CHAPTER 18

Configuring IP

The Internet Protocol (IP) is a packet-based protocol used to exchange data over computer networks. IP handles addressing, fragmentation, reassembly, and protocol demultiplexing. It is the foundation on which all other IP protocols, collectively referred to as the IP Protocol suite, are built. IP is a network-layer protocol that contains addressing and control information that allows data packets to be routed.

The Transmission Control Protocol (TCP) is built upon the IP layer. TCP is a connection-oriented protocol that specifies the format of data and acknowledgments used in the transfer of data. TCP also specifies the procedures that the computers use to ensure that the data arrives correctly. TCP allows multiple applications on a system to communicate concurrently because it handles all demultiplexing of the incoming traffic among the application programs.

This chapter describes how to configure the IP protocol. For a complete description of the commands in this chapter, refer to the "IP Commands" chapter of the *Access and Communication Servers Command Reference* publication. For information on configuring the various IP routing protocols, refer to the "Configuring IP Routing Protocols" chapter of this manual. For historical background and a technical overview of IP, see the *Internetworking Technology Overview* publication.

Cisco's Implementation of IP

Cisco's implementation of IP provides most of the major services contained in the various protocol specifications. Cisco communication servers also provide the TCP and User Datagram Protocol (UDP) services called Echo and Discard, which are described in RFCs 862 and 863, respectively.

Cisco supports both TCP and UDP at the transport layer, for maximum flexibility in services. Cisco also supports all standards for IP broadcasts.

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 IP Addressing Options
- Configure the Next Hop Resolution Protocol
- Disable IP Routing (if desired)

- Configure a Routing Process
- Configure Broadcast Packet Handling
- Enable Directed Broadcast-to-Physical Broadcast Translation
- Establish an IP Broadcast Address
- Configure IP Services
- Filter IP Packets
- 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 of the tasks in these sections are required. The tasks you need to 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

IP addresses identify locations to which IP datagrams can be sent. See the *Internetworking Technology Overview* publication for detailed information on IP addresses.

Some IP addresses are reserved for special uses and cannot be used for host, subnet, or network addresses. Table 18-1 lists ranges of IP addresses and shows which addresses are reserved and which are available for use.

Table 18-1 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.

To assign an IP address and a network mask to a network interface on the communication server, perform the following task in interface configuration mode:

Task	Command
Set an IP address for an interface.	**ip address** *ip-address mask*

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*. Subnets are described in the *Internetworking Technology Overview* publication.

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 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, your subnetting allows up to 254 hosts per logical subnet, but on one physical subnet you need to have 300 host addresses. Using secondary IP addresses on the communication 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, communication server-based network. Routers on an older, bridged segment can easily be made aware that there are many subnets 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 communication server at a time.

Note If any communication server on a network segment uses a secondary address, all other communication servers 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** *ip-address mask* **secondary**

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 "IP Configuration Examples" section at the end of the 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.	**ip subnet-zero**

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, and LAPB 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.

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 not to 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** *type number*

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 must also be enabled (listed as "up" in the **show interfaces** command display).

An example of how to configure serial interfaces can be found in the "IP Configuration Examples" section at the end of the chapter.

Configure IP Addressing Options

With our IP implementation, you can control interface-specific handling of IP addresses by facilitating address resolution, name services, and other functions. The following sections describe how to configure IP addressing options:

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

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 the device belongs to. 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 communication server 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 communication server uses three forms of address resolution: Address Resolution Protocol (ARP), proxy ARP, and Probe (which is similar to ARP). The communication server 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 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 communication server interface. RARP often is used by diskless nodes that do not know their IP addresses when they boot. Our communication servers attempt to use RARP if they do not know the IP address of an interface at startup. Also, the communication servers are able to act as RARP servers by responding to RARP requests that they are able to answer. See the "Loading System Images and Configuration Files" chapter to learn how to configure a communication server as a RARP server.

Perform the following tasks to set address resolution on the communication server:

- Define a static ARP, as necessary.
- Set ARP encapsulation.
- Set proxy ARP.

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 do need to define them, you can do so globally. Doing this task installs a permanent entry in the ARP cache. The communication server uses this entry to translate 32-bit IP addresses into 48-bit hardware addresses.

Optionally, you can specify that the communication server respond to ARP requests as if it were the owner of the specified IP address, and 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 dynamic 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** *ip-address hardware-address type*
Specify that the communication server respond to ARP requests as if it were the owner of the specified IP address.	**arp** *ip-address hardware-address type* **alias**

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** *seconds*

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 communication server 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 communication server 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 communication server 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 communication server receives an ARP request for a host that is not on the same network as the ARP request sender, and if the communication server has the best route to that host, then the communication server sends an ARP reply packet giving its own local data link address. The host that sent the ARP request then sends its packets to the communication server, 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**

Map Host Names to IP Addresses

Each unique IP address can have a host name associated with it. The communication server 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 Name System (DNS), the Internet's global naming scheme that uniquely identifies network devices. You do these by performing the following tasks:

- Map IP addresses to host names.
- Specify the domain name.
- Specify a Name Server.
- Disable the DNS.

The following sections describe these tasks.

Map IP Addresses to Host Names

The communication server maintains a table of host names and their corresponding addresses, also called host name-to-address mapping. Higher-layer protocols such as Telnet use host names to identify network devices (hosts). The communication server 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** *hostname* [*tcp-port-number*] *address1* [*address2...address8*]

Specify the Domain Name

You can specify a default domain name that the communication server 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 communication server will use to complete unqualified host names.	**ip domain-name** *name*
or	
Define a list of default domain names to complete unqualified host names.	**ip domain-list** *name*

See the "IP Configuration Examples" 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 Domain Name System (DNS), perform the following task in global configuration mode:

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

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.

