (.) as the delimiting characters. For example, Cisco Systems is a commercial organization that the IP identifies by a *com* domain name, so its domain name is *cisco.com*. A specific device in this domain, the File Transfer Protocol (FTP) system for example, is identified as *ftp.cisco.com*.

To keep track of domain names, IP has defined the concept of a *name server*, whose job it is to hold a cache (or database) of names mapped to IP addresses. To map domain names to IP addresses, you must first identify the host names, then specify a name server, and enable the Domain Naming System (DNS), the Internet's global naming scheme that uniquely identifies network devices. You do these by performing the tasks described in the following sections:

- Map IP Addresses to Host Names
- Specify the Domain Name
- Specify a Name Server
- Disable the DNS
- Use the DNS to Discover ISO CLNS Addresses

Map IP Addresses to Host Names

The Cisco IOS software maintains a table of host names and their corresponding addresses, also called a *host name-to-address mapping*. Higher-layer protocols such as Telnet use host names to identify network devices (hosts). The router and other network devices must be able to associate host names with IP addresses to communicate with other IP devices. Host names and IP addresses can be associated with one another through static or dynamic means.

Manually assigning host names to addresses is useful when dynamic mapping is not available.

To assign host names to addresses, perform the following task in global configuration mode:

Task	Command
Statically associate host names with IP addresses.	ip host <i>name</i> [<i>tcp-port-number</i>] <i>address1</i> [<i>address2...address8</i>]

Specify the Domain Name

You can specify a default domain name that the Cisco IOS software will use to complete domain name requests. You can specify either a single domain name or a list of domain names. Any IP host name that does not contain a domain name will have the domain name you specify appended to it before being added to the host table.

To specify a domain name or names, perform either of the following tasks in global configuration mode:

Task	Command
Define a default domain name that the Cisco IOS software will use to complete unqualified host names.	ip domain-name <i>name</i>
Define a list of default domain names to complete unqualified host names.	ip domain-list <i>name</i>

See the “IP Domains Example” section at the end of this chapter for an example of establishing IP domains.

Specify a Name Server

To specify one or more hosts (up to six) that can function as a name server to supply name information for the DNS, perform the following task in global configuration mode:

Task	Command
Specify one or more hosts that supply name information.	ip name-server <i>server-address1</i> [[<i>server-address2</i>]... <i>server-address6</i>]

Disable the DNS

If your network devices require connectivity with devices in networks for which you do not control name assignment, you can assign device names that uniquely identify your devices within the entire internetwork. The Internet’s global naming scheme, the DNS, accomplishes this task. This service is enabled by default.

To disable the DNS, perform the following task in global configuration mode:

Task	Command
Disable DNS-based host name-to-address translation.	no ip domain-lookup

See the “Dynamic Lookup Example” section at the end of this chapter for an example of enabling the DNS.

Use the DNS to Discover ISO CLNS Addresses

If your router has both IP and International Organization for Standardization Connectionless Network Service (ISO CLNS) enabled and you want to use ISO CLNS Network Service Access Point (NSAP) addresses, you can use the DNS to query these addresses, as documented in RFC 1348. This feature is enabled by default.

To disable DNS queries for ISO CLNS addresses, perform the following task in global configuration mode:

Task	Command
Disable DNS queries for ISO CLNS addresses.	no ip domain-lookup nsap

Configure HP Probe Proxy Name Requests

HP Probe Proxy support allows the Cisco IOS software to respond to HP Probe Proxy name requests. These requests are typically used at sites that have Hewlett-Packard equipment and are already using HP Probe Proxy. Tasks associated with HP Probe Proxy are shown in the following two tables.

To configure HP Probe Proxy, perform the following task in interface configuration mode:

Task	Command
Allow the Cisco IOS software to respond to HP Probe Proxy name requests.	ip probe proxy

Perform the following task in global configuration mode:

Task	Command
Enter the host name of an HP host (for which the router is acting as a proxy) into the host table.	ip hp-host <i>hostname ip-address</i>

See the “HP Hosts on a Network Segment Example” section at the end of this chapter for an example of configuring HP hosts on a network segment.

Configure the Next Hop Resolution Protocol

Routers, access servers, and hosts can use Next Hop Resolution Protocol (NHRP) to discover the addresses of other routers and hosts connected to a nonbroadcast, multiaccess (NBMA) network. Partially meshed NBMA networks are typically configured with multiple logical networks to provide full network layer connectivity. In such configurations, packets might make several hops over the NBMA network before arriving at the exit router (the router nearest the destination network). In addition, such NBMA networks (whether partially or fully meshed) typically require tedious static configurations. These static configurations provide the mapping between network layer addresses (such as IP) and NBMA addresses (such as E.164 addresses for Switched Multimegabit Data Service, or SMDS).

NHRP provides an ARP-like solution that alleviates these NBMA network problems. With NHRP, systems attached to an NBMA network dynamically learn the NBMA address of the other systems that are part of that network, allowing these systems to directly communicate without requiring traffic to use an intermediate hop.

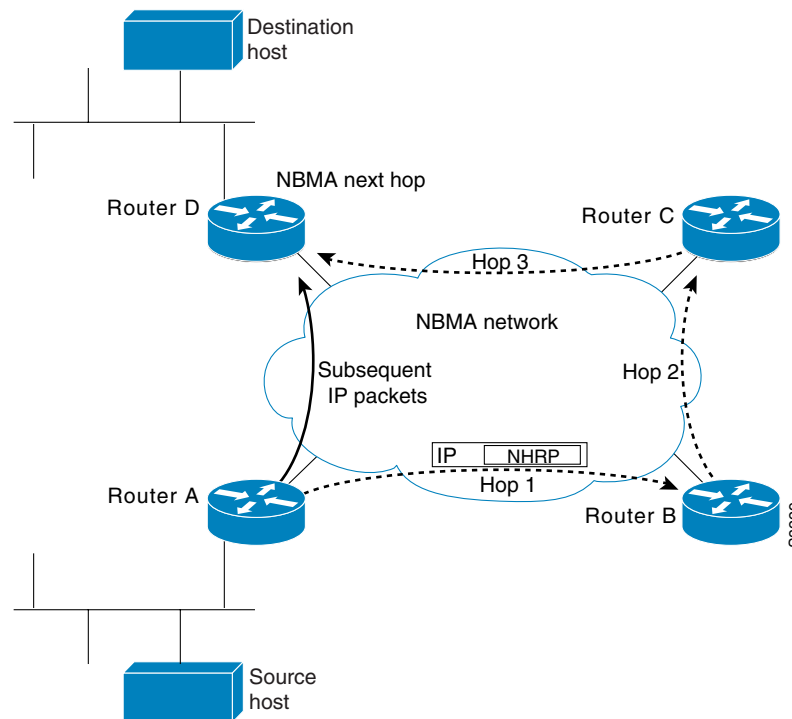
The NBMA network is considered nonbroadcast either because it technically does not support broadcasting (for example, an X.25 network) or because broadcasting is too expensive (for example, an SMDS broadcast group that would otherwise be too large).

Cisco's Implementation of NHRP

Cisco's implementation of NHRP supports IP Version 4, Internet Packet Exchange (IPX) network layers, and, at the link layer, ATM, Ethernet, SMDS, and multipoint tunnel networks. Although NHRP is available on Ethernet, it is not necessary to implement NHRP over Ethernet media because Ethernet is capable of broadcasting. Ethernet support is unnecessary (and not provided) for IPX.

Figure 4 illustrates four routers connected to an NBMA network. Within the network are ATM or SMDS switches necessary for the routers to communicate with each other. Assume that the switches have virtual circuit connections represented by hops 1, 2, and 3 of the figure. When Router A attempts to forward an IP packet from the source host to the destination host, NHRP is triggered. On behalf of the source host, Router A sends an NHRP request packet encapsulated in an IP packet, which takes three hops across the network to reach Router D, connected to the destination host. After receiving a positive NHRP reply, Router D is determined to be the "NBMA next hop," and Router A sends subsequent IP packets for the destination to Router D in one hop.

Figure 4 Next Hop Resolution Protocol (NHRP)



With NHRP, once the NBMA next hop is determined, the source either starts sending data packets to the destination (in a connectionless NBMA network such as SMDS) or establishes a virtual circuit connection to the destination with the desired bandwidth and quality of service (QOS) characteristics (in a connection-oriented NBMA network such as ATM).

Other address resolution methods can be used while NHRP is deployed. IP hosts that rely upon the LIS (Logical IP Subnet) model might require ARP servers and services over NBMA networks, and deployed hosts might not implement NHRP, but might continue to support ARP variations. NHRP is designed to eliminate the suboptimal routing that results from the LIS model, and can be deployed with existing ARP services without interfering with them.

NHRP is used to facilitate building a virtual private network. In this context, a virtual private network consists of a virtual Layer 3 network that is built on top of an actual Layer 3 network. The topology you use over the virtual private network is largely independent of the underlying network, and the protocols you run over it are completely independent of it.

Connected to the NBMA network are one or more stations that implement NHRP, and are known as *Next Hop Servers*. All routers running Release 10.3 or later are capable of implementing NHRP and, thus, can act as Next Hop Servers.

Each Next Hop Server serves a set of destination hosts, which might or might not be directly connected to the NBMA network. Next Hop Servers cooperatively resolve the NBMA next hop addresses within their NBMA network. In addition to NHRP, Next Hop Servers typically participate in protocols used to disseminate routing information across (and beyond the boundaries of) the NBMA network, and might support ARP service also.

A Next Hop Server maintains a “next-hop resolution” cache, which is a table of network layer address to NBMA address mappings. The table is created from information gleaned from NHRP register packets, extracted from NHRP request or reply packets that traverse the Next Hop Server as they are forwarded, or through other means such as ARP and preconfigured tables.

Protocol Operation

NHRP requests traverse one or more hops within an NBMA subnetwork before reaching the station that is expected to generate a response. Each station (including the source station) chooses a neighboring Next Hop Server to forward the request to. The Next Hop Server selection procedure typically involves performing a routing decision based upon the network layer destination address of the NHRP request. Ignoring error situations, the NHRP request eventually arrives at a station that generates an NHRP reply. This responding station either serves the destination, is the destination itself, or is a client that specified it should receive NHRP requests when it registered with its server. The responding station generates a reply using the source address from within the NHRP packet to determine where the reply should be sent.

NHRP Configuration Task List

To configure NHRP, perform the tasks described in the following sections. The first task is required, the remainder are optional.

- Enable NHRP on an Interface
- Configure a Station’s Static IP-to-NBMA Address Mapping
- Statically Configure a Next Hop Server
- Configure NHRP Authentication
- Control NHRP Rate
- Suppress Forward and Reverse Record Options
- Specify the NHRP Responder Address
- Change the Time Period NBMA Addresses Are Advertised as Valid
- Configure a GRE Tunnel for Multipoint Operation

Enable NHRP on an Interface

To enable NHRP for an interface on a router, perform the following task in interface configuration mode. In general, all NHRP stations within a logical NBMA network must be configured with the same network identifier.

Task	Command
Enable NHRP on an interface.	ip nhrp network-id <i>number</i>

See the “Logical NBMA Example” section and the “NHRP over ATM Example” section at the end of this chapter for examples of enabling NHRP.

Configure a Station’s Static IP-to-NBMA Address Mapping

To participate in NHRP, a station connected to an NBMA network should be configured with the IP and NBMA addresses of its Next Hop Server(s). The format of the NBMA address depends on the medium you are using. For example, ATM uses an NSAP address, Ethernet uses a MAC address, and SMDS uses an E.164 address.

These Next Hop Servers may also be the stations’s default or peer routers, so their addresses can be obtained from the station’s network layer forwarding table.

If the station is attached to several link layer networks (including logical NBMA networks), the station should also be configured to receive routing information from its Next Hop Server(s) and peer routers so that it can determine which IP networks are reachable through which link layer networks.

To configure static IP-to-NBMA address mapping on a station (host or router), perform the following task in interface configuration mode:

Task	Command
Configure static IP-to-NBMA address mapping.	ip nhrp map <i>ip-address nbma-address</i>

Statically Configure a Next Hop Server

A Next Hop Server normally uses the network layer forwarding table to determine where to forward NHRP packets, and to find the egress point from an NBMA network. A Next Hop Server may alternately be statically configured with a set of IP address prefixes that correspond to the IP addresses of the stations it serves, and their logical NBMA network identifiers.

To statically configure a Next Hop Server, perform the following task in interface configuration mode:

Task	Command
Statically configure a Next Hop Server.	ip nhrp nhs <i>nhs-address</i> [<i>net-address</i> [<i>netmask</i>]]

To configure multiple networks that the Next Hop Server serves, repeat the **ip nhrp nhs** command with the same Next Hop Server address, but different IP network addresses. To configure additional Next Hop Servers, repeat the **ip nhrp nhs** command.

Configure NHRP Authentication

Configuring an authentication string ensures that only routers configured with the same string can intercommunicate using NHRP. Therefore, if the authentication scheme is to be used, the same string must be configured in all devices configured for NHRP on a fabric. To specify the authentication string for NHRP on an interface, perform the following task in interface configuration mode:

Task	Command
Specify an authentication string.	ip nhrp authentication <i>string</i>

Control NHRP Rate

The three ways to control NHRP are

- Specify which IP packets trigger an NHRP request.
- Specify how many data packets have been sent to a particular destination before NHRP is attempted.
- Control the NHRP packet rate.

These methods are described in this section.

Triggering NHRP by IP Packets

You can specify an IP access list that is used to decide which IP packets can trigger the sending of NHRP requests. By default, all non-NHRP packets trigger NHRP requests. To limit which IP packets trigger NHRP requests, define an access list and then apply it to the interface.

To define an access list, perform one of the following tasks in global configuration mode:

Task	Command
Define a standard IP access list.	access-list <i>access-list-number</i> { deny permit } <i>source</i> [<i>source-wildcard</i>]
Define an extended IP access list.	access-list <i>access-list-number</i> { deny permit } <i>protocol source source-wildcard destination destination-wildcard</i> [precedence <i>precedence</i>] [tos <i>tos</i>] [established] [log]

Then apply the IP access list to the interface by performing the following task in interface configuration mode:

Task	Command
Specify an IP access list that controls NHRP requests.	ip nhrp interest <i>access-list-number</i>

Triggering NHRP on a Per-Destination Basis

By default, when the software attempts to transmit a data packet to a destination for which it has determined that NHRP can be used, it transmits an NHRP request for that destination. You can configure the system to wait until a specified number of data packets have been sent to a particular destination before NHRP is attempted. To do so, perform the following task in interface configuration mode:

Task	Command
Specify how many data packets are sent to a destination before NHRP is attempted.	ip nhrp use <i>usage-count</i>

Control NHRP Packet Rate

By default, the maximum rate at which the software sends NHRP packets is 5 packets per 10 seconds. The software maintains a per interface quota of NHRP packets (whether generated locally or forwarded) that can be transmitted. To change this maximum rate, perform the following task in interface configuration mode:

Task	Command
Change the NHRP packet rate per interface.	ip nhrp max-send <i>pkt-count every interval</i>

Suppress Forward and Reverse Record Options

To dynamically detect link-layer filtering in NBMA networks (for example, SMDS address screens), and to provide loop detection and diagnostic capabilities, NHRP incorporates a Route Record in requests and replies. The Route Record options contain the network (and link layer) addresses of all intermediate Next Hop Servers between source and destination (in the forward direction) and between destination and source (in the reverse direction).

By default, forward record options and reverse record options are included in NHRP request and reply packets. To suppress the use of these options, perform the following task in interface configuration mode:

Task	Command
Suppress forward and reverse record options.	no ip nhrp record

Specify the NHRP Responder Address

If an NHRP requestor wants to know which Next Hop Server generates an NHRP reply packet, it can request that information by including the responder address option in its NHRP request packet. The Next Hop Server that generates the NHRP reply packet then complies by inserting its own IP address in the NHRP reply. The Next Hop Server uses the primary IP address of the specified interface.

To specify which interface the Next Hop Server uses for the NHRP responder IP address, perform the following task in interface configuration mode:

Task	Command
Specify which interface the Next Hop Server uses to determine the NHRP responder address.	ip nhrp responder <i>type number</i>

If an NHRP reply packet being forwarded by a Next Hop Server contains that Next Hop Server's own IP address, the Next Hop Server generates an Error Indication of type "NHRP Loop Detected" and discards the reply.

