Chapter 14

Internet Protocol (IP)

Introduction

This chapter describes the main features of the Internet Protocol (IP), support for IP on the router, and how to configure and operate the router to route IP protocols.

IP protocols are widely used and available on nearly every host and PC system. They provide a range of services including remote login, file transfer, and Email. Using IP routers allows these services to be fully supported within an organisation and to other organisations internationally.

IP is often referred to as TCP/IP. The letters TCP refer to Transmission Control Protocol. This is a protocol that runs over IP and provides end-to-end reliability and control of IP network connections. A closely related protocol called UDP (User Datagram Protocol) also runs over IP and is used where reliable transport of datagrams is not required. Both TCP and UDP are used by modules in the router. TCP implements Telnet remote logins, while UDP downloads software.

The router is capable of routing IP data packets via the wide area network. This allows a group of remote LANs to be joined together as a single IP autonomous system and to be connected to other IP networks such as the Internet.

This chapter describes IPv4. For information on IPv6 and the router’s implementation of it, see Chapter 15, Internet Protocol Version 6 (IPv6).

Some interface and port types mentioned in this chapter may not be supported on your router. The interface and port types that are available vary depending on your product's model, and whether an expansion unit (PIC, NSM) is installed. For more information, see the Hardware Reference.

The Internet

The Internet (with a capital “I”) is the name given to the large, worldwide network of networks based on the original concepts of the ARPAnet. A large number of government, academic and commercial organisations are connected to the Internet, and use it to exchange traffic such as email. The Internet uses the TCP/IP protocols for all routing. In recent times the term internet (with a lowercase “i”) has also come to refer to any network (usually a wide area network) that uses the Internet Protocol. The remainder of this chapter concentrates on the latter definition, i.e. that of a generalised network that uses IP as the transport protocol.

The basic unit of data sent through an internet is a *packet* or *datagram*. An IP network functions by moving packets between routers and/or hosts. A packet consists of a *header* followed by the *data* (Figure 14-1 on page 14-6, Table 14-1 on page 14-7). The header contains the information necessary to move the packet across the internet. It must be able to cope with missing and duplicated packets as well as possible fragmentation (and reassembly) of the original packet.

Packets are sent using a *connectionless* transport mechanism. A connection is not maintained between the source and destination addresses; rather, the destination address is placed in the header and the packet is transmitted on a best effort basis. It is up to the intermediate systems (routers and gateways) to deliver the packet to the correct address, using the information in the header. Successive packets may take different routes through the network to the destination. There is a close analogy with the postal delivery system in that letters are placed in individually addressed envelopes and put into the system in the ‘hope’ that they will arrive. Like an internet, the postal system is very reliable. In an internet, higher layers (such as TCP and Telnet) are responsible for ensuring that packets are delivered in a reliable and sequenced way.

In contrast to a connectionless transport mechanism, a *connection-oriented* transport mechanism requires a connection to be maintained between the source and destination as long as necessary to complete the exchange of packets between source and destination. X.25 is an example of a connection-oriented protocol. A good analogy to a connection-oriented protocol is a telephone call in which both parties verify that they are talking to the correct person before exchanging highly sequenced data (because nothing intelligible results when both talk at the same time), and the connection is maintained until both parties have finished talking. It is not hard to imagine the chaos if the telephone system delivered words in the wrong order.

Figure 14-1: Format of an IP datagram

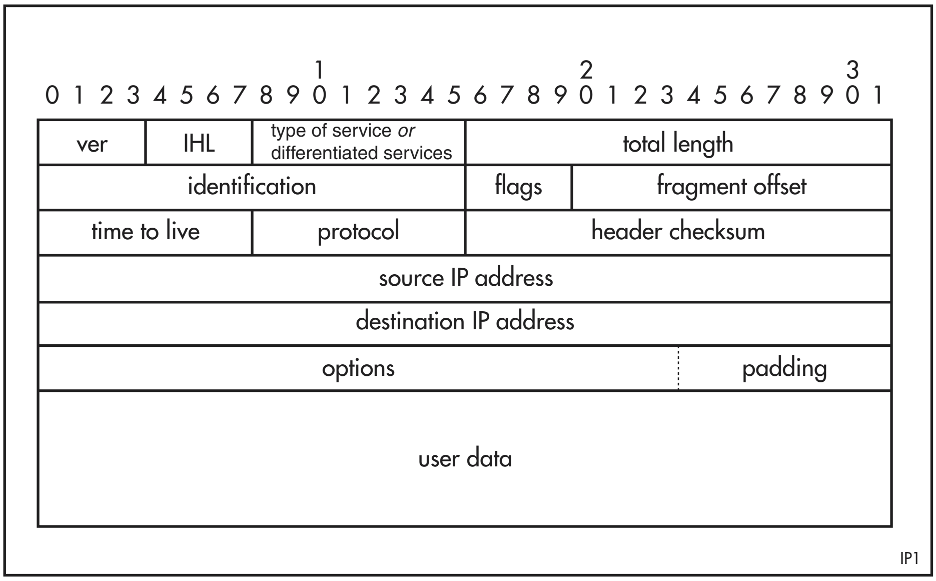


Table 14-1: Functions of the fields in an IP datagram

Field	Function
ver	Version of the IP protocol that created the datagram.
IHL	Length of the IP header in 32-bit words (5 is minimum value).
Type of service *or* differentiated services	Type of Service indicates the quality of service (precedence, delay, throughput, and reliability) desired for the datagram. Differentiated Services supersedes this, and contains the 6-bit DSCP and is used to sort traffic as part of a Quality of Service system. For more information, see RFC 2474, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*.
Total length	Length of the datagram (both header and user data), in octets.
Identification	16-bit value assigned by the originator of the datagram, used during reassembly.
Flags	Control bits indicating whether the datagram may be fragmented, and if so, whether other later fragments exist.
Fragment offset	For fragmented datagrams, offset in the original datagram of the data being carried in this datagram.
Time to live	Time in seconds the datagram is allowed to remain in the internet system.
Protocol	High level protocol used to create the message (analogous to the type field in an Ethernet packet).
Header checksum	Checksum of the header.
Source IP address	32-bit IP address of the sender.
Destination IP address	32-bit IP address of the recipient.
Options	Optional field primarily used for network testing or debugging.
Padding	All bits set to zero —used to pad the datagram header to a length that is a multiple of 32 bits.
User data	Actual data being sent.

Addressing

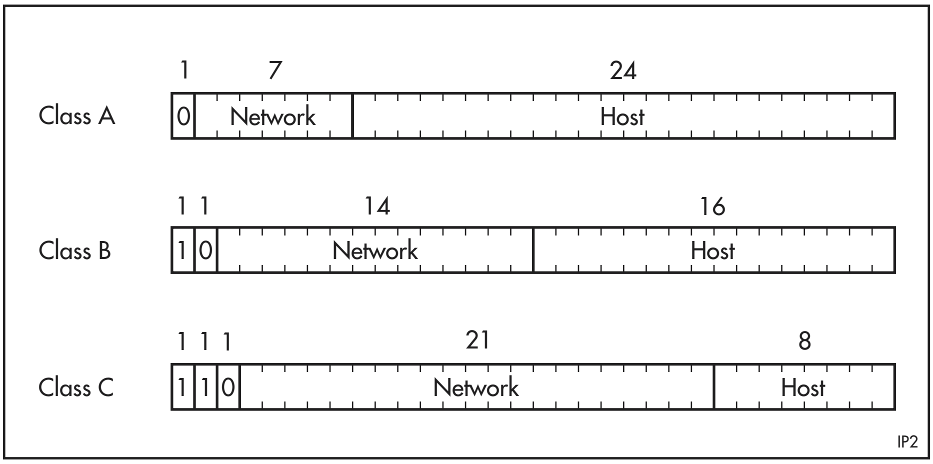
Internet addresses are fundamental to the operation of the TCP/IP internet. Each packet must contain an internet address to determine where to send the packet. Most packets also require a source address so that the sender of the packet is known. Addresses are 32-bit quantities that are logically divided into fields. They must not be confused with physical addresses (such as an Ethernet address); they serve to address Internet Protocol packets only. Addresses are organised into five classes described in the following table.

Table 14-2: Internet Protocol address classes and limits on numbers of networks and hosts

Class	Maximum number of possible networks	Maximum number of hosts per network
A	127	16,777,216
B	16,384	65,536
C	2,097,152	255
D	Reserved Class	
E	Reserved Class	

Each class differs in the number of bits assigned to the host and network portions of the address as shown in the following figure.

Figure 14-2: Subdivision of the 32 bits of an Internet address into network and host fields for class A, B, and C networks



The addressing scheme lets routers efficiently extract the host and network portions of an address. In general, a router is interested only in the network portion of an address.

Class A sets the Most Significant Bit (MSB) to 0 and allocates the next 7 bits to define the network and the remaining 24 bits to define the host. Class B sets the two MSBs to 10 and allocates the next 14 bits to designate the network while the remaining 16 refer to the host. Class C sets the three MSBs to 110 and allocates the next 21 bits to designate the network. The remaining 8 are left to the user to assign as host or subnet numbers.

The term *host* refers to any attached device on a subnet, including PCs, mainframes, and routers. Most hosts are connected to only one network; that is, they have a single IP address. Routers are connected to more than one network and can have multiple IP addresses. The IP address is expressed in *dotted decimal notation* by taking the 32 binary bits and forming 4 groups of 8 bits, each separated by a dot. For example:

```
10.4.8.2 is a class A address

10 is the DDN assigned network number
  .4.8 are (possibly) user assigned subnet numbers
      .2 is the user assigned host number


172.16.9.190 is a class B address

172.16 is the DDN assigned network number
      .9 is the user assigned subnet number
        .190 is the user assigned host number
```

The value 0.0.0.0 defines the default address, while a value of all ones in a host portion (such as 255) is reserved as the broadcast address. Some older versions of UNIX use a broadcast value of all zeros, therefore both the value ‘0’ and the value ‘255’ are reserved within any user assigned host portion. The address 172.16.0.0 refers to **any** host (**not** every host) on **any** subnet within the class B address 172.16. Similarly 172.16.9.0 refers to **any** host on subnet 9, whereas 172.16.9.255 is a packet addressed to **every** host on subnet 9. The router uses this terminology to indicate where packets are to be sent.

An address with 0 in the host portion refers to ‘this particular host’ while an address with 0 in the network portion refers to ‘this particular network’. As mentioned above, a value of all ‘1’ (255) is a broadcast. To reduce loading, IP consciously tries to limit broadcasts to the smallest possible set of hosts; hence, most broadcasts are *directed*. For example 172.16.56.255 is a broadcast to subnet 56 of network 172.16.

A major problem with the IP type of addressing is that it defines connections, not hosts. A particular address, although it is unique, defines a host by its connection to a particular network. Therefore, if the host is moved to another network, the address must also change. The situation is analogous to the postal system. A related problem can occur when an organisation with a class C address finds that they need to upgrade to class B. This involves a total change of every address for all hosts and routers. Thus the addressing system is not scalable.

Subnets

The growth of the Internet has meant a proliferation in the number of addresses that core routers must handle. More addresses mean more loading, which tends to slow the system down. This can be overcome by minimising the number of network addresses by sharing the same IP prefix (the assigned network number) with multiple physical networks. Generally these would all be within the same organisation although not required. There are two main ways of achieving this: subnetting and proxy ARP. Proxy Address Resolution Protocol (ARP) is discussed in "Address Resolution Protocol (ARP)" on page 14-12.

A subnet is formed by taking the host portion of the assigned address and dividing it into two parts. The first part is the 'set of subnets' while the second refers to the hosts on **each** subnet. For example, the DDN may assign a class B address as 172.16.0.0. The system manager would then assign the lower two octets in some way that makes sense for the network. A common method for class B is to simply use the higher octet to refer to the subnet. Thus there are 254 subnets (0 and 255 are reserved) each with 254 hosts. These subnets need not be physically on the same media. Generally they would be allocated geographically with subnet 2 being one site, subnet 3 another and so on. Some sites may have a requirement for multiple subnets on the same LAN. This could be to increase the number of hosts or simply to make administration easier. In this case it is normal (but not required) that the subnets be assigned contiguously for this site. This makes the allocation of a subnet mask easier. This mask is needed by the routers to ascertain which subnets are available at each site. Bits in the mask are set to 1 if the router is to treat the corresponding bit in the IP address as belonging to the network portion, or set to 0 if it belongs to the host portion. This allows a simple bit-wise logical "and" to determine if the address should be forwarded.

Although the standard does not require that the subnet mask select contiguous bits, it is normal practice to do so. To do otherwise can make the allocation of numbers rather difficult and prone to errors. Some example masks are:

```
11111111.11111111.11111111.00000000 = 255.255.255.0
<----network----> <subnet> <-host->
```

This would give 254 subnets on a class B network, each with 254 hosts.

```
11111111.11111111.11111111.11110000 = 255.255.255.240
<----network----> <--subnet--><host>
```

This would give 4094 subnets on a class B network, each with 14 hosts, or 14 subnets on a class C network each with 14 hosts.

The official description of subnetting is given in RFC 950. Subnet information and IP addresses are added to the router with the **add ip interface** command on page 14-77 and the **set ip interface** command on page 14-145.

Multihoming

The router can be configured as a *multihomed* device with multiple IP addresses. Up to 16 logical IP interfaces can be added to a single Layer 2 interface such as eth0, vlan1 or ppp0, and up to a total of 1280 logical interfaces per router.

An IP interface name is formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. For example, eth0-0 is the first logical IP interface assigned to the Ethernet 0 interface and ppp1-8 is the ninth logical IP interface assigned to the ppp1 interface. If a logical interface is not specified, 0 is assumed. For example, ‘ppp0’ is equivalent to ‘ppp0-0’.

Each logical interface has its own unique IP address and mask, and can be assigned its own traffic filter, policy filter, priority filter, GRE entity and security association. Each logical interface has its own interface counters and can be enabled, disabled, or reset independently of other logical interfaces assigned to the same Layer 2 interface. Each additional logical interface created on a Layer 2 interface adds an extra entry to the IP Address Table in the SNMP MIB-II MIB. See Chapter B, SNMP MIBs for a complete description of the objects in MIB-II.

The router does not support a single logical interface being associated with multiple physical interfaces in order to increase the reliability or throughput between directly connected machines by providing alternative physical paths between them. This functionality is provided by Layer 2 multiplexing schemes such as PPP multilink.

Local Interfaces

A local Interface is one that is always available for higher layer protocols to use and advertise to the network. Although a local interface is assigned an IP address, it does not have the usual requirement of connecting to a lower layer physical entity. This lack physical of attachment creates the perception of a local interface always being accessible via the network.

Local interfaces can be utilised by a number of protocols for various purposes. They can be used to improve access to a router, as well as increasing its reliability, security, scalability and protection. In addition, local interfaces can add flexibility and simplify management, information gathering and filtering.

One example of this increased reliability is for OSPF to advertise a local interface as a interface-route into the network irrespective of the physical links that may be “up” or “down” at the time. This provides a higher probability that the routing traffic will be received and subsequently forwarded. Further reliability and performance could be provided by configuring parallel BGP paths to a local interface on a peer device, which would result in improved load sharing.

Access and security can be improved through filtering. Incoming traffic can be filtered by rules that specify local interfaces as the only acceptable destination addresses.

Information gathering and filtering as well as management can potentially be simplified if protocols such as SNMP use local interfaces for receiving ender sending trap and log type information.

To add a new local interface, use the **add ip local** command on page 14-82.

To delete a new local interface, use the **delete ip local** command on page 14-102.

Address Resolution Protocol (ARP)

Most hosts also have a media-dependant physical address as well as the assigned IP address. This is a 6-byte, globally unique number for Ethernet LANs. Hosts need to know the physical address in order to communicate. The Address Resolution Protocol (ARP) lets a host find a target's physical address on the same media simply by knowing its IP address. It does this by sending out an ARP broadcast packet with both the source and destination IP address. The broadcast is media-dependant. For Ethernet LANs, the broadcast address is a packet whose destination address bits are all '1'. All stations on the LAN receive this address but only one host recognises its own IP address. It replies, thereby giving the original host its physical address.

The ARP protocol is defined in RFC 826 and is a simple but effective use of directed broadcasts. To reduce the number of broadcasts, each host generally keeps a cache of the IP address to physical address mappings (also called *bindings*). This cache is searched first before a broadcast is attempted to see if a mapping already exists. The ARP cache entries are aged to eliminate non-current connections. In the case of a packet destined for a local host, an initial ARP request is sent. If a response is not received, the ARP request is retried before an ICMP message is sent back to the packets sender.

A static entry can be added to the ARP cache to map hosts that do not support the ARP protocol using the **add ip arp** command on page 14-64. However, it is rarely necessary to add an ARP entry this way.

To modify existing static ARP entries, use the **set ip arp** command on page 14-135.

To delete existing static ARP entries, use the **delete ip arp** command on page 14-97.

To display the current contents of the router's ARP cache, use the **show ip arp** command on page 14-172.

Dynamic ARP entries are *aged* to ensure that the table does not fill with entries for hosts that are no longer active. Old entries are deleted. Static ARP entries are not aged.

The router uses a technique called *Proxy ARP* (defined in RFC 1027) to allow hosts that do not support routing (i.e. they have no knowledge of the network structure) to determine the physical addresses of hosts on other networks. The router intercepts ARP broadcast packets and substitutes its own physical address for that of the remote host. This occurs only when the router has the *best* route to the remote host. By responding to the ARP request, the router ensures that subsequent packets from the local host are directed to its physical address, and it can then forward these to the remote host. The process is symmetrical. Proxy ARP is enabled by default for Eth and VLAN interfaces. It can be enabled or disabled selectively using the command:

```
set ip interface=interface proxy={on|off}
```

To display details about interfaces assigned to the IP module, including whether Proxy ARP is enabled on each interface, use the **show ip interface** command on page 14-191.

MAC Address Logging

MAC Address Logging lets a user initiate logging of the MAC addresses of equipment connected to the router LAN interfaces, and which access the WAN interface. This provides an auditing trail in the event of anyone hacking into the system.

If NAT is used, the layer two MAC address of the equipment needs to be logged in addition to the IP address.

To enable MAC address logging, use the **enable ip arp log** command on page 14-120.

To disable MAC address logging, use the **disable ip arp log** command on page 14-111.

DHCP Client

IP interfaces can be configured either with a static IP address, or with a dynamic IP address assigned by DHCP (Dynamic Host Configuration Protocol). To configure an IP interface to use an address assigned by DHCP, set the **ipaddress** parameter to DHCP in the **add ip interface** command on page 14-77 and the **set ip interface** command on page 14-145.

When the **ipaddress** parameter of an IP interface is set to DHCP rather than a static IP address, the router’s DHCP client obtains the IP address and subnet mask for the interface, and other IP configuration parameters, from a DHCP server. See the description of the **add ip interface** command on page 14-77 for a list of the DHCP reply parameters the router uses to configure IP interfaces.

For example, to configure interface eth0 to obtain its IP address and subnet mask from DHCP, use the command

```
set ip interface=eth0 ipaddress=dhcp
```

If an IP interface is configured to obtain its IP address and subnet mask from DHCP, the interface does not take part in IP routing until the IP address and subnet mask have been set by DHCP.

Remote address assignment must be enabled with the **enable ip remoteassign** command on page 14-126 before IP interfaces can accept addresses dynamically assigned by DHCP.

ICMP

The Internet Control Message Protocol (ICMP) allows routers to send error and control messages to other routers or hosts. It provides the communication between IP software on one system and IP software on another. The router implements all non-obsolete ICMP functions (Table 14-3 on page 14-14). Some early systems may not fully implement all ICMP types. In particular type 11 (Time To Live Exceeded) is frequently not fully implemented.

The following ICMP messages can be disabled or enabled by the network manager:

- Network unreachable (RFC792 Type 3 Code 0)
- Host unreachable (RFC792 Type 3 Code 1)
- ICMP redirect messages (RFC792 Type 5 Code 0, 1, 2, 3)

To enable ICMP messages, use the **enable ip icmpreply** command on page 14-124.

To disable ICMP messages, use the **disable ip icmpreply** command on page 14-115.

Table 14-3: ICMP messages implemented by the router

ICMP Message Type	Router Response
Echo reply (0)	This is used to implement the **ping** command on page 14-129 that is common to most UNIX and TCP implementations. The router sends out an echo reply in response to an echo request.
Destination unreachable (3)	This message is sent when the router drops a packet because it did not have a route to the destination.
Source Quench (4)	The router sends this message when it must drop a packet due to limited internal resources. This could be because the source was sending data too fast to be forwarded.
Redirect (5)	The router issues this message to inform a local host that its target is located on the same LAN (no routing is required) or when it detects a host using a non-optimal route (usually because a link has failed or changed its status).
Echo request (8)	This is related to (1) and results in an echo reply being sent. The router can also generate an echo request as a result of the **ping** command on page 14-129.
Time to Live Exceeded (11)	If the TTL field in a packet falls to zero, the router sends this message.This could occur when a route is excessively long or when too many hops are in the path.

ICMP Router Discovery Advertisements

Router Discovery on the Router

The router supports all of *RFC 1256, ICMP Router Discovery Messages*, as it applies to routers. If this feature is configured, the router sends router advertisements periodically and in response to router solicitations. It does not support the Host Specification section of this RFC.

Benefits

Before an IP host can send an IP packet, it has to know the IP address of a neighbouring router that can forward it to its destination. ICMP Router Discovery messages allow routers to automatically advertise themselves to hosts. Other methods either require someone to manually keep these addresses up to date, or require DHCP to send the router address, or require the hosts to be able to eavesdrop on whatever routing protocol messages are being used on the LAN.

Router Discovery Process

The following table summarises what happens when Router Discovery advertisements are enabled for interfaces on the router.

Table 14-4: Router discovery process

When...	Then...
Router Discovery advertising starts on a router interface because: - the router starts up, or - advertisements are enabled on the router or on an interface	the router multicasts a router advertisement and continues to multicast them periodically until router advertising is disabled.
a host starts up	the host may send a router solicitation message.
the router receives a router solicitation	the router multicasts an early router advertisement on the multicast interface on which it received the router solicitation.
a host receives a router advertisement	the host stores the IP address and preference level for the advertisement lifetime.
the lifetime of all existing router advertisements on a host expires	the host sends a router solicitation.
a host does not receive a router advertisement after sending a small number of router solicitations	the host waits for the next unsolicited router advertisement
a host needs a default router address	the host uses the IP address of the router or L3 switch with the highest preference level.
Router Discovery advertising is deleted from the physical interface (DELETE IP ADVERTISE command), or the logical interface has ADVERTISE set to NO (SET IP INTERFACE command)	the router multicasts a router advertisement with the IP address(es) that stopped advertising, and a lifetime of zero (0). It continues to periodically multicast router advertisements for other interfaces.
the router receives a router advertisement from another router	the router does nothing but silently discards the message.

Router Advertisement Messages

A *router advertisement* is an ICMP (type 10) message containing:

- In the destination address field of the IP header, the interface's configured advertisement address, either 224.0.0.1 (ALL) or 255.255.255.255 (LIMITED).
- In the lifetime field, the interface's configured advertisement lifetime.
- In the Router Address and Preference Level fields, the addresses and preference levels of all the logical interfaces that are set to advertise.

Router Solicitation Messages

A *router solicitation* is an ICMP (type 10) message containing:

- Source Address: an IP address belonging to the interface from which the message is sent
- Destination Address: the configured Solicitation Address, and
- Time-to-Live: 1 if the Destination Address is an IP multicast address; at least 1 otherwise.

Router Advertisement Interval

The router advertisement *interval* is the time between router advertisements. For the first few advertisements sent from an interface (up to 3), the router sends the router advertisements at intervals of at most 16 seconds. After these initial transmissions, it sends router advertisements at random intervals between the minimum and maximum intervals that the user configures, to reduce the probability of synchronization with the advertisements from other routers on the same link. By default the minimum is 450 seconds (7.5 minutes), and the maximum is 600 seconds (10 minutes).

