CHAPTER 16

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 Internet protocols, collectively referred to as the Internet Protocol suite, are built. IP is a network-layer protocol that contains addressing information and some 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 *Communication Server Command Reference* publication. For information on configuring the various IP routing protocols, refer to the IP routing protocols chapter of this manual. For historical background and a technical overview of IP and other protocols, see the *Internetworking Technology Overview* publication.

Cisco's Implementation of IP

Cisco's implementation of TCP/IP provides most of the major services contained in the various protocol specifications. Cisco communication servers also provide the TCP and 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 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.

The following list summarizes these tasks and features:

- Assign IP Addresses to Network Interfaces
- Configure IP Addressing Options
- Enable IP Routing

- Configure a Routing Process
- Configure Broadcast Packet Handling
- Configure IP Services
- Filter IP Packets
- Configure IP Security Options
- Configure IP Accounting
- Configure Performance Parameters
- Configure IP over WANs
- Monitor and Maintain the IP Network

Each task is described in a following section. Remember that not all of these tasks are required. The tasks you perform will depend on your network and your needs.

At the end of this chapter, the section "IP Configuration Examples" illustrates 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.

To assign an Internet 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.

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 may 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 be easily 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**

See the "IP Configuration Examples" section at the end of the chapter for an example of creating a network from separated subnets.

Allowable Internet Addresses

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

Table 1-1 Reserved and Available Internet 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 Internet addresses is found in RFC 1166, "Internet Numbers."

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

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.

We do provide a way to 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:

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

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

Note Using an unnumbered serial line between different major networks requires special care. If at each end of the link there are different majornets assigned to the interfaces you specified as unnumbered, any routing protocols running across the serial line cannot 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** *interface-name*

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

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

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

Our IP implementation allows you to control interface-specific handling of IP addresses by facilitating address resolution, name services, and other functions. The following tasks are described in this section:

- Establish address resolution
- Map logical names to IP addresses
- Configure HP Probe Proxy name requests

The following sections provide additional information and describe these tasks.

Establish Address Resolution

A device in the Internet 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 Internet address is called *address resolution.* The process of determining the Internet 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 Internet 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 Stystem Images, Microcode Imasges, 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 cache, as necessary
- Set ARP encapsulations
- 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 Internet address with a media (hardware) address in the ARP cache.	**arp** *internet-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** *internet-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**}

Set 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 either disable or reenable proxy ARP, perform the following tasks in interface configuration mode, as necessary, for your network:

Task	Command
Disable proxy ARP on the interface.	**no ip proxy-arp**
Enable proxy ARP on the interface.	**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 Internet. This is a hierarchical naming scheme that provides for *domains*. Domain names are pieced together with periods (.) as the delimiting character. For example, Cisco Systems is a commercial organization that the Internet identifies by a *com* domain name, so that its domain name is *cisco.com*. A specific device in this domain, the 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
- Enable 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 a *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** *name* [*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 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*]

Enable the DNS

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

To enable or disable the DNS, perform the following tasks in global configuration mode:

Task	Command
Enable DNS-based host name-to-address translation.	**ip domain-lookup**
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.

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.

Enable IP Routing

Every communication server ships with IP routing automatically enabled. You may also at some later time want to reenable IP routing. To accomplish either of these goals, perform the following tasks in global configuration mode:

Task	Command
Enable IP routing.	**ip routing**
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.

Enable IP Host Routing

You can configure your communication server not to run routing protocols, but to use host routing methods to send packets to devices and networks that are not local. This command is used when the communication server is being used as a terminal server. To configure your communication server for host routing, perform the following task:

Enable IP host routing	**ip host-routing**

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 (Address Resolution Protocol)
- 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 discover 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.

Proxy ARP works as long as other communication servers support it. While this feature is supported on our communication servers, many other communication servers, especially host-based routing software, do not support it. This feature is enabled by default in our communication servers.

A 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 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 limitation of this method is 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 is possible 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** *address*

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

The Router Discovery Mechanism

The communication server software provides a third method, called *communication server discovery*, by which the communication server can dynamically learn about routes to other networks using the Gateway Discovery Protocol (GDP) for detecting communication servers or the ICMP Router Discovery Protocol (IRDP). The software is also capable of wiretapping RIP and IGRP routing updates and inferring the location of communication servers from those updates. The server/client implementation of communication server 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 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. Routers discovered using IGRP are assigned an arbitrary priority of 60. Routers 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:

- When a higher-priority communication server is discovered. The list of communication servers is polled at 5-minute intervals.
- When the current default communication server is declared down.