Change the Time Period NBMA Addresses Are Advertised as Valid

You can change the length of time that NBMA addresses are advertised as valid in positive and negative NHRP responses. In this context, *advertised* means how long the Cisco IOS software tells other routers to keep the addresses it is providing in NHRP responses. The default length of time for each response is 7,200 seconds (2 hours). To change the length of time, perform the following task in interface configuration mode:

Task	Command
Specify the number of seconds that NBMA addresses are advertised as valid in positive or negative NHRP responses.	ip nhrp holdtime <i>seconds-positive</i> [<i>seconds-negative</i>]

Configure a GRE Tunnel for Multipoint Operation

You can enable a generic routing encapsulation (GRE) tunnel to operate in multipoint fashion. A tunnel network of multipoint tunnel interfaces can be thought of as an NBMA network. To configure the tunnel, perform the following tasks in interface configuration mode:

Task	Command
Enable a GRE tunnel to be used in multipoint fashion.	tunnel mode gre multipoint
Configure a tunnel identification key.	tunnel key <i>key-number</i>

The tunnel key should correspond to the NHRP network identifier specified in the **ip nhrp network-id** command. See the "NHRP on a Multipoint Tunnel Example" section at the end of this chapter for an example of NHRP configured on a multipoint tunnel.

Disable IP Routing

IP routing is automatically enabled in the Cisco IOS software. If you choose to set up the router to bridge rather than route IP datagrams, you must disable IP routing. To disable IP routing, perform the following task in global configuration mode:

Task	Command
Disable IP routing.	no ip routing

When IP routing is disabled, the router will act as an IP end host for IP packets destined for or sourced by it, whether or not bridging is enabled for those IP packets not destined for the device. To reenabling IP routing, use the **ip routing** command.

Routing Assistance When IP Routing Is Disabled

The Cisco IOS software provides the following three methods by which the router can learn about routes to other networks when IP routing is disabled and the device is acting as an IP host:

- Proxy ARP
- Default Gateway (also known as *default router*)
- Router Discovery Mechanism

When IP routing is disabled, the default gateway feature and the router discovery client are enabled, and proxy ARP is disabled. When IP routing is enabled, the default gateway feature is disabled and you can configure proxy ARP and the router discovery servers.

Proxy ARP

The most common method of learning about other routes is by using proxy ARP. Proxy ARP, defined in RFC 1027, enables an Ethernet host with no knowledge of routing to communicate with hosts on other networks or subnets. Such a host assumes that all hosts are on the same local Ethernet, and that it can use ARP to determine their hardware addresses.

Under proxy ARP, if a device receives an ARP Request for a host that is not on the same network as the ARP Request sender, the Cisco IOS software evaluates whether it has the best route to that host. If it does, the device sends an ARP Reply packet giving its own Ethernet hardware address. The host that sent the ARP Request then sends its packets to the device, which forwards them to the intended host. The software treats all networks as if they are local and performs ARP requests for every IP address. This feature is enabled by default.

Proxy ARP works as long as other routers support it. Many other routers, especially those loaded with host-based routing software, do not support it.

Default Gateway

Another method for locating routes is to define a default router (or gateway). The Cisco IOS software sends all nonlocal packets to this router, which either routes them appropriately or sends an IP Control Message Protocol (ICMP) redirect message back, telling it of a better route. The ICMP redirect message indicates which local router the host should use. The software caches the redirect messages and routes each packet thereafter as efficiently as possible. The limitations of this method are that there is no means of detecting when the default router has gone down or is unavailable, and there is no method of picking another device if one of these events should occur.

To set up a default gateway for a host, perform the following task in global configuration mode:

Task	Command
Set up a default gateway (router).	<code>ip default-gateway ip-address</code>

To display the address of the default gateway, use the **show ip redirects EXEC** command.

Router Discovery Mechanism

The Cisco IOS software provides a third method, called *router discovery*, by which the router dynamically learns about routes to other networks using the Gateway Discovery Protocol (GDP) or the ICMP Router Discovery Protocol (IRDP). (GDP will not be supported in future Cisco IOS releases.) The software is also capable of wire-tapping Routing Information Protocol (RIP) and Interior Gateway Routing Protocol (IGRP) routing updates and inferring the location of routers from

those updates. The server/client implementation of router discovery does not actually examine or store the full routing tables sent by routing devices, it merely keeps track of which systems are sending such data.

You can configure the four protocols in any combination. When possible, we recommend that you use IRDP because it allows each router to specify *both* a priority and the time after which a device should be assumed down if no further packets are received. Devices discovered using IGRP are assigned an arbitrary priority of 60. Devices discovered through RIP are assigned a priority of 50. For IGRP and RIP, the software attempts to measure the time between updates, and assumes that the device is down if no updates are received for 2.5 times that interval.

Each device discovered becomes a candidate for the default router. The list of candidates is scanned and a new highest-priority router is selected when any of the following events occur:

- When a higher-priority router is discovered (the list of routers is polled at 5-minute intervals).
- When the current default router is declared down.
- When a TCP connection is about to time out because of excessive retransmissions. In this case, the server flushes the ARP cache and the ICMP redirect cache, and picks a new default router in an attempt to find a successful route to the destination.

To configure the router discovery feature using the GDP routing protocol, perform the following task in interface configuration mode:

Task	Command
Use the GDP protocol to configure router discovery.	ip gdp gdp

To configure the router discovery feature using the IRDP routing protocol, perform the following task in interface configuration mode:

Task	Command
Use the IRDP protocol to configure router discovery.	ip gdp irdp

To configure the router discovery feature using the RIP routing protocol, perform the following task in interface configuration mode:

Task	Command
Use the RIP protocol to configure router discovery.	ip gdp rip

To configure the router discovery feature using the IGRP routing protocol, perform the following task in interface configuration mode:

Task	Command
Use the IGRP protocol to configure router discovery.	ip gdp igrp

Enable IP Bridging

To transparently bridge IP on an interface, perform the following tasks beginning in global configuration mode:

Task	Command
Disable IP routing.	no ip routing
Specify an interface.	interface <i>type number</i>
Add the interface to a bridge group.	bridge-group <i>group</i> ¹

1. This command is documented in the “Transparent Bridging Commands” chapter of the *Bridging and IBM Networking Command Reference*.

You can route IP on some interfaces and transparently bridge it on other interfaces simultaneously. To enable concurrent routing and bridging, perform the following task in global configuration mode:

Task	Command
Enable concurrent routing and bridging.	bridge crb ¹

1. This command is documented in the “Transparent Bridging Commands” chapter of the *Bridging and IBM Networking Command Reference*.

Enable Integrated Routing and Bridging

With integrated routing and bridging (IRB), you can route IP traffic between routed interfaces and bridge groups, or route IP traffic between bridge groups. Specifically, local or unroutable traffic is bridged among the bridged interfaces in the same bridge group, while routable traffic is routed to other routed interfaces or bridge groups. Using IRB, you can

- Switch packets from a bridged interface to a routed interface
- Switch packets from a routed interface to a bridged interface
- Switch packets within the same bridge group

For more information about configuring integrated routing and bridging, refer to the “Configuring Transparent Bridging” chapter in the *Bridging and IBM Networking Configuration Guide*.

Configure a Routing Process

At this point in the configuration process, you can choose to configure one or more of the many routing protocols that are available based on your individual network needs. Routing protocols provide topology information of an internetwork. Refer to the “Configuring IP Routing Protocols” chapter for the tasks involved in configuring IP routing protocols. If you want to continue to perform basic IP configuration tasks, continue reading the following sections.

Configure Broadcast Packet Handling

A *broadcast* is a data packet destined for all hosts on a particular physical network. Network hosts recognize broadcasts by special addresses. Broadcasts are heavily used by some protocols, including several important Internet protocols. Control of broadcast messages is an essential part of the IP network administrator’s job.

The Cisco IOS software supports two kinds of broadcasting: *directed broadcasting* and *flooding*. A directed broadcast is a packet sent to a specific network or series of networks, while a flooded broadcast packet is sent to every network. A directed broadcast address includes the network or subnet fields.

Several early IP implementations do not use the current broadcast address standard. Instead, they use the old standard, which calls for all zeros instead of all ones to indicate broadcast addresses. Many of these implementations do not recognize an all-ones broadcast address and fail to respond to the broadcast correctly. Others forward all-ones broadcasts, which causes a serious network overload known as a *broadcast storm*. Implementations that exhibit these problems include systems based on versions of BSD UNIX prior to Version 4.3.

Routers provide some protection from broadcast storms by limiting their extent to the local cable. Bridges (including intelligent bridges), because they are Layer 2 devices, forward broadcasts to all network segments, thus propagating all broadcast storms.

The best solution to the broadcast storm problem is to use a single broadcast address scheme on a network. Most modern IP implementations allow the network manager to set the address to be used as the broadcast address. Many implementations, including the one in the Cisco IOS software, accept and interpret all possible forms of broadcast addresses.

For detailed discussions of broadcast issues in general, see RFC 919, “Broadcasting Internet Datagrams,” and RFC 922, “Broadcasting IP Datagrams in the Presence of Subnets.” The support for Internet broadcasts generally complies with RFC 919 and RFC 922; it does not support multisubnet broadcasts as defined in RFC 922.

The current broadcast address standard provides specific addressing schemes for forwarding broadcasts. Perform the tasks in the following sections to enable these schemes:

- Enable Directed Broadcast-to-Physical Broadcast Translation
- Forward UDP Broadcast Packets and Protocols
- Establish an IP Broadcast Address
- Flood IP Broadcasts

See the “Broadcasting Examples” section at the end of this chapter for broadcasting configuration examples.

Enable Directed Broadcast-to-Physical Broadcast Translation

To enable forwarding of directed broadcasts on an interface where the broadcast becomes a physical broadcast, perform one of the tasks that follow. By default, this feature is enabled only for those protocols configured using the **ip forward-protocol** global configuration command. You can specify an access list to control which broadcasts are forwarded. When an access list is specified, only those IP packets permitted by the access list are eligible to be translated from directed broadcasts to physical broadcasts.

Perform either of the following tasks in interface configuration mode as required for your network:

Task	Command
Enable directed broadcast-to-physical broadcast translation on an interface.	ip directed-broadcast [<i>access-list-number</i>]
Disable directed broadcast-to-physical broadcast translation on an interface.	no ip directed-broadcast [<i>access-list-number</i>]

Forward UDP Broadcast Packets and Protocols

Network hosts occasionally use UDP broadcasts to determine address, configuration, and name information. If such a host is on a network segment that does not include a server, UDP broadcasts are normally not forwarded. You can remedy this situation by configuring the interface of your router to forward certain classes of broadcasts to a helper address. You can use more than one helper address per interface.

You can specify a UDP destination port to control which UDP services are forwarded. You can specify multiple UDP protocols. You can also specify the Network Disk (ND) protocol, which is used by older diskless Sun workstations, and you can specify the network security protocol SDNS. By default, both UDP and ND forwarding are enabled if a helper address has been defined for an interface. The description for the **ip forward-protocol** command in the *Network Protocols Command Reference, Part 1* lists the ports that are forwarded by default if you do not specify any UDP ports.

If you do not specify any UDP ports when you configure the forwarding of UDP broadcasts, you are configuring the router to act as a BOOTP forwarding agent. BOOTP packets carry Dynamic Host Configuration Protocol (DHCP) information. (DHCP is defined in RFC 1531.) This means that the Cisco IOS software is now compatible with DHCP clients.

To enable forwarding and to specify the destination address, perform the following task in interface configuration mode:

Task	Command
Enable forwarding and specify the destination address for forwarding UDP broadcast packets, including BOOTP.	ip helper-address <i>address</i>

To specify which protocols will be forwarded, perform the following task in global configuration mode:

Task	Command
Specify which protocols will be forwarded over which ports.	ip forward-protocol { udp [<i>port</i>] nd sdns }

See the “Helper Addresses Example” section at the end of this chapter for an example of how to configure helper addresses.

Establish an IP Broadcast Address

The Cisco IOS software supports IP broadcasts on both LANs and WANs. There are several ways to indicate an IP broadcast address. Currently, the most popular way, and the default, is an address consisting of all ones (255.255.255.255), although the software can be configured to generate any form of IP broadcast address. Our software also receives and understands any form of IP broadcast.

To set the IP broadcast address, perform the following task in interface configuration mode:

Task	Command
Establish a different broadcast address (other than 255.255.255.255).	ip broadcast-address [<i>ip-address</i>]

If the router does not have nonvolatile memory, and you need to specify the broadcast address to use before the software is configured, you must change the IP broadcast address by setting jumpers in the processor configuration register. Setting bit 10 causes the device to use all zeros. Bit 10 interacts with bit 14, which controls the network and subnet portions of the broadcast address. Setting bit 14 causes the device to include the network and subnet portions of its address in the broadcast address. Table 2 shows the combined effect of setting bits 10 and 14.

Table 2 Configuration Register Settings for Broadcast Address Destination

Bit 14	Bit 10	Address (<net><host>)
Out	Out	<ones><ones>
Out	In	<zeros><zeros>
In	In	<net><zeros>
In	Out	<net><ones>

Some router platforms allow the configuration register to be set through the software; see the “Loading Images and Configuration Files” chapter of the *Configuration Fundamentals Configuration Guide* for details. For other router and access server platforms, the configuration register must be changed through hardware; see the appropriate hardware installation and maintenance manual for your system.

Flood IP Broadcasts

You can allow IP broadcasts to be flooded throughout your internetwork in a controlled fashion using the database created by the bridging spanning-tree protocol. Turning on this feature also prevents loops. In order to support this capability, the routing software must include the transparent bridging, and bridging must be configured on each interface that is to participate in the flooding. If bridging is not configured on an interface, it still will be able to receive broadcasts. However, the interface will never forward broadcasts it receives, and the router will never use that interface to send broadcasts received on a different interface.

Packets that are forwarded to a single network address using the IP helper address mechanism can be flooded. Only one copy of the packet is sent on each network segment.

In order to be considered for flooding, packets must meet the following criteria. (Note that these are the same conditions used to consider packets forwarding via IP helper addresses.)

- The packet must be a MAC-level broadcast.
- The packet must be an IP-level broadcast.
- The packet must be a TFTP, DNS, Time, NetBIOS, ND, or BOOTP packet, or a UDP protocol specified by the **ip forward-protocol udp** global configuration command.
- The packet’s time-to-live (TTL) value must be at least two.

A flooded UDP datagram is given the destination address you specified with the **ip broadcast-address** command on the output interface. The destination address can be set to any desired address. Thus, the destination address may change as the datagram propagates through the network. The source address is never changed. The TTL value is decremented.

After a decision has been made to send the datagram out on an interface (and the destination address possibly changed), the datagram is handed to the normal IP output routines and is, therefore, subject to access lists, if they are present on the output interface.

To use the bridging spanning-tree database to flood UDP datagrams, perform the following task in global configuration mode:

Task	Command
Use the bridging spanning-tree database to flood UDP datagrams.	ip forward-protocol spanning-tree

If no actual bridging is desired, you can configure a type-code bridging filter that will deny all packet types from being bridged. Refer to the “Configuring Transparent Bridging” chapter of the *Bridging and IBM Networking Configuration Guide* for more information about using access lists to filter bridged traffic. The spanning-tree database is still available to the IP forwarding code to use for the flooding.

Speed Up Flooding of UDP Datagrams

You can speed up flooding of UDP datagrams using the spanning-tree algorithm. Used in conjunction with the **ip forward-protocol spanning-tree** command, this feature boosts the performance of spanning tree-based UDP flooding by a factor of about four to five times. The feature, called *turbo flooding*, is supported over Ethernet interfaces configured for ARPA encapsulated, Fiber Distributed Data Interface (FDDI), and HDLC-encapsulated serial interfaces. However, it is not supported on Token Ring interfaces. As long as the Token Rings and the non-HDLC serial interfaces are not part of the bridge group being used for UDP flooding, turbo flooding will behave normally.

To enable turbo flooding, perform the following task in global configuration mode:

Task	Command
Use the bridging spanning-tree database to speed up flooding of UDP datagrams.	ip forward-protocol turbo-flood

Configure IP Services

The IP suite offers a number of services that control and manage IP connections. ICMP provides many of these services. ICMP messages are sent by routers or access servers to hosts or other routers when a problem is discovered with the Internet header. For detailed information on ICMP, see RFC 792.

To configure IP services, complete the tasks in the following sections:

- Disable ICMP Protocol Unreachable Messages
- Disable ICMP Redirect Messages
- Understand Path MTU Discovery
- Set the MTU Packet Size
- Enable ICMP Mask Reply Messages
- Disable IP Source Routing
- Configure Simplex Ethernet Interfaces

See the “ICMP Services Example” section at the end of this chapter for examples of ICMP services.

Disable ICMP Protocol Unreachable Messages

If the Cisco IOS software receives a nonbroadcast packet destined for itself that uses an unknown protocol, it sends an ICMP Protocol Unreachable message back to the source. Similarly, if the software receives a packet that it is unable to deliver to the ultimate destination because it knows of no route to the destination address, it sends an ICMP Host Unreachable message to the source. This feature is enabled by default.