Preference Level

The *preference level* is the preference of the advertised address as a default router address relative to other router addresses on the same subnet. By default, all routers and layer 3 switches have the same preference level, zero (0). While it is entered as a decimal from -2147483648 to 2147483647, it is encoded in router advertisements as a twos-complement hex integer from 0x8000000 to 0x7fffffff. A higher preference level is preferred over a lower value.

Lifetime

The *lifetime* of a router advertisement is how long the information in the advertisement is valid. By default, the lifetime of all advertisements is 1800 seconds (30 minutes).

Configuration Procedure

By default, the router does not send router advertisements.

To configure the router to send router advertisements

1. **Set the physical interface to advertise.**

   For each physical interface that is to send advertisements, add the interface. In most cases the default advertising parameters will work well, but you can change them if required. By default, the router sends router advertisements every 7.5 to 10 minutes, with a lifetime of 30 minutes. These settings are likely to work well in most situations, and will not cause a large amount of extra traffic, even if there are several routers on the LAN. If you change these settings, keep these proportions:

   ```
   lifetime=3 x maxadvertisementinterval
   ```

   ```
   minadvertisementinverval=0.75 x maxadvertisementinterval
   ```

   To change these settings, use one of the commands:

   ```
   add ip advertise interface
   ```

   ```
   set ip advertise interface
   ```

2. **Stop advertising on other logical interfaces.**

   By default, logical interfaces are set to advertise if their physical interface is set to advertise. If the physical interface has more than one logical interface (IP multihoming), and you only want some of them to advertise, set the other logical interfaces not to advertise with one of the commands:

```
add ip interface=interface ipaddress={ipadd|dhcp}
    advertise=no [other-ip-parameters]

set ip interface=interface advertise=no
    [other-ip-parameters]
```

3. **Set preference levels.**

   By default, every logical interface has the same preference for becoming a default router (mid range, 0). To give a logical interface a higher preference, increase **preferencelevel**. To give it a lower preference, decrease this value. If it should never be used as a default router, set it to **notdefault**.

```
add ip interfaceinterface ipaddress={ipadd|dhcp}
    preferencelevel={-2147483648..2147483647|notdefault}
    [other-ip-parameters]

set ip interface=interface
    [preferencelevel={-2147483648..2147483647|notdefault}]
    [other-ip-parameters]
```

4. **Enable advertising.**

   To enable router advertisements on all configured advertising interfaces, use the command:

```
enable ip advertise
```

5. **Check advertise settings.**

   To check the router advertisement settings, use the command:

```
show ip advertise
```

Routing

The process of routing packets consists of selectively forwarding data packets from one network to another. The router must determine which network to send each packet to, and over which interface to send the packet in order to reach the desired network. This information is contained in the router’s *routes*. For each packet, the router chooses the best route it has for that packet and uses that route to forward the packet. In addition, you can define filters to restrict the way packets are sent.

Types of Routes

The router learns routes from static information entered as part of the configuration process and by listening to any configured routing protocols. The following types of routes are available on the router:

- Interface

  The router creates an interface route when you create the interface. This route tells the router to send packets over that interface when the packets are addressed to the interface’s subnet.

- Dynamic

  The router learns dynamic routes from one or more routing protocols such as RIP or OSPF. The routing protocol updates these routes as the network topology changes.

- Static

  You can manually enter routes, which are then called static routes. For configuration instructions, see "Configuring Static Routes" on page 14-18. Uses of static routes include:

  - To specify the default route (to 0.0.0.0). If the router does not have another route to the packet's destination, it sends it out the default route. The default route normally points to an external network such as the Internet.
  - To set up multiple networks or subnets. In this case you define multiple routes for a particular interface, usually a LAN port. This is a method of supporting multiple subnets on a single physical media.

The Routing Table

The router maintains its routing information in a table of routes that tells the router how to find a remote network or host. Each route is uniquely identified in the table by its IP address, network mask, next hop, interface, protocol, and policy.

When the router receives an IP packet, and no filters are active that would exclude the packet, the router scans the routing table to find the most specific route to the destination, on an "up" interface. If multiple routes are equally specific, it selects the route with the lowest preference value. If multiple routes have equal preference, it selects the route with the lowest metric.

If the router does not find a direct route to the destination, and no default route exists, the router discards the packet and sends an ICMP message to that effect back to the source.

The router maintains the routing table dynamically by using one or more routing protocols such as RIP or OSPF. These protocols act to exchange routing information with other routers or hosts.

Configuring Static Routes

To create a static route, use the command:

```
add ip route=ipadd interface=interface nexthop=ipadd
    [circuit=miox-circuit] [dlci=dlci] [mask=ipadd]
    [metric=1..16] [metric1=1..16] [metric2=1..65535]
    [policy=0..7] [preference=0..65535] [tag=1..65535]
```

To define a default route, set **ipaddress** to 0.0.0.0 and **nexthop** to the network (router) where default packets are to be directed.

To define a subnet, set **ipaddress** to address of the new subnet, **nexthop** to 0.0.0.0, and **metric** to 1.

To modify an existing static route, use the command:

```
set ip route=ipadd interface=interface mask=ipadd
    nexthop=ipadd [circuit=miox-circuit] [dlci=dlci]
    [metric=1..16] [metric1=1..16] [metric2=1..65535]
    [policy=0..7] [preference=0..65535] [tag=1..65535]
```

To remove a static route altogether, use the command:

```
delete ip route=ipadd mask=ipadd interface=interface
    nexthop=ipadd
```

Caching Routes

By default, the router caches routes to improve route lookup performance. The route cache holds the most recently used routes. When the router is determining the best route to a destination, it searches the cache first, using a hash function calculated from the destination information. If the router does not find a route in the cache, it searches the entire route table. If the router then finds a route in the route table, it adds that route to the cache.

To disable the cache, use the command:

```
disable ip route cache
```

To enable the cache, use the command:

```
enable ip route cache
```

To see the current contents of the route cache, use the command:

```
show ip route=[ipadd] cache
```

Dynamic Routing Protocols

In all but the most simple networks, we recommend that you configure at least one dynamic routing protocol. Routing protocols enable the router to learn routes from other routers and switches on the network, and to respond automatically to changes in network topology. Options include:

- RIP—a relatively simple protocol which is particularly suitable for dynamically learning the interior structure of a network. *Interior* refers to routing within an organisation. For information about configuring RIP, see “RIP” on page 14-24.
- OSPF—a more complex protocol suitable for dynamically learning the interior and exterior structure of a network. *Exterior* refers to routing between organisations. For information about OSPF and configuring OSPF, see Chapter 23, Open Shortest Path First (OSPF).
- BGP—a complex protocol capable of managing thousands of routes efficiently. For information about BGP and configuring BGP, see Chapter 49, Border Gateway Protocol version 4 (BGP-4).

Setting Preference of Dynamically-Learned Routes

You can set the preference for all routes that the router learns from a particular protocol, using the command:

```
set ip route preference={default|1..65535} protocol={bgp-ext|
    bgp-int|ospf-ext1|ospf-ext2|ospf-inter|ospf-intra|ospf-
    other|rip}
```

This may be useful if you have more than one routing protocol defined because if the router has a choice of two valid routes it chooses the one with the lowest value for preference. For example, you can set all RIP routes to have a lower preference than OSPF routes.

This command does not change:

- Default preferences. Therefore, you can use **preference=default** to return to the original setting
- The preference for static routes. Use the **set ip route** command on page 14-156 instead.
- The preference for interface routes. These are always created with a lower preference value than dynamically learned routes, but can also be changed using the **set ip route** command.

Displaying Route Information

To see the number of routes, and other summary information, use the command:

```
show ip route general
```

To see all routes in the routing table, including both static and dynamic routes, use the command:

```
show ip route full
```

To see the number of octets sent and received using each route, use the command:

```
show ip route count
```

To see information about only routes to a particular subnet, specify the subnet address, use the command:

```
show ip route=ipadd [{cache|count|full}]
```

To see a list of all routes to a destination, with the most specific routes first, use the command:

```
show ip route=ipadd
```

The routes may have different metrics, next hops, policy or protocol. A list of routes is uniquely identified by its IP address and net mask.

Equal Cost Multipath Routing

Equal Cost Multipath Routing (ECMP) allows the router to distribute traffic over multiple equal-cost routes to a destination. When the router sends packets to that destination, it distributes the packets across all equal-cost routes. The router considers a route to be equal cost if it has the same destination IP address, mask, preference, and metric. You can have up to 16 individual routes to each destination.

Configuring ECMP

Table 14-5: Procedure for using ECMP

Step	Command	Action
1	**enable ip route** multipath	If ECMP has been disabled, enable it. ECMP is enabled by default.
2	**add ip route**=*ipadd* interface=*interface* nexthop=*ipadd* [*other-options...*]	Add static routes as required. You can create multiple static routes to the same destination.
3		Configure dynamic routing protocols as required.
4	**set ip route preference**=1..65535 protocol={bgp-ext|bgp-int|ospf-ext1|ospf-ext2|ospf-inter|ospf-intra|ospf-other|rip}	If you want routes from different routing protocols to have equal cost, give the protocols the same preference setting.

To disable ECMP, use the command:

```
disable ip route multipath
```

Routing Information Filters

Two mechanisms are provided to manage the process of learning dynamic routes via routing protocols:

- **Route filters**
- **Trusted routers**

Route filters

Route filters control which routes are received and sent by each routing protocol over each interface and to particular destinations. When routing information is received by the router, routes that match a filter are added to or omitted from the route table depending on the action defined for the route filter. When the router transmits routing information, routes that match a route filter are included or excluded from the transmission depending on the action defined for the route filter. Route filters do not apply to static or interface routes.

To create a route filter, use the **add ip route filter** command on page 14-90.

To destroy a route filter, use the **delete ip route filter** command on page 14-106.

To modify a route filter, use the **set ip route filter** command on page 14-158.

To list the current route filters, use the **show ip route filter** command on page 14-210.

When a route is received or transmitted by a routing protocol, the list of route filters is searched for a match to the route. The **ip**, **mask**, **interface**, **nexthop**, **policy**, and **protocol** parameters define a pattern to match against. The **direction** parameter determines whether the filter applies to route information received, transmitted or both. The **action** parameter determines whether routes matching the pattern are used or discarded.

Only one filter is ever applied to an individual route. Processing stops when a match is found to a filter, or the end of the filter list is reached. If at least one route filter is defined then the route filter list has an implicit “exclude all” entry after the last entry in the list. It may be necessary, therefore, to add an “include all” filter at the end of the list to allow all other routes that don't match.

Note that there are filtering limitations for the OSPF protocol. How the OSPF protocol is implemented affects how the route filter operation on OSPF Link State Advertisement (LSA) works. A route filter with **direction=send** filters matching routes regarded as Autonomous System (AS) external routes by OSPF. Also, the **interface** parameter is ignored, i.e. all interfaces are treated indifferently.

To filter OSPF inter-area routes (summary LSAs) define a ‘do not advertise’ OSPF range on an Area Border Router. This stops inter-area routes being advertised into another area. To do this, use the **set ospf range** command on page 23-47 of Chapter 23, Open Shortest Path First (OSPF) with the **effect** parameter set to **donotadvertise**.

Route filters defined in the **receive** direction may also have a subsequent effect on outgoing OSPF Link State Advertisements (LSAs). The behaviour depends on the router’s OSPF role. On an Area Border Router, a matching receive direction filter has the subsequent effect that no summary LSA is advertised into another area because summary LSA messages are derived from the filtered IP route table. In contrast, other OSPF area members derive their messages from their local OSPF configuration and their received LSA messages. Note that a OSPF design requirement is that LSA messages must not be filtered within an OSPF area.

IP route filters affect the interaction between the routing module and the IP routing table, but IP route filters do not filter receipt of routing protocol messages by the routing module and do not directly filter messages sent from the routing protocol. Messages sent from the routing protocol are affected if and only if they are derived from the IP routing table, which is true in most situations, including RIP, OSPF-ext messages, and OSPF summary Link State Advertisements (LSAs). Some types of OSPF LSAs, such as intra-area, cannot be filtered by route filters because they are not based on the local IP route table, and in order to meet OSPF design requirements of LSA propagation within areas.

The immediate effect of a route filter with **direction** set to **receive** and **action** set to **exclude** is that route advertisements received matching the filter do not result in a new entry in the local IP route table. However, routes already in the IP route table are not deleted even when they match the route filter. Therefore, to dynamically add a route filter at the manager prompt, it may be necessary to manually delete unwanted routes from the IP route table.

Trusted routers

The alternative mechanism is to define one or more trusted routers. A *trusted router* is a source of RIP broadcasts that can be trusted to provide up-to-date, valid routing information. If one or more trusted routers are defined, only routing information from the specified source is accepted by the router and included in the routing table. If no trusted routers are defined, routing information is accepted from any source, although RIP packets may be filtered (for example, with the **add ip filter** command on page 14-68 or the **add ip route filter** command on page 14-90) before reaching the RIP process.

To add a trusted router, use the **add ip trusted** command on page 14-95.

To delete a trusted router. use the **delete ip trusted** command on page 14-108

To display a list of trusted routers, use the **show ip trusted** command on page 14-215.

RIP

Routing Information Protocol (RIP) is described fully in RFC 1058. Extensions for RIP version 2 are described in RFC 1723. Extensions for RIP on demand is described in RFC 1582. RIP is a fairly simple distance vector protocol that defines networks based on how many hops they are from the router. Once a network is more than 15 hops away (one hop is one link), it is not included in the routing table.

The possible routes (there may be more than one) to a particular host are selected on the basis of the shortest one. If two routes have the same metric (hop count) or cost, the first one found is chosen. RIP does not cope with a meshed (multiply connected) network very well, but it suits star topologies very well.

RIP can have multiple links to a particular destination. It chooses the best one based simply on the metric, which for RIP, is either administratively assigned, or is the hop count (i.e. number of links). RIP cannot send data over multiple paths to a destination. Once a route is chosen, all data is sent over this path until the metric changes. OSPF routing is required if load balancing is required over multiple paths. Routes with equal cost are kept (and possibly used) for RIP, EGP, and OSPF.

Each router configured for RIP maintains a relatively simple route table as described earlier. The router periodically broadcasts its routing information to other routers. Similarly, it obtains this information from neighbouring routers to improve its own picture of the network. Routes are removed from the table when they are not kept up to date (refreshed) by the neighbouring routers.

The RIP version 2 extensions allow RIP updates to contain subnet masks and next hop information. The ability to carry subnet masks allows the use of different sized subnet masks on different subnets within the same network.

The RIP on demand extensions allow RIP to be used over demand links that are activated when there is traffic to send. Route information is exchanged when there is a change in the routing table and routes obtained over the link are not aged.

RIP broadcasts are automatically enabled when at least one RIP neighbour is defined. RIP neighbours are defined with the **interface** parameter in the **add ip rip** command on page 14-86.

The operation of RIP is controlled by four timers whose values are set globally with the **set ip riptimer** command on page 14-155.

To display current values of the RIP timers, use the **show ip riptimer** command on page 14-205.

To remove RIP neighbours, use the **delete ip rip** command on page 14-104. If no RIP neighbours are defined, RIP broadcasts are disabled.

To display the neighbours to which the router is sending RIP broadcasts, use the **show ip rip** command on page 14-201.

EGP

The Exterior Gateway Protocol (EGP) is a protocol that is used to exchange routes with routers on exterior networks. EGP operates by considering distinct networks, or groups of networks, as *autonomous systems*. The specification for EGP is contained in RFC 904. The use of EGP is usually confined to connections to a *trusted core* of routers, such as the Internet. In order for a router to make an EGP connection to a neighbouring router, the neighbour must either be on a network specified by an interface or on a network known to the router by Internal Gateway Protocols such as RIP or OSPF. The router forms EGP connections to neighbours that have been explicitly defined. EGP is used to implement what is called a *third party system*. It does this by notifying a neighbour router that another router (the third party) on the network has the best routes for a set of destinations.

To use EGP a network must be assigned an autonomous system number from the DDN Network Information Centre (see “Background Reading” on page xciv of Preface for address details).

To set the router’s autonomous system number, use the **set ip autonomous** command on page 49-105 of Chapter 49, Border Gateway Protocol version 4 (BGP-4).

To define EGP neighbour routers, use the **add ip egp** command on page 14-67.

To delete an EGP neighbour, use the **delete ip egp** command on page 14-98.

EGP is disabled by default. It can be enabled with the **enable ip egp** command on page 14-121.

To disable EGP, use the **disable ip egp** command on page 14-112.

Routing information derived from the RIP protocol can be transferred into outgoing EGP messages. This option can be enabled with the **enable ip exportrip** command on page 14-122. To disable this feature, use the **disable ip exportrip** command on page 14-112.

To display the defined EGP neighbours and the current status of the EGP links, use the **show ip egp** command on page 14-185.

OSPF

The Open Shortest Path First (OSPF) protocol is a relatively recent standard that is documented in RFC 1247. It has a number of significant benefits over older distance vector based protocols like RIP, including:

- OSPF is an open, published specification and not proprietary to any manufacturer.
- OSPF supports the concept of areas to allow networks to be administratively partitioned as they grow in size.
- Load balancing, in which multiple routes exist to a destination, is also supported. OSPF distributes traffic over these links.

See Chapter 23, Open Shortest Path First (OSPF) for more details.

Metrics

Metrics are used to determine the criteria for using one route over another route. In this sense they *measure* some aspect of the route. For RIP and EGP the metric is simply the *hop count*, which is a measure of the number of links it takes to get to the specified destination, or in other words, how far away it is. OSPF has a number of metrics. This, in part, accounts for its better performance in that it is not simply looking at one view of the network. For example, it can also allow for the fact that not all links have the same bandwidth. The router supports only one OSPF metric per interface.

OSPF Auto Cost Calculation

OSPF interfaces automatically set the OSPF metric of an IP interface on the basis of the bandwidth of the interface, instead of the system administrator manually setting the OSPF metric. Automatic setting takes into account that the speed of an interface can change over time, when ports change link state or change speed via auto negotiation or manual setting. If metrics are manually set, some interfaces are preferred when they should not be because the network configuration dynamically changes.

Note that the interface speed used in the cost calculation is the average interface speed. For example, if the interface is a VLAN with two ports up, and one port has a speed of 10 and the other a speed of 100, then the metric will be 18.

To configure auto cost calculation:

1. Do not set the OSPF metric manually in the **add ip interface** command. If you have, remove the manual setting, using the command:

   ```
   set ip interface=int ospfmetric=default
   ```

   The **ospfmetric** parameter specifies the cost of crossing the logical interface, for OSPF. If **default** is specified the interface is restored to the default metric value. The setting of the OSPF metric to a value other than **default** provides a mechanism to provide a metric for an interface that is preferred over the OSPF automatic metric setting (if enabled via **set ospf autocost=on**).

2. Set **autocost** to on and change the reference bandwidth if necessary, using the command:

   ```
   set ospf autocost=on [refbandwidth=10..10000]
   ```

   The **autocost** parameter specifies whether or not the switch will assign OSPF interface metrics based on the available interface bandwidth. If an OSPF metric has been manually assigned using the **add ip interface ospfmetric**=x command, the manual metric setting will take priority over an automatic metric setting. The default is **off**.

   The **refbandwidth** parameter specifies the reference bandwidth in megabits per second used for calculating the OSPF metric. The cost is calculated as **refbandwidth** / Interface Bandwidth. Using the default settings, the automatic cost calculation will result in an OSPF metric of 10 for a fast Ethernet (100M) interface. The **autocost** parameter must be set to **on** for the parameter **refbandwidth** to take effect. The default is 1000.

3. To check the settings, use the command:

   ```
   show ospf
   ```

Policy-Based Routing

Policy routing is a way to route packets that is based on policies or rules set by the network manager. It is an alternative to priority-based routing and to destination routing protocols such as RIP and OSPF that use metrics to determine the shortest or optimal path to the destination. Policy-based routing is useful in providing equal access, protocol-sensitive routing.

The Type of Service (TOS) octet in the IP header comprises three fields: precedence (bits 0 to 2), TOS (bits 3 to 6) and MBZ (bit 7). The precedence field is intended to denote the importance or priority of the datagram, but is not commonly used. The MBZ field should always be zero (0) and is currently unused. The TOS field denotes the type of service required and is used by the network to make trade-offs between throughput, delay, reliability, and cost. The TOS field is treated as an integer value between 0 and 15. RFC 1349 defines the semantics of five specific TOS values shown in the following table.

Table 14-6: TOS values defined by RFC 1349

Decimal	Binary	Meaning
8	1000	Minimise delay
4	0100	Maximise throughput
2	0010	Maximise reliability
1	0001	Minimise cost
0	0000	Normal service

Although the semantics of the other values are undefined, they are legal TOS values and network devices must not prevent the use of these values in any way.

TOS values may be considered when determining the route to use for an IP packet. All routes have an assigned TOS value. This is normally the default TOS value (0), unless the route has been learned using a routing protocol that supports TOS, or the TOS value has been statically assigned.

To forward an IP packet, a router uses the packet's destination address to search for a route to the destination. If a route is not found, or if the selected route has an infinite metric, the destination is considered unreachable and the packet is discarded. If a single route is found with a finite metric, it is used. If more than one route is found with a finite metric, the TOS values of the selected routes can be used to refine the selection. A route with a TOS value identical to the TOS value in the IP packet is used in preference to a route with the default TOS value (0).

The router uses the TOS field in IP routes to implement policy-based routing of IP packets. However, since the TOS field in IP packets is not set or used by many IP implementations, the router makes use of filters to assign the TOS values used for policy routing to IP packets as they are received.

To enable policy routing, the first step is to create a filter to select the IP packets to be routed according to policy with the **add ip filter** command on page 14-68.

The policy filter is then assigned to an interface with the **add ip interface** command on page 14-77 or the **set ip interface** command on page 14-145. The **policyfilter** parameter specifies the policy filter to apply. Packets received via the interface are checked against the entries in the policy filter and if a match is found, the packet is routed according to the policy specified in the matching filter entry.

Note that a traffic filter, a policy filter, and a priority filter can be assigned to an interface.

- Policy and priority filters *affect* packets as they are transmitted, but traffic filters *affect* packets as they are received.
- Policy and traffic filters are *configured* on the receiving interface, but priority filters are *configured* on the transmitting interface.
- Policy and traffic filters are *applied* to packets as they are received, but priority filters are *applied* to packets as they are queued for transmission.

An interface may have a maximum of one traffic filter, one policy filter, and one priority filter, but the same traffic, policy, or priority filter can be assigned to more than one interface.

The final step is to create static routes and assign policy numbers to the routes by using the **add ip route** command on page 14-88 or the **set ip route** command on page 14-156.

When a packet is received via an interface with an assigned policy filter, and the packet matches an entry in the filter, the packet is routed using a route with the same policy number specified in the matching policy filter entry. For example, when a packet matches a policy filter entry that specifies a **policy** value of 3, the packet is routed using a route with a **policy** value of 3.

For IP packets routed according to policy numbers 0 to 7, the TOS octet in the packet's IP header is not modified. For IP packets routed according to policy numbers 8 to 15, the TOS field (bits 3 to 6) in the packet's IP header are set to the policy number less 8 and the packet is routed using a route with a policy equivalent to the policy number less 8. For example, if the policy filter assigns an IP packet a policy number of 14, the packets TOS field is set to 6 (14-8) and the packet is routed using a route with a policy of 6.

Priority-Based Routing

Priority routing is a way to route packets according to priorities set by the network manager. It is an alternative to policy-based routing and to destination routing protocols such as RIP and OSPF that use metrics to determine the shortest or optimal path to the destination. Priority-based routing is useful in managing high priority interactive traffic and low priority batch traffic over the same link.

To enable priority routing, the network manager defines a set of priorities that make routing decisions. Each priority specifies the criteria by which to select IP packets, and routing actions to perform on the packets that match the criteria.

To create a filter to select IP packets based on priority and assign a priority, use the **add ip filter** command on page 14-68.

The **priority** parameter sets the priority of IP packets from p3 (highest) to p7 (lowest). The default is p5. Priority levels p0, p1, and p2 should not be used because they may conflict with router system activities.

Assign the priority filter to an interface by using either the **add ip interface** command on page 14-77 or the **set ip interface** command on page 14-145.