- When a TCP connection is about to time out because of excessive retransmissions. In this case, the server flushes the ARP cache and the ICMP redirect cache and picks a new default communication server in an attempt to find a successful route to the destination.

Detect Routers Using the IRDP Protocol

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**

Use the **no ip gdp irdp** command to turn the communication server discovery feature off.

Detect Routers Using the RIP Protocol

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**

Use the **no ip gdp rip** command to turn the communication server discovery feature off.

Detect Routers Using the IGRP Protocol

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**

Use the **no ip gdp igrp** command to turn the communication server discovery feature off.

Configure a Routing Process

At this point in the configuration process, you may choose to configure one or more of the many routing protocols that are available based on your individual network needs. Routing protocols provide topology information of an internetwork. Refer to the "Configuring 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 Internet 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 following tasks to enable these schemes:

- Enable directed broadcast to physical broadcast translation
- Forward UDP broadcast packets and protocols
- Establish an Internet broadcast address
- Flood IP broadcasts

Each task is described in a following section. 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 the following task. By default, this feature is enabled only for those protocols configured using the **ip forward-protocol** global configuration command. An access list may be specified 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 these 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*]
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 employ User Datagram Protocol (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.

A UDP destination port can be specified to control which UDP services are forwarded. Multiple UDP protocols can be specified. You also can specify the Network Disk (ND) protocol, which is used by older diskless Sun workstations. By default, both UDP and ND forwarding are enabled if a helper address has been defined for an interface. The *Communication Server Command Reference* publication lists the services that are forwarded by default if you do not specify any ports.

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

See the "IP Configuration Examples" section at the end of this chapter for an example of how to configure helper addresses.

Establish an Internet Broadcast Address

The communication server supports Internet broadcasts on both local- and wide-area networks. There are several ways to indicate an Internet 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 Internet broadcast address. Our communication servers also can receive and understand any form of Internet broadcast.

To set the communication server's Internet 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** [*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 Internet 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 1-2 shows the combined effect of setting bits 10 and 14.

Table 1-2 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, Microcode 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 guide for your communication server.

Configure IP Services

The Internet protocol 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 IP header. For detailed information on ICMP, see RFC 792.

This section summarizes the following tasks for configuring IP services:

- Generate ICMP Protocol Unreachable messages
- Generate ICMP Redirect messages
- Understand path MTU discovery
- Set the MTU packet size
- Enable ICMP Mask Reply messages

- Configure IP source-route header options
- Configure simplex Ethernet interfaces

Each task is described in a following section.

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

Generate 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 enable this service by performing the following task in interface configuration mode:

Task	Command
Enable the sending of ICMP Protocol Unreachable and Host Unreachable messages.	**ip unreachables**

To disable the sending of ICMP protocol unreachable messages, use the **no ip unreachables** command.

Generate 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.

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

Task	Command
Enable the sending of ICMP Redirect messages to learn routes.	**ip redirects**

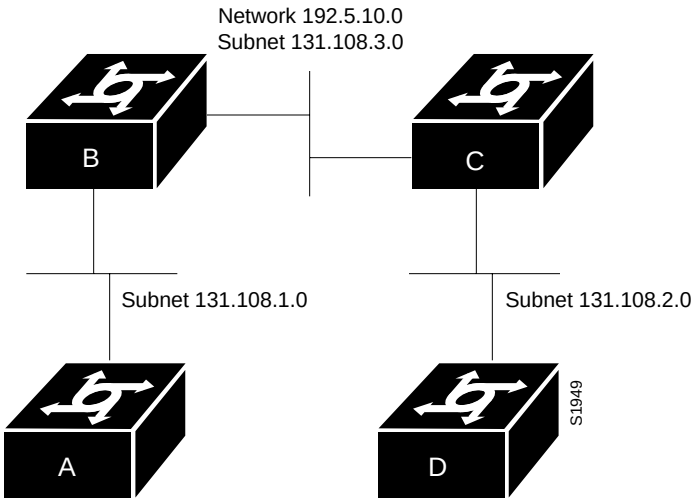
To disable the sending of ICMP Redirect messages, use the **no ip redirects** command.

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 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 destined for the receiving interface so that they fit the smallest packet size of all the links along the path. This technique is shown in Figure 1-1.

Figure 1-1 Path MTU Discovery



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). As shown in Figure 1-1, suppose one is trying 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 Path MTU Discovery, it also would include the MTU of the next-hop network link in the low-order bits of an unused header field.

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

Set the MTU Packet Size

All interfaces have a default maximum transmission unit (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**

Configure IP Source-Route Header Options

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 wish to force a packet to take a certain route through the network. The default is to perform source routing.