You can disable this service by performing the following task in interface configuration mode:

Task	Command
Disable the sending of ICMP Protocol Unreachable and Host Unreachable messages.	no ip unreachable

Disable ICMP Redirect Messages

Routes are sometimes less than optimal. For example, it is possible for the router to be forced to resend a packet through the same interface on which it was received. If this happens, the Cisco IOS software sends an ICMP Redirect message to the packet's originator telling it that it is on a subnet directly connected to the receiving device, and that it must forward the packet to another system on the same subnet. The software does this because the originating host presumably could have sent that packet to the next hop without involving this device at all. The Redirect message instructs the sender to remove the receiving device from the route and substitute a specified device representing a more direct path. This feature is enabled by default. However, when Hot Standby Router Protocol is configured on an interface, ICMP Redirect messages are disabled by default for the interface.

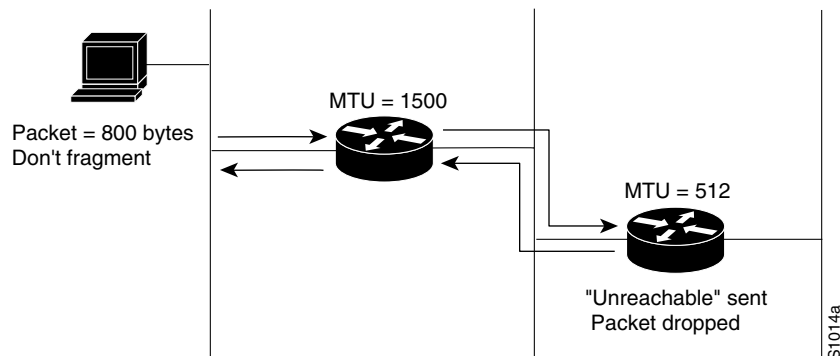
You can disable the sending of ICMP Redirect messages by performing the following task in interface configuration mode:

Task	Command
Disable the sending of ICMP Redirect messages to learn routes.	no ip redirects

Understand Path MTU Discovery

The Cisco IOS software supports the IP Path MTU Discovery mechanism, as defined in RFC 1191. IP Path MTU Discovery allows a host to dynamically discover and cope with differences in the maximum allowable maximum transmission unit (MTU) size of the various links along the path. Sometimes a router is unable to forward a datagram because it requires fragmentation (the packet is larger than the MTU you set for the interface with the **ip mtu** command), but the "don't fragment" (DF) bit is set. The Cisco IOS software sends a message to the sending host, alerting it to the problem. The host will have to fragment packets for the destination so that they fit the smallest packet size of all the links along the path. This technique is shown in Figure 5.

Figure 5 IP Path MTU Discovery



IP Path MTU Discovery is useful when a link in a network goes down, forcing the use of another, different MTU-sized link (and different routers). As shown in Figure 5, suppose a router is sending IP packets over a network where the MTU in the first router is set to 1,500 bytes, but the second router is set to 512 bytes. If the datagram's "Don't fragment" bit is set, the datagram would be dropped because the 512-byte router is unable to forward it. All packets larger than 512 bytes are dropped in this case. The second router returns an ICMP Destination Unreachable message to the source of the datagram with its Code field indicating, "Fragmentation needed and DF set." To support IP Path MTU Discovery, it would also include the MTU of the next-hop network link in the low-order bits of an unused header field.

IP Path MTU Discovery is also useful when a connection is first being established and the sender has no information at all about the intervening links. It is always advisable to use the largest MTU that the links will bear; the larger the MTU, the fewer packets the host must send.

Note IP Path MTU Discovery is a process initiated by end hosts. If an end host does not support IP Path MTU Discovery, the receiving device will have no mechanism available to avoid fragmenting datagrams generated by the end host.

The Cisco 7000 and Cisco 4000 routers support fast switching of IP packets between Ethernet and FDDI interfaces. When packets are being sent from FDDI to Ethernet interfaces and you are not using IP Path MTU Discovery, FDDI packets with data lengths larger than 1500 bytes will be fragmented into multiple Ethernet packets. This will slow performance. If the majority of your traffic travels off the FDDI ring, you might want to either lower the MTU size on your host FDDI interfaces to 1,500 bytes or run IP Path MTU Discovery on your hosts.

Because the CTR card does not support the switching of frames larger than 4,472 bytes, some interoperability problems may occur if CTR cards are intermixed with other Token Ring cards on the same network. You can minimize this by setting lower (and the same) IP maximum packet sizes for all devices on the network with the **ip mtu** interface command.

To enable Path MTU Discovery for connections initiated by the router (when the router is acting as a host), see the section "Enable TCP Path MTU Discovery" later in this chapter.

Set the MTU Packet Size

All interfaces have a default MTU packet size. You can adjust the IP MTU size so that if an IP packet exceeds the MTU set for an interface, the Cisco IOS software will fragment it.

Changing the MTU value (with the **mtu** interface configuration command) can affect the IP MTU value. If the current IP MTU value is the same as the MTU value, and you change the MTU value, the IP MTU value will be modified automatically to match the new MTU. However, the reverse is not true; changing the IP MTU value has no effect on the value for the **mtu** interface configuration command.

Also, all devices on a physical medium must have the same protocol MTU in order to operate.

To set the MTU packet size for a specified interface, perform the following task in interface configuration mode:

Task	Command
Set the IP MTU packet size for an interface.	ip mtu bytes

Enable ICMP Mask Reply Messages

Occasionally, network devices must know the subnet mask for a particular subnetwork in the internetwork. To achieve this information, such devices can send ICMP Mask Request messages. These messages are responded to by ICMP Mask Reply messages from devices that have the requested information. The Cisco IOS software can respond to ICMP Mask Request messages if this function is enabled.

To enable the sending of ICMP Mask Reply messages, perform the following task in interface configuration mode:

Task	Command
Enable the sending of ICMP Mask Reply messages.	ip mask-reply

Disable IP Source Routing

The Cisco IOS software examines IP header options on every packet. It supports the IP header options *Strict Source Route*, *Loose Source Route*, *Record Route*, and *Time Stamp*, which are defined in RFC 791. If the software finds a packet with one of these options enabled, it performs the appropriate action. If it finds a packet with an invalid option, it sends an ICMP Parameter Problem message to the source of the packet and discards the packet.

IP provides a provision that allows the source IP host to specify a route through the IP network. This provision is known as *source routing*. Source routing is specified as an option in the IP header. If source routing is specified, the software forwards the packet according to the specified source route. This feature is employed when you want to force a packet to take a certain route through the network. The default is to perform source routing.

You can disable IP source-route header options by performing the following task in global configuration mode:

Task	Command
Cause the software to discard any IP datagram containing a source-route option.	no ip source-route

Configure Simplex Ethernet Interfaces

You can configure simplex Ethernet interfaces. This feature is useful for setting up dynamic IP routing over a simplex circuit (a circuit that receives only or transmits only). When a route is learned on a receive-only interface, the interface designated as the source of the route is converted to the interface you specify. When packets are routed out this specified interface, they are sent to the IP address of the source of the routing update. To reach this IP address on a transmit-only Ethernet link, a static ARP entry mapping this IP address to the hardware address of the other end of the link is required.

To assign a transmit interface to a receive-only interface, perform the following task in interface configuration mode:

Task	Command
Assign a transmit interface to a receive-only interface.	transmit-interface <i>type number</i>

See the “Simplex Ethernet Interfaces Example” section at the end of this chapter for an example of configuring a simplex Ethernet interface.

Filter IP Packets

Packet filtering helps control packet movement through the network. Such control can help limit network traffic and restrict network use by certain users or devices. To permit or deny packets from crossing specified interfaces, we provide *access lists*.

You can use access lists in the following ways:

- To control the transmission of packets on an interface
- To control virtual terminal line access
- To restrict contents of routing updates

This section summarizes how to create IP access lists and how to apply them.

See the “IP Configuration Examples” section at the end of this chapter for examples of configuring IP access lists.

An access list is a sequential collection of permit and deny conditions that apply to IP addresses. The Cisco IOS software tests addresses against the conditions in an access list one by one. The first match determines whether the software accepts or rejects the address. Because the software stops testing conditions after the first match, the order of the conditions is critical. If no conditions match, the software rejects the address.

The two steps involved in using access lists are as follows:

- Step 1** Create an access list by specifying an access list number or name and access conditions.
- Step 2** Apply the access list to interfaces or terminal lines.

These steps are described in the next sections.

Create Standard and Extended Access Lists Using Numbers



Caution Release 11.1 and later releases introduced substantial changes to IP access lists. These extensions are backward compatible; migrating from a release earlier than Release 11.1 to the current image will convert your access lists automatically. However, previous releases are not upwardly compatible with these changes. Thus, if you save an access list with the current image and then use older software, the resulting access list will not be interpreted correctly. **This could cause you severe security problems.** Save your old configuration file before booting Release 11.1-or-later images.

The software supports the following styles of access lists for IP:

- Standard IP access lists use source addresses for matching operations.
- Extended IP access lists use source and destination addresses for matching operations, and optional protocol type information for finer granularity of control.
- Dynamic extended IP access lists grant access per user to a specific source or destination host basis through a user authentication process. In essence, you can allow user access through a firewall dynamically, without compromising security restrictions. Dynamic access lists and lock-and-key access are described in the “Managing the System” chapter of the *Configuration Fundamentals Configuration Guide*.

To create a standard access list, perform one of the following tasks in global configuration mode:

Task	Command
Define a standard IP access list using a source address and wildcard.	access-list <i>access-list-number</i> {deny permit} <i>source</i> [<i>source-wildcard</i>]
Define a standard IP access list using an abbreviation for the source and source mask of 0.0.0.0 255.255.255.255.	access-list <i>access-list-number</i> {deny permit} any

To create an extended access list, perform one of the following tasks in global configuration mode:

Task	Command
Define an extended IP access list number and the access conditions. Use the log keyword to get access list logging messages, including violations.	access-list <i>access-list-number</i> {deny permit} <i>protocol</i> <i>source</i> <i>source-wildcard</i> <i>destination</i> <i>destination-wildcard</i> [precedence <i>precedence</i>] [tos <i>tos</i>] [established] [log]
Define an extended IP access list using an abbreviation for a source and source wildcard of 0.0.0.0 255.255.255.255, and an abbreviation for a destination and destination wildcard of 0.0.0.0 255.255.255.255.	access-list <i>access-list-number</i> {deny permit} <i>protocol</i> any any
Define an extended IP access list using an abbreviation for a source and source wildcard of <i>source</i> 0.0.0.0, and an abbreviation for a destination and destination wildcard of <i>destination</i> 0.0.0.0.	access-list <i>access-list-number</i> {deny permit} <i>protocol</i> host <i>source</i> host <i>destination</i>

Task	Command
Define a dynamic access list. For information about lock-and-key access, refer to the “Managing the System” chapter in the <i>Configuration Fundamentals Configuration Guide</i> .	access-list <i>access-list-number</i> [dynamic <i>dynamic-name</i> [timeout <i>minutes</i>]] { deny permit } <i>protocol source source-wildcard destination destination-wildcard</i> [precedence <i>precedence</i>] [tos <i>tos</i>] [established] [log]

After an access list is created initially, any subsequent additions (possibly entered from the terminal) are placed at the end of the list. In other words, you cannot selectively add or remove access list command lines from a specific access list.

Note When creating an access list, remember that, by default, the end of the access list contains an implicit deny statement for everything if it did not find a match before reaching the end. Further, with standard access lists, if you omit the mask from an associated IP host address access list specification, 0.0.0.0 is assumed to be the mask.

Note Autonomous switching is not used when you have extended access lists.

After creating an access list, you must apply it to a line or interface, as shown in the section “Apply an Access List to an Interface or Terminal Line” later in this chapter.

See the “Implicit Masks in Access Lists Examples” section at the end of this chapter for examples of implicit masks.

Create Standard and Extended Access Lists Using Names



Caution Named access lists will not be recognized by any software release prior to Cisco IOS Release 11.2.

You can identify IP access lists with an alphanumeric string (a name) rather than a number (1 to 199). This feature allows you to configure more than 99 standard IP and 100 extended IP access lists in a router. If you identify your access list with a name rather than a number, the mode and command syntax are slightly different. Currently, only packet and route filters can use a named list.

Implementation Considerations

Consider the following before configuring named access lists:

- Access lists specified by name are not compatible with older releases.
- Not all access lists that accept a number will accept a name. Access lists for packet filters and route filters on interfaces can use a name.
- A standard access list and an extended access list cannot have the same name.
- Numbered access lists are also available, as described in the preceding section, “Create Standard and Extended Access Lists Using Numbers.”

To create a standard access list, perform the following tasks beginning in global configuration mode:

Task	Command
Step 1 Define a standard IP access list using a name.	ip access-list standard <i>name</i>
Step 2 In access-list configuration mode, specify one or more conditions allowed or denied. This determines whether the packet is passed or dropped.	deny { <i>source</i> [<i>source-wildcard</i>] any } or permit { <i>source</i> [<i>source-wildcard</i>] any }
Step 3 Exit access-list configuration mode.	exit

To create an extended access list, perform the following tasks beginning in global configuration mode:

Task	Command
Step 1 Define an extended IP access list using a name.	ip access-list extended <i>name</i>
Step 2 In access-list configuration mode, specify the conditions allowed or denied. Use the log keyword to get access list logging messages, including violations.	{ deny permit } <i>protocol source source-wildcard destination destination-wildcard</i> [precedence <i>precedence</i>] [tos <i>tos</i>] [established] [log]
or	
Define an extended IP access list using an abbreviation for a source and source wildcard of 0.0.0.0 255.255.255.255, and an abbreviation for a destination and destination wildcard of 0.0.0.0 255.255.255.255.	{ deny permit } <i>protocol any any</i>
or	
Define an extended IP access list using an abbreviation for a source and source wildcard of <i>source</i> 0.0.0.0, and an abbreviation for a destination and destination wildcard of <i>destination</i> 0.0.0.0.	{ deny permit } <i>protocol host source host destination</i>
or	
Define a dynamic access list. For information about lock-and-key access, refer to the “Configuring Traffic Filters” chapter in the <i>Security Configuration Guide</i> .	dynamic <i>dynamic-name</i> [timeout <i>minutes</i>] { deny permit } <i>protocol source source-wildcard destination destination-wildcard</i> [precedence <i>precedence</i>] [tos <i>tos</i>] [established] [log]

Note Autonomous switching is not used when you have extended access lists.

After you initially create an access list, you place any subsequent additions (possibly entered from the terminal) at the end of the list. In other words, you cannot selectively add access list command lines to a specific access list. However, you can use **no permit** and **no deny** commands to remove entries from a named access list.

Note When making the standard and extended access list, remember that, by default, the end of the access list contains an implicit deny statement for everything if it did not find a match before reaching the end. Further, with standard access lists, if you omit the mask from an associated IP host address access list specification, 0.0.0.0 is assumed to be the mask.

After creating an access list, you must apply it to a line or interface, as shown in the following section, “Apply an Access List to an Interface or Terminal Line.”

See the “Named Access List Example” section at the end of this chapter for an example of a named access list.

Apply an Access List to an Interface or Terminal Line

After you create an access list, you can apply it to one or more interfaces. Access lists can be applied on *either* outbound or inbound interfaces. The following two tables show how to accomplish this task for both terminal lines and network interfaces. Keep in mind that

- When controlling access to a line, you must use a number.
- When controlling access to an interface, you can use a name or number.

Perform the following task in line configuration mode. Only numbered access lists can be applied to lines. Set identical restrictions on all the virtual terminal lines, because a user can attempt to connect to any of them.

Task	Command
Restrict incoming and outgoing connections between a particular virtual terminal line (into a device) and the addresses in an access list.	access-class <i>access-list-number</i> { in out }

Perform the following task in interface configuration mode:

Task	Command
Control access to an interface.	ip access-group { <i>access-list-number</i> <i>name</i> } { in out }

For inbound access lists, after receiving a packet, the Cisco IOS software checks the source address of the packet against the access list. If the access list permits the address, the software continues to process the packet. If the access list rejects the address, the software discards the packet and returns an ICMP Host Unreachable message.

For outbound access lists, after receiving and routing a packet to a controlled interface, the software checks the source address of the packet against the access list. If the access list permits the address, the software transmits the packet. If the access list rejects the address, the software discards the packet and returns an ICMP Host Unreachable message.

When you apply an access list that has not yet been defined to an interface, the software will act as if the access list has not been applied to the interface and will accept all packets. Remember this behavior if you use undefined access lists as a means of security in your network.

Configure Network Address Translation (NAT)

Two of the key problems facing the Internet are depletion of IP address space and scaling in routing. Network Address Translation (NAT) is a feature that allows an organization's IP network to appear from the outside to use different IP address space than what it is actually using. Thus, NAT allows an organization with nonglobally routable addresses to connect to the Internet by translating those addresses into globally routable address space. NAT also allows a more graceful renumbering strategy for organizations that are changing service providers or voluntarily renumbering into CIDR blocks. NAT is also described in RFC 1631.