The **priorityfilter** parameter specifies the priority filter to use. Filter numbers 200 to 299 are treated as priority filters. Packets transmitted via the interface are checked against entries in this filter. When a match is found, the packet goes into a queue based on the packet's priority. Packets in higher priority queues are forwarded ahead of packets in lower priority queues.

Note that a policy filter, a priority filter, and a traffic filter can be assigned to an interface.

- Priority and policy filters *affect* packets as they are transmitted, whereas traffic filters *affect* packets as they are received.
- Priority filters are *configured* on the transmitting interface, whereas traffic and policy filters are *configured* on the receiving interface.
- Priority filters are *applied* to packets as they are queued for transmission, whereas traffic and policy filters are *applied* to packets as they are received

An interface can have only one traffic filter, one policy filter, and one priority filter. However, the same traffic, policy, or priority filter can be assigned to more than one interface.

Route Templates

The router uses IP route templates to add IP routes to IP subnetworks discovered during normal operation by other protocols such as IPsec. This is necessary when IP traffic going to the subnetwork must be routed via a route other than the default route.

When a software module other than a routing protocol such as IPsec adds a route to the IP routing table, and is configured to use an IP route template, the software module supplies the IP network address and mask. The IP route template provides all the other parameters for the entry in the IP route table.

To create a route template, use the **add ip route template** command on page 14-92.

To delete a route template, use the **delete ip route template** command on page 14-107.

To modify an existing route template, use the **set ip route template** command on page 14-161.

To display a list of the currently defined route templates or information about a specific route template, use the **show ip route template** command on page 14-212.

VLAN Tagging on Eth Interfaces

Eth ports on the router can route IP packets between VLANs, by applying a VLAN tag to frames that are transmitted out of the Eth port. You can configure multiple logical interfaces on the Eth port, so it can route frames to multiple VLANs. To configure this, use one of the commands:

```
add ip interface=interface ipaddress={ipadd|dhcp}
   vlantag={1..4094|none} [other-options...]
```

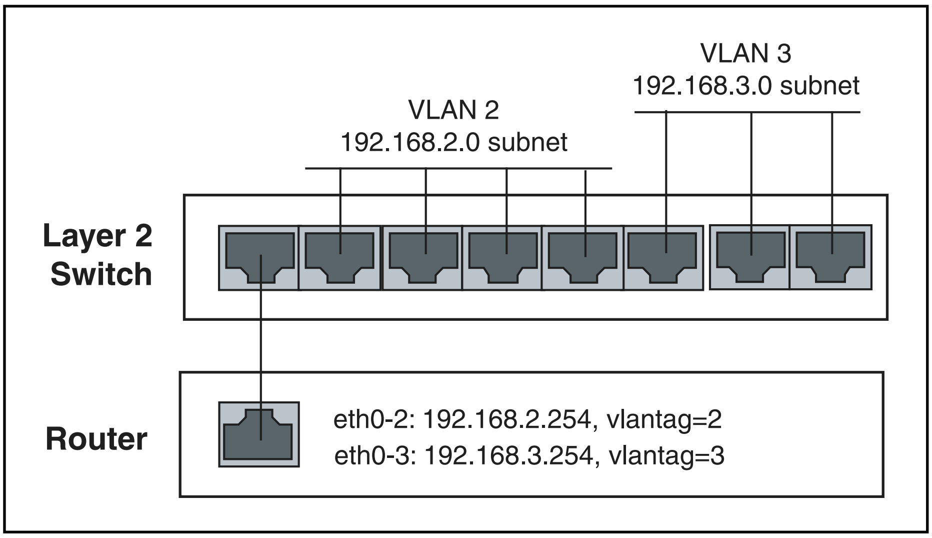
```
set ip interface=interface vlantag={1..4094|none}
   [other-options...]
```

The **vlantag** parameter specifies the VID (VLAN Identifier) to be included in the header of each frame that is transmitted over the logical interface. This parameter is only valid for Eth interfaces. Multiple logical interfaces on the same physical interface can share the same VLAN tag.

Example

In this scenario, the router is acting as both a gateway and a routing device for a Layer 2 LAN. The eth0 port on the router is connected to a Layer 2 switch through a port that is a tagged member of VLAN 2 and VLAN 3. Frames received on the eth0 port and destined for the Layer 2 switch are assigned the VID of the destination VLAN.

Figure 14-3: Example configuration for VLAN tagging on Eth interfaces



VLAN on L2 switch	IP subnet	logical ETH interface on router	Gateway address on router	VLAN tag on eth interface on router
vlan 2	192.168.2.0/24	eth0-2	192.168.2.254	2
vlan 3	192.168.3.0/24	eth0-3	192.168.3.254	3

To configure the router, use the following commands:

```
add ip interface=eth0-2 ipaddress=192.168.2.254
   mask=255.255.255.0 vlantag=2

add ip interface=eth0-3 ipaddress=192.168.3.254
   mask=255.255.255.0 vlantag=3
```

Note that only these logical Eth interfaces transmit tagged frames. Traffic that is transmitted from other interfaces (or logical interfaces) on the router are untagged.

Named Hosts

An important function of the IP module is the provision of access to Telnet services. Normally such services are accessed by specifying the IP address of the full domain name of the service provider in the **telnet** command on page 21-31 of Chapter 21, Terminal Server:

```
telnet payroll.admin.thecompany.com
telnet 172.16.8.5
```

A single router may provide access for users to many services. To make access to these services easier for users, the IP module provides a host *nickname* table that maps an IP address or a full domain name to a short, easy to remember nickname. To add entries to the host name table, use the **add ip host** command on page 14-76.

For example, to add the nickname “payroll” for the IP host with IP address 172.16.8.5 and domain name “payroll.thecompany.com”, use the command:

```
add ip host=payroll ipaddress=172.16.8.5
```

To modify an entry, use the **set ip host** command on page 14-144.

To delete an entry altogether, use the **delete ip host** command on page 14-101.

If a domain name is specified in the **telnet** command on page 21-31 of Chapter 21, Terminal Server, when a user tries to access the service, the router sends a Domain Name System (DNS) request to a defined name server to translate the host name into an IP address.

Primary and secondary name servers must be defined with the **add ip dns** command on page 14-65 and **set ip dns** command on page 14-137.

The primary and secondary name server’s addresses can be either statically configured with the **primary** and **secondary** parameters, or learned dynamically over an interface. Name servers can be learned via DHCP over an Ethernet interface (eth or vlan) or via IPCP over a PPP interface. The interface is specified with the **interface** parameter.

If no name servers have been manually configured, and name server configuration is assigned to an interface by either PPP or DHCP, this configuration is automatically used for the default name servers. Name servers configured in this way are identified by an “*” in the “Domain” column of the **show ip dns** output table (Figure 14-27 on page 14-183).

Automatically-configured name servers can be deleted with the **delete ip dns** command on page 14-97 or replaced with the **set ip dns** command on page 14-137. A deleted automatic configuration may subsequently reappear if the interface concerned is reset.

Note that from the viewpoint of the ISP, the ISP router can be configured to offer a specific DNS server address to the local router by using the command:

```
set ppp dnsprpimary=ipadd dnssecondary=ipadd
```

When the router performs a DNS lookup, it first sends the request to the primary name server. If a response is not received within 20 seconds the request is sent to the secondary name server.

Users can now access the service using any of the commands:

```
telnet payroll
telnet payroll.admin.thecompany.com
telnet 172.16.8.5
```

If the *sysName* MIB object is set to the router’s fully qualified domain name (such as router.company.com) with the **set system name** command on page 1-124 of Chapter 1, Operation, and a name server has been defined using the **set ip nameserver** command on page 14-151, then the **telnet mainhost** command attempts a Telnet connection to the host “mainhost.company.com”, provided “mainhost” is not an IP nickname (IP nicknames take precedence).

The **add ip dns** command on page 14-65 specifies the Eth, VLAN or PPP interface used to learn primary or secondary DNS server addresses. Typically the PPP interface is a dial-up connection to an ISP that provides the DNS name server PPP option. For example, a local router acting as a DNS relay for connecting a PC to the Internet (see Figure 14-4 on page 14-33).

When the PPP interface is already up when the host receives the DNS request, and a DNS server address was not set during IPCP negotiation, the DNS request is discarded. When the PPP interface is down, the interface is activated and IPCP negotiation is used to learn the DNS server address. When a DNS server address is learned as a result of the IPCP negotiation, the DNS request is forwarded to that address. Otherwise, the DNS request is discarded.

The above explanation applies to the local router acting as a DNS relay for a PC connecting to the Internet via an ISP.

If the router was originally configured to learn name servers dynamically over a particular interface for use in resolving host names in the specified domain, this configuration can be overridden by specifying values for one or both of the static name server parameters (**primary** and **secondary**). Similarly, if static name server addresses were originally configured, use of the **interface** parameter causes name server information learned dynamically to overwrite the static name server configuration. Static name server addresses are lost.

Note that from the ISP’s point of view, the ISP router can be configured to offer a specific DNS server address to the local router via IPCP by using the command:

```
set ppp dnsprimary=ipadd dnssecondary=ipadd
```

DNS servers offered by the router when acting as a DHCP server are configured with the **dnsserver** parameter in the **add dhcp policy** command on page 35-5 of Chapter 35, Dynamic Host Configuration Protocol (DHCP).

DNS Relay Agent

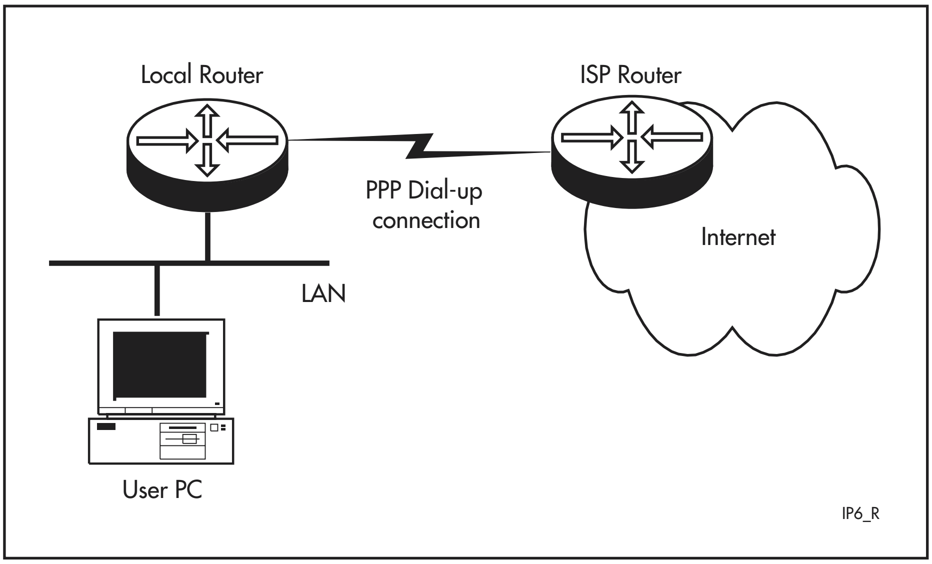
The DNS relay agent receives DNS requests from hosts and forwards them to the router's own configured DNS server. The DNS relay agent is disabled by default, and can be enabled **enable ip dnsrelay** command on page 14-121. To disable it, use the **disable ip dnsrelay** command on page 14-111.

To display the current state of the DNS relay agent, use the **show ip** command on page 14-168.

DNS requests are forwarded to the router's own DNS server. The DNS server's address can be set using the **add ip dns** command on page 14-65 and **set ip dns** command on page 14-137.

The **set ip nameserver** and **set ip dnsrelay** commands have been made obsolete by the **add ip dns** and **set ip dns** commands. These commands no longer appear in dynamically generated configuration scripts. Router-generated configuration scripts replace **set ip nameserver** and **set ip dnsrelay** commands with **add ip dns** and **set ip dns** commands.

Figure 14-4: Local router acting as a DNS relay for an Internet connection



DNS Caching

DNS caching allows the router to store recently requested domain or host addresses so they can be quickly retrieved if an identical DNS request is received. DNS caching reduces traffic on the Internet and improves performance for both DNS and DNS relay under heavy usage. The DNS cache has a limited size, and times out entries after a specified period of up to 60 minutes.

When a domain or host is requested, the cache is searched for a matching entry. If a match is found, a response is sent to the requesting PC or host. If a matching entry is not found, a request is sent to a remote server.

To add a DNS server to the list of DNS servers used to resolve host names into IP addresses, use the **add ip dns** command on page 14-65.

Once the DNS servers have been configured, set the configuration information with the **set ip dns** command on page 14-137.

For example, to add or set the IP addresses of the default primary and secondary name servers to 192.168.20.1 and 192.168.20.2 respectively, use the commands:

```
add ip dns primary=192.168.20.1 secondary=192.168.20.2
```

```
set ip dns primary=192.168.20.1 secondary=192.168.20.2
```

To set the DNS cache size and timeout values, use the **set ip dns cache** command on page 14-138.

To delete name server information from the DNS server, use the **delete ip dns** command on page 14-97.

Server Selection

The router can be configured to use a range of DNS servers with different servers being selected based on the host name being resolved.

The **domain** parameter in the **add ip dns** command allows the user to specify a suffix that must be present on a host name in order for the name servers specified by the command to be used.

If the **domain** parameter is not specified, the name servers are used as the default name servers. All DNS requests that do not match another specified domain are sent to the default name servers. This is equivalent to specifying **domain=any**.

To add primary and secondary name servers with IP addresses of 202.36.163.1 and 202.36.163.3 respectively, for use as default name servers, use the command:

```
add ip dns domain=any primary=202.36.163.1
   secondary=202.36.1.3
```

These servers are used for all host names that do not match any of the domains that are configured with their own set of name servers.

For example, to add primary and secondary name servers with IP addresses of 192.168.10.1 and 192.168.10.2 respectively, for use when resolving host names in the domain *apples.com*, use the command:

```
add ip dns domain=apples.com primary=192.168.1.1
   secondary=192.168.1.2
```

If a request is sent for the domain *www.fruit.apples.com*, the DNS servers at 192.168.1.1 or 192.168.1.2 are used because the domain matches *apples.com*.

If a request is sent for the domain *ftp.fruitpunch.apples.com*, the DNS servers at 192.168.1.1 or 192.168.1.2 are also used because the domain matches *apples.com*.

If a request is sent for the domain *www.armadillo.com*, the domain does not match *apples.com* so the ANY servers 202.36.1.1 or 202.36.1.3 are used.

Traffic Filters

Filters provide a mechanism for determining whether to process IP packets received over network interfaces. When an IP packet matches one of the patterns in a filter, the filter determines whether the packet is discarded or passed to the IP routing module.

Filtering is configured on a per-interface basis for packets received over the interface. Filtering decisions can be based on combinations of source address, destination address, TCP port, and protocol.

A *filter* is a list of *patterns*. A *pattern* consists of the following:

- A half pattern used to compare against the source address and port of an IP packet.
- A half pattern used to compare against the destination address and port of an IP packet.
- A protocol used to compare against the protocol of an IP packet.
- An ICMP message type used to compare against the type field of an ICMP packet.
- A flag used to compare against the presence or absence of an IP options field in an IP packet header.
- A maximum reassembly size used to compare against the reassembled packet size for IP fragments.
- A flag used to compare against the initiating end of a TCP session.
- An action, either *inclusion* or *exclusion*. Inclusion is the action of allowing the IP packet to be processed further and forwarded. Exclusion is the action of discarding the IP packet.

The filter is terminated by an implicit *match all* pattern, with an exclusion action. This pattern cannot be removed and does not appear in any displays.

A *half pattern* is a combination of an *IP address*, *network mask* and *port*. The IP address and network mask are represented in dotted decimal notation. The port is a TCP or UDP port number.

A specific half pattern matches exactly one address and port combination. A general half pattern matches a range of addresses and/or ports. When two specific half patterns are combined, the resulting specific pattern matches exactly one connection between two specific address/port pairs. Any other combination of specific and/or general half patterns produces a general pattern matching more than one address/port pair.

A linear search is performed on the filter. Searching stops at the first match found, so the order of patterns is important. Specific patterns always appear before general patterns. Within the specific patterns the order of patterns does not affect filter results since each pattern matches a specific and exclusive case. Within the general patterns, the order of patterns affects filter results since each pattern matches a range of address/port combinations that may overlap with another pattern.

For example, if the aim of the filter is to include all connections from a particular network except for a small range of addresses (e.g. a particular subnet), the exclusion pattern for the subnet must appear before the inclusion pattern for the network. Otherwise, packets from the subnet are included (and forwarded for processing) by the inclusion pattern without being compared against the exclusion pattern.

Regardless of whether a pattern is specific or general, its position in the filter effects the efficiency of the filter. Patterns that match the most commonly expected conditions should appear ahead of patterns matching less common conditions. This reduces the number of comparisons required to get a match.

To add an entry to a filter, use the **add ip filter** command on page 14-68. The **sport**, **dport**, **icmpcode**, and **icmptype** parameters can be a decimal number or one of a list of predefined names. The **log** parameter determines whether matches to a filter entry result in a message being sent to the router's Logging facility, and the content of the log messages.

To modify an entry in a filter, use the **set ip filter** command on page 14-140.

To delete an entry in a filter, use the **delete ip filter** command on page 14-99.

To display filters and patterns currently defined and the number of matches, use the **show ip filter** command on page 14-186.

For overall efficiency, most traffic received by the router should be forwarded. The router should not be filtering out most of the traffic it receives. The efficiency of the filtering process can be maximised by careful ordering of all filters, including general filters, to reduce the number of comparisons required for the majority of IP packets. The counts of matches displayed in the output of the **show ip filter** command on page 14-186 can aid in determining the most efficient ordering of patterns within filters.

Defining a filter does not automatically enable it. To enable it, you must assign it to a network interface on the router with the **add ip interface** command on page 14-77.

To change the filter used on an interface, use the **set ip interface** command on page 14-145.

To display information about the interfaces assigned to the IP module, including the associated filter (if any) for each interface, use the **show ip interface** command on page 14-191.

A traffic filter, policy filter, and priority filter can be assigned to an interface. Traffic filters are configured on receiving interfaces and are applied to packets as they are received (packets are checked for a match to a filter entry). Traffic filters either discard packets or allow them into the router for processing and forwarding.

Priority filters are configured on transmitting interfaces and are applied to packets as they are queued for transmission (packets are checked for a match to a filter entry). Priority filters are applied to packets that have been received, processed, and assigned to an interface for transmission. Priority filters determine the order in which packets are sent.

Policy filters are configured on receiving interfaces and are applied to packets as they are received (packets are checked for a match to a filter entry). The policy filter's effect, however, is seen in the way filtered packets are transmitted. After packets have passed any traffic filters, the policy filter preferentially selects a route from the route table on which to forward the packet.

An interface may have a maximum of one traffic filter, one policy filter and one priority filter, but the same traffic, policy or priority filter can be assigned to more than one interface.

SNMP

SNMP (Simple Network Management Protocol) is defined in RFCs 1155–1157, 1213, 1351, and 1352. The router's implementation of SNMP is based on RFC 1157, *A Simple Network Management Protocol (SNMP)*, and RFC 1812, *Requirements for IP Version 4 Routers*. SNMP provides a mechanism for management entities, or stations, to extract information from the *Management Information Base* (MIB) of a managed device.

SNMP can use a number of different protocols as its underlying transport mechanism, but the most common transport protocol, and the only one supported by the router, is UDP. Therefore the IP module must be enabled and properly configured in order to use SNMP. SNMP *trap* messages are sent to UDP port 162; all other SNMP messages are sent to UDP port 161. The router's SNMP agent accepts SNMP messages up to the maximum UDP length the router can receive.

The router implements an enterprise MIB (enterprise number 293), and a number of other standard MIBs including MIB-II (RFC 1213), Frame Relay DTE MIB (RFC 1315), Ethernet-like Interface Types MIB (RFC 1398), Bridge MIB (RFC 1493) and the Host Resources MIB (RFC 1514). See Chapter 38, Simple Network Management Protocol (SNMP) for a detailed description of the router's SNMP agent and the commands required to configure SNMP on the router. See Chapter B, SNMP MIBs for a detailed description of the MIBs and objects supported by the router's SNMP agent.

The router's standard **set** and **show** commands can also be used to access objects in the MIBs supported by the router.

Control and Debug Commands

Several commands control the overall operation of the IP module. The IP module is disabled by default. To enable the IP module, use the **enable ip** command on page 14-119. To disable it, use the **disable ip** command on page 14-110.

All setup information is retained if the module is shut down. It is not necessary to enter new setup information after turning on the module.

The IP module operates in one of two modes, server mode or forwarding mode. In server mode the router does not route IP packets, but provides Telnet services, responds to SNMP requests, and uses TFTP to download software upgrades. In forwarding mode, the router routes IP packets, as well as performing all the functions of server mode. The default operational mode is forwarding. The operational mode is set with the **enable ip forwarding** command on page 14-123 and the **disable ip forwarding** command on page 14-114.

The current operational mode is retained when the IP module is disabled, and restored when the IP module is re-enabled. To display a snapshot of the current state of the IP module, use the **show ip** command on page 14-168.

The router stores all setup information (such as IP addresses) in non-volatile memory. To purge information stored in the IP module, use the **purge ip** command on page 14-131.

To enable the IP debugging facility, use the **enable ip debug** command on page 14-120. To disable it, use the **disable ip debug** command on page 14-111.

When the debugging facility is enabled, invalid IP packets that are received are stored in a circular buffer for later analysis. The buffer can store up to 40 packets. Subsequent new packets overwrite the oldest existing packets. To examine the buffer, use the **show ip debug** command on page 14-182

To display information about active TCP sessions, including the state and port number, use the **show tcp** command on page 14-218.

If a TCP connection is specified, detailed debugging information for that connection is displayed. To display similar information for active UDP sessions, use the **show ip udp** command on page 14-215.

Ping and Trace Route

The ping (*Packet Internet Groper*) and trace route functions verify connections between networks and network devices.

Ping tests the connectivity between two network devices to determine whether each network device can “see” the other device. The traditional PING command (found on most UNIX systems, for example) can be used only between two systems running the Internet Protocol (IP), and uses ICMP *Echo Request* messages. The router’s extended **ping** command on page 14-129 supports IPv4, IPv6, OSI, IPX, and AppleTalk protocols. Native Echo request packets are sent to the destination addresses and responses are recorded. To initiate the transmission of ping packets, use a **ping** command on page 14-129.

Any parameters not specified use the defaults configured with a previous invocation of the **set ping** command on page 14-163.

As each response packet is received, a message is displayed on the terminal device from which the command was entered and details are recorded. To display default configuration and summary information, use the **show ping** command on page 14-216.

To halt a ping in progress, use the **stop ping** command on page 14-224.

Trace route is used to discover the route used to pass packets between two systems running the IP protocol. It sends UDP packets with the Time To Live (TTL) field in the IP header set starting at 1 and increased by one for each subsequent packet sent until the destination is reached. Each hop along the path responds with a TTL exceeded packet and from this the path can be determined. To initiate a trace route, use the **trace** command on page 14-225.

Any parameters not specified use defaults configured with a previous invocation of the **set trace** command on page 14-165.

As each response packet is received a message is displayed on the terminal device from which the command was entered and the details are recorded. To display default configuration and summary information, use the **show trace** command on page 14-223.

To halt a trace route that is in progress, use the **stop trace** command on page 14-225.

Finger

The finger user information protocol provides a mechanism for exchanging user information between a finger client and a finger server.

A finger client is used to query a finger server for information about a specific user, or to request a list of all logged in users on the server. The information returned depends on the implementation of the finger server. However, in most cases the information returned includes the user's login name, real name, home directory, shell type, login details, and mail status. Other information may also be returned, and signature files may also be appended to the reply. To send a finger query to the finger server on the specified host(s), use the **finger** command on page 14-128.

The response from the finger server is sent to the terminal or telnet session from which the command was entered.

A finger server may also be configured so that when a query is received from a remote client, the finger query initiates a script file on the server that can be set to run system commands. While this opens a significant hole in system security, it makes it possible to control a remote host from anywhere within a network without having to log in to the host. IP filtering and careful selection of the commands that may be run via this method should provide basic security, although users should enable this function only when they are aware of the security implications.