To configure IP source-route header options or to disable this feature, perform the following tasks in global configuration mode:

Task	Command
Handle IP packets with source routing information.	**ip source-route**
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.

Create an Access List

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:

Step 1 Create an access list by specifying an access list number and access conditions.

Step 2 Apply the access list to interfaces.

These steps are described in the next sections.

Create Standard and Extended Access Lists

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 either a standard or an extended access list, perform one of the following tasks in global configuration mode:

Task	Command
Define an IP access list number and the access conditions.	**access-list** *access-list-number* {**permit** | **deny**} *source source-mask*
Define an extended IP access list number and the access conditions.	**access-list** *access-list-number* {**permit** | **deny**} *protocol source source-mask destination destination-mask* [*operator operand*] [**established**]

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 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, 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

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 following 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 to an interface an access list (standard or extended) that has not yet been defined, 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 connect to any of them.

Configure 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 used generally to comply with U.S. DoD security policy.

IPSO 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
- Accepts or rejects messages with extended security options

Two general steps are required to configure IPSO-level security:

Step 1 Enable IPSO and set the security classifications.

Step 2 Select the security levels.

See the following sections for descriptions of each task. See the *Communication Server Command Reference* publication for more specific details on the security options. See the "IP Configuration Examples" section at the end of this chapter for an example of configuring IPSO options.

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.

Select the Security Levels

To select the various IPSO security levels, 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**
Remove any basic security option that may 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**

Default Values for Minor Keywords

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

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

Table 1-3 provides a list of all default values.

Table 1-3 Default Security Keyword Values

Interface Type	Level	Authority	Implicit Labelling	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 may 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 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.

You enable IP accounting on a per-interface basis. The following two tables list the tasks to configure IP accounting functions.

Perform the following task in interface configuration mode:

Task	Command
Enable IP accounting.	**ip accounting**

Perform the following tasks in global configuration mode:

Task	Command
Enable IP accounting for transit traffic outbound on an interface.	**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*

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

You can tune IP performance in several ways using the following methods:

- Compress TCP packet headers
- Set the TCP connection-attempt time

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 only supported on serial lines using HDLC 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 (with the **passive** keyword), 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**]
or	
Specify the total number of header compression connections that can exist on an interface.	**ip tcp compression-connections** *number*

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 dialup asynchronous connections, such as a network consisting of Dial-on-Demand links that are implemented over modems, because it will affect your ability to Telnet over the link (from the communication server) if the link must be brought up.

Because this 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*

Configure IP over WANs

You can configure IP over X.25, SMDS, Frame Relay, and DDR networks. To do this, configure the appropriate address mappings.

Monitor and Maintain the IP Network

You can perform the following tasks to monitor and maintain your network:

- Clear caches, tables, and databases
- Display system and network statistics

Each task is described in a following section.

Clear Caches, Tables, and Databases

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

The following table lists the tasks associated with clearing caches, tables, and databases. 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*] | ***}**

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 *Communication Server Command Reference* for details about the commands listed in these tasks. Perform the following tasks in 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 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 active IP accounting or checkpointed database.	**show ip accounting** [**checkpoint**]
Display Internet addresses mapped to TCP ports (aliases).	**show ip aliases**
Display the IP ARP cache.	**show ip arp**
Display the usability status of interfaces.	**show ip interface** [*interface unit*]
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* | **summary**] [*process-id*]]
Show statistics on TCP header compression.	**show ip tcp header-compression**
Display IP protocol statistics.	**show ip traffic**
Test network node reachability (privileged).	**ping** [*protocol*] {*host* | *address*}
Test network node reachability using a simple ping facility (unprivileged).	**ping** [*protocol*] {*host* | *address*}
Trace packet routes through the network (privileged).	**trace** [*destination*]
Trace packet routes through the network (unprivileged).	**trace ip** *destination*

IP Configuration Examples

This section shows complete configuration examples for the most common configuration situations. The configuration examples provided include the following topics:

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

Serial Interfaces Configuration Example

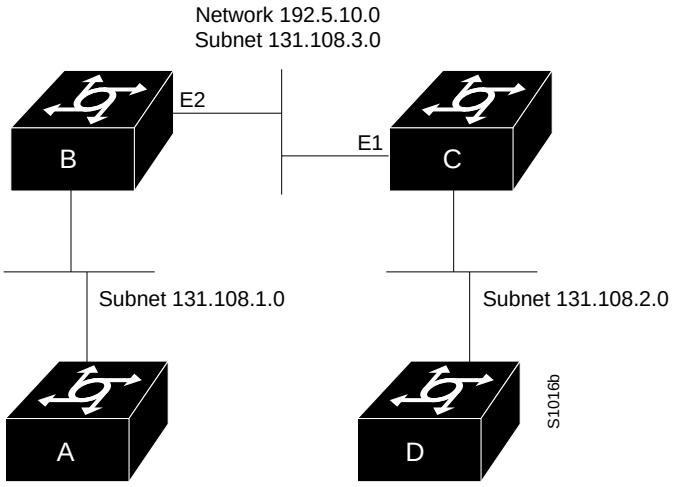
In the following example, the second serial interface (serial 1) is given ethernet 0's address. The serial interface is unnumbered.