NAT has several applications. Use it for the following purposes:

- You want to connect to the Internet, but not all your hosts have globally unique IP addresses. NAT enables private IP internetworks that use nonregistered IP addresses to connect to the Internet. NAT is configured on the router at the border of a stub domain (referred to as the *inside network*) and a public network such as the Internet (referred to as the *outside network*). NAT translates the internal local addresses to globally unique IP addresses before sending packets to the outside network.
- You must change your internal addresses. Instead of changing them, which can be a considerable amount of work, you can translate them by using NAT.
- You want to do basic load sharing of TCP traffic. You can map a single global IP address to many local IP addresses by using the TCP load distribution feature.

As a solution to the connectivity problem, NAT is practical only when relatively few hosts in a stub domain communicate outside of the domain at the same time. When this is the case, only a small subset of the IP addresses in the domain must be translated into globally unique IP addresses when outside communication is necessary, and these addresses can be reused when no longer in use.

A significant advantage of NAT is that it can be configured without requiring changes to hosts or routers other than those few routers on which NAT will be configured. As discussed previously, NAT may not be practical if large numbers of hosts in the stub domain communicate outside of the domain. Furthermore, some applications use embedded IP addresses in such a way that it is impractical for a NAT device to translate. These applications may not work transparently or at all through a NAT device. NAT also hides the identity of hosts, which may be an advantage or a disadvantage.

A router configured with NAT will have at least one interface to the inside and one to the outside. In a typical environment, NAT is configured at the exit router between a stub domain and backbone. When a packet is leaving the domain, NAT translates the locally significant source address into a globally unique address. When a packet is entering the domain, NAT translates the globally unique destination address into a local address. If more than one exit point exists, each NAT must have the same translation table. If the software cannot allocate an address because it has run out of addresses, it drops the packet and sends an ICMP Host Unreachable packet.

A router configured with NAT must not advertise the local networks to the outside. However, routing information that NAT receives from the outside can be advertised in the stub domain as usual.

As mentioned previously, the term *inside* refers to those networks that are owned by an organization and that must be translated. Inside this domain, hosts will have address in the one address space, while on the outside, they will appear to have addresses in a another address space when NAT is configured. The first address space is referred to as the *local* address space while the second is referred to as the *global* address space.

Similarly, *outside* refers to those networks to which the stub network connects, and which are generally not under the organization's control. As will be described later, hosts in outside networks can be subject to translation also, and can, thus, have local and global addresses.

To summarize, NAT uses the following definitions:

- **Inside local address**—The IP address that is assigned to a host on the inside network. The address is probably not a legitimate IP address assigned by the Network Information Center (NIC) or service provider.
- **Inside global address**—A legitimate IP address (assigned by the NIC or service provider) that represents one or more inside local IP addresses to the outside world.
- **Outside local address**—The IP address of an outside host as it appears to the inside network. Not necessarily a legitimate address, it was allocated from address space routable on the inside.
- **Outside global address**—The IP address assigned to a host on the outside network by the host's owner. The address was allocated from globally routable address or network space.

NAT Configuration Task List

Before configuring any NAT translation, you must know your inside local addresses and inside global addresses. The following sections discuss how you can use NAT to perform optional tasks:

- Translate Inside Source Addresses
- Overload an Inside Global Address
- Translate Overlapping Addresses
- TCP Load Distribution
- Change Translation Timeouts
- Monitor and Maintain NAT

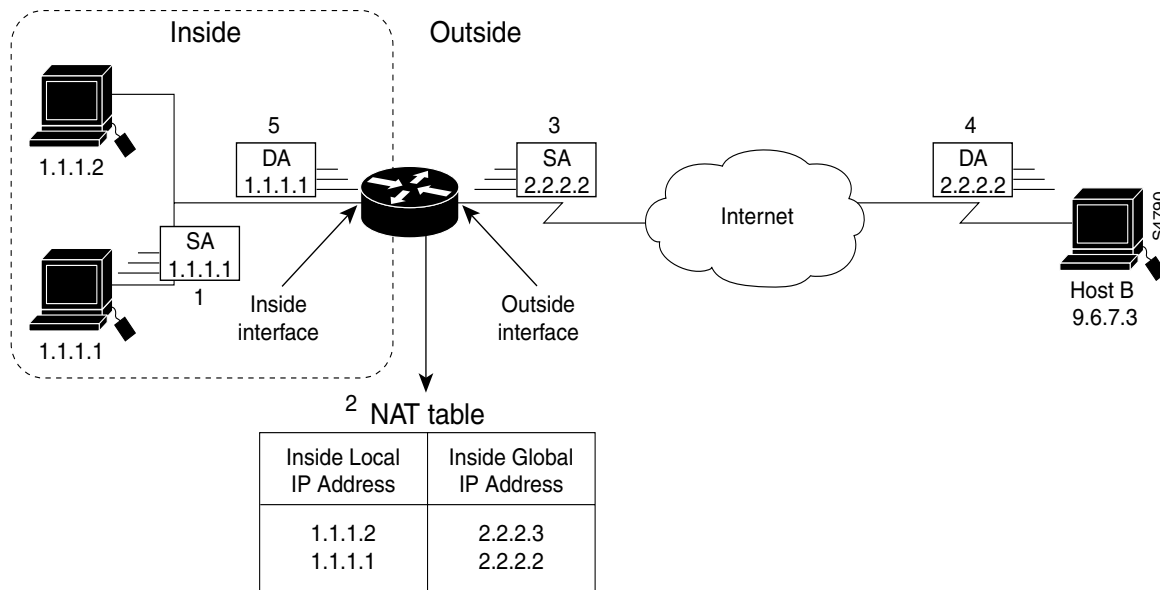
Translate Inside Source Addresses

Use this feature to translate your own IP addresses into globally unique IP addresses when communicating outside of your network. You can configure static or dynamic inside source translation as follows:

- *Static translation* establishes a one-to-one mapping between your inside local address and an inside global address. Static translation is useful when a host on the inside must be accessible by a fixed address from the outside.
- *Dynamic translation* establishes a mapping between an inside local address and a pool of global addresses.

Figure 6 illustrates a router that is translating a source address inside a network to a source address outside the network.

Figure 6 NAT Inside Source Translation



The following steps describe inside source address translation, as shown in Figure 6:

- Step 1** The user at Host 1.1.1.1 opens a connection to Host B.
- Step 2** The first packet that the router receives from Host 1.1.1.1 causes the router to check its NAT table.
 - If a static translation entry was configured, the router goes to Step 3.
 - If no translation entry exists, the router determines that source address (SA) 1.1.1.1 must be translated dynamically, selects a legal, global address from the dynamic address pool, and creates a translation entry. This type of entry is called a *simple entry*.
- Step 3** The router replaces the inside local source address of Host 1.1.1.1 with the translation entry's global address, and forwards the packet.
- Step 4** Host B receives the packet and responds to Host 1.1.1.1 by using the inside global IP destination address (DA) 2.2.2.2.
- Step 5** When the router receives the packet with the inside global IP address, it performs a NAT table lookup by using the inside global address as a key. It then translates the address to the inside local address of Host 1.1.1.1 and forwards the packet to Host 1.1.1.1.
- Step 6** Host 1.1.1.1 receives the packet and continues the conversation. The router performs Steps 2 through 5 for each packet.

Static

To configure static inside source address translation, perform the following tasks beginning in global configuration mode:

Task	Command
Establish static translation between an inside local address and an inside global address.	<code>ip nat inside source static local-ip global-ip</code>

Task	Command
Specify the inside interface.	interface <i>type number</i>
Mark the interface as connected to the inside.	ip nat inside
Specify the outside interface.	interface <i>type number</i>
Mark the interface as connected to the outside.	ip nat outside

The previous steps are the minimum you must configure. You could configure multiple inside and outside interfaces.

Dynamic

To configure dynamic inside source address translation, perform the following tasks beginning in global configuration mode:

Task	Command
Define a pool of global addresses to be allocated as needed.	ip nat pool <i>name start-ip end-ip {netmask netmask prefix-length prefix-length}</i>
Define a standard access list permitting those addresses that are to be translated.	access-list <i>access-list-number permit source [source-wildcard]</i>
Establish dynamic source translation, specifying the access list defined in the prior step.	ip nat inside source list <i>access-list-number pool name</i>
Specify the inside interface.	interface <i>type number</i>
Mark the interface as connected to the inside.	ip nat inside
Specify the outside interface.	interface <i>type number</i>
Mark the interface as connected to the outside.	ip nat outside

Note The access list must permit only those addresses that are to be translated. (Remember that there is an implicit “deny all” at the end of each access-list.) An access list that is too permissive can lead to unpredictable results.

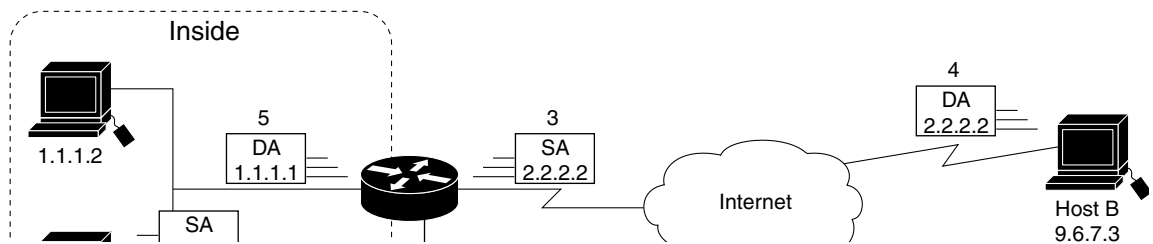
See the “Dynamic Inside Source Translation Example” section at the end of this chapter for an example of dynamic inside source translation.

Overload an Inside Global Address

You can conserve addresses in the inside global address pool by allowing the router to use one global address for many local addresses. When this overloading is configured, the router maintains enough information from higher-level protocols (for example, TCP or UDP port numbers) to translate the global address back to the correct local address. When multiple local addresses map to one global address, each the TCP or UDP port numbers of each inside host distinguish between the local addresses.

Figure 7 illustrates NAT operation when one inside global address represents multiple inside local addresses. The TCP port numbers act as differentiators.

Figure 7 NAT Overloading Inside Global Addresses



The following steps are taken in overloading inside global addresses, as shown in Figure 7. Both Host B and Host C think they are talking to a single host at address 2.2.2.2. They are actually talking to different hosts; the port number is the differentiator. In fact, many inside hosts could share the inside global IP address by using many port numbers.

- Step 1** The user at Host 1.1.1.1 opens a connection to Host B.
- Step 2** The first packet that the router receives from Host 1.1.1.1 causes the router to check its NAT table.

If no translation entry exists, the router determines that address 1.1.1.1 must be translated, and sets up a translation of inside local address 1.1.1.1 to a legal global address. If overloading is enabled, and another translation is active, the router reuses the global address from that translation and saves enough information to be able to translate back. This type of entry is called an *extended entry*.
- Step 3** The router replaces the inside local source address 1.1.1.1 with the selected global address and forwards the packet.
- Step 4** Host B receives the packet and responds to Host 1.1.1.1 by using the inside global IP address 2.2.2.2.
- Step 5** When the router receives the packet with the inside global IP address, it performs a NAT table lookup, using the protocol, inside global address and port, and outside address and port as a key, translates the address to inside local address 1.1.1.1, and forwards the packet to Host 1.1.1.1.
- Step 6** Host 1.1.1.1 receives the packet and continues the conversation. The router performs Steps 2 through 5 for each packet.

To configure overloading of inside global addresses, perform the following tasks beginning in global configuration mode:

Task	Command
Define a pool of global addresses to be allocated as needed.	<code>ip nat pool name start-ip end-ip {netmask netmask prefix-length prefix-length}</code>

Task	Command
Define a standard access list.	access-list <i>access-list-number</i> permit <i>source</i> [<i>source-wildcard</i>]
Establish dynamic source translation, identifying the access list defined in the prior step.	ip nat inside source list <i>access-list-number</i> pool <i>name</i> overload
Specify the inside interface.	interface <i>type number</i>
Mark the interface as connected to the inside.	ip nat inside
Specify the outside interface.	interface <i>type number</i>
Mark the interface as connected to the outside.	ip nat outside

Note The access list must permit only those addresses that are to be translated. (Remember that there is an implicit “deny all” at the end of each access-list.) An access list that is too permissive can lead to unpredictable results.

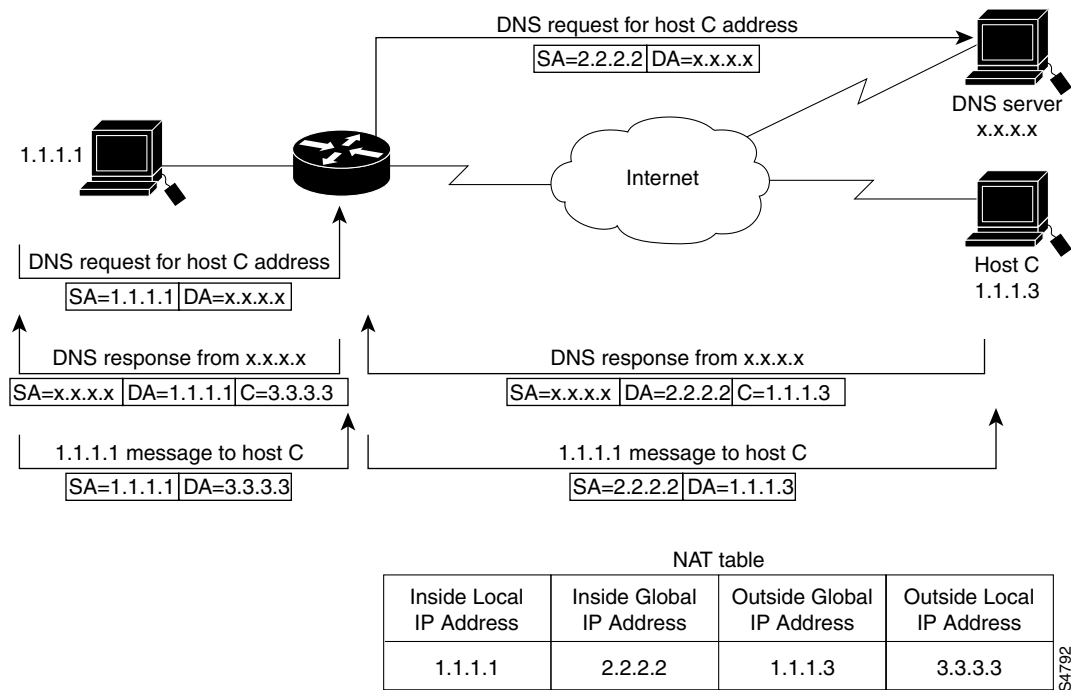
See the “Overloading Inside Global Addresses Example” section at the end of this chapter for an example of overloading inside global addresses.

Translate Overlapping Addresses

The NAT overview discusses translating IP addresses, perhaps because your IP addresses are not legal, officially assigned IP addresses. Perhaps you chose IP addresses that officially belong to another network. The case of an address used both illegally and legally is called *overlapping*. You can use NAT to translate inside addresses that overlap with outside addresses. Use this feature if your IP addresses in the stub network are legitimate IP addresses belonging to another network, and you want to communicate with those hosts or routers.

Figure 8 shows how NAT translates overlapping networks.

Figure 8 NAT Translating Overlapping Addresses



The router takes the following steps when translating overlapping addresses:

- Step 1** The user at Host 1.1.1.1 opens a connection to Host C by name, requesting a name-to-address lookup from a DNS server.
- Step 2** The router intercepts the DNS reply and translates the returned address if there is an overlap (that is, the resulting legal address resides illegally in the inside network). To translate the return address, the router creates a simple translation entry mapping the overlapping address 1.1.1.3 to an address from a separately configured, outside local address pool.

The router examines every DNS reply from everywhere, ensuring that the IP address is not in the stub network. If it is, the router translates the address.
- Step 3** Host 1.1.1.1 opens a connection to 3.3.3.3.
- Step 4** The router sets up translations mapping inside local and global addresses to each other, and outside global and local addresses to each other.
- Step 5** The router replaces the source address with the inside global address, and replaces the destination address with the outside global address.
- Step 6** Host C receives the packet and continues the conversation.
- Step 7** The router does a lookup, replaces the destination address with the inside local address, and replaces the source address with the outside local address.
- Step 8** Host 1.1.1.1 receives the packet and the conversation continues, using this translation process.

Static

To configure static outside source address translation, perform the following tasks beginning in global configuration mode:

Task	Command
Establish static translation between an outside local address and an outside global address.	ip nat outside source static <i>global-ip local-ip</i>
Specify the inside interface.	interface <i>type number</i>
Mark the interface as connected to the inside.	ip nat inside
Specify the outside interface.	interface <i>type number</i>
Mark the interface as connected to the outside.	ip nat outside

Dynamic

To configure dynamic outside source address translation, perform the following tasks beginning in global configuration mode.