Disable the DNS

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 "IP Configuration Examples" section at the end of this chapter for an example of enabling the DNS.

Configure HP Probe Proxy Name Requests

HP Probe Proxy support allows a communication server to respond to HP Probe Proxy name requests. These requests are typically used at sites that have Hewlett-Packard (HP) equipment and are already using HP Probe. 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 communication server 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 communication server is acting as a proxy) into the host table.	**ip hp-host** *hostname ip-address*

See the "IP Configuration Examples" section at the end of this chapter for an example of configuring HP hosts on a network segment.

Configure the Next Hop Resolution Protocol

Communication Servers and hosts can use Next Hop Resolution Protocol (NHRP) to discover the addresses of other communication servers and hosts connected to a nonbroadcast, multiaccess (NBMA) network. In the past, partially meshed NBMA networks had to be configured with overlapping LIS (logically independent IP subnets). In such configurations, packets might have had to make several hops over the NBMA network before arriving at the exit communication server (the communication server nearest the destination network). In addition, such NBMA networks (whether partially or fully meshed) have typically required tedious static configurations. These static configurations provided the mapping between network layer addresses (such as IP) and NBMA addresses (such as E.164 addresses for SMDS).

NHRP provides an ARP-like solution that alleviates these NBMA network problems. With NHRP, systems attached to an NBMA network can dynamically learn the NBMA address of the other systems that are part of that network. These systems can then directly communicate without using an intermediate hop, which reduces traffic.

The NBMA network can be considered anonbroadcast network either because it technically does not support broadcasting (for example, an X.25 network) or because broadcasting is not feasible (for example, an SMDS broadcast group or an extended Ethernet that would be too large).

Cisco's Implementation of NHRP

Cisco's initial implementation of NHRP supports only IP Version 4. Currently, NHRP can run on 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.

Figure 18-1 Next Hop Resolution Protocol NHRP)

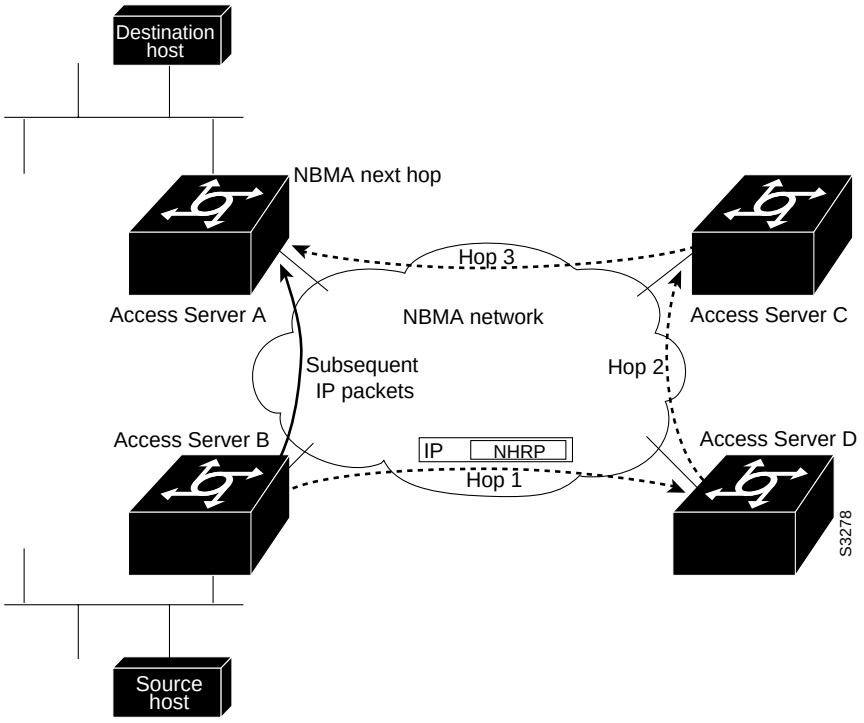


Figure 18-1 illustrates four communication servers connected to an NBMA network. Within the network are ATM or SMDS switches necessary for the communication servers to communicate with each other. Assume that the switches have virtual circuit connections represented by hops 1, 2, and 3 of the figure. When Communication Server A attempts to forward an IP packet from the source host to the destination host, NHRP is triggered. On behalf of the source host, Communication Server A sends an NHRP request packet encapsulated in an IP packet, which takes three hops across the network to reach Communication Server D, connected to the destination host. After receiving a positive NHRP reply, Communication Server D is determined to be the "NBMA next hop," and Communication Server A sends subsequent IP packets for the destination to Communication Server D in one hop.

With NHRP, once the NBMA next hop is determined, the source can either start sending IP packets to the destination (in a connectionless NBMA network such as SMDS) or first establish a 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 in use while NHRP is deployed. Hosts that can use only the LIS 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 uses a virtual private network, which is a virtual Layer 3 network that is built on top of an actual Layer 3 network. The topology you can use over the virtual private network can be largely independent of the underlying network, and the protocols you run over it can be completely independent of it.

Connected to the NBMA network are one or more Next Hop servers, which implement NHRP. All of our communication servers are capable of implementing NHRP and thus can act as Next Hop servers. A host or communication server that is not an NHRP speaker must be configured with the identity of the Next Hop server that serves it.

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 support 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 IP-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.