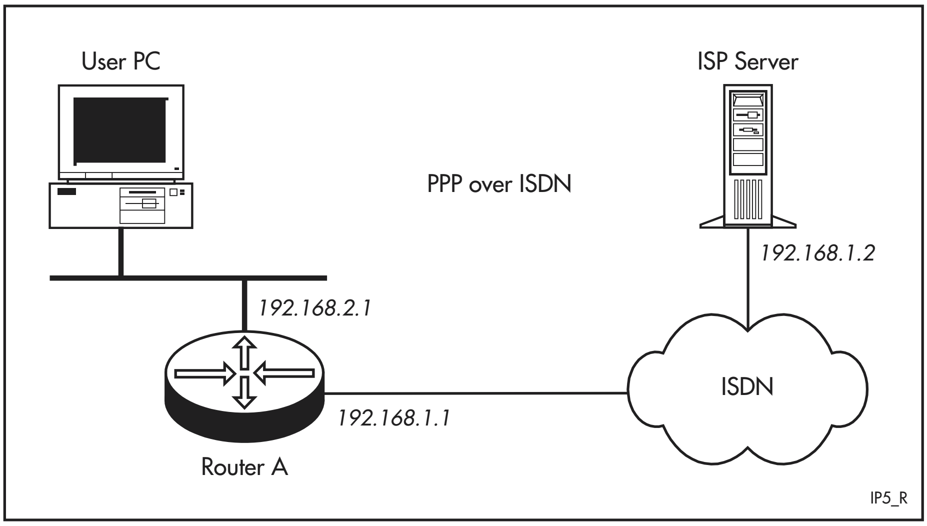
Example

The following example shows how to use the finger client to obtain new mail from a local ISP whenever a link to that ISP is brought up.

In this example, the local ISP provider's mail host is running a finger server configured to respond to a finger query from a subscriber by downloading any new mail for the subscriber's account. Whenever the link to the ISP is brought up, the mail host is automatically polled to see if there is any new mail, without any user intervention.

The user's PC is connected to the LAN. A router on the LAN is used to access the ISP via a Basic Rate ISDN interface (BRI), which brings up the link on demand. On the router, a trigger is created that is activated when the ISDN call comes up and becomes active. The trigger runs a script that sends a finger query to the ISP with the username set to the account name of the subscriber (Figure 14-5 on page 14-40).

Figure 14-5: Using finger to trigger mail downloads from a mail host.



To configure finger to trigger mail downloads from a mail host

1. **Configure the ISDN connection to the ISP.**

   Configure the Q.931 profile for the local territory:

   ```
   set q931=0 profile=nz
   ```

   Create an ISDN call, named "myisp", with ISDN number 123456:

   ```
   add isdn call=myisp number=123456 prec=out outsub=local
       searchsub=local
   ```

   Create a PPP interface to use the ISDN call, and enable the idle timer to make the link a dial-on-demand connection with a timeout period of 60 seconds (the default):

   ```
   create ppp=0 over=isdn-myisp

   set ppp=0 idle=on
   ```

2. **Configure IP.**

   Enable IP and add IP interfaces for the Ethernet LAN and the PPP connection to the ISP:

   ```
   enable ip

   add ip interface=ppp0 IP=192.168.1.1

   add ip interface=eth0 ip=192.168.2.1
   ```

3. **Create a trigger to send the finger query.**

   Create a script file to send the finger command to the ISP:

   ```
   add script=finger.scp text="finger
       myaccountname@192.168.1.2"
   ```

   Create the trigger to activate this script:

   ```
   create trigger=1 interface=ppp0 event=up script=finger.scp
   ```

   When the PPP link is brought up by a subscriber trying to access the ISP from anywhere on the LAN, the router sends a finger query to the ISP, which causes new mail on the server to return to the subscriber.

Security Options

As well as the security features provided by IP traffic filters (see "Traffic Filters" on page 14-35) and restrictions on access to the router's SNMP agent (see "SNMP" on page 14-37), the IP module provides a number of features for securing networks.

To enable source routing of IP packets, use the **enable ip srcroute** command on page 14-127. To disable it, use the **disable ip srcroute** command on page 14-118.

By default, source routing is disabled. Source routing is rarely used for legitimate purposes and is commonly used to circumvent packet-filtering firewalls.

To enable filtering of IP packets with a small fragment offsets or overlapping fragments, use the **enable ip fofilter** command on page 14-122. To disable it, use the **disable ip fofilter** command on page 14-113.

Attacks using tiny or overlapping fragments are designed to foil security schemes based on packet filtering mechanisms. Tiny fragments are too small to contain the full TCP header, making filter pattern matching difficult, while overlapping fragments can be used to 'replace' portions of preceding valid fragments with data that would otherwise be considered 'invalid'.

Security Associations

A Security Association (SA) defines a security transform to be applied to all traffic between any of its members. A security association exists on one or more routers and defines a Virtual Private Network (VPN). A Security Association is identified by its Security Parameters Index (SPI), which must be the same on all instances of the security association throughout the VPN.

The members of an IP security association are IP addresses or contiguous groups of IP addresses. A member is defined by a base IP address and a network mask. A member is local to a router when it is separated from the Internet by the router, otherwise it is remote. All members of a security association are local to one router and remote to the other routers in the VPN.

The IP routing module uses security associations to implement IP payload encryption. AT-VPNet uses hardware encryption resources, security associations and IP payload encryption to create secure virtual private networks across the Internet.

To create a Security Association, use the **create sa** command on page 45-58 of Chapter 45, IP Security (IPsec).

To add members to the security association, specify the member's IP addresses with the **add sa member** command on page 45-46 of Chapter 45, IP Security (IPsec),

The MASK parameter can be used to add a contiguous range of IP addresses as members of the security association. See the **create sa** and **add sa member** commands in Chapter 45, IP Security (IPsec).

To assign an IP interface with zero or more security associations, use the **add ip sa** command on page 14-94.

The IP SA commands provide support for RFCs 1825, 1827, and 1829, which have been superseded by IP Security. See Chapter 45, IP Security (IPsec) and RFCs 240–2412 for more information about IPsec.

To remove a security association from an IP interface, use the **delete ip sa** command on page 14-107.

To display a list of security associations currently assigned to an IP interface, use the **show ip sa** command on page 14-214.

If an IP interface does not have an assigned security association, all IP packets transiting the interface are processed normally by the IP routing module. If an IP interface uses one or more security associations it passes all packets transiting the interface to the SA module. The SA module examines the source and destination addresses of the packet to determine if the packet is in any of the security associations used by the interface. If a match is found, the SA module passes the packet to the ENCO module on the channel to which the security association is attached. In this way the configured transform is applied to the packet. If a packet is not in one of the security associations, it is discarded or sent through the interface depending on the setting of the IP interface's **samode** parameter. To set this parameter, use either the **add ip interface** command on page 14-77 or the **set ip interface** command on page 14-145.

IP datagrams discarded by a security association are logged by the Logging facility with a message type/subtype of IPFIL/SA.

Broadcast Forwarding

The broadcast forwarding facility provides a mechanism for redirecting UDP broadcast packets to other hosts, routers or networks in an internet. A typical example would be the redirection of NETBIOS broadcasts between a Windows NT server on a central head office LAN and Windows NT workstations attached to remote LANs. NETBIOS is just one of a number of UDP broadcast packets that may require forwarding. Others include TFTP, DNS, BOOTP and Time. BOOTP forwarding, defined in RFC 1542, is a special case and is handled separately by the router (see "BOOTP Relay Agent" on page 14-45).

Broadcast forwarding is configured by defining, for each interface, a list of one or more UDP ports to listen on, and the destination IP addresses to which any UDP broadcasts are to be forwarded. By default, broadcast forwarding is disabled. When broadcast forwarding is enabled and configured, UDP listen ports are opened for each of the UDP ports on which UDP broadcasts are to be forwarded. When a UDP broadcast packet is received on an interface for one of the configured ports, it is forwarded to each of the destinations listed for that interface.

Examples

The following examples illustrate two different approaches to configuring broadcast forwarding.

Forwarding to a Unicast Address

In this example, a number of Windows NT workstations on a remote office LAN are attached to a single Windows NT server on the head office LAN (Figure 14-6 on page 14-43, Table 14-7 on page 14-43). Since all broadcasts originate from or are intended for a single NT server on the head office LAN, the destination address for the NETBIOS port is set to the NT server's unicast IP address. In this case, broadcast forwarding needs to be configured on the remote router. This method may become administratively difficult when many destinations on the same network must be specified.

Figure 14-6: Example configuration for broadcast forwarding to a unicast address

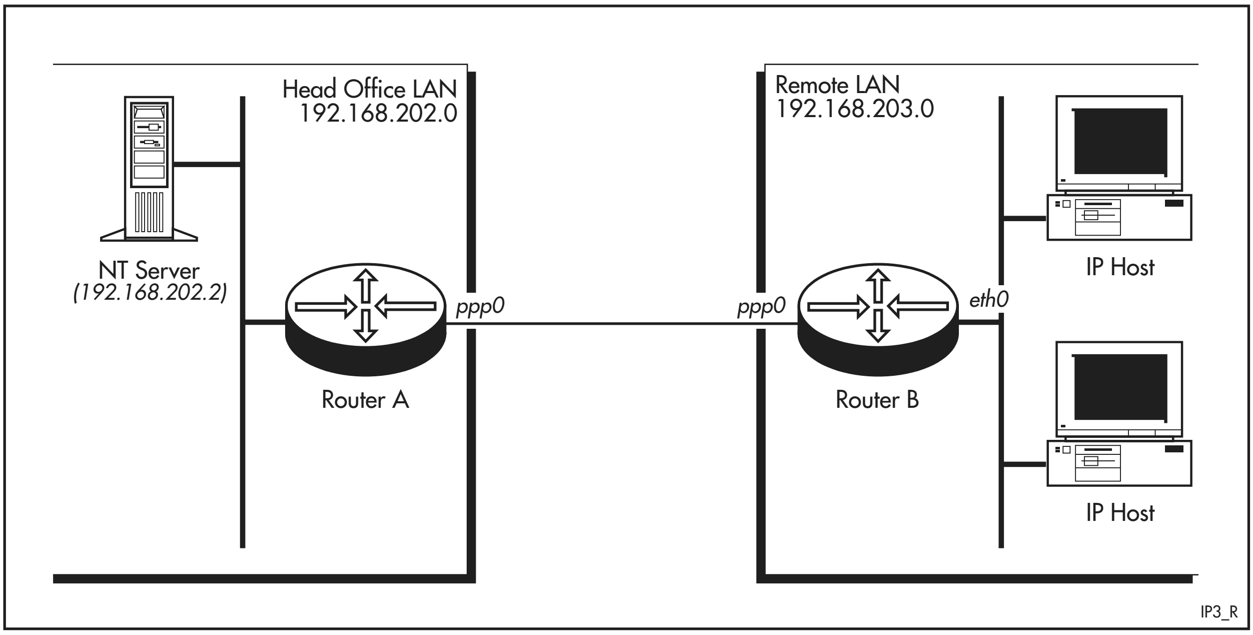


Table 14-7: Example configuration parameters for broadcast forwarding to a unicast address

Parameter	Head Office	Remote Office
Router Name	A	B
IP address of LAN	192.168.202.0	192.168.203.0
Ethernet interface	eth0	eth0
PPP (WAN link) interface	ppp0	ppp0
IP address of NT Server	192.168.202.2	-
UDP protocol to forward	NETBIOS	-

To configure broadcast forwarding to a unicast address

1. **Enable broadcast forwarding.**

```
ENABLE IP
ENABLE IP HELPER
```

2. **Configure the UDP protocols to be forwarded.**

All NETBIOS broadcasts received by Router B’s Ethernet interface are to be forwarded to the NT server with IP address 192.168.202.2.

```
ADD IP INTERFACE=eth0 IPADDRESS=192.168.20.1
ADD IP HELPER PORT=NETBIOS DESTINATION=192.168.202.2
    INTERFACE=eth0
```

Forwarding to a Broadcast Address

In this example, a number of Windows NT workstations on a remote office LAN are attached to several Windows NT servers on the head office LAN (Figure 14-7 on page 14-44, Table 14-8 on page 14-45). Since broadcasts originate from or are intended for several NT servers on the head office LAN, the destination address for the specified port is set to the subnet broadcast address for the head office LAN. In this case, broadcast forwarding must be configured on both the remote router and the head office router. When the head office router receives the UDP packet it re-broadcasts the packet on to the remote LAN. This method has two consequences that need to be considered. First, broadcast traffic increases on the head office LAN. Second, if care is not taken with the configuration, broadcast loops may be created. The broadcast forwarding facility in this mode is acting like a pseudo-bridge, but without the protection of protocols such as Spanning Tree to detect loops.

Figure 14-7: Example configuration for broadcast forwarding to a multicast address

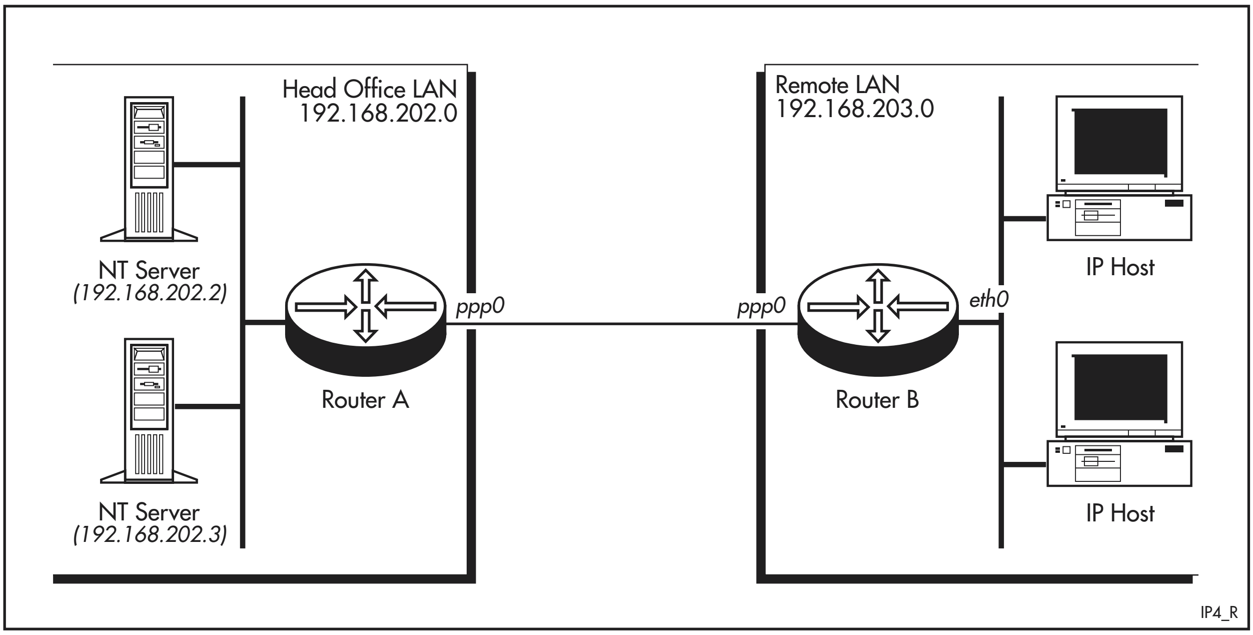


Table 14-8: Example configuration parameters for broadcast forwarding to a multicast address

Parameter	**Head Office**	**Remote Office**
Router Name	A	B
IP address of LAN	192.168.202.0	192.168.203.0
Ethernet interface	eth0	eth0
PPP (WAN link) interface	ppp0	ppp0
IP address of NT Servers	192.168.203.2 192.168.202.3	-
UDP protocol to forward	NETBIOS	-

To configure broadcast forwarding to a multicast address

1. **Enable broadcast forwarding.**

   Enable broadcast forwarding on Router A and Router B, using the following command on each router:

   ```
   enable ip
   enable ip helper
   ```

2. **Configure the UDP protocols to be forwarded.**

   All NETBIOS broadcasts received by the remote router’s Ethernet interface are to be forwarded to the head office LAN with IP address 192.168.202.0. On Router B, use the command:

   ```
   add ip interface=eth0 ipaddress=192.168.203.2
   add ip helper port=netbios destination=192.168.202.255
      interface=eth0
   ```

   All NETBIOS broadcasts received by the head office router’s PPP interface ppp0 are to be broadcast on to the head office LAN. On Router A, use the command:

   ```
   create ppp=0 over=syn0
   add ip interface=ppp0 ipaddress=0.0.0.0
   add ip helper port=netbios destination=192.168.203.255
      interface=PPP0
   ```

BOOTP Relay Agent

BOOTP is a UDP-based protocol that allows a booting host to configure itself dynamically without external interventions. A BOOTP server responds to requests from BOOTP clients for configuration information, such as the IP address the client should use. BOOTP is defined in RFC 951, *Bootstrap Protocol (BOOTP).*

RFC 1542, *Clarifications and Extensions for the Bootstrap Protocol*, defines extensions to the BOOTP protocol, including the behaviour of a BOOTP Relay Agent.

The router’s BOOTP Relay Agent relays BOOTREQUEST messages originating from any of the router’s interfaces to a user-defined destination, and relays BOOTREPLY messages addressed to BOOTP clients on networks directly connected to the router. BOOTREPLY messages addressed to clients on networks not directly connected to the router are ignored by the relay agent and treated as ordinary IP packets for forwarding.

A BOOTREQUEST message may be relayed via unicast, multicast or broadcast methods. In the last case, the message does not re-broadcast to the interface from which it was received. The relay destinations are configured independently of other broadcast forwarders' destinations (e.g. TFTP).

The 'hops' field in a BOOTP message is used to record the number of hops (routers) the message has been through. If the value of the 'hops' field exceeds a predefined threshold (normally 16), the message is discarded by the relay agent. The threshold may be set to a value from 1 to 16.

To enable the BOOTP Relay Agent, use the **enable bootp relay** command on page 14-119.

The agent must currently be disabled. To disable the agent, use the **disable bootp relay** command on page 14-109.

To define a relay destination, use the **add bootp relay** command on page 14-62.

More than one relay destination may be defined, with successive commands. Request messages are relayed to all defined relay destinations so messages may be duplicated.

To delete a relay destination, use the **delete bootp relay** command on page 14-96.

The destination must exactly match a destination previously defined with the **add bootp relay** command on page 14-62.

To purge the BOOTP configuration (including the relay destination list), use the **purge bootp relay** command on page 14-131. The BOOTP module is disabled, all configuration data (including non-volatile storage) is purged, and then BOOTP is re-enabled with default settings.

When the 'hops' field in a BOOTP message exceeds a predefined threshold the BOOTP message is discarded. The default of the threshold is 4. To set the threshold to any value from 1 to 16, use the **set bootp maxhops** command on page 14-133.

To display the current configuration of the BOOTP Relay Agent, use the **show bootp relay** command on page 14-166.

IP Multicasting

IP multicasting, defined in RFC 1112, *Host Extensions for IP Multicasting*, and RFC 1812, *Requirements for IP version 4 Routers*, is the process of transmitting an IP datagram to a group of hosts. A *host group* may contain zero or more hosts. A multicast datagram is delivered to each member of the group as if the datagram had been sent individually to each host as a unicast datagram.

A host group is identified by a single IP address. IP addresses from 224.0.0.0 to 239.255.255.255 are reserved for use as multicast addresses, and each address identifies a host group. The IP address 224.0.0.0 is guaranteed not to be assigned to any host group. The IP address 224.0.0.1 is assigned to the permanent group of all IP hosts and gateways, and is used to address all multicast hosts on the directly connected network. There is no multicast IP address for all hosts on the Internet.

Host groups are dynamic – hosts can join or leave host groups at any time. Any host on the Internet can be a member of any host group, and can be a member of any number of groups at the same time. A host does not need to be a member of a host group to send a multicast datagram to the group.

A multicast datagram can be transmitted to both the local network and all remote networks that are reachable within the IP TTL (time-to-live) value for the datagram. To send an IP multicast datagram, a host transmits the datagram as a local multicast datagram to all members of the host group on the directly connected network. Multicast routers on the local network forward the multicast datagram to all other networks with members in the host group. On the remote destination network, the local multicast router transmits the datagram as a local multicast onto the directly connected network.

Multicasting can be performed dynamically using DVMRP, PIM Sparse Mode or PIM Dense Mode, or statically. Both dynamic and static multicasting use IGMP to manage group membership. IGMP, DVMRP, PIM Sparse Mode and PIM Dense Mode are described in Chapter 17, IP Multicasting.

Static Multicast Forwarding

If neither DVMRP nor PIM are enabled for full dynamic multicasting, IP interfaces can be statically set to receive or send all multicast packets. The default is for all interfaces to receive all multicast packets, but not to forward any. If either DVMRP or PIM are enabled, they dynamically determine the forwarding behaviour of interfaces, and the static multicast setting of an interface is ignored (Chapter 14, IP Multicasting).

The router can be configured to send and receive multicast datagrams; to send only or receive only; or to neither send nor receive them. IP multicasting when the IP interface is created, use the command:

```
add ip interface=interface IPaddress={ipadd|DHCP}
    [MULticast={OFF|SENd|RECeive|BOTH|ON}] [other-options]
```

To configure on an existing IP interface, use the command:

```
set ip interface=interface MULticast={BOTH|OFF|ON|RECeive|
    SENd} [other-options]
```

To display the state of static IP multicasting and counts of multicast packets processed, use the **show ip interface** command on page 14-191.

Static multicast routing is configured on a per-interface basis. All logical IP interfaces on the same IP interface use the same multicast setting, so changing the setting for multicasting on one logical interface affects all other logical interfaces in the IP interface. For multicast datagrams being forwarded by the router, an IP interface with more than one logical interface forwards one multicast datagram out the interface. However, this does not necessarily apply to multicast packets originating from the router. For example, in the case of OSPF on a router with a number of logical interfaces, each Hello packet sent must be sent on a per logical interface basis, since the packet checking code at the destination checks the source address of the packet for a match.

Network Address Translation

Network Address Translation (NAT), defined in RFC 1631, provides a solution to one of the major problems facing the Internet – IP address depletion. NAT allows the reuse of IP addresses by taking advantage of the fact that a small percentage of hosts in a stub domain communicate outside the domain at any one time. A *stub* domain is a domain such as a corporate network that handles traffic originating from or destined for hosts in the domain. Many hosts never communicate outside of their stub domain. Only a subset of the IP addresses inside a stub domain needs to be translated into globally unique IP addresses when outside communication is required.

NAT allows IP addresses inside a stub domain to be reused by other stub domains, meaning they need not be unique. For instance, a single Class A address could be used by a stub domain. RFC 1597 specifies a list of network addresses that are reserved for such a purpose. NAT is configured at each exit point router between a stub domain and the Internet backbone. When there is more than one exit point, each NAT entity must use the same translation table. The router at the exit point is configured with a group of IP addresses that are globally unique on the Internet. The source IP address of a packet received at the exit point is translated to use one of the globally unique addresses, and then forwarded to the Internet.

Enhanced NAT (ENAT) is a variation that uses a single global Internet IP address. Each new session crossing the exit point router is assigned a set of unique TCP/UDP port numbers. For example, consider a TELNET packet sent from a private network to a host on the Internet. The source IP address is changed to the single global IP address and the source TCP port number is substituted for one that is unique. The router maintains a list of current sessions using the IP source address, original source port, substituted port, destination port, and destination IP address information. The advantage of this scheme is that only a single IP address is required from an Internet Service Provider (ISP) to connect an entire private network. The private network can easily be shifted to another ISP simply by changing the one global IP address.

A disadvantage of NAT and ENAT is that both require intervention into the IP packets themselves. In particular, the source or destination address must be modified for each packet traversing the NAT entity. This means patching the IP checksum and also their pseudo checksums when TCP or UDP traffic is being forwarded. Some IP protocols embed IP information into the data stream that also requires patching. As a result, only payload encryption is possible on such links.

However, a major benefit of NAT and ENAT technologies is greatly enhanced security. Many firewalls use ENAT technology.