Task	Command
Define a pool of local addresses to be allocated as needed.	ip nat pool <i>name start-ip end-ip {netmask netmask prefix-length prefix-length}</i>
Define a standard access list.	access-list <i>access-list-number permit source [source-wildcard]</i>
Establish dynamic outside source translation, specifying the access list defined in the prior step.	ip nat outside source list <i>access-list-number pool name</i>
Specify the inside interface.	interface <i>type number</i>
Mark the interface as connected to the inside.	ip nat inside
Specify the outside interface.	interface <i>type number</i>
Mark the interface as connected to the outside.	ip nat outside

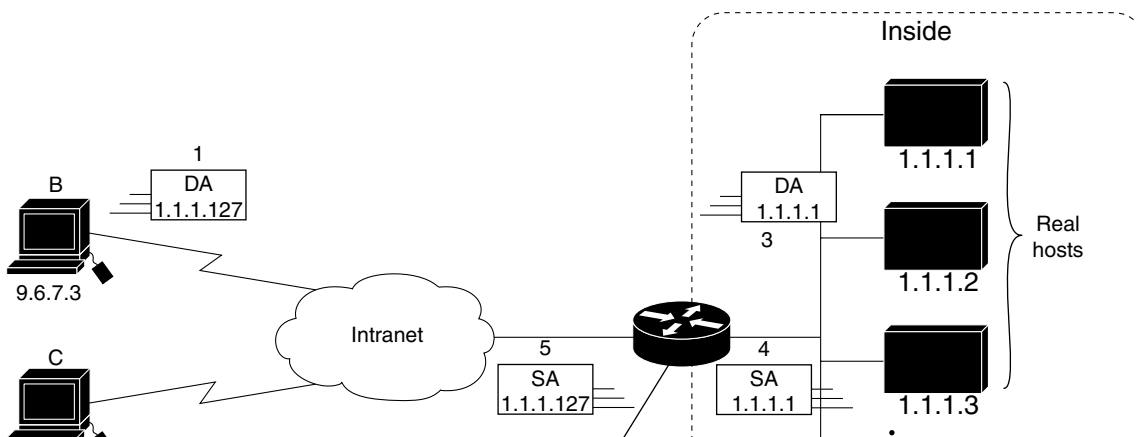
Note The access list must permit only those addresses that are to be translated. (Remember that there is an implicit “deny all” at the end of each access-list.) An access list that is too permissive can lead to unpredictable results.

See the “Translating Overlapping Address Example” section at the end of this chapter for an example of translating an overlapping address.

TCP Load Distribution

Another use of NAT is unrelated to Internet addresses. Your organization may have multiple hosts that must communicate with a heavily used host. Using NAT, you can establish a virtual host on the inside network that coordinates load sharing among real hosts. Destination addresses that match an access list are replaced with addresses from a rotary pool. Allocation is done in a round-robin basis, and only when a new connection is opened from the outside to the inside. Non-TCP traffic is passed untranslated (unless other translations are in effect). Figure 9 illustrates this feature.

Figure 9 NAT TCP Load Distribution



The router takes the following steps when translating rotary addresses:

- Step 1** The user on Host B (9.6.7.3) opens a connection to virtual host at 1.1.1.127.
- Step 2** The router receives the connection request and creates a new translation, allocating the next real host (1.1.1.1) for the inside local IP address.
- Step 3** The router replaces the destination address with the selected real host address and forwards the packet.
- Step 4** Host 1.1.1.1 receives the packet and responds.
- Step 5** The router receives the packet, performs a NAT table lookup using the inside local address and port number, and the outside address and port number as the key. The router then translates the source address to the address of the virtual host and forwards the packet.

The next connection request will cause the router to allocate 1.1.1.2 for the inside local address.

To configure destination address rotary translation, perform the following tasks beginning in global configuration mode. This allows you to map one virtual host to many real hosts. Each new TCP session opened with the virtual host will be translated into a session with a different real host.

Task	Command
Define a pool of addresses containing the addresses of the real hosts.	ip nat pool <i>name start-ip end-ip {netmask netmask prefix-length prefix-length}</i> type rotary
Define an access list permitting the address of the virtual host.	access-list <i>access-list-number</i> permit <i>source [source-wildcard]</i>

Task	Command
Establish dynamic inside destination translation, identifying the access list defined in the prior step.	ip nat inside destination list <i>access-list-number</i> pool <i>name</i>
Specify the inside interface.	interface <i>type number</i>
Mark the interface as connected to the inside.	ip nat inside
Specify the outside interface.	interface <i>type number</i>
Mark the interface as connected to the outside.	ip nat outside

Note The access list must permit only those addresses that are to be translated. (Remember that there is an implicit “deny all” at the end of each access-list.) An access list that is too permissive can lead to unpredictable results.

See the “TCP Load Distribution Example” section at the end of this chapter for an example of rotary translation.

Change Translation Timeouts

By default, dynamic address translations time out after some period of non-use. You can change the default values on timeouts, if necessary. When overloading is not configured, simple translation entries time out after 24 hours. To change this value, perform the following task in global configuration mode:

Task	Command
Change the timeout value for dynamic address translations that do not use overloading.	ip nat translation timeout <i>seconds</i>

If you have configured overloading, you have finer control over translation entry timeout because each entry contains more context about the traffic that is using it. To change timeouts on extended entries, perform one or more of the following tasks in global configuration mode:

Task	Command
Change the UDP timeout value from 5 minutes.	ip nat translation udp-timeout <i>seconds</i>
Change the DNS timeout value from 1 minute.	ip nat translation dns-timeout <i>seconds</i>
Change the TCP timeout value from 24 hours.	ip nat translation tcp-timeout <i>seconds</i>
Change the Finish and Reset timeout value from 1 minute.	ip nat translation finrst-timeout <i>seconds</i>

Monitor and Maintain NAT

By default, dynamic address translations will time out from the NAT translation table at some point. You can clear the entries before the timeout by performing one of the following tasks in EXEC mode:

Task	Command
Clear all dynamic address translation entries from the NAT translation table.	clear ip nat translation *

Task	Command
Clear a simple dynamic translation entry containing an inside translation, or both inside and outside translation.	clear ip nat translation inside <i>global-ip local-ip</i> [outside <i>local-ip global-ip</i>]
Clear a simple dynamic translation entry containing an outside translation.	clear ip nat translation outside <i>local-ip global-ip</i>
Clear an extended dynamic translation entry.	clear ip nat translation protocol inside <i>global-ip global-port local-ip local-port</i> [outside <i>local-ip local-port global-ip global-port</i>]

You can display translation information by performing one of the following tasks in EXEC mode:

Task	Command
Display active translations.	show ip nat translations [verbose]
Display translation statistics.	show ip nat statistics

Configure the Hot Standby Router Protocol

The Hot Standby Router Protocol provides high network availability because it routes IP traffic from hosts on Ethernet, FDDI, or Token Ring networks without relying on the availability of any single router.

This feature is useful for hosts that do not support a router discovery protocol (such as IRDP) and do not have the functionality to switch to a new router when their selected router reloads or loses power. Because existing TCP sessions can survive the *failover*, this protocol also provides a more transparent means of recovery for hosts that dynamically select a next hop for routing IP traffic.

When the Hot Standby Router Protocol is configured on a network segment, it provides a virtual MAC address and an IP address that is shared among routers in a group of routers that is running the Hot Standby Router Protocol. One of these devices is selected by the protocol to be the active router. The active router receives and routes packets destined for the group's MAC address. For n routers running the Hot Standby Router Protocol, there are $n + 1$ IP and MAC addresses assigned.

The Hot Standby Router Protocol detects when the designated active router fails, at which point a selected standby router assumes control of the Hot Standby group's MAC and IP addresses. A new standby router is also selected at that time.

Devices that are running the Hot Standby Router Protocol send and receive multicast UDP-based hello packets to detect router failure and to designate active and standby routers.

When the Hot Standby Router Protocol is configured on an interface, ICMP Redirect messages are disabled by default for the interface.

You can configure multiple Hot Standby groups on an interface, thereby making fuller use of the redundant routers. To do so, specify a group number for each Hot Standby command you configure for the interface.

Note Token Ring interfaces allow up to three Hot Standby groups each.

Note The Cisco 1000 series, Cisco 2500 series, Cisco 3000 series, and Cisco 4000 series that use Lance Ethernet hardware do not support multiple Hot Standby groups on a single Ethernet interface.

To enable the Hot Standby Router Protocol on an interface, perform the following task in interface configuration mode:

Task	Command
Enable the Hot Standby Router Protocol.	standby [<i>group-number</i>] ip [<i>ip-address</i> [secondary]]

To configure other Hot Standby group attributes that affect how the local router participates in the Hot Standby Router Protocol, perform one or more of the following tasks in interface configuration mode:

Task	Command
Configure the time between hello packets and the hold time before other routers declare the active router to be down.	standby [<i>group-number</i>] timers <i>hellotime holdtime</i>
Set the Hot Standby priority, used in choosing the active router.	standby [<i>group-number</i>] priority <i>priority-number</i>
Specify that, if the local router has priority over the current active router, the local router should attempt to take its place as the active router.	standby [<i>group-number</i>] preempt
Configure the interface to track other interfaces, so that if one of the other interfaces goes down, the device's Hot Standby priority is lowered.	standby [<i>group-number</i>] track <i>type number</i> [<i>interface-priority</i>]
Select an authentication string to be carried in all Hot Standby Router Protocol messages.	standby [<i>group-number</i>] authentication <i>string</i>
Configure Hot Standby Router Protocol to use the interface's burned-in address as its virtual MAC address instead of the preassigned MAC address (on Ethernet and FDDI) or the functional address (on Token Ring).	standby use-bia

Configure Basic IP Security Options

Our IP Security Option (IPSO) support addresses both the basic and extended security options as described in RFC 1108. Our implementation is only minimally compliant with RFC 1108, because the Cisco IOS software only accepts and generates a four-byte IPSO. IPSO is generally used to comply with the U.S. Government's Department of Defense security policy.

Our basic IPSO support provides the following features:

- Defines security level on a per-interface basis
- Defines single-level or multilevel interfaces
- Provides a label for incoming packets
- Strips labels on a per-interface basis
- Reorders options to put any basic security options first

To configure basic IPSO, complete the tasks in the following sections:

- Enable IPSO and Set the Security Classifications
- Specify How IP Security Options Are Processed

Enable IPSO and Set the Security Classifications

To enable IPSO and set security classifications on an interface, perform either of the following tasks in interface configuration mode:

Task	Command
Set an interface to the requested IPSO classification and authorities.	ip security dedicated <i>level authority [authority...]</i>
Set an interface to the requested IPSO range of classifications and authorities.	ip security multilevel <i>level1 [authority1...] to level2 authority2 [authority2...]</i>

Use the **no ip security** command to reset an interface to its default state.

Specify How IP Security Options Are Processed

To specify how IP security options are processed, perform any of the following optional tasks in interface configuration mode:

Task	Command
Enable an interface to ignore the authorities field of all incoming packets.	ip security ignore-authorities
Classify packets that have no IPSO with an implicit security label.	ip security implicit-labelling [<i>level authority [authority...]</i>]
Accept packets on an interface that has an extended security option present.	ip security extended-allowed
Ensure that all packets leaving the router on an interface contain a basic security option.	ip security add
Remove any basic security option that might be present on a packet leaving the router through an interface.	ip security strip
Prioritize security options on a packet.	ip security first
Treat as valid any packets that have Reserved1 through Reserved4 security levels.	ip security reserved-allowed

Default Values for Command Keywords

In order to fully comply with IPSO, the default values for the minor keywords have become complex. Default value usages include the following:

- The default for all of the minor keywords is *off*, with the exception of **implicit-labelling** and **add**.
- The default value of **implicit-labelling** is *on* if the interface is unclassified Genser; otherwise, it is *off*.
- The default value for **add** is *on* if the interface is not “unclassified Genser”; otherwise, it is *off*.

Table 3 provides a list of all default values.

Table 3 Default Security Keyword Values

Interface Type	Level	Authority	Implicit Labeling	Add IPSO
None	None	None	On	Off
Dedicated	Unclassified	Genser	On	Off
Dedicated	Any	Any	Off	On
Multilevel	Any	Any	Off	On

The default value for any interface is “dedicated, unclassified Genser.” Note that this implies implicit labeling. This might seem unusual, but it makes the system entirely transparent to packets without options. This is the setting generated when you specify the **no ip security** interface configuration command.

Configure Extended IP Security Options

Our extended IPSO support is compliant with the Department of Defense Intelligence Information System Network Security for Information Exchange (DNSIX) specification documents. Extended IPSO functionality can unconditionally accept or reject Internet traffic that contains extended security options by comparing those options to configured allowable values. This support allows DNSIX networks to use additional security information to achieve a higher level of security than that achievable with basic IPSO.

We also support a subset of the security features defined in the DNSIX Version 2.1 specification. Specifically, we support DNSIX definitions of the following:

- How extended IPSO is processed
- Audit trail facility

There are two kinds of extended IPSO fields defined by the DNSIX 2.1 specification and supported by our implementation of extended IPSO—Network Level Extended Security Option (NLESO) and Auxiliary Extended Security Option (AESO) fields.

NLESO processing requires that security options be checked against configured allowable information, source, and compartment bit values, and requires that the router be capable of inserting extended security options in the IP header.

AESO is similar to NLESO, except that its contents are not checked and are assumed to be valid if its source is listed in the AESO table.

To configure extended IPSO, complete the tasks in the following sections:

- Configure Global Default Settings
- Attach ESOs to an Interface
- Attach AESOs to an Interface

DNSIX Version 2.1 causes slow-switching code.

See the “IPSO Configuration Examples” section at the end of this chapter.

Configure Global Default Settings

To configure global default setting for extended IPSO, including AESOs, perform the following task in global configuration mode:

Task	Command
Configure system-wide default settings.	ip security eso-info <i>source compartment-size default-bit</i>

Attach ESOs to an Interface

To specify the minimum and maximum sensitivity levels for an interface, perform the following tasks in interface configuration mode:

Task	Command
Set the minimum sensitivity level for an interface.	ip security eso-min <i>source compartment-bits</i>
Set the maximum sensitivity level for an interface.	ip security eso-max <i>source compartment-bits</i>

Attach AESOs to an Interface

To specify the extended IPSO sources that are to be treated as AESO sources, perform the following task in interface configuration mode:

Task	Command
Specify AESO sources.	ip security aeso <i>source compartment-bits</i>

Configure the DNSIX Audit Trail Facility

The audit trail facility is a UDP-based protocol that generates an audit trail of IPSO security violations. This facility allows the system to report security failures on incoming and outgoing packets. The Audit Trail Facility sends DNSIX audit trail messages when a datagram is rejected because of IPSO security violations. This feature allows you to configure organization-specific security information.

The DNSIX audit trail facility consists of two protocols:

- DNSIX Message Deliver Protocol (DMDP) provides a basic message-delivery mechanism for all DNSIX elements.
- Network Audit Trail Protocol (NAT) provides a buffered logging facility for applications to use to generate auditing information. This information is then passed on to DMDP.

To configure the DNSIX auditing facility, complete the tasks in the following sections:

- Enable the DNSIX Audit Trail Facility
- Specify Hosts to Receive Audit Trail Messages
- Specify Transmission Parameters

Enable the DNSIX Audit Trail Facility

To enable the DNSIX audit trail facility, perform the following task in global configuration mode:

Task	Command
Start the audit writing module.	dnsix-nat source <i>ip-address</i>

Specify Hosts to Receive Audit Trail Messages

To define and change primary and secondary addresses of the host to receive audit messages, perform the following tasks in global configuration mode:

Task	Command
Specify the primary address for the audit trail	dnsix-nat primary <i>ip-address</i>
Specify the secondary address for the audit trail.	dnsix-nat secondary <i>ip-address</i>
Specify the address of a collection center that is authorized to change primary and secondary addresses. Specified hosts are authorized to change the destination of audit messages.	dnsix-nat authorized-redirection <i>ip-address</i>

Specify Transmission Parameters

To specify transmission parameters, perform the following tasks in global configuration mode:

Task	Command
Specify the number of records in a packet before it is sent to a collection center.	dnsix-nat transmit-count <i>count</i>
Specify the number of transmit retries for DMDP.	dnsix-dmdp retries <i>count</i>

Configure IP Accounting

Our IP accounting support provides basic IP accounting functions. By enabling IP accounting, users can see the number of bytes and packets switched through the Cisco IOS software on a source and destination IP address basis. Only transit IP traffic is measured and only on an outbound basis; traffic generated by the software or terminating in the software is not included in the accounting statistics. To maintain accurate accounting totals, the software maintains two accounting databases: an active and a checkpointed database.

Our IP accounting support also provides information identifying IP traffic that fails IP access lists. Identifying IP source addresses that violate IP access lists alerts you to possible attempts to breach security. The data also indicates that you should verify IP access list configurations. To make this feature available to users, you must enable IP accounting of access list violations using the **ip accounting access-violations** command. Users can then display the number of bytes and packets from a single source that attempted to breach security against the access list for the source destination pair. By default, IP accounting displays the number of packets that have passed access lists and were routed.