Modes of Operation

NHRP supports two modes of operation: server mode and fabric mode. The modes differ in the way the Next Hop server updates the destination address in the IP packet containing the NHRP Request.

Hosts attached directly to the NBMA network have no knowledge of whether NHRP is deployed in server or fabric mode and host configuration is the same in each case. Regardless of which mode is used, NHRP clients must be configured with the IP address and NBMA address of at least one Next Hop server. In practice, a host’s default communication server should also be its Next Hop server.

Fabric Mode

In fabric mode, it is expected that all communication servers within the NBMA network are NHRP-capable. A Next Hop server serving a destination must lie along the routed path to that destination. In practice, this means that all egress communication servers must double as Next Hop servers serving the destinations beyond them, and that hosts on the NBMA network are served by communication servers that double as Next Hop servers.

Server Mode

In server mode, few Next Hop servers exist in an NBMA network. This might occur in networks having communication servers that do not support NHRP or networks that have many directly attached hosts and relatively few communication servers.

Server mode requires static configuration of Next Hop server identity in the client stations (hosts or communication servers). The client station must be configured with the IP address of one or more Next Hop servers, and there must be a path to that Next Hop server (either directly, in which case the Next Hop server’s NBMA address must be known, or indirectly, through a communication server whose NBMA address is known). If there are multiple Next Hop servers, they must be configured with each others’ addresses, the identities of the destinations they each serve, and a logical NBMA network identifier. (This static configuration requirement, which might also involve authentication, tends to limit the number of Next Hop servers.)

If the NBMA network offers a group addressing or multicast feature, the client station can be configured with a group address assigned to the group of Next Hop servers. The client might then submit NHRP requests to the group address, eliciting a response from one or more Next Hop servers, depending on the response strategy selected.

The servers can also be configured with the group or multicast address of their peers, and a Next Hop server might use this address to forward NHRP requests that its peers cannot satisfy. This might elicit a response (to the Next Hop server) from one or more Next Hop servers, depending on the response strategy. The Next Hop server would then forward the NHRP reply to the NHRP request originator. The purpose of using group addressing or a similar multicast mechanism in this scenario is to eliminate the need to preconfigure each Next Hop server in a logical NBMA network with both the individual identities of other Next Hop servers and the destinations they serve. It reduces the number of Next Hop servers that might be used to process an NHRP request (in those configurations where Next Hop servers either respond or forward via the multicast, only two Next Hop servers would be traversed) and allows the Next Hop server that serves the NHRP request originator to cache next hop information associated with the reply.

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 (Server Mode)
- Configure NHRP Authentication
- Control NHRP Initiation
- 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
- Monitor and Maintain NHRP

For NHRP configuration examples, see the section "NHRP Configuration Examples" later in this chapter.

Enable NHRP on an Interface

To enable NHRP for an interface on a communication server, 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** *number*

For an example of enabling NHRP, see the section "Enabling NHRP Example" at the end of this chapter.

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).

Alternatively, the station should be configured with a means of acquiring those addresses, that is, the group address that can be used to reach the Next Hop servers.

A third possibility is that the Next Hop server(s) can be physically located on the stations's default or peer communication servers, so their IP addresses can be obtained from the station's IP 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 communication servers 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 communication server), perform the following task in interface configuration mode:

Task	Command
Configure static IP-to-NBMA address mapping.	**ip nhrp map** *ip-address nbma-address*

Statically Configure a Next Hop Server (Server Mode)

A Next Hop server is configured with its own identity, a set of IP address prefixes that correspond to the IP addresses of the stations it serves, a logical NBMA network identifier, and in the case of server mode, the identities of other Next Hop servers in the same logical NBMA network. If a served station is attached to several link layer networks, the Next Hop server might also need to be configured to advertise routing information to such stations.

If a Next Hop server acts as an egress communication server for stations connected to link layer networks other than the NBMA network, the Next Hop server must also be configured to exchange routing information between the NBMA network and these other link layer networks.

In all cases, routing information is exchanged using conventional intradomain or interdomain routing protocols.

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** *nhs-address* [*net-address* [*netmask*]]

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.

In the absence of static address configurations, a Next Hop server operates in fabric mode and must itself learn the NBMA addresses of the stations it serves dynamically through NHRP.

Configure NHRP Authentication

Configuring an authentication string ensures that only communication servers 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 communication servers that are 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** *string*

Control NHRP Initiation

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 can trigger NHRP requests. To limit which IP packets trigger NHRP requests, you must 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** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*]
Define an extended IP access list.	**access-list** *access-list-number* {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**tos** *tos*][**established**]

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** *access-list-number*

Suppress Forward and Reverse Record Options

To dynamically detect link-layer filtering in NBMA networks (for example, X.25 closed user group facility, or 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** *interface-type interface-number*

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 communication server tells other communication servers to keep the addresses it is providing in NHRP responses. The default length of time for each response is 7200 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** *seconds-positive* [*seconds-negative*]

Configure a GRE Tunnel for Multipoint Operation

You can enable a generic route 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** *key-number*

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

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**] [*interface-type interface-number*]
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 communication server 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**

Disable IP Routing

Every communication server ships with IP routing automatically enabled. If you choose to set up the communication server 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 communication server 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 communication server. To reenable IP routing, use the **ip routing** command.