The term *private* network refers to stub domains, as they are typically assigned IP addresses using RFC 1597. Traffic or routing information from these networks should never be sent directly to the Internet. Officially assigned IP addresses are globally unique. The terms *global interface* and *global IP address* specify the interface and officially assigned IP address for that interface, which connects the private network to the Internet.

The router supports the following NAT methodologies.

Methodology	Description
Static NAT	One fixed IP address is translated to one fixed global address. The benefit of this scheme is that all the hosts on a private network can be numbered using RFC 1597, but certain servers' IP addresses can be statically translated to external global IP addresses, when a session traverses the router running NAT. All possible TCP/UDP and ICMP sessions and flows to the specified host can originate from the global Internet.
Dynamic NAT	Any host from a specified range of hosts can use a pool of global IP addresses to connect to the global Internet. Addresses are allocated on a first come first served basis. When no sessions or flows for a particular private IP-to-global-IP translation remain, the global IP address is returned to the pool for re-use.
Dynamic ENAT	A single global Internet IP address is required. Both the private IP address and protocol dependent port numbers are translated. The router maintains a list of all currently used port numbers and allocates a unique number for each new session or flow being translated. Sessions and flows can originate from the private network.
Static ENAT	A private IP address and port number are mapped to a global IP address and port number. This method allows a server on the private network to be made available to the global Internet. The server still remains very secure since sessions or flows to the specified port can originate from the global Internet.
Interface ENAT	A variation of dynamic ENAT in which the global IP address for translation is determined dynamically from a specified interface. This method allows a private network to use ENAT translation on a router with a PPP link to an ISP who dynamically issues global IP addresses. Sessions and flows can originate from the private network.

By default, NAT is disabled on the router. NAT must be explicitly enabled with the **enable ip nat** command on page 14-125.

NAT is automatically disabled when the firewall is enabled because the firewall provides NAT services. However, the NAT configuration is retained so that it can be (manually) re-enabled if the firewall is disabled.

To add translations to the network address translation table, use the **add ip nat** command on page 14-83.

The state of TCP sessions using the NAT gateway is maintained. The router monitors the flag bits of TCP packets passing through the NAT gateway. When the NAT gateway detects a TCP session has closed, the entry is deleted after a timeout period.

Handling UDP packets is based on the concept of a *flow*. Although UDP is a connectionless protocol, typically when two hosts communicate using UDP, there is an exchange of information from one UDP port on one host to another UDP port on the other host. This traffic can be thought of as a flow because the port numbers do not change during this exchange of information. When a flow starts, a timer is started. Each received packet resets the timer. If the timer expires, the flow entry is deleted.

The method for handling ICMP messages is dependent on the NAT method that is used. With static or dynamic NAT, IP addresses in the messages are translated as expected. With ENAT, the use of a single global IP address necessitates special handling of ICMP messages. ICMP messages can be categorised as follows:

- ICMP Destination Unreachable, Time Exceeded, Parameter Problem, and Source Quench packets sent to the global IP address are handled by matching the returned 64 bytes of header to a current session or flow. If no match is found the message is silently discarded. Outbound ICMP messages are fixed as normal with the source IP address being replaced with the global address.
- ICMP Redirect messages are ignored as they have no meaning across a NAT gateway.
- ICMP echo requests and Timestamps are handled slightly differently. An entry is created for the packet and a timer started. The sequence number is replaced with a unique port number, the original sequence number stored, and the fixed up packet forwarded on. When a reply is returned it is matched, fixed up and returned to the sender. The entry is then discarded. If no reply is received then the entry is deleted when the timer expires.

The global IP address for a PPP interface can be assigned dynamically. This is the case for a router connected to an ISP who dynamically assigns IP addresses. If an interface ENAT has been created over the PPP interface, NAT monitors all PPP interface state changes. If the global IP address associated with a PPP interface is marked as “dynamically set”, then if the PPP interface is down and a packet is ready to be transmitted, NAT calls PPP to bring up the interface and queue the packet for later handling. When the PPP interface comes up, NAT determines the current IP address for the global interface, and based on that, forwards the stored packets. Note, if the global interface is marked as “dynamically set” and the interface goes down, then all current sessions or flows are deleted as they are no longer valid.

In this implementation all received traffic on the global interface is dropped (and possibly logged) unless either the connection or flow was initiated from the private network, or a specific configuration entry exists to explicitly allow the traffic.

Remote Address Assignment

The remote IP address assignment facility enables unnumbered PPP interfaces (such as PPP interfaces with an IP address of 0.0.0.0) to be dynamically assigned an IP address during the PPP link's negotiation process.

If a PPP interface is created with an IP address of 0.0.0.0, and remote IP address assignment is enabled, during the IP control protocol (IPCP) negotiation process the router allows the remote PPP peer to set the IP address of the local PPP interface.

If the local PPP interface has an IP number other than 0.0.0.0, or if remote IP address assignment is disabled, the router does not allow the remote PPP peer to set the IP address of the local PPP interface.

To enable a remote IP address assignment, use the **enable ip remoteassign** command on page 14-126. To disable one, use the **disable ip remoteassign** command on page 14-117. The current status of the remote IP assignment option is displayed in the output of the **show ip** command on page 14-168.

IP Address Pools

An IP address pool is a named collection of IP addresses that ACC, PPP and other modules can use to assign IP addresses to dynamic connections. The advantage of an address pool is that a finite number of IP addresses can be re-used by many clients. When a client is finished with the IP address (for example, when a dial-in SLIP connection terminates) the IP address is returned to the pool and is available for another client to use.

The router supports multiple methods for assigning IP addresses to dynamic dial-in calls. The following procedure is used to select the IP address assigned to a dial-in call:

1. If the user is authenticated via RADIUS, and the RADIUS response supplies an IP address, then that IP address is used.
2. If the user is authenticated using TACACS and an ACC domain name has been specified with the **add acc domainname** command on page 28-19 of Chapter 28, Asynchronous Call Control then the domain name is appended to the login name and a Domain Name Service (DNS) request is issued to resolve the name to an IP address.
3. If the user is authenticated using TACACS and an ISDN domain name has been specified with the **add isdn domainname** command on page 11-67 of Chapter 11, Integrated Services Digital Network (ISDN), then the domain name is appended to the login name and a Domain Name Service (DNS) request is issued to resolve the name to an IP address.
4. If the user is authenticated via the router's internal User Authentication Database, and an IP address is set in the User Authentication Database for that user, then that IP address is used.
5. If the call is an ACC call on an asynchronous port with an IP address set, then that IP address is used.
6. If the ACC or PPP call has an IP pool set, and the request to the IP pool is successful, then that IP address is used.

To create an IP address pool, use the **create ip pool** command on page 14-96.

To destroy an IP address pool, use the **destroy ip pool** command on page 14-109.

To display the currently configured IP address pools, and the status of the IP addresses in the pools, use the **show ip pool** command on page 14-199.

To associate an IP address pool with an ACC call so that SLIP dial-in connections using that call uses IP addresses from the IP address pool, use either of the commands:

```
add acc call=call-name ippool=pool-name
    [other-acc-options...]

set acc call=call-name ippool=pool-name
    [other-acc-options...]
```

To disassociate an IP address pool from an ACC call so that dial-in connections using that call no longer use IP addresses from the IP address pool, use the command:

```
set acc call=call-name ippool=none
```

To associate an IP address pool with a PPP interface so that connections using that interface use IP addresses from the IP address pool, use either of the commands:

```
create ppp=ppp-interface over=physical-interface
    ippool=pool-name [other-ppp-options...]

set ppp=ppp-interface ippool=pool-name [other-ppp-options...]
```

To disassociate an IP address pool from a PPP interface so that connections using that interface no longer use IP addresses from the IP address pool, use the command:

```
set ppp=ppp-interface ippool=none
```

To associate an IP address pool with a PPP template so that dynamic PPP interfaces created using the PPP template use IP addresses from the IP address pool, use either of the commands:

```
create ppp template=template ippool=pool-name
    [other-template-options...]

set ppp template=template ippool=pool-name
    [other-template-options...]
```

To disassociate an IP address pool from a PPP template so that dynamic PPP interfaces created using the PPP template no longer use IP addresses from the IP address pool, use the command:

```
set ppp template=template ippool=none
```

Configuration Examples

The following examples shows how to configure IP on the router. The first example shows how to configure basic IP routing. The second example shows how to configure IP filtering on the router to perform firewall functions.

A Basic TCP/IP Setup

In this example, two routers are to be connected. Each acts as a router rather than just a Telnet server. The routers are connected to each other using the Point-to-Point Protocol (PPP) over a wide area data communications link. Each router has a single Ethernet LAN segment attached, on which are located local hosts and PCs shown in the following figure:

Figure 14-8: Example configuration for a basic TCP/IP network

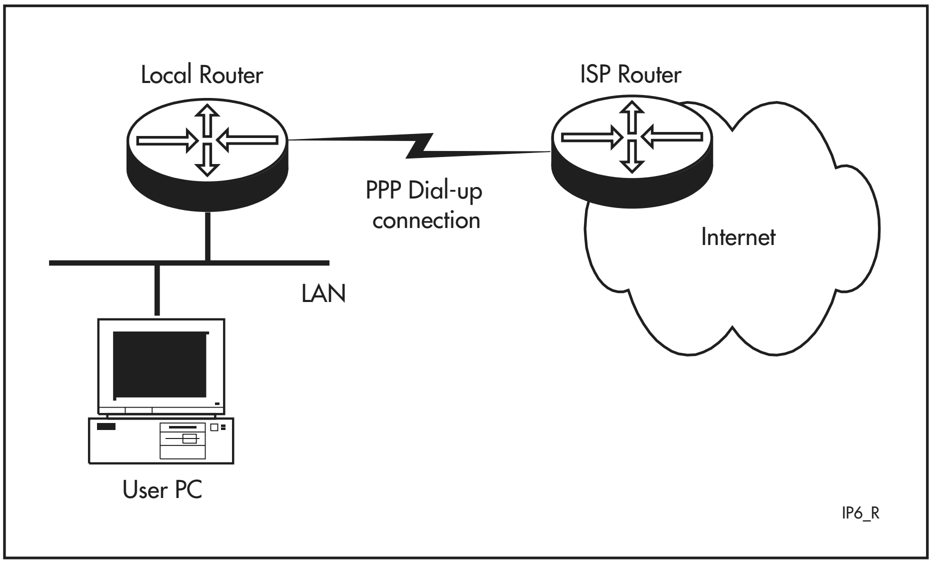


Table 14-9: Example configuration parameters for a basic TCP/IP network

Parameter	Router A	Router B
LAN IP subnet address	172.16.8.0	192.168.31.16
LAN network class	B	C
LAN number of subnet bits	8	4
LAN IP network mask	255.255.255.0	255.255.255.240
Ethernet IP address	172.16.8.33	192.168.31.30
Synchronous port	0	0
PPP interface	0	0
PPP IP subnet address	172.16.254.0	172.16.254.0
PPP interface IP address	172.16.254.1	172.16.254.2

To configure a basic IP network

1. **Configure the PPP Link.**

   See "The Command Processor" on page 1-5 of Chapter 1, Operation for a step-by-step example of how to establish MANAGER level access. Use the following command on each router:

   ```
   create ppp=0 over=syn0
   ```

2. **Initialise and enable the IP routing module.**

   Use the following commands on both routers to initialise the IP routing database and enable the IP routing module. The **purge ip** command disables the IP routing module, so the module must be explicitly enabled:

   ```
   purge ip
   enable ip
   ```

3. **Add interfaces to the IP routing module.**

   The interfaces must now be assigned to the IP routing module. Use the following commands on Router A:

   ```
   add ip interface=eth0 ip=172.16.8.33 mask=255.255.255.0
   add ip interface=ppp0 ip=172.16.254.1 mask=255.255.255.0
   ```

   The IP routing module on Router B must now be configured, using a similar sequence of commands. The main difference is that Router B has a Class C network on the Ethernet interface. This requires a different network mask. Use the following commands for Router B:

   ```
   add ip interface=eth0 IP=192.168.31.30
      mask=255.255.255.240
   add ip interface=ppp0 IP=172.16.254.2 mask=255.255.255.0
   ```

   The metrics for the interfaces defaults to 1. The IP module is now enabled, linked to the physical interfaces, and operational. By default, the router does not receive or transmit route information until it has been configured to use a routing protocol. For this example assume RIP is used.

4. **Configure RIP as the routing protocol.**

   A routing protocol must now be enabled to allow the routers to communicate and to update the internal routing tables. For this example RIP is used. This is to be broadcast onto the Ethernet LAN, but is to be directed explicitly to each end of the PPP link. For Router A, use the following commands:

   ```
   add ip rip interface=eth0
   add ip rip interface=ppp0
   show ip rip
   ```

   Specifying only the interface causes RIP to be broadcast to the whole network or subnet.

   For Router B use:

   ```
   add ip rip interface=eth0
   add ip rip interface=ppp0
   show ip rip
   ```

   The router configuration is now complete.

To test the configuration

1. **Check the operational mode of the IP routing module.**

   The IP module operates in one of two modes, SERVER or FORWARDING. In SERVER mode, the router does not route IP packets, but provides Telnet services, responds to SNMP requests, and uses TFTP to download software upgrades. In FORWARDING mode, the router routes IP packets, as well as performing all the functions of SERVER mode. The default operational mode is FORWARDING. To examine the current setting, use the command:

   ```
   show ip
   ```

   This command displays the general status of the IP module. To change the operational mode, use the commands:

   ```
   disable ip forwarding
   enable ip forwarding
   ```

   For more information on using the router as a Telnet server, see Chapter 21, Terminal Server.

2. **Check the routes.**

   Provided the interfaces are connected to other systems acting as routers, the router obtains IP routes after a short period (up to 60 seconds). These routes show the network from the point of view of the router. The route table can be checked to verify the correct operation of the IP module using the following command on either router:

   ```
   show ip route
   ```

   The display (on Router A) should be similar to the following output.

```
IP Routes
--------------------------------------------------------------------------------
Destination       Mask                NextHop              Interface         Age
DLCI/Circ.        Type      Policy    Protocol             Metrics    Preference
--------------------------------------------------------------------------------
172.16.8.0        255.255.255.0       0.0.0.0              eth0             8372
-                 direct    0         static               1                 100
172.16.254.0      255.255.255.0       0.0.0.0              ppp0             8372
-                 direct    0         static               1                 100
192.168.31.16     255.255.255.240     172.16.254.2         ppp0             8369
-                 remote    0         rip                  2                 100
--------------------------------------------------------------------------------
```

The route table should contain easily verifiable data and should indicate that this router can communicate with other router systems. The **ping** command on page 14-129 (common to most TCP/IP implementations) can be used on a host to test that paths to remote hosts are available through the router.

The router’s **ping** command can be used to verify that hosts respond on both links:

```
ping ipadd
```

or:

```
ping nickname
```

if *nickname* has been added to the host name table using the **add ip host** command on page 14-76. ICMP echo request packets are sent to the host IP address and the response time for each is listed when the command is successful.

3. **Check the ARP cache.**

   The ARP cache starts to show binding information (especially from the LAN link) for each active host on the links. The ARP cache can be checked using the command:

   ```
   show ip arp
   ```

   The router should have entries for some known hosts in the ARP cache. This means that it communicates correctly with these hosts.

   Entries appear only in the ARP cache when a local host attempts to access a host on another subnet or when it uses a protocol like BOOTP. It is easy to force this by attempting to ping a host on another subnet from a local host.

4. **Try using Telnet to access the remote router.**

   To Telnet from Router A to Router B, on Router A use the command:

   ```
   telnet 192.168.31.30
   ```

   To Telnet from Router B to Router A, on Router B use the command:

   ```
   telnet 172.16.8.33
   ```

   You can use any of the assigned interface IP addresses as the target for a Telnet access.

Troubleshooting

No Route Exists to the Remote Router

1. Wait for at least one minute to ensure that a RIP update has been received.
2. Repeat step 4. Check that the link is OPENED for both LCP and IP by typing:

   ```
   show ppp
   ```

   The display should be similar to the following output.

```
Name            Enabled  ifIndex  Over                  CP           State
------------------------------------------------------------------------------
ppp0              YES      04                           IPCP         OPENED
                                  syn0                  LCP          OPENED
------------------------------------------------------------------------------
```

   See Chapter 9, Point-to-Point Protocol (PPP) for further details on how to check the PPP link.

3. Try restarting the IP routing module (a warm restart), by typing:

   ```
   reset ip
   ```

   If the route still does not appear, contact your authorised distributor or reseller for assistance.

Telnet Fails

1. If Telnet into the remote router fails, check that the IP address being used matches the one assigned to this router. Check that RIP is configured correctly (step 4).

2. If Telnet into a host on the remote LAN fails, but works into the remote router, check the IP address you are using is correct. Check that both routers are gateways, not servers by typing:

   ```
   show ip
   ```

   The "IP Packet Forwarding" entry in the output should be enabled.

3. Ensure that the remote host is running a Telnet daemon and is correctly configured. Check that RIP is being broadcast (i.e. to '.255') on the remote LAN by typing (on the remote router):

   ```
   show ip rip
   ```

   The display on Router A should be similar to the following output.

```
Interface  Circuit/DLCI    IP Address     Send  Receive  Demand  Auth  Password
--------------------------------------------------------------------------------
eth0       -               -              COMP  BOTH     NO      NO
ppp0       -               172.16.249.34  RIP1  RIP2     YES     NO
ppp1                       172.16.250.2   RIP2  NONE     YES     NO
--------------------------------------------------------------------------------
```

4. Check that the ARP cache on the remote router contains an entry for the remote host. This indicates that the host has been active. Use the command:

   ```
   show ip arp
   ```

5. Check that a route exists to the subnet that the target host is on, with:

   ```
   show ip route
   ```

   For sites with multiple subnets on a single LAN, static routes may be required.

6. Try using the **ping** command on page 14-129 on the remote router to check that the host responds. For example, type:

   ```
   ping 172.16.8.2
   ```

   The response should be similar to the following output.

```
Echo reply 1 from 172.16.8.2 time delay 20 ms

Echo reply 2 from 172.16.8.2 time delay 40 ms

Echo reply 3 from 172.16.8.2 time delay 0 ms

Echo reply 4 from 172.16.8.2 time delay 0 ms

Echo reply 5 from 172.16.8.2 time delay 60 ms
```

7. Contact your authorised distributor or reseller for assistance.

Configuring IP Filters

With the increase in connections to the Internet, and the interconnection of networks from different organisations, filtering data packets is an important mechanism to ensure that only legitimate connections are allowed. Security can never be perfect while connections to other networks exist, but filters allow network managers to manage the permissible free access, while restricting users without permission.

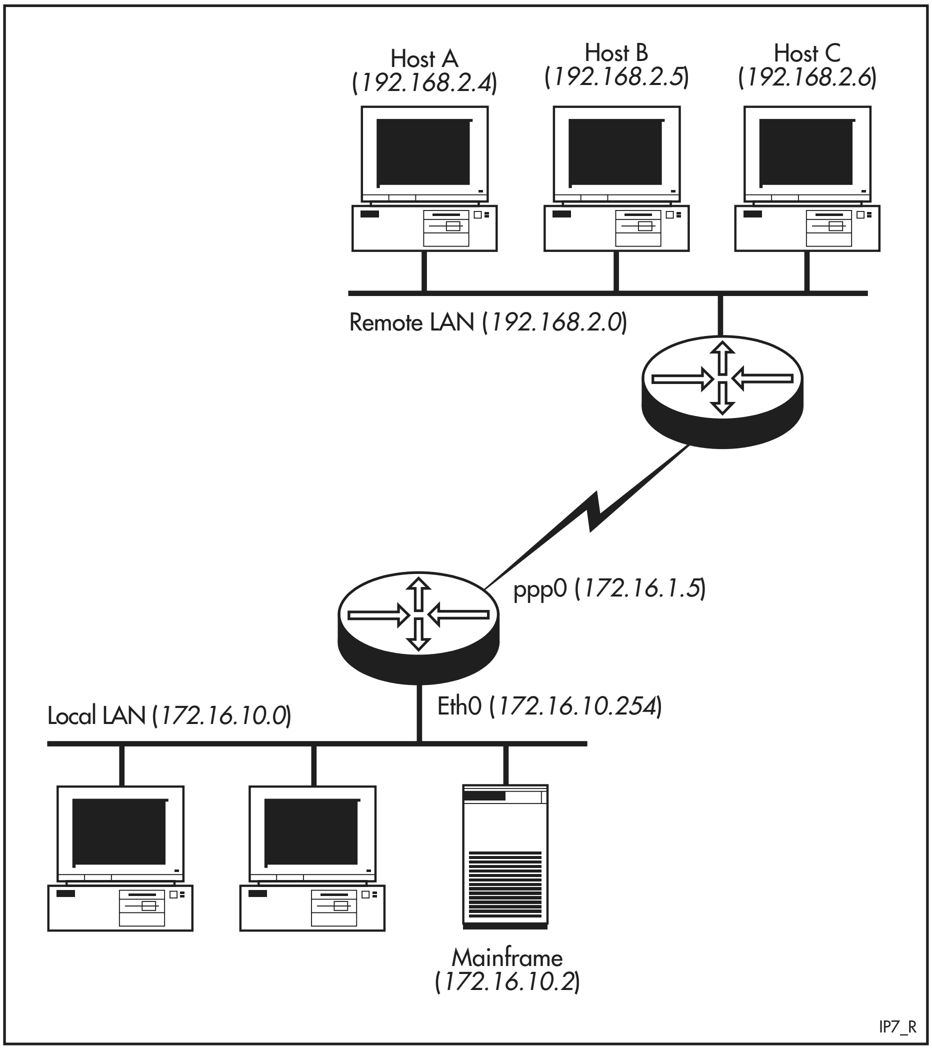
The router has firewall functionality and can restrict traffic on the basis of source/destination IP address, source/destination ports, IP protocol type and TCP flags. The choice of filters depends on an organisation's particular requirements. However, extensive filtering and large filter lists reduce the performance of the router, so filtering design needs to ensure that lists are simple, but effective.

In this example, an organisation wishes to allow access to its mainframe for users from another organisation. Access from the remote network is controlled by filters defined on the local router (Figure 14-9 on page 14-59). On the remote network there are three hosts. Host A can connect via Telnet to the mainframe. Host B can connect via Telnet and FTP. Host C can connect via FTP. Table 14-10 on page 14-58 lists parameter values used in the example. A static route exists for the PPP link between the local and remote routers.