To enable IP accounting, perform one of the following tasks for each interface in interface configuration mode:

Task	Command
Enable basic IP accounting.	ip accounting

Task	Command
Enable IP accounting with the ability to identify IP traffic that fails IP access lists.	ip accounting access-violations

To configure other IP accounting functions, perform one or more of the following tasks in global configuration mode:

Task	Command
Set the maximum number of accounting entries to be created.	ip accounting-threshold <i>threshold</i>
Filter accounting information for hosts.	ip accounting-list <i>ip-address wildcard</i>
Control the number of transit records that will be stored in the IP accounting database.	ip accounting-transits <i>count</i>

To display IP access violations for a specific IP accounting database, perform the following task in EXEC mode:

Task	Command
Display IP access-violation information.	show ip accounting [checkpoint] access-violations

To display IP access violations, you must give the **access-violations** keyword on the command. If you do not specify the keyword, the command defaults to displaying the number of packets that have passed access lists and were routed. The access violations output displays the number of the access list failed by the last packet for the source and destination pair. The number of packets reveals how aggressive the attack is upon a specific destination.

Use the EXEC command **show ip accounting** to display the active accounting database. To display the checkpointed database, use the **show ip accounting checkpoint** EXEC command. The **clear ip accounting** EXEC command clears the active database and creates the checkpointed database.

Configure Performance Parameters

To tune IP performance, complete the tasks in the following sections:

- Compress TCP Packet Headers
- Set the TCP Connection Attempt Time
- Enable TCP Path MTU Discovery
- Set the TCP Maximum Read Size
- Set the TCP Window Size
- Set the TCP Outgoing Queue Size
- Enable Fast Switching
- Enable Fast Switching on the Same Interface
- Enable SSE Fast Switching
- Configure IP Distributed and NetFlow Switching on VIP Interfaces
- Enable IP Autonomous Switching

- Disable Optimum Switching
- Control Route Cache Invalidation

Compress TCP Packet Headers

You can compress the headers of your TCP/IP packets in order to reduce their size, thereby increasing performance. Header compression is particularly useful on networks with a large percentage of small packets (such as those supporting many Telnet connections). This feature only compresses the TCP header, so it has no effect on UDP packets or other protocol headers. The TCP header compression technique, described fully in RFC 1144, is supported on serial lines using HDLC or PPP encapsulation. You must enable compression on both ends of a serial connection.

You can optionally specify outgoing packets to be compressed only if TCP incoming packets on the same interface are compressed. If you do not specify this option, the Cisco IOS software will compress all traffic. The default is no compression.

You also can specify the total number of header compression connections that can exist on an interface. You should configure one connection for each TCP connection through the specified interface.

To enable compression, perform either of the following optional tasks in interface configuration mode:

Task	Command
Enable TCP header compression.	ip tcp header-compression [passive]
Specify the total number of header compression connections that can exist on an interface.	ip tcp compression-connections <i>number</i>

Note When compression is enabled, fast switching is disabled. Fast processors can handle several fast interfaces, such as T1s, that are running header compression. However, you should think carefully about your network's traffic characteristics before compressing TCP headers. You might want to use the monitoring commands to help compare network utilization before and after enabling header compression.

Set the TCP Connection Attempt Time

You can set the amount of time the Cisco IOS software will wait to attempt to establish a TCP connection. In previous versions of software, the system would wait a fixed 30 seconds when attempting to establish a connection. This amount of time is not sufficient in networks that have dial-up asynchronous connections (such as a network consisting of dial-on-demand links that are implemented over modems) because it will affect your ability to Telnet over the link (from the router) if the link must be brought up.

Because the connection attempt time is a host parameter, it does not pertain to traffic going through the device, just to traffic originated at the device.

To set the TCP connection attempt time, perform the following task in global configuration mode:

Task	Command
Set the amount of time the Cisco IOS software will wait to attempt to establish a TCP connection.	ip tcp synwait-time <i>seconds</i>

Enable TCP Path MTU Discovery

Path MTU Discovery is a method for maximizing the use of available bandwidth in the network between the end points of a TCP connection, and is described in RFC 1191. By default, this feature is disabled. Existing connections are not affected when this feature is turned on or off. To enable Path MTU Discovery, perform the following task in interface configuration mode:

Task	Command
Enable Path MTU Discovery.	ip tcp path-mtu-discovery [age-timer { <i>minutes</i> infinite }]

Customers using TCP connections to move bulk data between systems on distinct subnets would benefit most by enabling this feature. This might include customers using RSRB with TCP encapsulation, STUN, X.25 Remote Switching (also known as XOT or X.25 over TCP), and some protocol translation configurations.

The **ip tcp path-mtu-discovery** command is to enable Path MTU Discovery for connections initiated by the router when it is acting as a host. For a discussion of how the Cisco IOS software supports Path MTU Discovery when the device is acting as a router, see the section “Understand Path MTU Discovery” earlier in this chapter.

The age-timer is a time interval for how often TCP should re-estimate the Path MTU with a larger maximum segment size (MSS). The default path MTU Discovery age-timer is 10 minutes; its maximum is 30 minutes. You can turn off the age-timer by setting it to infinite.

Set the TCP Maximum Read Size

By default, for Telnet and rlogin, the maximum number of characters that TCP reads from the input queue at once is a very large number (the largest possible 32-bit positive number). We do not recommend that you change this value. However, you could change that value by performing the following task in global configuration mode.

Task	Command
Set the TCP maximum read size for Telnet or rlogin.	ip tcp chunk-size <i>characters</i>

Set the TCP Window Size

The default TCP window size is 2144 bytes. We recommend you keep the default value, unless you know your router is sending large packets (greater than 536 bytes). To change the default window size, perform the following task in global configuration mode.

Task	Command
Set the TCP window size.	ip tcp window-size <i>bytes</i>

Set the TCP Outgoing Queue Size

The default TCP outgoing queue size per connection is 5 segments if the connection has a TTY associated with it (like a Telnet connection). If there is no TTY connection associated with it, the default queue size is 20 segments. To change the 5-segment default value, perform the following task in global configuration mode.

Task	Command
Set the TCP outgoing queue size.	<code>ip tcp queuemax packets</code>

Enable Fast Switching

Fast switching involves the use of a high-speed switching cache for IP routing. With fast switching, destination IP addresses are stored in the high-speed cache so that some time-consuming table lookups need not be done. Our routers generally offer better packet transfer performance when fast switching is enabled. However, if you have bursts of traffic, or if slow-speed serial links (64K and below) are being fed from higher-speed media such as T1 or Ethernet, then disabling fast switching can reduce the packet drop rate to some extent. Fast switching allows outgoing packets to be load balanced on a *per-destination* basis.

To enable or disable fast switching, perform either of the following tasks in interface configuration mode:

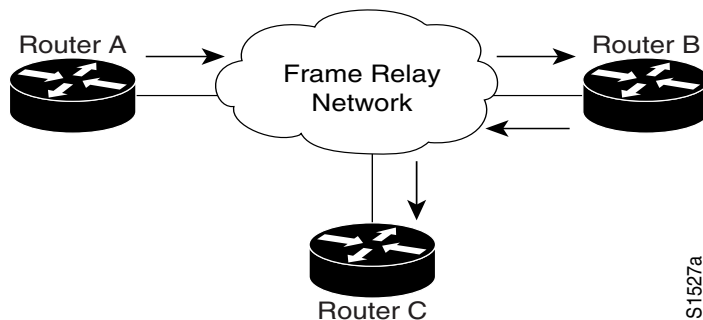
Task	Command
Enable fast switching (use of a high-speed route cache for IP routing).	<code>ip route-cache</code>
Disable fast switching and enable load balancing on a per-packet basis.	<code>no ip route-cache</code>

Enable Fast Switching on the Same Interface

You can enable IP fast switching when the input and output interfaces are the same interface. This normally is not recommended, though it is useful when you have partially meshed media such as Frame Relay. You could use this feature on other interfaces, although it is not recommended because it would interfere with redirection.

Figure 10 illustrates a scenario where this is desirable. Router A has a data link connection identifier (DLCI) to Router B, and Router B has a DLCI to Router C. There is no DLCI between Routers A and C; traffic between them must go in and out of Router B through the same interface.

Figure 10 IP Fast Switching on the Same Interface



Perform the following task in interface configuration mode to allow IP fast switching on the same interface:

Task	Command
Enable the fast switching of packets out of the same interface on which they arrived.	ip route-cache same-interface

Enable SSE Fast Switching

The silicon switching engine (SSE) is on the Silicon Switch Processor (SSP) board in the Cisco 7000. SSE switching contributes to very fast packet processing by allowing the SSE to perform switching independently of the system processor.

To enable SSE fast switching, perform the following task in interface configuration mode:

Task	Command
Enable the SSE fast switching.	ip route-cache sse

The SSE can perform both standard and extended IP access list checking. That is, packets passing either standard or extended IP output access list checks can be SSE-switched. SSE switching of IP packets is not supported if input access lists are used. Prior to Release 10.3, when extended IP access lists were configured on a router having an SSE, IP traffic was fast-switched instead of SSE-switched.

Configure IP Distributed and NetFlow Switching on VIP Interfaces

On Cisco RSP7000 and Cisco 7500 series routers with a Route Switch Processor (RSP) and with Versatile Interface Processor (VIP) controllers, the VIP hardware can be configured to switch packets received by the VIP with no per-packet intervention on the part of the RSP. This process is called *distributed switching*. Distributed switching decreases the demand on the RSP.

The VIP hardware can also be configured for NetFlow switching, a new high-performance feature that identifies initiation of traffic flow between internet endpoints, caches information about the flow, and uses this cache for high-speed switching of subsequent packets within the identified stream.

NetFlow switching data can also be exported to network management applications.

Note Release 11.2 supports distributed switching of IP packets only.

Distributed switching by the VIP is supported for the following *input* VIP port adapters:

- VIP Fast Ethernet (ARPA encapsulation)
- VIP 4E/8E (ARPA encapsulation)
- VIP 4T/8T (HDLC and PPP encapsulations)
- VIP FDDI (SNAP encapsulation)
- VIP 4R Token Ring (SNAP encapsulation with no RIF)

Packets received by the VIP may be distributed switched to the following *output* interface types:

- VIP Fast Ethernet (ARPA encapsulation)
- VIP 4E/8E (ARPA encapsulation)
- VIP 4T/8T (HDLC, Frame Relay, and PPP encapsulations)
- VIP FDDI (SNAP encapsulation)
- FEIP—Fast Ethernet Interface Processor
- EIP—Ethernet Interface Processor
- FIP—FDDI Interface Processor (SNAP encapsulation)
- HIP—High-Speed Serial Interface (HSSI) Interface Processor (HDLC, PPP, or Frame Relay encapsulation)
- FSIP—Fast Serial Interface Processor (HDLC, PPP, or Frame Relay encapsulation)
- AIP—ATM Interface Processor (AAL5MUX and LLC-SNAP encapsulations only)
- TRIP—Token Ring Interface Processor
- MIP—MultiChannel Interface Processor (HDLC, PPP, or Frame Relay encapsulation)

To configure distributed switching on the VIP, first configure the router for IP routing as described in this chapter and the “Configuring IP Routing Protocols” chapter.

After you configure IP routing, perform the following tasks beginning in global configuration mode:

Task	Command
Step 1 Specify the interface, and enter interface configuration mode.	interface <i>type slot/port-adapter lport</i>
Step 2 Enable VIP distributed switching of IP packets on the interface.	ip route-cache distributed
Step 3 Specify either flow or optimum switching.	ip route-cache [flow optimum]

Flow switching is faster than the default optimum fast-switching on Cisco 7507 and 7513 platforms when extended access lists are configured. When the RSP or VIP is flow switching, it uses a flow cache instead of a destination network cache to switch IP packets. The flow cache uses source and destination network address, protocol, and source and destination port numbers to distinguish entries.

To export NetFlow switching cache entries to a workstation when a flow expires, perform the following task in global configuration mode:

Task	Command
Configure the router to export NetFlow cache entries to a workstation.	ip flow-export <i>ip-address udp-port</i>

Enable IP Autonomous Switching

Autonomous switching gives a router faster packet processing by allowing the ciscoBus to switch packets independently without interrupting the system processor. It works only in Cisco 7000 series systems with a switch processor controller card running microcode Version 1.4 or later. (See the information on microcode revisions in the microcode release notes accompanying this publication for other microcode revision requirements.)

By default, IP autonomous switching is not enabled.

Perform any of the following tasks in interface configuration mode as needed for your network:

Task	Command
Enable both fast switching and autonomous switching.	ip route-cache cbus
Disable both fast switching and autonomous switching on an interface.	no ip route-cache
Disable only autonomous switching on an interface.	no ip route-cache cbus

Disable Optimum Switching

Optimum switching is enabled by default for IP on Ethernet, FDDI, and serial interfaces on the Cisco 7500 series RSP. On serial interfaces, it is supported for HDLC encapsulation only. If you want to use fast-switching or process switching, disable optimum switching by performing the following task in interface configuration mode:

Task	Command
Disable optimum switching.	no ip route-cache optimum

Control Route Cache Invalidation

The high-speed route cache used by IP fast switching and autonomous switching is invalidated when the IP routing table changes. By default, the invalidation of the cache is delayed slightly to avoid excessive CPU load while the routing table is changing.

To control route cache invalidation, perform the following tasks in global configuration mode as needed for your network:

Task	Command
Allow immediate invalidation of the cache.	no ip cache-invalidate-delay
Delay invalidation of the cache.	ip cache-invalidate-delay [<i>minimum maximum quiet threshold</i>]

Note This task normally should not be necessary. It should be performed only under the guidance of technical staff. Incorrect configuration can seriously degrade the performance of your router.

Configure IP over WANs

You can configure IP over X.25, SMDS, Frame Relay, and DDR networks. To do this, configure the address mappings, as described in the appropriate chapters of the *Wide-Area Networking Configuration Guide*.

Monitor and Maintain the IP Network

To monitor and maintain your network, perform the tasks in the following sections:

- Clear Caches, Tables, and Databases
- Clear the Access List Counters

- Specify the Format of Network Masks
- Display System and Network Statistics
- Monitor and Maintain NHRP

Clear Caches, Tables, and Databases

You can remove all contents of a particular cache, table, or database. Clearing a cache, table, or database can become necessary when the contents of the particular structure have become or are suspected to be invalid.

The following table lists the tasks associated with clearing caches, tables, and databases. Perform the following tasks as needed in EXEC mode:

Task	Command
Clear the IP ARP cache and the fast-switching cache.	clear arp-cache
Remove one or all entries from the host name and address cache.	clear host { <i>name</i> *}
Clear the active IP accounting or checkpointed database when IP accounting is enabled.	clear ip accounting [checkpoint]
Remove one or more routes from the IP routing table.	clear ip route { <i>network</i> [<i>mask</i>] *}
Cause the SSE route processor on the Cisco 7000 to recompute the program for IP and download it again.	clear ip sse
Cause the route processor on the Cisco 7000 to be reinitialized.	clear sse

Clear the Access List Counters

The system counts how many packets pass each line of an access list; the counters are displayed by the **show access-lists** command. You can clear the counters of an access list by performing the following task in EXEC mode:

Task	Command
Clear the access list counters.	clear access-list counters <i>access-list-number</i> <i>name</i>

Specify the Format of Network Masks

IP uses a 32-bit mask that indicates which address bits belong to the network and subnetwork fields, and which bits belong to the host field. This is called a *netmask*. By default, **show** commands display an IP address and then its netmask in dotted decimal notation. For example, a subnet would be displayed as 131.108.11.55 255.255.255.0.

You might find it more convenient to display the network mask in hexadecimal format or bitcount format instead. The hexadecimal format is commonly used on UNIX systems. The previous example would be displayed as 131.108.11.55 0XFFFFFF00.

The bitcount format for displaying network masks is to append a slash (/) and the total number of bits in the netmask to the address itself. The previous example would be displayed as 131.108.11.55/24.

To specify the format in which netmasks appear for the current session, perform the following task in EXEC mode:

Task	Command
Specify the format of network masks for the current session.	term ip netmask-format { bitcount decimal hexadecimal }

To configure the format in which netmasks appear for an individual line, perform the following task in line configuration mode:

Task	Command
Configure the format of network masks for a line.	ip netmask-format { bitcount decimal hexadecimal }

Display System and Network Statistics

You can display specific statistics such as the contents of IP routing tables, caches, and databases. The resulting information can be used to determine resource utilization and to solve network problems. You also can display information about node reachability and discover the routing path that your device's packets are taking through the network.