Routing Assistance When IP Routing Is Disabled

The communication server software provides three methods by which the communication server can learn about routes to other networks when IP routing is disabled and the communication server is acting as an IP host:

- Proxy ARP
- A default gateway (also known as default communication server)
- The 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 communication server receives an ARP Request for a host that is not on the same network as the ARP Request sender, the communication server evaluates whether it has the best route to that host. If the communication server does have the best route, it sends an ARP Reply packet giving its own Ethernet hardware address. The host that sent the ARP Request then sends its packets to the communication server, 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 communication servers support it. Many other communication servers, especially host-based routing software, do not support it.

Default Gateway

Another method for locating routes is to define a default communication server (or gateway). The software sends all nonlocal packets to this communication server, which either routes them appropriately or sends an Internet Control Message Protocol (ICMP) redirect message back to the communication server, telling it of a better route. The ICMP redirect message indicates which local communication server 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 communication server has crashed or is unavailable, and no method of picking another communication server 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 (communication server).	**ip default-gateway** *ip-address*

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

Router Discovery Mechanism

The communication server software provides a third method, called *router discovery*, by which the communication server can dynamically learn about routes to other networks using the Gateway Discovery Protocol (GDP) or the ICMP Router Discovery Protocol (IRDP) for detecting routers. 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 communication servers from those updates. The server/client implementation of router discovery does not actually examine or store the full routing tables sent by communication servers, it merely keeps track of which systems are sending such data.

This mechanism supports the following protocols:

- Gateway Discovery Protocol (GDP)
- ICMP Router Discovery Protocol (IRDP)
- Routing Information Protocol (RIP)
- Interior Gateway Routing Protocol (IGRP)

You can configure these protocols in any combination. When possible, we recommend that you use GDP or IRDP because they allow each communication server to specify *both* a priority and the time after which a communication server should be assumed down if no further packets are received. Communication servers discovered using IGRP are assigned an arbitrary priority of 60.

Communication servers discovered through RIP are assigned a priority of 50. For IGRP and RIP, the software attempts to measure the time between updates and will assume that the communication server is down if no updates are received for 2.5 times that interval.

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

- A higher-priority communication server is discovered (the list of communication servers is polled at 5-minute intervals).
- The current default communication server is declared down.
- 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 communication server in an attempt to find a successful route to the destination.

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

Task	Command
Use the GDP protocol to configure communication server discovery.	**ip gdp gdp**

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

Task	Command
Use the IRDP protocol to configure communication server discovery.	**ip gdp irdp**

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

Task	Command
Use the RIP protocol to configure communication server discovery.	**ip gdp rip**

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

Task	Command
Use the IGRP protocol to configure communication server discovery.	**ip gdp igrp**

Configure a Routing Process

At this point in the configuration process, you can configure one or more of the many routing protocols 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.

Our communication servers support 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 on our communication server, can 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 communication server support for Internet broadcasts generally complies with RFC 919 and RFC 922; however, 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

See the "IP Configuration 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** [*access-list-number*]

Task	Command
Disable directed broadcast-to-physical broadcast translation on an interface.	**no ip directed-broadcast** [*access-list-number*]

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 communication server to forward certain classes of broadcasts to a helper address. You can have 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 *Router Products Command Reference* publication 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 communication server 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 communication server 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** *address*

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 [***port***] | nd | sdns}**

See the “IP Configuration Examples” section in this publication for an example of how to configure helper addresses.

Establish an IP Broadcast Address

The communication server supports IP broadcasts on both local- and wide-area networks. 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 communication servers can be configured to generate any form of IP broadcast address. Our communication servers also can receive and understand any form of IP broadcast.

To set the communication server's 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** [*ip-address*]

If the communication server does not have nonvolatile memory, and you need to specify the broadcast address to use before the communication server has been configured, you have to change the IP broadcast address by setting jumpers in the processor configuration register. Setting bit 10 causes the communication server 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 communication server to include the network and subnet portions of its address in the broadcast address. Table 18-2 shows the combined effect of setting bits 10 and 14.

Table 18-2 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 communication server platforms allow the configuration register to be set through the software; see the "Loading System Images and Configuration Files" chapter for details. For other communication server platforms, the configuration register can only be changed through hardware; see the appropriate hardware installation and maintenance manual for your system.

Configure IP Services

The IP suite offers a number of services that control and manage IP connections. Many of these services are provided by the Internet Control Message Protocol (ICMP). ICMP messages are sent by communication servers to hosts or other communication servers 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

See the "IP Configuration Examples" section at the end of this chapter for examples of ICMP services.

Disable ICMP Protocol Unreachable Messages

If the communication server 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 communication server 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 unreachables**

Disable ICMP Redirect Messages

Routes sometimes can become less than optimal. For example, it is possible for the communication server to be forced to resend a packet through the same interface on which it was received. If this happens, the communication server sends an ICMP Redirect message to the packet's originator telling it that it is on a subnet directly connected to the communication server, and that it must forward the packet to another system on the same subnet. It does so because the originating host presumably could have sent that packet to the next hop without involving the communication server at all. The Redirect message instructs the sender to remove the communication server from the route and substitute a specified device representing a more direct path. This feature is enabled by default.