Table 14-10: Example configuration parameters for IP filtering

Site	**Local LAN**	**Remote LAN**
LAN subnet	172.16.10.0	192.168.2.0
LAN network mask	255.255.255.0	255.255.255.0
Eth0 interface IP address	172.16.10.254	-
ppp0 interface IP address	172.16.1.5	-
ppp0 network mask	255.255.255.0	255.255.255.0
Mainframe IP address	172.16.10.2	-
Remote Host A IP address	-	192.168.2.4
Remote Host B IP address	-	192.168.2.5
Remote Host C IP address	-	192.168.2.6

Figure 14-9: Example configuration for IP filtering



To configure IP filters

1. **Create a filter to control the access of hosts A, B and C to the mainframe.**

   Create filter 1 for interface ppp0 to control the access of hosts A, B and C on the remote network to the mainframe on the local network. To enable Telnet connections from host A, use the command:

```
enable ip
add ip filter=1 so=192.168.2.4 sm=255.255.255.255
   destination=172.16.10.2 dm=255.255.255.255
   dport=telnet protocol=tcp sess=any action=include
```

   To enable Telnet and FTP access from host B, use the commands:

```
add ip filter=1 so=192.168.2.5 sm=255.255.255.255
   destination=172.16.10.2 dm=255.255.255.255 dp=ftpdata
   protocol=tcp sess=esta action=include
add ip filter=1 so=192.168.2.5 sm=255.255.255.255
   destination=172.16.10.2 dm=255.255.255.255 dp=ftp
   protocol=tcp sess=any action=include
add ip filter=1 so=192.168.2.5 sm=255.255.255.255
   destination=172.16.10.2 dm=255.255.255.255 dp=telnet
   protocol=tcp sess=any action=include
```

To enable FTP access from host C, use the commands:

```
add ip filter=1 so=192.168.2.6 sm=255.255.255.255
   destination=172.16.10.2 dm=255.255.255.255 dp=ftp
   protocol=tcp sess=esta action=include
add ip filter=1 so=192.168.2.6 sm=255.255.255.255
   destination=172.16.10.2 dm=255.255.255.255 dp=ftpdata
   protocol=tcp sess=esta action=include
```

The last entry in a filter is always an implicit entry (one which you do not have to enter) to exclude all sources, destinations, and ports. It is equivalent to the command:

```
add ip filter=1 so=0.0.0.0 smask=0.0.0.0
   destination=0.0.0.0. dmask=0.0.0.0 sport=all
   action=exclude
```

2. **Create a filter to allow replies from the mainframe to reach hosts A, B and C.**

   Create filter 2 for interface eth0 to allow the replies from the mainframe to remote hosts A, B and C, but prevent other users on the local network from accessing remote hosts A, B and C:

```
add ip filter=2 so=172.16.10.2 sm=255.255.255.255
   sp=telnet destination=192.168.2.4 dm=255.255.255.255
   protocol=tcp sess=esta action=include
add ip filter=2 so=172.16.10.2 sm=255.255.255.255
   sp=telnet destination=192.168.2.5 dm=255.255.255.255
   protocol=tcp sess=esta action=include
add ip filter=2 so=172.16.10.2 sm=255.255.255.255
   sp=ftpdata destination=192.168.2.5 dm=255.255.255.255
   protocol=tcp sess=any action=include
add ip filter=2 so=172.16.10.2 sm=255.255.255.255 sp=ftp
   destination=192.168.2.5 dm=255.255.255.255
   protocol=tcp sess=esta action=include
add ip filter=2 so=172.16.10.2 sm=255.255.255.255
   sp=ftpdata destination=192.168.2.65 dm=255.255.255.255
   protocol=tcp sess=any action=include
add ip filter=2 so=172.16.10.2 sm=255.255.255.255 sp=ftp
   destination=192.168.2.6 dm=255.255.255.255
   protocol=tcp sess=esta action=include
```

   The explicit exclusion is not required. Other hosts on the local network are not able to communicate with hosts on the remote network.

3. **Add the filters to the interfaces.**

   The filters that have been defined must be assigned to interfaces in order for them to take affect. Assign filter 1 to interface ppp0 and filter 2 to interface eth0, using the commands:

```
create ppp=0 over=syn0
add ip interface=ppp0 ip=172.16.10.54 mask=255.255.255.0
   filter=1
add ip interface=eth0 ip=172.16.1.5 mask=255.255.255.0
   filter=2
```

4. **Test the configuration.**

   The definitions of the filters can be checked with the command:

```
show ip filter
```

   This command produces output similar to Figure 14-10 on page 14-61.

   To display details of the IP interfaces defined, including the filter assigned to each interface (Figure 14-11 on page 14-61), use the command:

```
show ip interface
```

Figure 14-10: Example output from the **show ip filter** command for IP filtering

```
IP Filters
--------------------------------------------------------------------------------
No. Ent. Source Port   Source Address    Source Mask      Session            Size
         Dest. Port    Dest. Address     Dest. Mask       Prot.(C/T)      Options
         Type          Act/Pol/Pri       Logging                          Matches
--------------------------------------------------------------------------------
 1    1  Any           192.168.2.4       255.255.255.255  Start               Any
         23:23         172.16.10.2       255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      2  Any           192.168.2.5       255.255.255.255  Established         Any
         20:20         172.16.10.2       255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      3  Any           192.168.2.5       255.255.255.255  Any                 Any
         21:21         172.16.10.2       255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      4  Any           192.168.2.5       255.255.255.255  Start               Any
         23:23         172.16.10.2       255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      5  Any           192.168.2.6       255.255.255.255  Start               Any
         21:21         172.16.10.2       255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      6  Any           192.168.2.6       255.255.255.255  Established         Any
         20:20         172.16.10.2       255.255.255.255  TCP                 Any
         General       Include           Off                                    0

     Requests: 0            Passes: 0            Fails: 0
--------------------------------------------------------------------------------
 2    1  23:23         172.16.10.2       255.255.255.255  Established         Any
         Any           192.168.2.4       255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      2  23:23         172.16.10.2       255.255.255.255  Established         Any
         Any           192.168.2.5       255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      3  20:20         172.16.10.2       255.255.255.255  Any                 Any
         Any           192.168.2.5       255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      4  21:21         172.16.10.2       255.255.255.255  Established         Any
         Any           192.168.2.5       255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      5  20:20         172.16.10.2       255.255.255.255  Any                 Any
         Any           192.168.2.65      255.255.255.255  TCP                 Any
         General       Include           Off                                    0
      6  21:21         172.16.10.2       255.255.255.255  Established         Any
         Any           192.168.2.6       255.255.255.255  TCP                 Any
         General       Include           Off                                    0

     Requests: 0            Passes: 0            Fails: 0
--------------------------------------------------------------------------------
```

Figure 14-11: Example output from the **show ip interface** command for IP filtering

```
Interface     Type     IP Address       Bc Fr PArp  Filt RIP Met.   SAMode  IPSc
Pri. Filt     Pol.Filt Network Mask     MTU   VJC   GRE  OSPF Met.  DBcast  Mul.
--------------------------------------------------------------------------------
eth0          Static   172.16.10.254    1  n  On    002  01         Pass    --
---           ---      255.255.255.0    1500  -     ---  0000000000 No      ---
ppp0          Static   172.16.1.5       1  n  -     001  01         Pass    --
---           ---      255.255.255.0    1500  Off   ---  0000000001 No      ---
--------------------------------------------------------------------------------
```

Command Reference

This section describes the commands available on the router to configure and manage the IP routing module. The extended **ping** command on page 14-129 requires that the IPX and AppleTalk routing modules be enabled and correctly configured if IPX or AppleTalk addresses are used. See Chapter 19, Novell IPX and Chapter 31, AppleTalk for a detailed description of the commands required to enable and configure IPX or AppleTalk routing.

Some interface and port types mentioned in this chapter may not be supported on your router. The interface and port types that are available vary depending on your product's model, and whether an expansion unit (PIC, NSM) is installed. For more information, see the *AR400 Series Router Hardware Reference*.

The shortest valid command is denoted by capital letters in the Syntax section. See "Conventions" on page xcv of Preface in the front of this manual for details of the conventions used to describe command syntax. See Appendix A, Messages for a complete list of error messages and their meanings.

add bootp relay

Syntax `ADD BOOTp RELAy=ipadd`

where *ipadd* is an IP address in dotted decimal notation

Description This command adds a BOOTP relay destination. The **relay** parameter specifies the IP address of a BOOTP server in dotted decimal notation. Up to 50 relay destinations can be defined, using successive commands. BOOTP request messages are relayed to all defined relay destinations, so messages may be duplicated.

Examples To add the BOOTP server with IP address 192.168.13.11, use:

```
add boot rela=192.168.13.11
```

Related Commands

delete bootp relay
disable bootp relay
enable bootp relay
purge bootp relay
set bootp maxhops
show bootp relay

add ip advertise interface

Syntax

```
ADD IP ADVertise INTerface=interface
    [ADVertisementaddress={ALL|LIMited}]
    [MAXadvertisementinterval=4..1800]
    [MINadvertisementinterval=3..MAXadvertisementinterval]
    [LIFetime=MAXadvertisementinterval..9000]
```

where *interface* is an interface name formed by concatenating an interface type and an interface instance (e.g. vlan1).

Description

This command adds ICMP Router Discovery advertising to a single physical IP interface. The interface sends router advertisements when it has been globally enabled with the **enable ip advertise** command.

The **advertisementinterval** parameter specifies the IP destination address to be used for multicast advertisements sent from the interface. If **all** is specified, the destination is the all-systems multicast address, 224.0.0.1. If **limited** is specified, the destination is the limited-broadcast address, 255.255.255.255. The default is **all**.

The **maxadvertisementinterval** parameter specifies the maximum time between sending multicast advertisements from the interface. The default is 600 seconds.

The **minadvertisementinterval** parameter specifies the minimum time between sending multicast advertisements from the interface. The default is 450 seconds.

The **lifetime** parameter specifies the maximum length of time that the advertised addresses are to be considered as valid router addresses by hosts. The default is 1800 seconds.

If you change the advertising intervals, keep these proportions:
lifetime=3 x maxadvertisementinterval
minadvertisementinterval=0.75 x maxadvertisementinterval

Examples

To add Router Discovery advertising to VLAN2, modify the default advertisement address to the more limited broadcast address 255.255.255.255, and to modify the maxadvertisement interval to 1000 seconds, use the command:

```
add ip adv int=VLAN2 adv=lim max=1000 min=750 lif=3000
```

Related Commands

add ip interface
delete ip advertise interface
disable ip advertise
enable ip advertise
set ip advertise interface
set ip interface
show ip advertise

add ip arp

Syntax

```
ADD IP ARP=ipadd INTerface=interface
    [{CIRCuit=miox-circuit|DLCI=dlci|ETHernet=macadd|
    [POrt=port-number]}]
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit).
- *macadd* is the physical Ethernet (MAC) address of a host.
- *port-number* is the physical switch port number. Port numbers start at 1 and end at m, where m is the highest numbered Ethernet switch port, including uplink ports.

Description

This command adds a static ARP entry to the ARP cache. This is typically used to add entries for hosts that do not support ARP or to speed up the address resolution function for a host. The ARP entry must not already exist.

The ARP parameter specifies the IP address of the host.

The **interface** parameter specifies the interface over which the host can be reached. The specified interface must already exist. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces.

If the interface type is ETH, ER, VLAN, or X.25 DTE, then one of the following parameters must be present on the command line. If the interface type is ATM or PPP, the following parameters do not apply and must not be present. If the interface type is ATM inverse ARP must be turned off for the interface. This can be done with the **add ip interface** command on page 14-77 or the **set ip interface** command on page 14-145.

The **circuit** parameter specifies the MIOX circuit on an X.25 interface.

The **dlci** parameter specifies the physical address for the host on a Frame Relay interface.

The **ethernet** parameter specifies the physical (MAC) address for the host on the following:

- ETH interfaces
- VLAN interfaces

The **port** parameter specifies the physical switch port number in a VLAN. If the **interface** parameter specifies a VLAN interface, both the **ethernet** and **port** parameters are required. Otherwise, the **port** parameter is invalid. When configuring VLAN interfaces, both the **ethernet** and **port** parameters must be used.

Examples To add a static ARP entry for a host with an Ethernet address of 00-00-00-08-31-9F and an IP address of 172.16.9.197 on interface eth0, use:

```
add ip arp=172.16.9.197 int=eth0 eth=00-00-00-08-31-9F
```

To add a static ARP entry for a Frame Relay station with an IP address of 172.16.249.5 at the remote end of DLC 23, use:

```
add ip arp=172.16.249.5 int=fr0 dlci=23
```

Related Commands **delete ip arp**
set ip arp
show ip arp

add ip dns

Syntax

```
ADD IP DNS [DOMain={ANY|domain-name}]
   {INTerface=interface|PRIMary=ipadd [SECOndary=ipadd]}
```

where:

- *domain-name* is a character string of up to 255 characters. Valid characters are uppercase and lowercase letters, digits (0-9), and the underscore character (“_”).
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *ipadd* is an IP address in dotted decimal notation.

Description This command adds a DNS server to the list of DNS servers used to resolve host names into IP addresses.

The **domain** parameter specifies the domain for which this DNS server is to be used to resolve host names. DNS requests for hosts in this domain are sent to this server. If **any** is specified, the name server is the default name server, and is used for domains not otherwise matched by another DNS entry. The default is **any**.

The default name server must be configured before domain-specific name servers can be configured. The maximum number of domain-specific name servers is 10.

The **interface** parameter specifies the interface over which the router learns the address of a primary and/or a secondary name server. The primary and secondary name server's addresses can be either statically configured using the **primary** and **secondary** parameters, or learned dynamically over an interface. Name servers can be learned via DHCP over an Ethernet or VLAN interface or via IPCP over a PPP interface. If the **interface** parameter is specified, the **primary** and **secondary** parameters are not required. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces.

The **primary** parameter specifies the IP address of the name server to be used as the primary name server for resolving hosts in the specified domain. If the **primary** parameter is specified, the **interface** parameter must not be specified.

The **secondary** parameter specifies the IP address of the name server to be used as the secondary name server for resolving hosts in the specified domain. If the **secondary** parameter is specified, the **interface** parameter must not be specified.

Examples

To add primary and secondary name servers, with IP addresses of 192.168.20.1 and 192.168.20.2 respectively, for use as default name servers when the domain to be resolved does not match any of the domain suffixes specifically configured, use the command:

```
add ip dns prim=192.168.20.1 seco=192.168.20.2
```

The name servers are used when the name being resolved does not match any DNS domain suffixes specifically configured by commands such as in the following example.

To add primary and secondary name servers, with IP addresses of 192.168.10.1 and 192.168.10.2 respectively, for use when resolving host names in the domain "oranges.com", use the command:

```
add ip dns dom=oranges.com prim=192.168.10.1
   seco=192.168.10.2
```

Related Commands

delete ip dns
set ip dns
show ip dns

add ip egp

Syntax

```
ADD IP EGP=ipadd
```

where *ipadd* is an IP address in dotted decimal notation

Description

This command adds an EGP neighbour to the list of EGP neighbours with which to exchange exterior EGP routing information. If EGP is already enabled (using the **enable ip egp** command on page 14-121), the router tries to start an EGP connection to the neighbour. An attempt is made to start a connection to all defined EGP neighbours at boot or reinitialisation of the IP module. A maximum of 8 neighbours can be defined.

The **egp** parameter specifies the IP address of the EGP neighbour. The EGP neighbour must not already be defined.

Examples

To add the router with IP address 172.16.248.33 as an EGP neighbour, use:

```
add ip egp=172.16.248.33
```

Related Commands

add ip rip
delete ip egp
delete ip rip
disable ip egp
disable ip exportrip
enable ip egp
enable ip exportrip
set ip autonomous in Chapter 49, Border Gateway Protocol version 4 (BGP-4)
show ip
show ip egp

add ip filter

Syntax

```
ADD IP FILter=filter-number SOurce=ipadd [SMask=ipadd]
    [SPort={port-name|port-id}] [DEStination=ipadd
    [DMask=ipadd]] [DPort={port-name|port-id}]
    [ICMPCode={icmp-code-name|icmp-code-id}]
    [ICmptype={icmp-type-name|icmp-type-id}] [LOG={4..1600|
    Dump|Header|None}] [OPtions={False|OFF|ON|NO|True|YES}]
    [PROTocol={protocol|Any|Egp|Icmp|Ospf|Tcp|Udp}]
    [SEssion={Any|Established|Start}] [SIze=size]
    [ENTry=1..255] {ACtion={INCLude|EXCLude}|POLIcy=0..15|
    PRIOrity=P0..P7}
```

where:

- *filter-number* is a number from 0 to 399.
- *ipadd* is an IP address in dotted decimal notation.
- *port-name* is the predefined name for an IP port.
- *port-id* is an IP port number or a range in the format *low:high*.
- *icmp-code-name* is the predefined name for an ICMP reason code.
- *icmp-code-id* is the number of an ICMP reason code.
- *icmp-type-name* is the predefined name of an ICMP message type.
- *icmp-type-id* is the number of an ICMP message type.
- *protocol* is an IP protocol number.
- *size* is a number from 64 to 65535.

Description

This command adds a pattern to an IP traffic filter, policy filter, routing filter, or priority filter. The exact pattern should not already exist in the filter.

The **filter** parameter specifies the number of the filter to which the pattern is to be added.

- Filters with numbers from 0 to 99 are treated as traffic filters, and use the **action** parameter to specify the action to take with a packet that matches the pattern.
- Filters with numbers from 100 to 199 are treated as policy filters, and use the **policy** parameter to specify the policy to use when routing a packet that matches the pattern.
- Filters with numbers from 200 to 299 are treated as priority filters, and use the **priority** parameter to specify the priority to assign to a packet that matches the pattern.
- Filters from 300 to 399 are treated as routing filters, and use the **action** parameter to specify the action to take with a route that matches the pattern.

An interface may have a maximum of one traffic filter, one policy filter, and one priority filter, but the same traffic, policy or priority filter can be assigned to more than one interface. Traffic and routing policy filters are applied to packets received via the interface, whereas policy and priority filters are applied to packets as they are transmitted. Routing filters are used in commands that manipulate the passing of IP routing information in and out of the router.

All parameters are valid for traffic (0-99), policy (100-199) and priority (200-299) filters. The **source**, **smask**, **entry**, and **action** parameters are valid for routing filters (300-399).

The **source** parameter specifies the source IP address, in dotted decimal notation, for the pattern.

The **smask** parameter specifies the mask, in dotted decimal notation to apply to source addresses for this pattern. The mask is used to determine the portion of the source IP address in the IP packet that is significant for comparison with this pattern.

The values of **source** and **smask** must be compatible. For each bit in **smask** that is set to zero, the equivalent bit in **source** must also be zero (0). If either **source** or **smask** is 0.0.0.0, then both must be 0.0.0.0. The default is 255.255.255.255.

The **sport** parameter specifies the source port to check against for this pattern as the recognised name of a well-known UDP or TCP port (Table 14-11 on page 14-70), a decimal value from 0 to 65535, or a range of numbers formatted *low:high*. If *low* is omitted, 0 is assumed; if *high* is omitted, the maximum port number is assumed. If a port other than **any** is specified, the **protocol** parameter is required and must be TCP or UDP. The default is **any**.

The **destination** parameter specifies the destination IP address for the pattern in dotted decimal notation. The default is 0.0.0.0.

The **dmask** parameter specifies the mask in dotted decimal notation to apply to the destination address for this pattern. The mask determines the portion of the destination IP address in the IP packet that is significant for comparison with this pattern. If **dmask** is specified, **destination** must also be specified.

The values of **destination** and **dmask** must be compatible. For each bit in **dmask** that is set to zero (0), the equivalent bit in **destination** must also be zero (0). If either **destination** or **dmask** is 0.0.0.0, then both must be 0.0.0.0. If **destination** is specified, the default for **dmask** is 255.255.255.255. If **destination** is not specified, the default for **dmask** is 0.0.0.0.

The **dport** parameter specifies the destination port to check against for this pattern as the recognised name of a well-known UDP or TCP port (Table 14-11 on page 14-70), a decimal value from 0 to 65535, or a range of numbers formatted *low:high*. If *low* is omitted, 0 is assumed; if *high* is omitted, the maximum port number is assumed. If a port other than **any** is specified, the **protocol** parameter is required and must be TCP or UDP. The default is **any**.

If a pattern for Telnet is not explicitly added to a filter assigned to an interface, all Telnet traffic received over the specified interface is discarded. This prevents Telnet connections to the router itself via the interface. To enable access to the router’s command prompt via Telnet, a pattern for Telnet must be added to the filter for the interface.

Table 14-11: Predefined port names used by the IP filtering process

Port Name	Number	Protocol[1]	Description
ANY	-	-	Any port
BOOTPC	68	UDP	Bootstrap Protocol Client
BOOTPS	67	UDP	Bootstrap Protocol Server
DOMAIN	53	TCP/UDP	Domain Name Server
FINGER	79	TCP	Finger
FTP	21	TCP	File Transfer [Control]
FTPDATA	20	TCP	File Transfer [Default Data]
GOPHER	70	TCP	Gopher
HOSTNAME	101	TCP/UDP	NIC Host Name Server
IPX	213	TCP/UDP	IPX
KERBEROS	88	UDP	Kerberos
LOGIN	49	UDP	Login Host Protocol
MSGICP	29	TCP/UDP	MSG ICP
NAMESERVER	42	UDP	Host Name Server
NEWS	144	TCP	NewS
NNTP	119	TCP	Network News Transfer Protocol
NTP	123	TCP	Network Time Protocol
RTELNET	107	TCP/UDP	Remote Telnet Service
SFTP	115	TCP/UDP	Simple File Transfer Protocol
SMTP	25	TCP	Simple Mail Transfer
SNMP	161	UDP	SNMP
SNMPTRAP	162	UDP	SNMPTRAP
SYSTAT	11	TCP	Active Users
TELNET	23	TCP	Telnet
TFTP	69	UDP	Trivial File Transfer
TIME	37	TCP/UDP	Time
UUCP	540	TCP	uucpd
UUCPRLOGIN	541	TCP/UDP	uucp-rlogin
WWWHTTP	80	TCP	World Wide Web HTTP
XNSTIME	52	TCP/UDP	XNS Time Protocol

[1] *The protocol typically used with the port.*

The **icmptype** and **icmpcode** parameters specify the ICMP message type and ICMP message reason code to match against the ICMP type and code fields in an ICMP packet. The **icmptype** parameter specifies the ICMP message type to match as a decimal value from 0 to 255, or the recognised name of an ICMP type (Table 14-12 on page 14-71). The **icmpcode** parameter specifies the ICMP message reason code to match as a decimal value from 0 to 255, or the recognised name of an ICMP reason code (Table 14-13 on page 14-71). Both parameters are valid when the **protocol** parameter is set to **icmp**.

Table 14-12: Predefined ICMP type names used by the IP filtering process

ICMP Type Name	ICMP Type Value	ICMP Type Description	ICMP Codes Supported
ECHORPLY	0	Echo reply messages	No
UNREACHABLE	3	Unreachable messages	Yes
QUENCH	4	Source quench messages	No
REDIRECT	5	Redirect messages	Yes
ECHO	8	Echo request messages	No
ADVERTISEMENT	9	Router advertisement messages	No
SOLICITATION	10	Router solicitation messages	No
TIMEEXCEED	11	Time exceeded messages	Yes
PARAMETER	12	Parameter problem messages	Yes
TSTAMP	13	Timestamp request messages	No
TSTAMPRPLY	14	Timestamp reply messages	No
INFOREQ	15	Information request messages	No
INFOREP	16	Information reply message	No
ADDRREQ	17	Address mask request messages	No
ADDRREP	18	Address mask reply messages	No
NAMEREQ	37	Name request messages	No
NAMERPLY	38	Name reply messages	No

Table 14-13: Predefined ICMP code names used by the IP filtering process

ICMP Code Name	ICMP Code Value	ICMP Code Description	Applies to ICMP Type Name...
ANY	(any)	Any ICMP code	(any)
NETUNREACH	0	Network unreachable	UNREACHABLE
HOSTUNREACH	1	Host unreachable	UNREACHABLE
PROTUNREACH	2	Protocol unreachable	UNREACHABLE
PORTUNREACH	3	Port unreachable	UNREACHABLE
FRAGMENT	4	Fragmentation is needed but “do not fragment” flag is set	UNREACHABLE
SOURCEROUTE	5	Source route failed	UNREACHABLE
NETUNKNOWN	6	Destination network unknown	UNREACHABLE
HOSTUNKNOWN	7	Destination host unknown	UNREACHABLE
HOSTISOLATED	8	Source host isolated	UNREACHABLE
NETCOMM	9	Communication with destination network administratively prohibited	UNREACHABLE
HOSTCOMM	10	Communication with destination host administratively prohibited	UNREACHABLE
NETTOS	11	Network unreachable for selected TOS	UNREACHABLE
HOSTTOS	12	Host unreachable for selected TOS	UNREACHABLE

Table 14-13: Predefined ICMP code names used by the IP filtering process

ICMP Code Name	ICMP Code Value	ICMP Code Description	Applies to ICMP Type Name...
FILTER	13	Communication administratively prohibited due to filtering	UNREACHABLE
HOSTPREC	14	Host precedence violation	UNREACHABLE
PRECEDENT	15	Precedence cutoff in effect	UNREACHABLE
NETREDIRECT	0	Redirect datagrams for the network	REDIRECT
HOSTREDIRECT	1	Redirect datagram for the host	REDIRECT
NETRTOS	2	Redirect datagrams for the TOS and network	REDIRECT
HOSTRTOS	3	Redirect datagrams for the TOS and host	REDIRECT
TTL	0	TTL exceeded in transit	TIMEEXCEED
FRAGREASSM	1	Fragment reassembly time exceeded	TIMEEXCEED
PTRPROBLEM	0	Pointer value referencing the octet in the original IP packet caused problem	PARAMETER
NOPTR	1	No pointer present	PARAMETER

The **log** parameter specifies whether matches to a filter entry result in a message being sent to the router's Logging facility, and the content of the log messages. This parameter enables logging of the IP packet filtering process down to the level of an individual filter entry.

If a number from 4 to 1600 is specified, the filter number, entry number, and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/PASS (for patterns with an **include** action) or IPFIL/FAIL (for patterns with an **exclude** action). In addition, the first 4 to 1600 octets of the data portion of TCP, UDP, and ICMP packets or the first 4 to 1600 octets after the IP header of other protocol packets are logged with a message type/subtype of IPFIL/DUMP.

If **dump** is specified, the filter number, entry number, and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/PASS (for patterns with an **include** action) or IPFIL/FAIL (for patterns with an **exclude** action). In addition, the first 32 octets of the data portion of TCP, UDP, and ICMP packets or the first 32 octets after the IP header of other protocol packets are logged with a message type/subtype of IPFIL/DUMP.

If **header** is specified, the filter number, entry number, and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/PASS (for patterns with an **include** action) or IPFIL/FAIL (for patterns with an **exclude** action). If **none** is specified, matches to the filter entry are not logged. The default is **none**.

The **options** parameter specifies the IP options field is used to check against the pattern. If **yes**, the pattern matches IP packets with options set; if **no**, the pattern matches packets without options set. The default is to match IP packets with or without IP options set.

The **protocol** parameter specifies the protocol to check against for this pattern as a decimal value from 0 to 65534. Valid protocol names are:

- Exterior Gateway Protocol (EGP)
- Internet Control Message Protocol (ICMP)
- Open Shortest Path First Protocol (OSPF)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

If either **sport** or **dport** is specified, **protocol** must be defined as TCP or UDP. Specifying TCP or UDP filters packets from companion protocols, for example ICMP, RIP and OSPF, that do not use TCP or UDP as a transport mechanism. The default is **any**.

The **session** parameter specifies the type of TCP packet to match, and can be used when the **protocol** parameter specifies TCP. If **start** is specified, the pattern matches TCP packets with the SYN bit set and the ACK bit clear. If **established** is specified, the pattern matches TCP packets with either the SYN bit clear or the ACK bit set. If **any** is specified, the pattern matches any TCP packet. The default is **any**.

The **size** parameter specifies the maximum reassembled size to match against for each IP fragment. If the fragment's offset plus size is greater than the value specified, the fragment is discarded.

The **entry** parameter specifies the entry number in the filter that this new pattern occupies. Existing patterns with the same or higher entry numbers are pushed down the filter. The default is to add the new pattern to the end of the filter.

The **action** parameter is used for traffic and routing filters and specifies the action to take when the pattern is matched. If **include** is specified, the IP packet is processed and forwarded for traffic filters, or the IP route is selected, for routing filters. If **exclude** is specified, the IP packet is discarded for traffic filters, or the IP route is excluded for routing filters. The **action**, **policy**, and **priority** parameters are mutually exclusive so only one may be specified.

The **policy** parameter is used for policy-based routing and specifies the policy to use when the pattern is matched.

- For policy numbers from 0 to 7, routes with a matching policy are considered first.
- For policy numbers from 8 to 15, routes with a policy of *n*-8 (where *n* is the filter policy) are considered first, and the policy value *n*-8 is written into the TOS field of the packet.

The policy number is assigned to incoming packets but employed during forwarding (transmission). When no route is matched to the policy, the packet is routed as if no policies are present; only routes with no policy are considered. The **action**, **policy**, and **priority** parameters are mutually exclusive so only one may be specified.

The **priority** parameter is used for priority routing and specifies the priority when the pattern is matched. The priority number is assigned to incoming packets but employed during forwarding (transmission). Packets can be assigned a priority from p3 (highest) to p7 (lowest). The default is p5. Priority levels p0, p1, and p2 should not be used because they may conflict with router system activities. The **action**, **policy**, and **priority** parameters are mutually exclusive so only one may be specified.

Examples To create filters to allow only FTP traffic between two hosts with IP addresses 172.16.10.2 and 192.168.2.6, use the commands:

```
add ip fil=1 so=192.168.2.6 sm=255.255.255.255
   des=172.16.10.2 dp=ftp prot=t ac=incl
add ip fil=1 so=192.168.2.6 sm=255.255.255.255
   des=172.16.10.2 dp=ftpdata prot=t ac=incl
add ip fil=2 so=172.16.10.2 sm=255.255.255.255 sp=ftp
   des=192.168.2.6 prot=t ac=incl
add ip fil=2 so=172.16.10.2 sm=255.255.255.255 sp=ftpdata
   des=192.168.2.6 prot=t ac=incl
```

Related Commands

add bgp peer
add ip route filter
delete ip filter
delete ip route filter
set bgp peer
set ip filter
show ip filter
show ip route filter

add ip helper

Syntax `ADD IP HElper DEStination=`*`ipadd`* `INTerface=`*`interface`* `POrt=`*`port-number`*

where:

- *ipadd* is an IP address in dotted decimal notation.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *port-number* is a UDP port number from 1 to 65535, or one of the predefined UDP port names DNS (port 53), NT or NETBIOS (ports 137 and 138), TACACS (port 49), TIME (port 37) or TFTP (port 69).

Description This command adds a port or a set of named ports to the list of UDP ports to listen for on the specified interface. When a broadcast UDP packet is received on the specified interface with the specified destination port number it is redirected to the destination IP address. This allows all network broadcast packets to be delivered across the internet to appropriate servicing hosts. Multiple invocations of this command can be used for forward packets for several UDP ports to the same IP address, to forward packets for a single UDP port to multiple IP addresses.

The **destination** parameter specifies the IP address to which the UDP broadcast traffic is forward.

The **interface** parameter specifies the interface to which the UDP port list is assigned. UDP broadcasts are forwarded that are received for the specified interface for one of the UDP ports in the UDP port list. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces.

The **port** parameter specifies the UDP port, as a decimal number from 1 to 65535, or the recognised name of a UDP port set. Broadcast traffic received by the router on the specified port or set of ports is redirected to the IP host at the destination address. Up to 32 ports can be specified.

Examples

To forward all NETBIOS broadcasts received via interface eth0 to IP address 192.168.202.3, use the command:

```
add ip he po=netbios des=192.168.202.3 int=eth0
```

To forward all broadcasts to UDP port 3001 received via interface eth0 to IP address 192.168.100.2, use the command:

```
add ip he po=3001 int=eth0 des=192.168.100.2
```

Related Commands

delete ip helper
disable ip helper
enable ip helper
show ip helper

add ip host

Syntax

```
ADD IP HOst=name IPaddress=ipadd
```

where:

- *name* is a character string up to 60 characters long. If the string contains spaces, it must be in double quotes.
- *ipadd* is an IP address in dotted decimal notation.

Description

This command adds a user-defined name for an IP host to the host name table. The host name table makes it easier to Telnet to commonly accessed hosts by enabling the user to enter a shorter, easier to remember name for the host rather than the host's full IP address or domain name. The name can also be used with the **ping** command on page 14-129.

The **host** parameter specifies the user-defined name for the IP host. A host with the same name must not already exist in the host name table. When a host name is specified in the Telnet command, the entire name is used to match a name in the host name table. All characters are used in the comparison, including nonalphabetic characters if they are present.

The **ipaddress** parameter specifies the IP address of the host.

Examples

To add the host name “zaphod” to the host name table for an IP host with an IP address of 172.16.1.5 and the domain name “zaphod.company.com”, use:

```
add ip host=Zaphod IP=172.16.1.5
```

To Telnet to the host, use any of the following commands:

```
telnet zaphod
telnet zaphod.company.com
telnet 172.16.1.5
```

Related Commands

delete ip host
set ip host
set ip nameserver
set ip secondarynameserver
show ip host

add ip interface

Syntax

```
ADD IP INTerface=interface IPaddress={ipadd|DHCP}
    [ADVertise={YES|NO}] [BROadcast={0|1}]
    [DIRectedbroadcast={False|NO|OFF|ON|True|YES}]
    [FILter={0..99|NONE}] [FRAgment={NO|OFF|ON|YES}]
    [GRAtuitousarp={ON|OFF}] [GRE={0..100|NONE}]
    [IGMPProxy={OFF|UPstream|DOWNstream}] [INVersearp={ON|
    OFF}] [MASK=ipadd] [METric=1..16] [MULticast={BOTH|NO|
    OFF|ON|RECeive|SENd|YES}] [OSPFmetric=1..65534]
    [POLicyfilter={100..199|NONE}]
    [PREferencelevel={-2147483648..2147483647|NOTDEFAULT}]
    [PRIorityfilter={200..299|NONE}]
    [[PROxyarp={False|NO|OFF|ON|True|YES|STrict|DEFRoute}]
    [RIPMetric=1..16]
    [SAMode={Block|Passthrough}] [VJC={False|NO|OFF|ON|
    True|YES}] [VLANTAG={1..4094|NONE}]
```

where:

- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *ipadd* is an IP address in dotted decimal notation.

Description

This command adds a logical interface to the IP module. A maximum of 1280 interfaces can be added.

When the router is in security mode, this command can be issued only by a user with security officer privilege.

The **interface** parameter specifies the name of the logical interface, and implicitly, the attached Layer 2 interface. The Layer 2 interface must already be configured. The IP interface must not already be assigned to the IP module. At least two interfaces must be defined before the router can route IP packets, but only one interface (usually eth0) needs to be defined when the router is acting as a server. When an interface is added, it is automatically enabled. Only one logical interface may be configured to the same IP network or subnet. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces.

The **advertise** parameter specifies whether or not the address is to be advertised using the Router Discovery feature as described in RFC 1256. Advertise can only be set if a valid IP address is also specified. The default advertise value is Yes.

The **broadcast** parameter specifies whether to use a broadcast address with all 1s or all 0s. The default is 1. An all 0s setting contradicts current RFCs and is only provided for backwards compatibility with some older UNIX systems.

The **directedbroadcast** parameter specifies whether the router allows network or subnet broadcasts to be forwarded to the network directly attached to the logical interface. The default is **no**.

The **filter** parameter specifies the traffic filter to apply to IP packets transmitted or received over the logical interface. The filter must already have been defined with the **add ip filter** command on page 14-68. A logical interface may have a maximum of one traffic filter, one policy filter and one priority filter, but the same traffic, policy or priority filter can be assigned to more than one interface. Traffic filters are applied to packets received via the logical interface. The default is to not apply a filter.

The **fragment** parameter specifies whether the "*Do not fragment*" bit is obeyed for outgoing IP packets that are larger than the MTU of the interface. If **yes**, the "*Do not fragment*" bit is ignored and outgoing IP packets larger than the MTU of the interface are fragmented. This is particularly useful for interfaces configured with GRE, SA and/or IPsec encapsulation, which can potentially increase packet sizes beyond the MTU of the interface. If **no**, the "*Do not fragment*" bit is obeyed and IP packets larger than the MTU are discarded. This is normal behaviour for IP. The **fragment** parameter has no effect on packets smaller than the interface MTU. The default is **no**.

The **gratuitousarp** parameter enables or disables the acceptance of gratuitous ARP request or gratuitous ARP reply. The default is **on**.

The **gre** parameter specifies the GRE (Generic Routing Encapsulation) entity associated with the logical interface. The GRE entity must have been created previously with the **add gre** command on page 29-10 of Chapter 29, Generic Routing Encapsulation (GRE). The default is **none**.

The **igmpproxy** parameter specifies the status of IGMP proxying for the specified interface. If **off**, the interface does not do IGMP Proxy. If **upstream**, the interface passes IGMP messages in the upstream direction. A router can have only one interface when the IGMP proxy direction is upstream. If **downstream**, the interface can receive IGMP messages from the downstream direction. The default is **off**. To display information about IGMP and multicast group membership for each IP interface, use the **show ip igmp** command on page 17-73 of Chapter 17, IP Multicasting.

The **inversearp** parameter enables or disables the operation of the Inverse Address Resolution Protocol (INVARP) on ATM interfaces. The **inversearp** parameter must be set to **on** for IPoA configurations, and to **off** for RFC 1483 Routed configurations. The default is **off**. (Inverse ARP is always on for Frame Relay interfaces.)

The **ipaddress** parameter specifies the IP address of the logical interface. If **dhcp** is specified, the router acts as a DHCP client and obtains the configuration of the IP interface via DHCP. Table 14-14 on page 14-79 lists the parameters from the DHCP reply that the router uses. If an IP interface is configured to use DHCP to obtain its IP address and subnet mask, the interface does not take part in IP routing until the IP address and subnet mask have been set by DHCP.

If different interfaces on a device need to be uniquely distinguished, then extended DHCP identification is needed, and the **extendid** parameter in the **set**

dhcp command must be on before DHCP clients are created. See the **set dhcp** command on page 35-20 of Chapter 35, Dynamic Host Configuration Protocol (DHCP).

Table 14-14: DHCP reply parameters used by the router for configuring IP

DHCP Parameter	Purpose
IP address and mask	IP address and subnet mask for the IP interface.
DNS Servers	DNS server addresses added to the list of IP name servers. A primary name server and a secondary name server are supported. Name servers are normally added with the **set ip nameserver** command on page 14-151 and the **set ip secondarynameserver** command on page 14-162.
Gateway	Default route is added over the specified interface with the next hop set to the gateway address. If a default route does already exist on the router, the gateway parameter in the DHCP reply is ignored.
Domain Name	Domain name of the router.

Remote address assignment must be enabled using the **enable ip remoteassign** command on page 14-126 before IP interfaces accept addresses dynamically assigned by DHCP.

The **mask** parameter specifies the subnet mask for the logical interface. The value must be consistent with the value specified for the **ipaddress** parameter. The default is the network mask for the address class of the IP address (for example, 255.255.0.0 for a Class B address, 255.255.255.0 for a Class C address). If **ipaddress** is set to **dhcp**, the **mask** parameter should not be set because the subnet mask received from the DHCP server is used.

The **multicast** parameter specifies whether the interface receives and forwards multicast packets when DVMRP and PIM are not enabled. If **both** or **on** is specified, the router both sends and receives multicast packets. If **off**, the router neither sends nor receives multicast packets. If **receive** is specified, the router receives but does not send multicast packets. If **send** is specified, the router sends but does not receive multicast packets. Note that this parameter applies to the entire IP interface, not an individual logical interface. Setting it on one logical interface sets it on all other logical interfaces associated with the same IP interface. This parameter determines the interface's static behaviour for multicast packets. When DVMRP or PIM-SM is enabled, it determines the forwarding behaviour of interfaces dynamically, and this parameter has no effect (Chapter 14, IP Multicasting). The default is **receive**.

The **ospfmetric** parameter specifies the cost of crossing the logical interface for OSPF. The default is 1.

The **policyfilter** parameter specifies the policy filter to apply to IP packets received over the logical interface. The filter must already have been defined with the **add ip filter** command on page 14-68. A logical interface may have a maximum of one traffic filter, one policy filter, and one priority filter. However, the same traffic, policy or priority filter can be assigned to more than one interface. Policy filters are applied to packets when they are transmitted. The default is not to apply a filter.

The **preferencelevel** parameter specifies the preference of the address as a default router address relative to other router addresses on the same subnet, as a decimal integer. If the minimum value (-2147483648) or **notdefault** is specified, the address is not used by neighbouring hosts as a default address, even though it may be advertised. The default is the mid range 0.

The **priorityfilter** parameter specifies the priority filter to apply to IP packets transmitted over the logical interface. The filter must already have been defined with the **add ip filter** command on page 14-68. A logical interface may have a maximum of one traffic filter, one policy filter, and one priority filter. However, the same traffic, policy or priority filter can be assigned to more than one interface. Priority filters are applied to packets as they are transmitted. The default is not to apply a filter.

The **proxyarp** parameter enables or disables proxy ARP responses to ARP requests. This parameter is valid for Eth and VLAN interfaces. The default is **on**.

If the **on/true/yes** option is specified, the device will respond to proxy ARP Requests using specific routes if they exist. If the **off/false/no** option is specified, the device will not respond to ARP requests. If the **defroute** option is specified, the device will respond to proxy ARP Requests using specific routes if they exist or a default route (0.0.0.0) if it exists. If the **strict** option is specified, the router will only respond to ARP requests using specific routes if they exist.

*If the **defroute** option is currently enabled, any other **proxyarp** option selected will disable the **defroute** mode of operation.*

When the device is operating in defroute mode, it is non-compliant with RFC 1027.

The **ripmetric** parameter specifies the cost of crossing the logical interface for RIP. The default is 1. The **metric** parameter is also accepted for backwards compatibility.

The **samode** parameter specifies how the logical interface handles IP packets that do not belong to one of the security associations assigned to the interface. If **block** is specified, IP packets that do not belong to a security association assigned to the logical interface are blocked from transiting the interface and are discarded. If **passthrough** is specified, IP packets that do not belong to a security association assigned to the logical interface are allowed to transit the interface and are forwarded as normal by the IP routing software. The default is **block**. This parameter has affect when one or more security associations have been assigned to the logical interface with the **add ip sa** command on page 14-94.

The **vjc** parameter is valid for Point-to-Point Protocol (PPP) and X25T interfaces, and specifies whether Van Jacobson header compression is to be used on the Layer 2 interface. The **vjc** parameter applies to all logical interfaces attached to the same Layer 2 interface. Changing the setting on one logical interface alters the setting on the others attached to the Layer 2 interface. Compression provides the most advantage on slower link speeds (up to 48 kbps). At speeds of 64 kbps and higher, compression actually reduces efficiency and so should be disabled. Van Jacobson's TCP/IP header compression should not be enabled on a multilink PPP interface. The default is **off**.

The **vlantag** parameter specifies the VID (VLAN Identifier) to be included in the header of each frame that is transmitted over the logical interface. This parameter is valid for Eth interfaces only. Multiple logical interfaces on the same physical interface can share the same VLAN tag. The default is **none**, which means no VID is included. For more information, see “VLAN Tagging on Eth Interfaces” on page 14-30.

Examples

To add PPP interface 0 (logical interface ppp0-0) with an IP address of 172.16.248.33, a subnet mask of 255.255.255.0, a metric of 5 and Van Jacobson’s header compression, use:

```
add ip int=ppp0 ip=172.16.248.33 mask=255.255.255.0 ripm=5
   vjc=on
```

To add a second logical interface to PPP interface 0 (logical interface ppp0-1) with an IP address of 172.16.200.1, a subnet mask of 255.255.255.0, a metric of 5 and Van Jacobson’s header compression, use:

```
add ip int=ppp0-1 ip=172.16.200.1 mask=255.255.255.0 ripm=5
   vjc=on
```

To add Ethernet interface 0 with an IP address of 202.36.163.1, a mask of 255.255.255.192 and associate the IP interface with GRE entity 1, use:

```
add ip int=eth0 ip=202.36.163.1 mask=255.255.255.192 gre=1
```

Related Commands

add ip advertise interface
delete ip advertise interface
delete ip interface
disable ip advertise
disable ip interface
enable ip advertise
enable ip interface
reset ip interface
set ip advertise interface
set ip interface
show ip igmp
show ip interface

add ip local

Syntax

```
ADD IP LOCAL=[1..15][FILTER={filter-number|NONE}]
    [GRE=[0..100 NONE}][IPADDRESS=ipadd]
    [POLICYFILTER={filter-numbering}]
    [PRIORITYFILTER={filter-number|NONE}]
```

where:

- *filter-number* is a number from 0 to 299.
- *ipadd* is an IP address in dotted decimal notation.

Description

This command adds a local interface to the router. Up to fifteen local interfaces can be added to a single router. These are in addition to the default local interface that is automatically added at start up, and can be configured through the **set ip local** command. A local interface is *virtual* in the sense that it is not associated with a physical interface. Each local interface can be assigned an IP address, which can then be used as the source address of IP packets generated internally by IP protocols such as RIP, OSPF, PING and NTP. Higher layer protocols such as RIP, OSPF, PING and NTP must assign a source IP address to packets passed to IP for forwarding.

The following rules are used to determine which IP address to use as the source address:

1. If the higher layer protocol's configuration specifies the use of either a source IP address or a local interface, then the configured address is used as the packet's source IP address. For example, the **sipaddress** parameter of the **ping** command specifies the source IP address to use in ping packets. While the **local** parameter of the **add bgp peer** command specifies a local interface to use to obtain a source IP address.
2. If the default local interface has been assigned an IP address, then this will be used as the packet's source IP address. Otherwise, the IP routing module determines the interface over which the packet is to be transmitted, and assigns the IP address of the interface as the packet's source IP address.

The **local** parameter specified is a unique identifying number that is used to identify a particular local interface. The naming convention, or alias, for this interface is the concatenation of the word *local* along with this identifying number.

The **filter** parameter specifies which filter will be applied to IP packets transmitted or received over the interface. The filter must already have been defined with the **add ip filter** command on page 14-68. An interface may have a maximum of one traffic filter, one policy filter and one priority filter, but the same traffic, policy or priority filter can be assigned to more than one interface. Traffic filters are applied to packets received via the interface. The default is not to apply a filter.

The **gre** parameter specifies the GRE (Generic Routing Encapsulation) entity associated with the interface. The specified GRE entity must have been created previously using the **add gre** command on page 29-10 of Chapter 29, Generic Routing Encapsulation (GRE). The default is NONE.

The **ipaddress** parameter specifies the IP address of the interface. This must be the IP address of one of the switch's active IP interfaces. Note that specifying an IP address of 0.0.0.0 effectively 'unsets' the IP address of the local interface specified.

The **policyfilter** parameter specifies the policy filter that will be applied to IP packets received over the interface. The filter must already have been defined using the ADD IP FILTER command on page 8-57. Although an interface can only have one traffic filter, one policy filter and one priority filter; each of these filters can be assigned to more than one interface. Policy filters are applied to packets as they are transmitted. The default setting is NONE, that is, not to apply a filter.

The **priorityfilter** parameter specifies the filter that will be applied to IP packets received over the interface. The filter must have already been defined with the **add ip filter** command on page 14-68. Although an interface can only have one traffic filter, policy filter, and priority filter; each of these filters can be assigned to more than one interface. Priority filters are applied to packets as they are transmitted. The default setting is NONE, that is, not to apply a filter.

Examples To add the local interface 3 with an IP address of 192.168.33.1, use:

```
add ip loc=3 ip=192.168.33.1
```

Related Commands

delete ip local
set ip local
show ip interface

add ip nat

Syntax

```
ADD IP NAT IP=ipadd [MASK=ipadd] [GBLIPaddress=ipadd]
    [GBLMask=ipadd] [GBLPort=port] [GBLINterface=interface]
    [POrt=port] [PROTocol={protocol|ALL|EGP|GRE|ICmp|OSPF|
    SA|TCp|UDp}]
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *port* is an IP port number or the predefined name for an IP service.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *protocol* is an IP protocol number.

Description This command adds a local private IP network to the address translation table that NAT uses. The method of NAT translation depends on parameters in this command.

- To create a static NAT, use the **ip** and **gblipaddress** parameters.
- To create a dynamic NAT, use the **ip**, **mask**, **gblipaddress**, and **gblmask** parameters.
- To create a dynamic ENAT, use the **ip**, **mask**, and **gblipaddress**.

- To create a static ENAT, use the **ip**, **protocol**, **port**, **gblipaddress**, and **gblport** parameters.
- To create an interface ENAT, use the parameters **ip**, **mask**, and **gblinterface**.

NAT must be explicitly enabled with the **enable ip nat** command on page 14-125 before these translations take effect.

The **ip** parameter specifies either a host or network IP address for the private network. This parameter can be used with the **mask** parameter to specify a range of IP address for the private network.

The **protocol** parameter specifies the IP protocol number or the name of a predefined protocol type to be used with a static ENAT entry. If **tcp** or **udp** is specified, then **port** must also be specified.

The **port** parameter specifies the port number or service name (Table 14-15) for the port used on the private IP host when specifying a static ENAT entry.