These tasks are summarized in the table that follows. See the "IP Commands" chapter in the *Network Protocols Command Reference, Part 1* for details about the commands listed in these tasks. Perform any of the following tasks in privileged EXEC mode:

Task	Command
Display the contents of one or all current access lists.	show access-lists [<i>access-list-number</i> <i>name</i>]
Display the entries in the ARP table.	show arp
Display state information and the current configuration of the DNSIX audit writing module.	show dnsix
Display the default domain name, style of lookup service, the name server hosts, and the cached list of host names and addresses.	show hosts
Display the contents of current IP access lists.	show ip access-list [<i>access-list-number</i> <i>name</i>]
Display the active IP accounting or checkpointed database.	show ip accounting [checkpoint]
Display IP addresses mapped to TCP ports (aliases).	show ip aliases
Display the IP ARP cache.	show ip arp
Display the routing table cache used to fast switch IP traffic.	show ip cache [<i>prefix mask</i>] [<i>type number</i>]
Display the usability status of interfaces.	show ip interface [<i>type number</i>]
Display the masks used for network addresses and the number of subnets using each mask.	show ip masks <i>address</i>
Display the address of a default gateway.	show ip redirects
Display the current state of the routing table.	show ip route [<i>address [mask]</i>] [<i>protocol</i>]
Display the current state of the routing table in summary form.	show ip route summary

Task	Command
Show statistics on TCP header compression.	show ip tcp header-compression
Display IP protocol statistics.	show ip traffic
Display a summary of SSP statistics.	show sse summary
Display the status of the standby router.	show standby
Test network node reachability (privileged).	ping [<i>protocol</i>] { <i>host</i> <i>address</i> }
Test network node reachability using a simple ping facility (user).	ping [<i>protocol</i>] { <i>host</i> <i>address</i> }
Trace packet routes through the network (privileged).	trace [<i>destination</i>]
Trace packet routes through the network (user).	trace ip <i>destination</i>

See the “Ping Command Example” section at the end of this chapter for an example of pinging.

Monitor and Maintain NHRP

To monitor the NHRP cache or traffic, perform either of the following tasks in EXEC mode:

Task	Command
Display the IP NHRP cache, optionally limited to dynamic or static cache entries for a specific interface.	show ip nhrp [dynamic static] [<i>type number</i>]
Display NHRP traffic statistics.	show ip nhrp traffic

The NHRP cache can contain static entries caused by statically configured addresses and dynamic entries caused by the Cisco IOS software learning addresses from NHRP packets. To clear static entries, use the **no ip nhrp map** command. To clear the NHRP cache of dynamic entries, perform the following task in EXEC mode:

Task	Command
Clear the IP NHRP cache of dynamic entries.	clear ip nhrp

IP Configuration Examples

The following sections provide IP configuration examples:

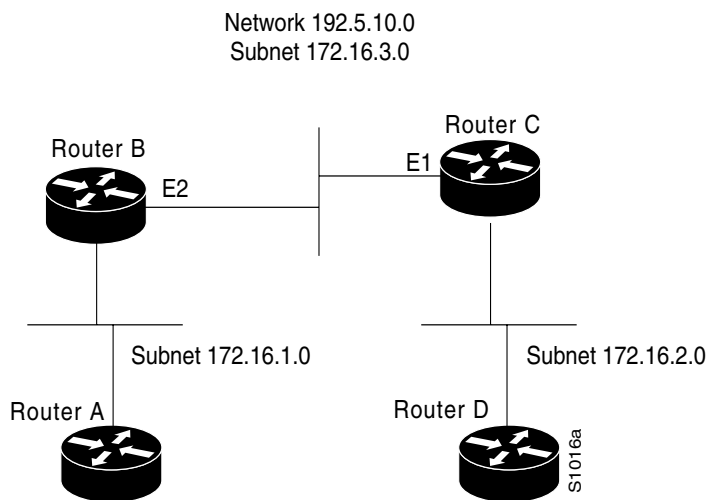
- Creating a Network from Separated Subnets Example
- Serial Interfaces Configuration Example
- IP Domains Example
- Dynamic Lookup Example
- HP Hosts on a Network Segment Example
- Logical NBMA Example
- NHRP over ATM Example
- NHRP on a Multipoint Tunnel Example
- Broadcasting Examples

- Helper Addresses Example
- ICMP Services Example
- Simplex Ethernet Interfaces Example
- Numbered Access List Examples
- Named Access List Example
- NAT Configuration Examples
- IPSO Configuration Examples
- Ping Command Example

Creating a Network from Separated Subnets Example

In the following example, subnets 1 and 2 of network 131.108.0.0 are separated by a backbone, as shown in Figure 11. The two networks are brought into the same logical network through the use of secondary addresses.

Figure 11 Creating a Network from Separated Subnets



The following examples show the configurations for Routers B and C.

Configuration for Router B

```
interface ethernet 2
ip address 192.5.10.1 255.255.255.0
ip address 131.108.3.1 255.255.255.0 secondary
```

Configuration for Router C

```
interface ethernet 1
ip address 192.5.10.2 255.255.255.0
ip address 131.108.3.2 255.255.255.0 secondary
```

Serial Interfaces Configuration Example

In the following example, the second serial interface (serial 1) is given Ethernet 0's address. The serial interface is unnumbered.

```
interface ethernet 0
ip address 145.22.4.67 255.255.255.0
interface serial 1
ip unnumbered ethernet 0
```

IP Domains Example

The example that follows establishes a domain list with several alternate domain names.

```
ip domain-list csi.com
ip domain-list telecomprog.edu
ip domain-list merit.edu
```

Dynamic Lookup Example

A cache of host name-to-address mappings is used by **connect**, **telnet**, **ping**, **trace**, **write net**, and **configure net EXEC** commands to speed the process of converting names to addresses. The commands used in this example specify the form of dynamic name lookup to be used. Static name lookup also can be configured.

The following example configures the host name-to-address mapping process. IP DNS-based translation is specified, the addresses of the name servers are specified, and the default domain name is given.

```
! IP Domain Name System (DNS)-based host name-to-address translation is enabled
ip domain-lookup
! Specifies host 131.108.1.111 as the primary name server and host 131.108.1.2
! as the secondary server
ip name-server 131.108.1.111 131.108.1.2
! Defines cisco.com as the default domain name the router uses to complete
! unqualified host names
ip domain-name cisco.com
```

HP Hosts on a Network Segment Example

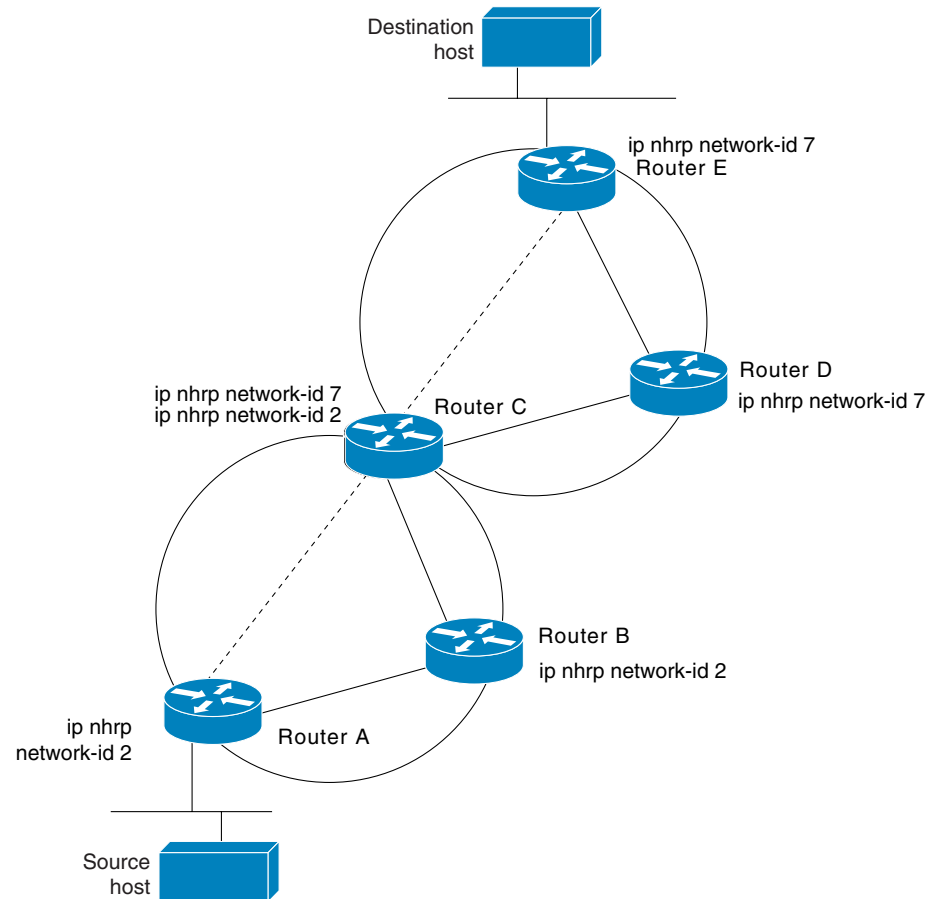
The following example has a network segment with Hewlett-Packard devices on it. The commands in this example customize the first Ethernet port to respond to Probe name requests for bl4zip and to use Probe as well as ARP.

```
ip hp-host bl4zip 131.24.6.27
interface ethernet 0
arp probe
ip probe proxy
```

Logical NBMA Example

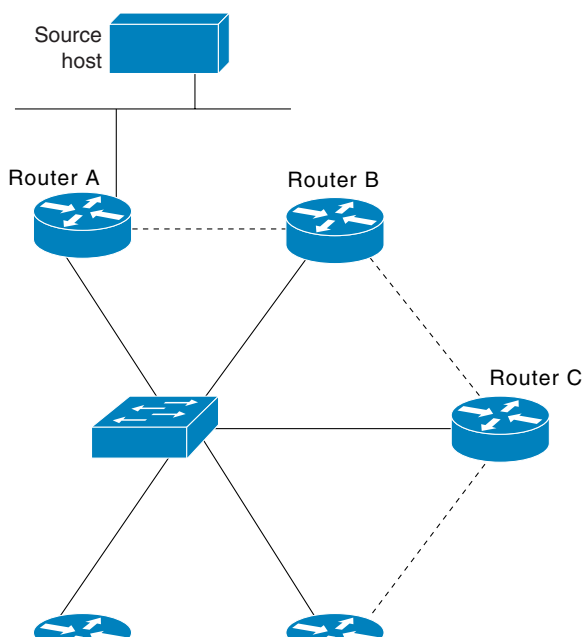
A logical NBMA network is considered the group of interfaces and hosts participating in NHRP and having the same network identifier. Figure 12 illustrates two logical NBMA networks (shown as circles) configured over a single physical NBMA network. Router A can communicate with Routers B and C because they share the same network identifier (2). Router C can also communicate with Routers D and E, as they share network identifier 7. After address resolution is complete, Router A can send IP packets to Router C in one hop, and Router C can send them to Router E in one hop, as shown by the dotted lines.

Figure 12 Two Logical NBMA Networks over One Physical NBMA Network



The physical configuration of the five routers in Figure 12 might actually be that shown in Figure 13. The source host is connected to Router A and the destination host is connected to Router E. The same switch serves all five routers, making one physical NBMA network.

Figure 13 Physical Configuration of a Sample NBMA Network



Refer again to Figure 12. Initially, before NHRP has resolved any NBMA addresses, IP packets from the source host to the destination host travel through all five routers connected to the switch before reaching the destination. When Router A first forwards the IP packet toward the destination host, Router A also generates an NHRP request for the destination host's IP address. The request is forwarded to Router C, whereupon a reply is generated. Router C replies because it is the egress router between the two logical NBMA networks.

Similarly, Router C generates an NHRP request of its own, to which Router E replies. In this example, subsequent IP traffic between the source and the destination still requires two hops to traverse the NBMA network, since the IP traffic must be forwarded between the two logical NBMA networks. Only one hop would be required if the NBMA network were not logically divided.

NHRP over ATM Example

The following example shows a configuration of three routers using NHRP over ATM. Additionally, subinterfaces and dynamic routing are used. Router A obtains an OSPF route that it can use to reach the LIS where Router B resides. Router A can then initially reach Router B through Router C. Router A and Router B are able to directly communicate without Router C once NHRP has resolved Router A's and Router C's respective NSAP addresses.

The significant portions of the configurations for Routers A, B, and C follow.

Router A

```
interface ATM0/0
 ip address 10.1.0.1 255.255.0.0
 ip nhrp network-id 1
 map-group a
 atm nsap-address 11.1111.11.111111.1111.1111.1111.1111.1111.11
 atm rate-queue 1 10
 atm pvc 1 0 5 qsaal

router ospf 1
 network 10.0.0.0 0.255.255.255 area 0

map-list a
 ip 10.1.0.3 atm-nsap 33.3333.33.333333.3333.3333.3333.3333.3333.33
```

Router B

```
interface ATM0/0
 ip address 10.2.0.2 255.255.0.0
 ip nhrp network-id 1
 map-group a
 atm nsap-address 22.2222.22.222222.2222.2222.2222.2222.2222.22
 atm rate-queue 1 10
 atm pvc 2 0 5 qsaal

router ospf 1
 network 10.0.0.0 0.255.255.255 area 0

map-list a
 ip 10.2.0.3 atm-nsap 33.3333.33.333333.3333.3333.3333.3333.3333.33
```

Router C

```
interface ATM0/0
 no ip address
 atm rate-queue 1 10
 atm pvc 2 0 5 qsaal

interface ATM0/0.1 multipoint
 ip address 10.1.0.3 255.255.0.0
 ip nhrp network-id 1
 map-group a
 atm nsap-address 33.3333.33.333333.3333.3333.3333.3333.3333.33
 atm rate-queue 1 10

interface ATM0/0.2 multipoint
 ip address 10.2.0.3 255.255.0.0
 ip nhrp network-id 1
 map-group b
 atm nsap-address 33.3333.33.333333.3333.3333.3333.3333.3333.33
 atm rate-queue 1 10

router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
 neighbor 10.1.0.1 priority 1
 neighbor 10.2.0.2 priority 1

map-list a
 ip 10.1.0.1 atm-nsap 11.1111.11.111111.1111.1111.1111.1111.1111.11

map-list b
 ip 10.2.0.2 atm-nsap 22.2222.22.222222.2222.2222.2222.2222.2222.22
```

NHRP on a Multipoint Tunnel Example

With multipoint tunnels, a single tunnel interface may be connected to multiple neighboring routers. Unlike point-to-point tunnels, a tunnel destination need not be configured. In fact, if configured, the tunnel destination must correspond to an IP multicast address. Broadcast or multicast packets to be sent over the tunnel interface can then be transmitted by sending the GRE packet to the multicast address configured as the tunnel destination.

Multipoint tunnels require that you configure a tunnel key. Otherwise, unexpected GRE traffic could easily be received by the tunnel interface. For simplicity, it is recommended that the tunnel key correspond to the NHRP network identifier.

In the following example, Routers A, B, C, and D all share a common Ethernet segment. Minimal connectivity over the multipoint tunnel network is configured, thus creating a network that can be treated as a partially meshed NBMA network. Due to the static NHRP map entries, Router A knows how to reach Router B, Router B knows how to reach Router C, Router C knows how to reach Router D, and Router D knows how to reach Router A.

When Router A initially attempts to send an IP packet to Router D, the packet is forwarded through Routers B and C. Through NHRP, the routers quickly learn each other's NBMA addresses (in this case, IP addresses assigned to the underlying Ethernet network). The partially meshed tunnel network readily becomes fully meshed, at which point any of the routers can directly communicate over the tunnel network without their IP traffic requiring an intermediate hop.

The significant portions of the configurations for Routers A, B, C, and D follow.

Router A

```
interface tunnel 0
no ip redirects
ip address 11.0.0.1 255.0.0.0
ip nhrp map 11.0.0.2 10.0.0.2
ip nhrp network-id 1
ip nhrp nhs 11.0.0.2
tunnel source ethernet 0
tunnel mode gre multipoint
tunnel key 1

interface ethernet 0
ip address 10.0.0.1 255.0.0.0
```

Router B

```
interface tunnel 0
no ip redirects
ip address 11.0.0.2 255.0.0.0
ip nhrp map 11.0.0.3 10.0.0.3
ip nhrp network-id 1
ip nhrp nhs 11.0.0.3
tunnel source ethernet 0
tunnel mode gre multipoint
tunnel key 1

interface ethernet 0
ip address 10.0.0.2 255.0.0.0
```

Router C

```
interface tunnel 0
no ip redirects
```

```

ip address 11.0.0.3 255.0.0.0
ip nhrp map 11.0.0.4 10.0.0.4
ip nhrp network-id 1
ip nhrp nhs 11.0.0.4
tunnel source ethernet 0
tunnel mode gre multipoint
tunnel key 1

interface ethernet 0
ip address 10.0.0.3 255.0.0.0

```

Router D

```

interface tunnel 0
no ip redirects
ip address 11.0.0.4 255.0.0.0
ip nhrp map 11.0.0.1 10.0.0.1
ip nhrp network-id 1
ip nhrp nhs 11.0.0.1
tunnel source ethernet 0
tunnel mode gre multipoint
tunnel key 1

interface ethernet 0
ip address 10.0.0.4 255.0.0.0

```

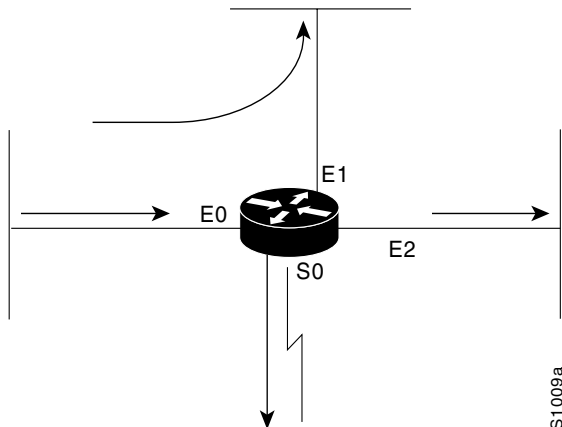
Broadcasting Examples

The Cisco IOS software supports two types of broadcasting: directed broadcasting and flooding. A directed broadcast is a packet sent to a specific network or series of networks, while a flooded broadcast packet is sent to every network. The following examples describe configurations for both types of broadcasting.