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

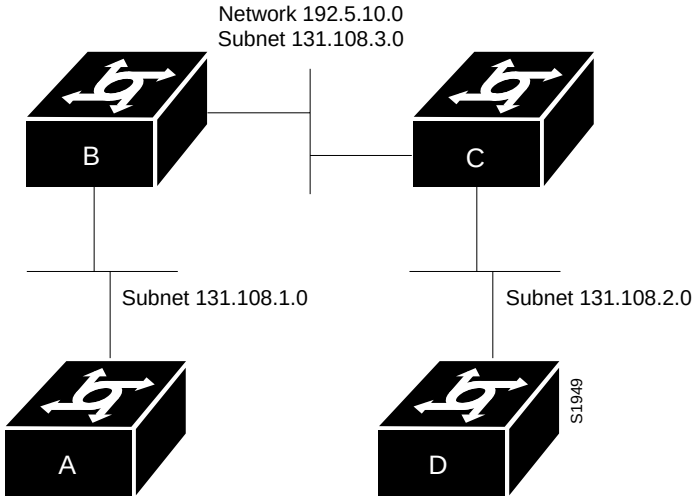
Task	Command
Disable the sending of ICMP Redirect messages to learn routes.	**no ip redirects**

Understand Path MTU Discovery

Our communication servers support 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 communication server 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 communication server 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 18-2.

Figure 18-2 IP Path MTU Discovery

IP Path MTU Discovery is useful when a link in a network goes down, forcing use of another, different MTU-sized link (and different communication servers). Figure 18-2 shows an attempt to send IP packets over a network where the MTU in the first communication server is set to 1500 bytes, but then reaches a communication server where the MTU is set to 512 bytes. If the datagram's "Don't fragment" bit is set, the datagram would be dropped because the 512-byte communication server is unable to forward it. All packets larger than 512 bytes will be dropped in this case. The



second communication server 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 needs to send.

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

Because the CTR card does not support the switching of frames larger than 4472 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.

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 a communication server's interface, the communication server 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 need to 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 communication server 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 communication server 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 communication server 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 allowing 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 communication server 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 communication server to discard any IP datagram containing a source-route option.	**no ip source-route**

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 communication server interfaces, we provide *access lists*.

You can use access lists in several 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 access lists and how to apply them.

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

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

Caution This release introduces substantial changes to IP access lists. These extensions are backward compatible; migrating from existing releases to the 10.3 Release 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 10.3 Release 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 beta IOS Release 10.3 images.

The software supports two 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, as well as optional protocol type information for finer granularity of control.

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** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*]
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** *access-list-number* {**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.	**access-list** *access-list-number* {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**tos** *tos*][**established**]
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** *access-list-number* {**deny** | **permit**} *protocol* **any**
Define an extended IP access list using an abbreviation for a source and source wildcard of *source* 0.0.0.0 and an abbreviation for a destination and destination wildcard of *destination* 0.0.0.0.	**access-list** *access-list-number* {**deny** | **permit**} *protocol* **host** *source* **host** *destination*

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 Keep in mind when making the standard and extended access list 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.

Refer to the "IP Configuration Examples" section at the end of this chapter for examples of implicit masks.

Apply an Access List to an Interface or Terminal Line

After an access list is created, you can apply it to one or more interfaces. Access lists can be applied on *either* outbound or inbound interfaces. The next two tables show how this task is accomplished for both terminal lines and network interfaces.

Perform the following task in line configuration mode:

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** *access-list-number* {**in** | **out**}

Perform the following task in interface configuration mode:

Task	Command
Control access to an interface.	**ip access-group** *access-list-number* {**in** | **out**}

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

When you apply an access list (standard or extended) that has not yet been defined to an interface, the communication server 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.

Note Set identical restrictions on all the virtual terminal lines, because a user can attempt to connect to any of them.

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 our communication server only accepts and generates a four-byte IPSO. IPSO is generally used to comply with the U.S. Government's DoD 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** *level authority* [*authority...*]
or	
Set an interface to the requested IPSO range of classifications and authorities.	**ip security multilevel** *level1* [*authority1...*] **to** *level2 authority2* [*authority2...*]

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** [*level authority* [*authority...*]]
Accept packets on an interface that has an extended security option present.	**ip security extended-allowed**
Ensure that all packets leaving the communication server on an interface contain a basic security option.	**ip security add**

Task	Command
Remove any basic security option that might be present on a packet leaving the communication server 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**

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 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 *off* if the interface is "unclassified Genser"; otherwise it is *on.*

Table 18-3 provides a list of all default values.

Table 18-3 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 communication server 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.

Configure Global Default Settings

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

Task	Command
Configure system-wide default settings.	**ip security eso-info** *source compartment-size default-bit*

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** *source compartment-bits*
Set the maximum sensitivity level for an interface.	**ip security eso-max** *source compartment-bits*

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** *source compartment-bits*

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** *ip-address*

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 task in global configuration mode:

Task	Command
Specify the primary address for the audit trail.	**dnsix-nat primary** *ip-address*
Specify the secondary address for the audit trail.	**dnsix-nat secondary** *ip-address*
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** *ip-address*

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** *count*
Specify the number of transmit retries for DMDP.	**dnsix-dmdp retries** *count*

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 communication server on a source and destination IP address basis. Only transit IP traffic is measured and only on an outbound basis; traffic generated by the communication server or terminating in the communication server is not included in the accounting statistics. To maintain accurate accounting totals, the communication server software maintains two accounting databases: an active and a checkpointed database.

Our IP accounting support provides information to identify 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**
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** *threshold*
Filter accounting information for hosts.	**ip accounting-list** *ip-address mask*
Control the number of transit records that will be stored in the IP accounting database.	**ip accounting-transits** *count*

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 Fast 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 communication server 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** *number*

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 communication server will wait to attempt to establish a TCP connection. In previous versions of communication server software, the system would wait a fixed 30 seconds when attempting to do so. 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. Your ability to make a Telnet connection over the link (from the communication server) will be affected 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 communication server, just to traffic originated at the communication server.