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

Example of Creating a Network from Separated Subnets

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

Figure 1-2 Creating a Network from Separated Subnets



The following is Router B's configuration:

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

The following is Router C's configuration:

```
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
```

Example of Establishing IP Domains

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

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

Example of Configuring HP Hosts on a Network Segment

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
```

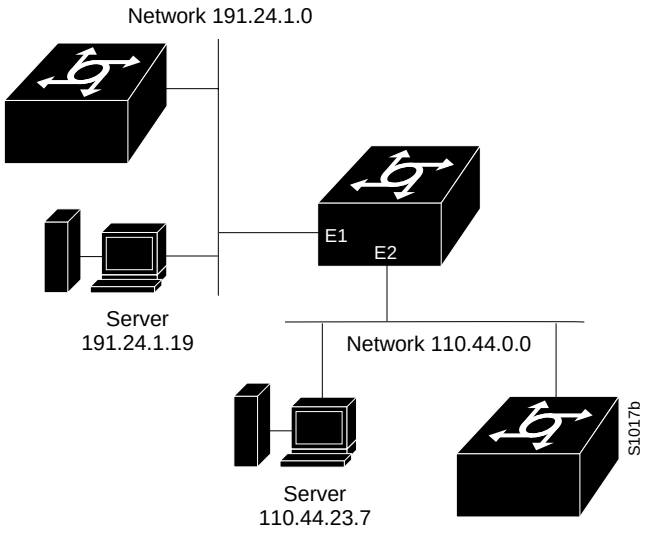
Helper Addresses Example

In this 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 1-3 illustrates how to configure the communication server that connects network 110 to network 191.24.1.

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

Figure 1-3 IP Helper Addresses



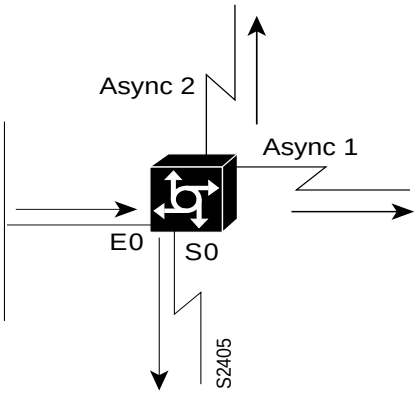
Broadcasting Examples

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. The following examples show configurations for both types of broadcasting.

Flooded Broadcast Example

Figure 1-4 shows a flooded broadcast packet being sent to every network. The packet that is incoming from interface E0 is flooded to interfaces Async 0, Async 1, and Serial 0.

Figure 1-4 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. 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 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 this 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 this example, the first line permits any incoming TCP connections with destination port 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 Internet 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.

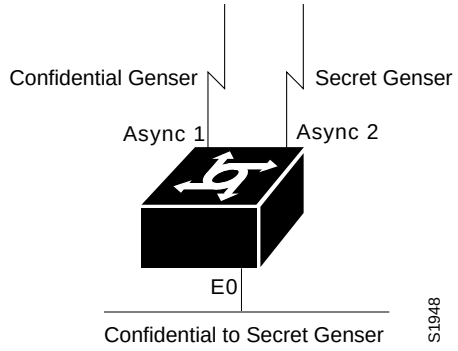
In the following example, the Ethernet network is a Class B network with the address 128.88.0.0, and the mail host's address is 128.88.1.2. The keyword **established** is used only for the TCP protocol to indicate an established connection. A match occurs if the TCP datagram has the ACK or RST bits set,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

IPSO Configuration Examples

In this example, three interfaces are presented. These interfaces are running at security levels of Confidential Genser, Secret Genser, and Confidential to Secret Genser, as shown in Figure 1-5.

Figure 1-5 IPSO Security Levels



The following commands set up interfaces for the configuration in Figure 1-5:

```
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 this next example, there are devices on Ethernet 0 that cannot generate a security option, and so must accept packets without a security option. These hosts do not understand security options; therefore, never place one on such interfaces. Furthermore, there are hosts on the other two networks that are using the extended security option to communicate information, so you must allow these to pass through the system. Finally, there also is a host (a Blacker Front End) 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
```