Flooded Broadcast Example

Figure 14 shows a flooded broadcast packet being sent to every network. The packet that is incoming from interface E0 is flooded to interfaces E1, E2, and S0.

Figure 14 IP Flooded Broadcast



A directed broadcast address includes the network or subnet fields. For example, if the network address is 128.1.0.0, the address 128.1.255.255 indicates all hosts on network 128.1.0.0. This would be a directed broadcast. If network 128.1.0.0 has a subnet mask of 255.255.255.0 (the third octet is the subnet field), the address 128.1.5.255 specifies all hosts on subnet 5 of network 128.1.0.0—another directed broadcast.

Flooding of IP Broadcasts Example

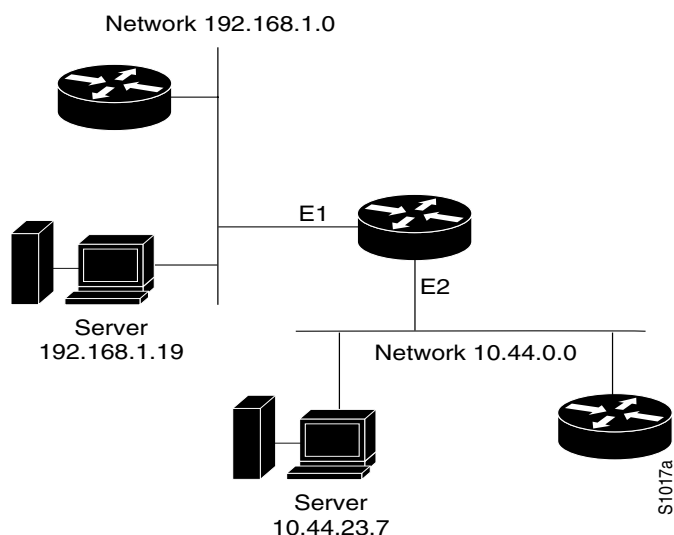
In the following example, flooding of IP broadcasts is enabled on all interfaces (two Ethernet and two serial). No bridging is permitted. The access list denies all protocols. No specific UDP protocols are listed by a separate **ip forward-protocol udp** interface configuration command, so the default protocols (TFTP, DNS, Time, NetBIOS, and BOOTP) will be flooded.

```
ip forward-protocol spanning-tree
bridge 1 protocol dec
access-list 201 deny 0x0000 0xFFFF
interface ethernet 0
bridge-group 1
bridge-group 1 input-type-list 201
interface ethernet 1
bridge-group 1
bridge-group 1 input-type-list 201
interface serial 0
bridge-group 1
bridge-group 1 input-type-list 201
interface serial 1
bridge-group 1
bridge-group 1 input-type-list 201
```

Helper Addresses Example

In the following example, one router is on network 191.24.1.0 and the other is on network 110.44.0.0, and you want to permit IP broadcasts from hosts on either network segment to reach both servers. Figure 15 illustrates how to configure the router that connects network 110 to network 191.24.1.

Figure 15 IP Helper Addresses



The following example shows the configuration:

```
ip forward-protocol udp
!
interface ethernet 1
ip helper-address 110.44.23.7
interface ethernet 2
ip helper-address 191.24.1.19
```

ICMP Services Example

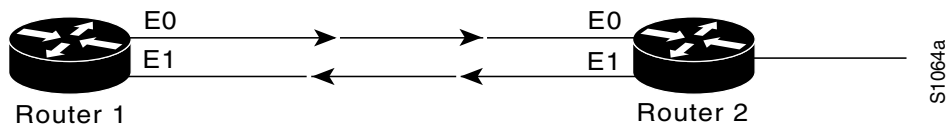
The example that follows changes some of the ICMP defaults for the first Ethernet interface 0. Disabling the sending of redirects could mean that you do not think your devices on this segment will ever have to send a redirect. Disabling the Unreachables messages will have a secondary effect—it also will disable IP Path MTU Discovery, because path discovery works by having the Cisco IOS software send Unreachables messages. If you have a network segment with a small number of devices and an absolutely reliable traffic pattern—which could easily happen on a segment with a small number of little-used user devices—you would be disabling options that your device would be unlikely to use anyway.

```
interface ethernet 0
no ip unreachable
no ip redirects
```

Simplex Ethernet Interfaces Example

The following is an example of configuring a simplex Ethernet interface. Figure 16 illustrates how to configure IP on two routers sharing transmit-only and receive-only Ethernet connections.

Figure 16 Simplex Ethernet Connections



Configuration for Router 1

```
interface ethernet 0
ip address 128.9.1.1
!
interface ethernet 1
ip address 128.9.1.2
transmit-interface ethernet 0
!
!use show interfaces command to find router2-MAC-address-E0
arp 128.9.1.4 router2-MAC-address-E0 arpa
```

Configuration for Router 2

```
interface ethernet 0
ip address 128.9.1.3
transmit-interface ethernet 1
!
```

```
interface ethernet 1
ip address 128.9.1.4
!
!use show interfaces command to find router1-MAC-address-E1
arp 128.9.1.1 router1-MAC-address-E1 arpa
```

Numbered Access List Examples

In the following example, network 36.0.0.0 is a Class A network whose second octet specifies a subnet; that is, its subnet mask is 255.255.0.0. The third and fourth octets of a network 36.0.0.0 address specify a particular host. Using access list 2, the Cisco IOS software would accept one address on subnet 48 and reject all others on that subnet. The last line of the list shows that the software would accept addresses on all other network 36.0.0.0 subnets.

```
access-list 2 permit 36.48.0.3
access-list 2 deny 36.48.0.0 0.0.255.255
access-list 2 permit 36.0.0.0 0.255.255.255
interface ethernet 0
ip access-group 2 in
```

Implicit Masks in Access Lists Examples

IP access lists contain *implicit* masks. For instance, if you omit the mask from an associated IP host address access list specification, 0.0.0.0 is assumed to be the mask. Consider the following example configuration:

```
access-list 1 permit 0.0.0.0
access-list 1 permit 131.108.0.0
access-list 1 deny 0.0.0.0 255.255.255.255
```

For this example, the following masks are implied in the first two lines:

```
access-list 1 permit 0.0.0.0 0.0.0.0
access-list 1 permit 131.108.0.0 0.0.0.0
```

The last line in the configuration (using the deny keyword) can be left off, because IP access lists implicitly *deny* all other access. This is equivalent to finishing the access list with the following command statement:

```
access-list 1 deny 0.0.0.0 255.255.255.255
```

The following access list only allows access for those hosts on the three specified networks. It assumes that subnetting is not used; the masks apply to the host portions of the network addresses. Any hosts with a source address that does not match the access list statements will be rejected.

```
access-list 1 permit 192.5.34.0 0.0.0.255
access-list 1 permit 128.88.0.0 0.0.255.255
access-list 1 permit 36.0.0.0 0.255.255.255
! (Note: all other access implicitly denied)
```

To specify a large number of individual addresses more easily, you can omit the address mask that is all zeros from the **access-list** global configuration command. Thus, the following two configuration commands are identical in effect:

```
access-list 2 permit 36.48.0.3
access-list 2 permit 36.48.0.3 0.0.0.0
```

Extended Access List Examples

In the following example, the first line permits any incoming TCP connections with destination ports greater than 1023. The second line permits incoming TCP connections to the SMTP port of host 128.88.1.2. The last line permits incoming ICMP messages for error feedback.

```
access-list 102 permit tcp 0.0.0.0 255.255.255.255 128.88.0.0 0.0.255.255 gt 1023
access-list 102 permit tcp 0.0.0.0 255.255.255.255 128.88.1.2 0.0.0.0 eq 25
access-list 102 permit icmp 0.0.0.0 255.255.255.255 128.88.0.0 255.255.255.255
interface ethernet 0
ip access-group 102 in
```

For another example of using an extended access list, suppose you have a network connected to the Internet, and you want any host on an Ethernet to be able to form TCP connections to any host on the Internet. However, you do not want IP hosts to be able to form TCP connections to hosts on the Ethernet except to the mail (SMTP) port of a dedicated mail host.

SMTP uses TCP port 25 on one end of the connection and a random port number on the other end. The same two port numbers are used throughout the life of the connection. Mail packets coming in from the Internet will have a destination port of 25. Outbound packets will have the port numbers reversed. The fact that the secure system behind the router always will be accepting mail connections on port 25 is what makes it possible to separately control incoming and outgoing services. The access list can be configured on either the outbound or inbound interface.

In the following example, the Ethernet network is a Class B network with the address 128.88.0.0, and the mail host's address is 128.88.1.2. The keyword **established** is used only for the TCP protocol to indicate an established connection. A match occurs if the TCP datagram has the ACK or RST bits set, which indicate that the packet belongs to an existing connection.

```
access-list 102 permit tcp 0.0.0.0 255.255.255.255 128.88.0.0 0.0.255.255 established
access-list 102 permit tcp 0.0.0.0 255.255.255.255 128.88.1.2 0.0.0.0 eq 25
interface ethernet 0
ip access-group 102 in
```

Named Access List Example

The following configuration creates a standard access list named `Internet_filter` and an extended access list named `marketing_group`:

```
interface Ethernet0/5
ip address 2.0.5.1 255.255.255.0
ip access-group Internet_filter out
ip access-group marketing_group in
...
ip access-list standard Internet_filter
permit 1.2.3.4
deny any
ip access-list extended marketing_group
permit tcp any 171.69.0.0 0.0.255.255 eq telnet
deny tcp any any
permit icmp any any
deny udp any 171.69.0.0 0.0.255.255 lt 1024
deny ip any any log
```

NAT Configuration Examples

The following are NAT configuration examples.

Dynamic Inside Source Translation Example

The following example translates all source addresses passing access list 1 (having a source address from 192.168.1.0/24) to an address from the pool named net-208. The pool contains addresses from 171.69.233.208 to 171.69.233.233.

```
ip nat pool net-208 171.69.233.208 171.69.233.233 netmask 255.255.255.240
ip nat inside source list 1 pool net-208
!
interface serial 0
ip address 171.69.232.182 255.255.255.240
ip nat outside
!
interface ethernet 0
ip address 192.168.1.94 255.255.255.0
ip nat inside
!
access-list 1 permit 192.168.1.0 0.0.0.255
```

Overloading Inside Global Addresses Example

The following example creates a pool of addresses named net-208. The pool contains addresses from 171.69.233.208 to 171.69.233.233. Access list 1 allows packets having the source address from 192.168.1.0 to 192.168.1.255. If no translation exists, packets matching access list 1 are translated to an address from the pool. The router allows multiple local addresses (192.168.1.0 to 192.168.1.255) to use the same global address. The router retains port numbers to differentiate the connections.

```
ip nat pool net-208 171.69.233.208 171.69.233.233 netmask 255.255.255.240
ip nat inside source list 1 pool net-208 overload
!
interface serial0
ip address 171.69.232.182 255.255.255.240
ip nat outside
!
interface ethernet0
ip address 192.168.1.94 255.255.255.0
ip nat inside
!
access-list 1 permit 192.168.1.0 0.0.0.255
```

Translating Overlapping Address Example

In the following example, the addresses in the local network are being used legitimately by someone else on the Internet. An extra translation is required to access that external network. Pool net-10 is a pool of outside local IP addresses. The statement `ip nat outside source list 1 pool net-10` translates the addresses of hosts from the outside overlapping network to addresses in that pool.

```
ip nat pool net-208 171.69.233.208 171.69.233.233 prefix-length 28
ip nat pool net-10 10.0.1.0 10.0.1.255 prefix-length 24
ip nat inside source list 1 pool net-208
ip nat outside source list 1 pool net-10
!
interface serial 0
ip address 171.69.232.192 255.255.255.240
ip nat outside
!
```

```

interface ethernet0
ip address 192.168.1.94 255.255.255.0
ip nat inside
!
access-list 1 permit 192.168.1.0 0.0.0.255

```

TCP Load Distribution Example

In the following example, the goal is to define a virtual address, connections to which are distributed among a set of real hosts. The pool defines the addresses of the real hosts. The access list defines the virtual address. If a translation does not already exist, TCP packets from serial 0 (the outside interface) whose destination matches the access list are translated to an address from the pool.

```

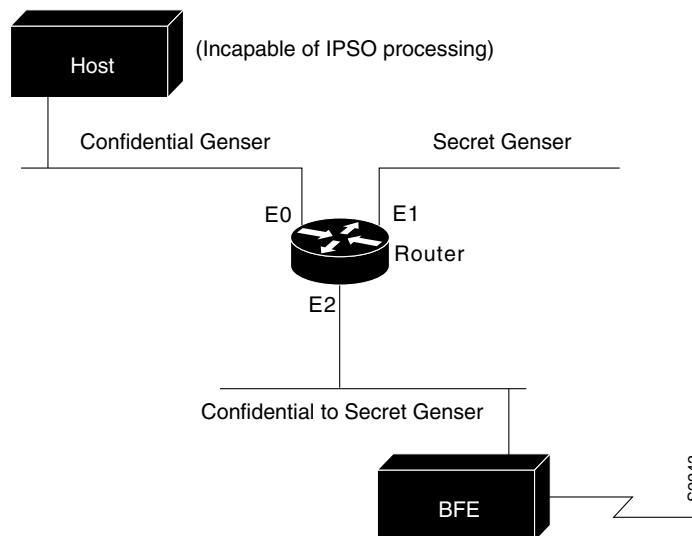
ip nat pool real-hosts 192.168.15.2 192.168.15.15 prefix-length 28 type rotary
ip nat inside destination list 2 pool real-hosts
!
interface serial 0
ip address 192.168.15.129 255.255.255.240
ip nat outside
!
interface ethernet 0
ip address 192.168.15.17 255.255.255.240
ip nat inside
!
access-list 2 permit 192.168.15.1

```

IPSO Configuration Examples

In the following example, three Ethernet interfaces are presented. These interfaces are running at security levels of Confidential Genser, Secret Genser, and Confidential to Secret Genser, as shown in Figure 17.

Figure 17 IPSO Security Levels



The following commands set up interfaces for the configuration in Figure 17:

```
interface ethernet 0
ip security dedicated confidential genser
interface ethernet 1
ip security dedicated secret genser
interface ethernet 2
ip security multilevel confidential genser to secret genser
```

It is possible for the setup to be much more complex.

In the following example, there are devices on Ethernet 0 that cannot generate a security option, and so must accept packets without a security option. These hosts do not understand security options; therefore, never place one on such interfaces. Furthermore, there are hosts on the other two networks that are using the extended security option to communicate information, so you must allow these to pass through the system. Finally, there also is a host (a Blacker Front End; see the “Configuring X.25 and LABP” chapter of the *Wide-Area Networking Configuration Guide* for more information about Blacker emergency mode) on Ethernet 2 that requires the security option to be the first option present, and this condition also must be specified. The new configuration follows.

```
interface ethernet 0
ip security dedicated confidential genser
ip security implicit-labelling
ip security strip
interface ethernet 1
ip security dedicated secret genser
ip security extended-allowed
!
interface ethernet 2
ip security multilevel confidential genser to secret genser
ip security extended-allowed
ip security first
```

The following example configures a Cisco router with HP-UX CMW DNSIX hosts. The following commands should be configured on each LAN interface of the router in order for two DNSIX hosts to communicate:

```
ip security multilevel unclassified nsa to top secret nsa
ip security extended allowed
```

DNSIX hosts do not need to know the router’s IP addresses, and DNSIX hosts do not need to set up M6RHDB entries for the routers.

Ping Command Example

You can specify the address to use as the source address for ping packets. In the following example, it is 131.108.105.62:

```
Sandbox# ping
Protocol [ip]:
Target IP address: 131.108.1.111
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: yes
Source address: 131.108.105.62
Type of service [0]:
Set DF bit in IP header? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 131.108.1.111, timeout is 2 seconds:
!!!!
Success rate is 100 percent, round-trip min/avg/max = 4/4/4 ms
```