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

Task	Command
Set the amount of time the communication server will wait to attempt to establish a TCP connection.	**ip tcp synwait-time** *seconds*

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 communication servers generally offer better packet transfer performance when fast switching is enabled.

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

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

Control Route Cache Invalidation

The high-speed route cache used by IP fast 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** [*minimum maximum quiet threshold*]

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 wide-area networking chapters.

Monitor and Maintain the IP Network

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

- Clear Caches, Tables, and Databases
- Specify the Format of Network Masks
- Display System and Network Statistics

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 are suspected to be invalid.

The following table lists the tasks associated with clearing caches, tables, and databases. All are performed in EXEC mode.

Task	Command
Remove one or all entries from the host name and address cache.	**clear host** {*name* | ***}**
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** {*network* **[***mask***]** | ***}**

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 network mask. By default, **show** commands display an IP address and then its network mask 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 above 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 network mask to the address itself. The above example would be displayed as 131.108.11.55/24.

To specify the format in which network masks 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 network masks 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 communication server statistics such as the contents of IP routing tables, caches, and databases. Information provided can be used to determine resource utilization and solve network problems. You also can display information about node reachability and discover the routing path that your communication server's packets are taking through the network.

These tasks are summarized in the table that follows. See the "IP Commands" chapter in the *Access and Communication Servers Command Reference* for details about the commands listed in these tasks. Perform the following tasks in privileged EXEC mode:

Task	Command
Display the contents of all current access lists.	**show access-lists**
Display the entries in the ARP table for the communication server.	**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** [*access-list-number*]
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** [*prefix mask*] [*interface-type interface-number*]
Display the usability status of interfaces.	**show ip interface** [*type number*]
Display the masks used for network addresses and the number of subnets using each mask.	**show ip masks** *address*
Display the address of a default gateway.	**show ip redirects**
Display the current state of the routing table.	**show ip route** [*address* [*mask*]] | [*protocol*]
Display the current state of the routing table in summary form.	**show ip route summary**
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**
Test network node reachability (privileged).	**ping** [*protocol*] {*host* | *address*}
Test network node reachability using a simple ping facility (user).	**ping** [*protocol*] {*host* | *address*}
Trace packet routes through the network (privileged).	**trace** [*destination*]
Trace packet routes through the network (user).	**trace ip** *destination*

IP Configuration Examples

The following sections provide IP configuration examples:

- Serial Interfaces Configuration Example
- Creating a Network from Separated Subnets Example
- Dynamic Lookup Example
- Establishing IP Domains Example
- Configuring HP Hosts on a Network Segment Example
- Helper Addresses Example
- Broadcasting Example
- Customizing ICMP Services Example
- Access List Examples
- IPSO Configuration Examples
- Ping Command Example

Serial Interfaces Configuration Example

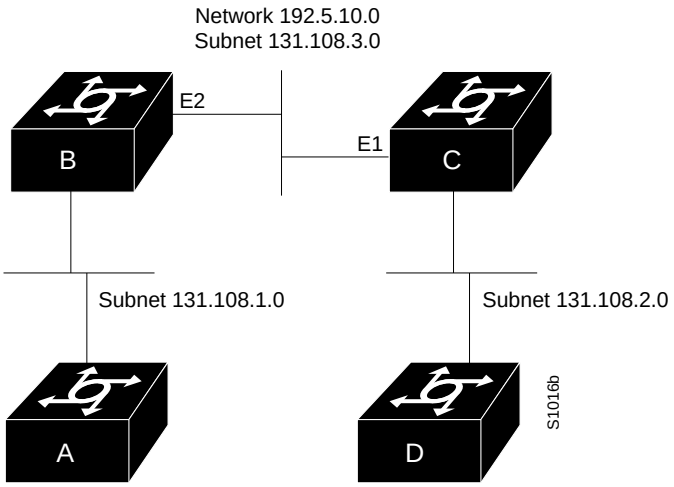
In the following example, the second serial interface (serial 1) is given Ethernet 0's address. The serial interface is unnumbered.

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

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 18-3. The two networks are brought into the same logical network through the use of secondary addresses.

Figure 18-3 Creating a Network from Separated Subnets



The following examples show the configurations for communication servers B and C.

Configuration for Communication Server 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 Communication Server 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
```

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 for the communication server. 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 communication server uses to complete
! unqualified host names
ip domain-name cisco.com
```

Establishing IP Domains Example

The following example establishes a domain list with several alternate domain names:

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

Configuring HP Hosts on a Network Segment Example

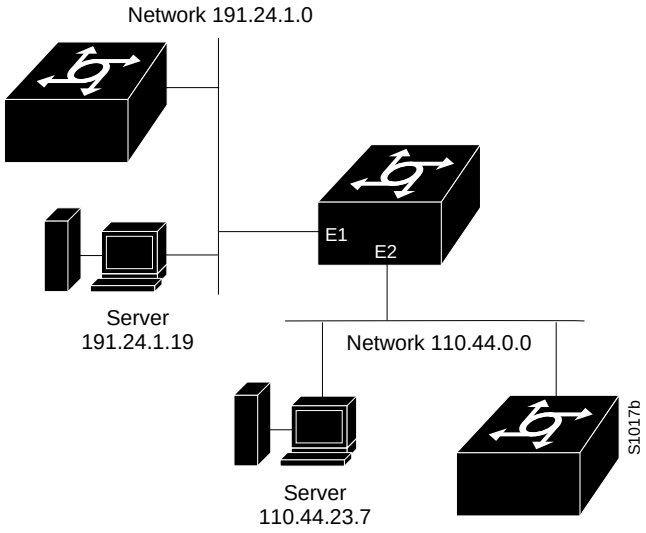
The following example has a network segment with Hewlett-Packard devices on it. The commands listed customize the communication server's first Ethernet port to respond to Probe name requests for bl4zip and to use Probe as well as ARP.

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

Helper Addresses Example

In the following example, one communication server 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 18-7 illustrates how to configure the communication server that connects network 110 to network 191.24.1.

Figure 18-4 IP Helper Addresses



The following example shows the configuration:

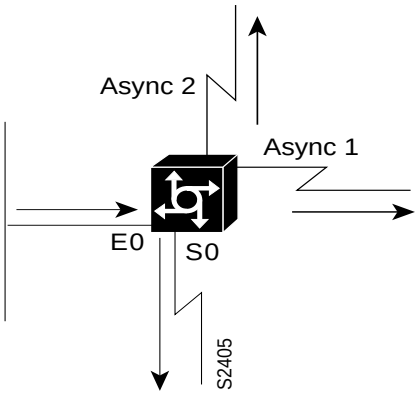
```
ip forward-protocol udp
!
interface ethernet 1
ip helper-address 110.44.23.7
interface ethernet 2
ip helper-address 191.24.1.19
```

Broadcasting Example

Our communication servers support 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.

Figure 18-8 shows a flooded broadcast packet being sent to every network. The packet that is incoming from interface E0 is flooded to interfaces Async 1, Async 2 and S0 (Serial 0).

Figure 18-5 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.

Customizing 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 communication servers 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 communication servers 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 communication server would be unlikely to use anyway.

```
interface ethernet 0
no ip unreachables
no ip redirects
```

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 communication server would accept one address on subnet 48 and reject all others on that subnet. The last line of the list shows that the communication server 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
```

Examples of Implicit Masks in Access Lists

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
```

Examples of Configuring Extended Access Lists

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 communication server 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.

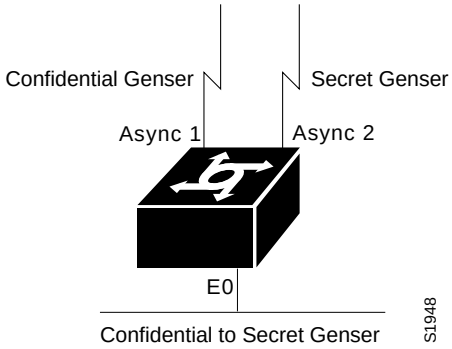
In the following example, the Ethernet network is a Class B network with the address 128.88.0.0, and the mail host's address is 128.88.1.2. The keyword **established** is used only for the TCP protocol to indicate an established connection. A match occurs if the TCP datagram has the ACK or RST bits set,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

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 18-6.

Figure 18-6 IPSO Security Levels



The following commands set up interfaces for the configuration in Figure 18-6:

```
interface async 1
ip security dedicated confidential genser
interface async 2
ip security dedicated secret genser
interface ethernet 0
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 an interface. 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 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 async 1
ip security dedicated secret genser
ip security extended-allowed
!
```

```
interface async 2
ip security multilevel confidential genser to secret genser
ip security extended-allowed
ip security first
```

Ping Command Example

You can specify the communication server 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
```

NHRP Configuration Examples

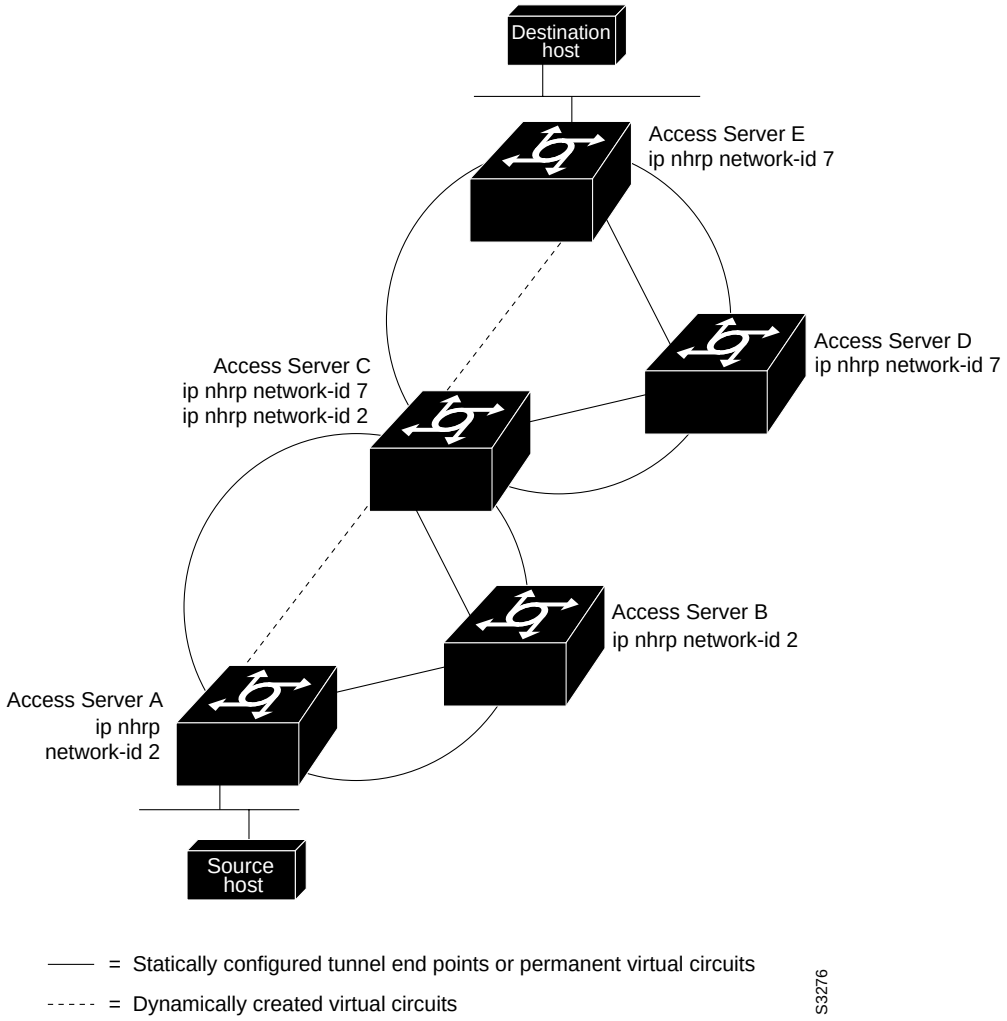
This section provides the following NHRP configuration examples:

- Enabling NHRP Example
- NHRP on a Multipoint Tunnel Example

Enabling NHRP Example

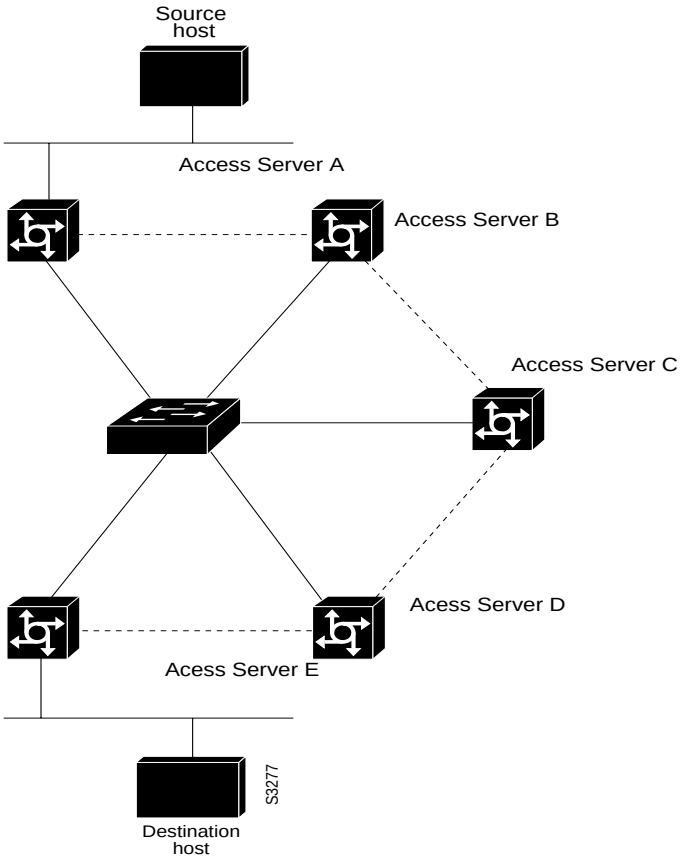
A logical NBMA network is considered to be the group of interfaces and hosts that participate in NHRP and have the same network identifier. Figure 18-2 illustrates two logical NBMA networks (shown as circles) configured over a single physical NBMA network. Communication server A can communicate with communication server B and C because they share the same network identifier (2). Communication server C can also communicate with communication server D and E, as they share network identifier 7. After address resolution is complete, communication server A can send IP packets to communication server C in one hop, and communication server C can send them to communication server E in one hop, as shown by the dotted lines.

Figure 18-7 Two Logical NBMA Networks over One Physical NBMA Network



The physical configuration of the five communication servers in Figure 18-2 might be that shown in Figure 18-3. The source host is connected to communication server A and the destination host is connected to communication server E. The same switch serves all five communication servers, making one physical NBMA network.

Figure 18-8 Physical Configuration of a Sample NBMA



Refer again to Figure 18-2. Initially, before NHRP has resolved any NBMA addresses, IP packets from the source host to the destination host travel through all five communication servers connected to the switch before reaching the destination. When communication server A first forwards the IP packet toward the destination host, communication server A also generates an NHRP request for the destination host's IP address. The request is forwarded to communication server C, whereupon a reply is generated. Communication server C replies because it is the egress communication server between the two logical NBMA networks.

Similarly, communication server C generates an NHRP request of its own, which communication server E replies to. In this example, subsequent IP traffic between the source and the destination still requires two hops to traverse the NBMA network, because 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 on a Multipoint Tunnel Example

With multipoint tunnels, a single tunnel interface may be connected to multiple neighboring communication servers. Unlike point-to-point tunnels, a tunnel destination does not need to be configured. In fact, if configured, the tunnel destination must correspond to an IP multicast address. Broadcast or multicast packets sent over the tunnel interface can be then 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 generic route encapsulation (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, communication serverrs 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, Communication Server A knows how to reach communication server B, communication server B knows how to reach communication server C, communication server C know how to reach communication server D, and communication server D knows how to reach communication server A.

When communication server A initially attempts to send an IP packet to communication server D, the packet is forwarded through communication server B and C. Through NHRP, the communication servers 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 communication servers can directly communicate over the tunnel network without their IP traffic requiring an intermediate hop.

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

Communication Server 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
```

Communication Server 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
```

Communication Server 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
```

Communication Server 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
```