Table 14-15: Service names for use with Network Address Translation (NAT)

Service Names	**Value**
ECHO	7
DISCARD	9
FTP	21
TELNET	23
SMTP	25
TIME	37
DOMAIN	53
BOOTPS	67
BOOTPC	68
TFTP	69
GOPHER	70
FINGER	79
WWW	80
KERBEROS	88
RTELNET	107
POP2	109
POP3	110
SNMP	161
SNMPTRAP	162
BGP	179
RIP	520
PPTP	1723

The **gblipaddress** parameter specifies either an officially assigned global IP address or the start of a range of officially assigned global IP addresses. This parameter can be used with the **gblmask** parameter to specify the range of global IP addresses for an entry.

The **gblport** parameter specifies the port number or service name (Table 14-15 on page 14-84) for the port available to global Internet access when creating a static ENAT.

The **gblinterface** parameter specifies the interface that has or will dynamically obtain an officially assigned global IP address. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces.

A static ENAT entry can be added to an existing dynamic ENAT entry only.

Examples To configure a static NAT to translate between the private IP address 10.1.1.2 and the officially assigned global IP address 202.1.1.1, use the command:

```
add ip nat IP=10.1.1.2 gblip=202.1.1.1
```

To configure a dynamic NAT entry to allow an entire private network with the IP address 10.1.1.0 to use a block of 32 officially assigned global IP addresses starting at 202.1.1.1, use the command:

```
add ip nat ip=10.1.1.0 mask=255.0.0.0 gblip=202.1.1.1
    gblm=255.255.255.224
```

To configure a dynamic ENAT entry to allow an entire private network with the IP address 192.168.100.0 to use the single officially assigned global IP address 202.1.1.1, use the command:

```
add ip nat ip=192.168.100.0 mask=255.255.255.0
    gblip=202.1.1.1
```

To add a static ENAT entry to the above example to allow access to a WWW server at IP address 192.168.100.54 on the private network, use the command:

```
add ip nat ip=192.168.100.54 prot=TCP po=80 gblip=202.1.1.1
    gblp=80
```

To re-map the WWW port on the server (192.168.100.54) to port 8001 in the private network, use the command:

```
add ip nat ip=192.168.100.54 prot=tcp po=8001 gblip=202.1.1.1
    gblp=80
```

To allow two different TELNET servers on the private network at IP addresses 192.168.100.54 and 192.168.100.92 to be accessed from the Internet, use the commands:

```
add ip nat ip=192.168.100.54 prot=tcp po=23 gblip=202.1.1.1
    gblp=23
add ip nat IP=192.168.100.92 prot=tcp po=23 gblip=202.1.1.2
    gblp=23
```

To add an interface ENAT entry to allow an entire private network with the IP address 10.1.1.0 to use a single dynamically assigned global IP addresses on interface ppp0, use the command:

```
add ip nat ip=10.1.1.0 mask=255.0.0.0 gblin=PPP0
```

Related Commands

delete ip nat
disable ip nat
enable ip nat
show ip nat

add ip rip

Syntax

```
ADD IP RIP INTerface=interface [CIRCuit=miox-circuit]
    [DLCi=dlci] [IP=ipadd] [NEXThop=ipadd] [SENd={NOne|
    RIP1|RIP2|COmpatible}] [RECeive={NOne|RIP1|RIP2|BOth}]
    [DEMand={False|NO|OFF|ON|True|YES}] [AUth={NOne|
    PASSword|MD5}] [PASSword=password] [STATicexport={YES|
    NO}]
```

where:

- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit) from 0 to 1023.
- *ipadd* is an IP address in dotted decimal notation.
- *password* is a character string 1 to 63 characters long. It may contain uppercase and lowercase letters, digits (0-9), the hyphen (-), and the underscore character (“_”).

Description

This command adds a RIP neighbour so that RIP packets are sent to and received from an IP address on an interface.

The **interface** parameter specifies an existing interface on which to send or receive RIP packets. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces.

The **circuit** parameter specifies the X.25 circuit on which to send or receive RIP packets. It is a required parameter for X25T interfaces and is valid when the interface is an X25T interface.

The **dlci** parameter specifies the Frame Relay DLCI on which to send or receive RIP packets. It is a required parameter for Frame Relay interfaces and is valid for Frame Relay only.

The **ip** parameter specifies the IP address of the RIP neighbour. If an IP address is specified, then RIP packets received on the interface are accepted from this address. If no IP address is specified, then the source address of RIP packets is not checked. RIP updates generated by the device being configured are sent to the specific IP address. If no **ip** parameter is specified, RIP packets are sent to the RIP multicast address 224.0.0.9 (if the **send** parameter is **rip2** or **compatible**), or the broadcast address (if the **send** parameter is **rip1**).

The **nexthop** parameter is carried in RIP v2 packets to inform the destination of the next hop address returning to the device being configured. The default is 0.0.0.0, indicating the (local) source of the RIP route update. If **nexthop** is specified, **ip** must also be specified, and **send** must not indicate that RIPv1 packets are to be sent.

The **send** parameter specifies the version of RIP packet to send. If **none** is specified, then no RIP packets are sent. If **rip1** is specified, RIP version 1 packets are sent; if **rip2**, version 2 packets are sent. If **compatible** is specified, RIP version 2 packets are sent without routes that a router receiving only RIP version 1 treats as host routes. The default is **rip1**.

The **receive** parameter specifies the version of RIP packets to receive. If **none**, then no RIP packets are accepted from the IP address on the interface. If **rip1** is specified, RIP version 1 packets are accepted; if **rip2**, version 2 packets are accepted. If **both** is specified, then either RIP version 1 or RIP version 2 packets are accepted (as long as a version compatibility rule is not violated). The default is **both**.

The **demand** parameter specifies whether to use RIP demand procedures when send and receiving RIP, and for routes received from this neighbour. If **no**, demand procedures are not used; if **yes**, they are used. The default is **no**.

The **authentication** parameter specifies the method used to authenticate RIP packets. This must be **none** unless using RIP version 2. If **none**, no authentication is used. If **password** is specified, a plaintext password is used to authenticate RIP packets; if **md5**, an encrypted password is used. The default is **none**.

The **password** parameter specifies the password to use if the **authentication** parameter is set to **password** or **md5**. This parameter is required when authentication is used. Although 63 characters are allowed as a password, only the first 16 are used. A warning to this effect is generated when the command is entered.

The **staticexport** parameter specifies whether static routing information is propagated from this interface. If **yes**, static routes are included in routing exports; if **no**, they are omitted. The default is **yes**.

Examples To broadcast RIP version 1 on an Ethernet interface (eth0), use the command:

```
add ip rip int=eth0
```

To send RIP version 2 on a demand interface (ppp0) with password authentication, but not accept any RIP packets on the interface, use the command:

```
add ip rip int=ppp0 sen=rip2 rec=no dem=yes au=pass
   pass=hanselandgretal
```

To receive RIP version 2 packets on an Ethernet interface (eth0) from one and only one host (172.16.248.33) and broadcast RIP version 1 packets on the interface, use the commands:

```
add ip rip int=eth0 ip=172.16.248.33 rec=rip sen=no
add ip rip int=eth0 rec=no
```

Related Commands

add ip egp
delete ip egp
delete ip rip
disable ip egp
disable ip exportrip
enable ip egp
enable ip exportrip
set ip rip
show ip
show ip rip

add ip route

Syntax

```
ADD IP ROUte=ipadd INTerface=interface NEXThop=ipadd
    [CIRCuit=miox-circuit] [DLCi=dlci] [MASK=ipadd]
    [METric=1..16] [METRIC1=1..16] [METRIC2=1..65535]
    [POLIcy=0..7] [PREFerence=0..65535] [TAG=1..65535]
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit).

Description

This command adds a static route to the IP route table. Static routes can be used to define default routes to external routers or networks. A default route is one with a network address of 0.0.0.0. When the router receives data and cannot find a route for it, it sends the data to the default route. To define a default route, **ipaddress** is set to 0.0.0.0 and **nexthop** points to the network (router) where default packets are to be directed. The static route must not already exist. However, if the route exists as a dynamic route (such as RIP-derived), the static route can still be added.A recommended limit of 300 static routes applies to devices with 16MB of DRAM or less.

This command also defines subnets. Multiple routes can be defined for a single interface (usually a LAN). This is useful for configuring more than one network or subnet on a particular interface. A common problem is when hosts exceed the capacity of a single subnet. Additional subnets can be assigned by adding static routes. In this case **ipaddress** is set to the new subnet, **nexthop** is set to 0.0.0.0, and **metric** set to 1.

The **route** parameter specifies the IP address of the static route.

The **interface** parameter specifies the IP interface with which the route is associated. The interface must already exist and be assigned to the IP module. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces. If the interface is a Frame Relay interface, the **dlci** parameter is required and specifies the DLC to use on the Frame Relay interface. If the interface is an X.25 DTE interface, the **circuit** parameter is required and specifies the name of a MIOX circuit already defined for the X.25 DTE interface.

The **nexthop** parameter specifies the IP address of the next hop (router) for the route. The default is the IP address of the interface specified by the **interface** parameter. For a PPP link, **nexthop** should be the IP address of the remote end of the PPP link.

The **mask** parameter specifies the subnet mask for the route. The default mask is determined using the following:

1. If **mask** is specified, use the specified mask.
2. If the route is the default route, use a mask of 0.0.0.0.
3. If the route is for a network to which the router is not attached, use the unsubnetted mask for the network class (A, B or C).
4. Otherwise, use the subnet mask of the specified interface. The subnet mask does not need to be specified in most cases.

In all cases a check is performed on the route and mask to verify that the route is the same before and after masking. This ensures that a static route is not specified to more than its subnet mask.

The **metric1** parameter specifies the cost of traversing the route for RIP. The default is 1. The normal range is from 2 to 16. A metric of 1 should be used if adding a subnet to an interface.The **metric** parameter is also accepted for backwards compatibility.

The **metric2** parameter specifies the cost of traversing the route for OSPF. The default is 1.

The **policy** parameter specifies the type of service for the route. The default is 0.

The **preference** parameter specifies the preference for the route. When more than one route in the route table matches the destination address in an IP packet, the route with the lowest preference value is used to route the packet. If two or more routes have the same preference, the route with the longest subnet mask is used. Interface routes have a preference of 0 and RIP routes have a preference of 100. The default preference for static routes other than 0.0.0.0 is 60. The default for the default static route 0.0.0.0 is 360.

The **tag** parameter specifies an integer to tag the route with. You can then match against this number in a route map and only import the appropriately-tagged routes into BGP.

Examples

To create a default route that points to a router at the remote end of a PPP link attached to interface ppp0 with the IP address 172.16.8.82, use the command:

```
add ip rou=0.0.0.0 int=ppp0 next=172.16.8.82 met=1
```

To add the subnet 172.16.9.0 to the existing subnet on interface eth0:

```
add ip rou=172.16.9.0 int=eth0 next=0.0.0.0 met=1
```

Adding static routes to get more local address space can cause problems with PC-based TCP/IP software. You may need to change the subnet mask on the PC so that the PC can see hosts on other subnets.

Related Commands

delete ip route
set ip route
show ip route

add ip route filter

Syntax

```
ADD IP ROUte FILter[=filter-id] IP=ipadd MASK=ipadd
   ACtion={INCLude|EXCLude} [DIrection={RECeive|SENd|
   BOTH}] [INTerface=interface] [NEXThop=ipadd]
   [POLIcy=0..7] [PROTocol={ANY|EGP|OSPF|RIP}]
```

where:

- *filter-id* is a number from 1 to 100.
- *ipadd* is an IP address in dotted decimal notation.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description

This command adds a route filter. A route filter controls which routes are sent and received by the routing protocols. Note that there are some filtering limitations. For more information, see “Routing Information Filters” on page 14-22. Route filters do not apply to static or interface routes.

When a route is received or transmitted by a routing protocol, the list of route filters is searched for a match to the route. Processing stops when a match is found or the end of the list is reached. If at least one route filter is defined, then the filter list has an implicit “exclude all” after the last entry in the list.

Therefore, it may be necessary to add an "include all" filter at the end of the list to allow all other routes that do not match.

The **filter** parameter specifies where the filter is inserted in the list. If this parameter is not specified, the filter is added to the end of the list.

The **ip** parameter specifies the network address to match. The wildcard character ("*") can be used to match a network range. For example, 192.168.*.* matches all destination networks that start with 192.168. The wildcard character can replace a complete number; for example, 192.168.*.* is valid but 192.16*.*.* is not.

The **mask** parameter specifies the network mask of the network to match. The wildcard character ("*") can be used to match a network mask range. For example, 255.255.*.* matches all destination network masks that start with 255.255. The wildcard character can replace a complete number; for example, 255.255.*.* is valid but 255.25*.*.* is not.

The **action** parameter specifies what to do with routes that match the filter. If **include** is specified, the route is included; if **exclude** is specified, it is excluded.

The **direction** parameter specifies whether to filter the route when receiving or sending it. If **receive** is specified, the **protocol** parameter specifies the routing protocol that receives the route information; if it is **send**, the **protocol** parameter specifies the routing protocol that advertises the routes.

The **interface** parameter specifies the interface to which the filter applies. If specified, the route is filtered when the route is sent or received on the interface. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces.

The **nexthop** parameter specifies the IP address of the next hop router to match. If specified, the route is filtered when the route is sent or received to or from the next hop.

The **policy** parameter specifies the type of service to filter. If not specified, all types of service are filtered.

The **protocol** parameter specifies the routing protocol to which the filter applies. The default is **any**. When **direction** is **receive**, then **protocol** specifies the routing protocol that receives the route information. If **direction** is **send**, **protocol** specifies the routing protocol that advertises the routes.

The way that the OSPF protocol works affects how the route filter operation on OSPF Link State Advertisement (LSA) works. A route filter with **direction=send** filters only matching routes regarded as Autonomous System (AS) external routes by OSPF. Also, the **interface** parameter is ignored, meaning all interfaces are treated indifferently.

Examples To add a route filter that includes RIP-derived routes from all sources, use the command:

```
add ip rou fil=1 prot=rip ac=incl di=both ip=*.*.*.*
   mask=*.*.*.*
```

To exclude all routes received from the 10.0.0.0 network from the route table, but include all other received routes in the route table, use the commands:

```
add ip rou fil=1 ip=10.0.0.0 mask=255.0.0.0 ac=excl di=rec
```

```
add ip rou fil=2 ip=*.*.*.* mask=*.*.*.* ac=incl
```

The second filter is necessary to override the effect of the implicit "exclude all" following the last entry in a filter list.

Related Commands

delete ip route filter
set ip route filter
show ip route filter

add ip route template

Syntax

```
ADD IP ROUte TEMPlate=name INTerface=interface
   NEXThop=ipadd [CIRCuit=miox-circuit] [DLCi=dlci]
   [METric=1..16] [METRIC1=1..16] [METRIC2=1..65535]
   [POLIcy=0..7] [PREFerence=0..65535]
```

where:

- *name* is a character string 1 to 31 characters long, and is not case-sensitive. Valid characters are any printable character. If *name* contains spaces, it must be in double quotes.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *ipadd* is an IP address in dotted decimal notation.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit).

Description This command adds an IP route template. IP route templates are used by the router to add IP routes to IP subnetworks discovered during normal operation by other protocols, such as IPsec. This is required when IP traffic to the discovered IP subnetwork needs to be routed via a route other than the default route.

The **interface** parameter specifies the IP interface with which any route added using this template is associated. The interface must already exist and be assigned to the IP module. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)

- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces.

If the interface is a Frame Relay interface, the **dlci** parameter is required and specifies the DLC to use on the Frame Relay interface. If the interface is an X.25 DTE interface, the **circuit** parameter is required and specifies the name of a MIOX circuit already defined for the X.25 DTE interface.

The **nexthop** parameter specifies the IP address of the next hop (router) for routes added with this template. The default is the IP address specified by the **interface** parameter. For a PPP link, **nexthop** should be the IP address of the remote end of the PPP link.

The **metric1** parameter specifies the cost of traversing routes added with this template for RIP. The default is 1. The normal range is from 2 to 16. A metric of 1 should be used when adding a subnet to an interface.The **metric** parameter is also accepted for backwards compatibility.

The **metric2** parameter specifies the cost of traversing any route added with this template for OSPF. The default is 1.

The **policy** parameter specifies the type of service for any route added using this template. The default is 0.

The **preference** parameter specifies the preference for routes added with this template. When more than one route in the route table matches the destination address in an IP packet, the route with the lowest preference value is used. If two or more routes have the same preference, the route with the longest subnet mask is used. Interface routes have a preference of 0 and RIP routes have a preference of 100. The default preference for static routes other than 0.0.0.0 is 60. The default for the default static route 0.0.0.0 is 360.

Examples To add an IP route template named “branch_office”, use the command:

```
add ip rou temp=branch_office int=vlan1 next=192.168.23.3
```

Related Commands

create ipsec policy
delete ip route template
set ip route template
show ip route template

add ip sa

Syntax `ADD IP SA=sa-id INTerface=interface`

where:

- *sa-id* is a number from 0 to 100.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description This command adds a security association to the list of security associations for an IP interface. IP SA commands provide support for RFCs 1825, 1827, and 1829, which have been superseded by IP Security. See Chapter 45, IP Security (IPsec) and RFCs 2401–2412 for more information about IPsec.

The **sa** parameter specifies the identifier of the security association. The security association must have been created previously using the **create sa** command on page 45-58 of Chapter 45, IP Security (IPsec).

The **interface** parameter specifies the name of the interface. The interface must already be assigned to the IP routing module. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces.

Examples To add security association 3 to the IP interface ppp0, use the command:

```
add ip sa=3 int=ppp0
```

Related Commands

create sa
delete ip sa
set ip interface
show ip interface
show ip sa

add ip trusted

Syntax

```
ADD IP TRusted=ipadd
```

where *ipadd* is an IP address in dotted decimal notation

Description

This command adds an entry to the trusted router table. This table acts as a filter that determines which sources of routing information (RIP) are to be accepted. It would be used in the situation where, for instance, the router is connected to a LAN to which several other routers are connected. It may be desirable for the router to route packets from networks known to the other routers (the usual case). In this case, the other routers broadcast routing information onto the LAN (such as RIP, EGP or OSPF), which is then picked up by the router and used to develop the internal routing table.

In the default case where no trusted routers have been specified, the router accepts all routing information unless the source has been filtered in some way. For example, it could be filtered using the **add ip filter** command on page 14-68. However, this blocks all information from being processed, including routing information. The related **add ip route filter** command on page 14-90 should be used for filtering routing information.

The trusted table ensures that the router's routing table is updated by *trusted* sources of routing information. Other routers are not filtered, but their routing information is not used until they are added to the table. A maximum of 32 trusted host addresses can be defined.

The **trusted** parameter specifies the IP address of a host from which RIP information is accepted. Adding one or more trusted routers automatically enables the trusted router option. If no trusted routers are defined, the router accepts routing information from any source.

Examples

To specify the host with an IP address of 172.16.8.33 as a trusted source of RIP information, use:

```
add ip tr=172.16.8.33
```

Related Commands

add ip filter
delete ip filter
delete ip trusted
set ip filter
show ip filter
show ip trusted

create ip pool

Syntax

```
CREate IP POOL=pool-name IP=ipadd[-ipadd]
```

where:

- *pool-name* is a character string 1 to 15 characters long. Valid characters are any printable characters. If *pool-name* contains spaces, it must be in double quotes.
- *ipadd* is an IP address in dotted decimal notation.

Description

This command creates a pool of IP addresses that can be used by ACC, PPP and other modules to assign IP addresses.

The **pool** parameter specifies a name for the IP address pool. The name is used in other commands to identify the pool.

The **ip** parameter specifies a range of IP addresses or a single one assigned to the pool. They should not overlap with IP address or ranges in other pools.

Examples

To create an IP pool named “dialin” with the IP addresses 192.168.1.1 to 192.168.1.16, use the command:

```
cre ip pool=dialin ip=192.168.1.1-192.168.1.16
```

Related Commands

destroy ip pool
show ip pool

delete bootp relay

Syntax

```
DELete BOOTp RELAy=ipadd
```

where *ipadd* is an IP address in dotted decimal notation

Description

This command deletes a BOOTP relay destination. The RELAY parameter specifies the IP address of a BOOTP server in dotted decimal notation.

Examples

To delete the BOOTP server with IP address 192.168.13.11, use:

```
del boot rela=192.168.13.11
```

Related Commands

add bootp relay
disable bootp relay
enable bootp relay
purge bootp relay
set bootp maxhops
show bootp relay

delete ip advertise interface

Syntax

```
DELete IP ADVertise INTerface=interface
```

where *interface* is an interface name formed by concatenating an interface type and an interface instance (such as vlan1)

Description

This command deletes ICMP Router Discovery advertising from a single physical IP interface and its configuration from a physical IP interface.

Example

To delete Router Discovery from vlan1, use the command:

```
del ip adv int=vlan1
```

Related Commands

add ip advertise interface
disable ip advertise
enable ip advertise
set ip advertise interface

delete ip arp

Syntax

```
DELete IP ARP=ipadd
```

where *ipadd* is an IP address in dotted decimal notation

Description

This command deletes a dynamic or static ARP entry from the ARP cache. The ARP entry must already exist. The ARP parameter specifies the IP address of the ARP entry to be deleted.

Examples

To delete an ARP entry for a host with an IP address of 172.16.9.197, use:

```
del ip arp=172.16.9.197
```

Related Commands

add ip arp
set ip arp
show ip arp

delete ip dns

Syntax

```
DELete IP DNS [DOMain={ANY|domain-name}]
```

where *domain-name* is a character string of up to 255 characters. Valid characters are uppercase and lowercase letters, digits (0-9), and the underscore character (“_”).

Description

This command deletes name server information from the DNS servers used by the router to resolve host names. When name server information is deleted, all DNS cache entries that were learned from those servers are removed from the cache.

The **domain** parameter specifies a domain name suffix for the name server configuration information to be deleted. If the **domain** parameter is not specified, the default DNS server configuration is deleted.

You cannot delete the default name server configuration while domain-specific name servers are configured.

Examples

To delete name server configuration information used for hosts in the domain “oranges.com”, use the command:

```
del ip dns dom=oranges.com
```

To delete the default name server configuration use the command:

```
del ip dns
```

Related Commands

add ip dns
set ip dns
show ip dns

delete ip egp

Syntax

DELete IP EGP=*ipadd*

where *ipadd* is an IP address in dotted decimal notation

Description

This command deletes an EGP neighbour so that exterior EGP routing information is longer exchanged with the specified EGP neighbour. If EGP is already enabled (with the **enable ip egp** command on page 14-121), the router disconnects the EGP connection to the neighbour.

Examples

To delete the router with IP address 172.16.248.33 as an EGP neighbour, use:

```
del ip egp=172.16.248.33
```

Related Commands

add ip egp
add ip rip
delete ip rip
disable ip egp
disable ip exportrip
enable ip egp
enable ip exportrip
set ip autonomous in Chapter 49, Border Gateway Protocol version 4 (BGP-4)
show ip
show ip egp
show ip rip

delete ip filter

Syntax

```
DELete IP FILter=0..399 ENTry={entry-number|ALL}
```

where *entry-number* is the position of this entry in the filter

Description

This command deletes an existing pattern from an IP traffic filter, policy filter, or priority filter.

The **filter** parameter specifies the number of the filter where the pattern is to be deleted.

- Filters with numbers from 0 to 99 are treated as traffic filters, and use the **action** parameter to specify the action to take with a packet that matches the pattern.
- Filters with numbers from 100 to 199 are treated as policy filters, and use the **policy** parameter to specify the policy to use when routing a packet that matches the pattern.
- Filters with numbers from 200 to 299 are treated as priority filters, and use the **priority** parameter to specify the priority to assign to a packet that matches the pattern.
- Filters from 300 to 399 are treated as routing filters, and use the **action** parameter to specify the action to take with a route that matches the pattern.
- Filters from 300 to 399 cannot be deleted if a BGP peer is using it, therefore the command fails.

The **entry** parameter specifies the entry number in the filter that is to be deleted. If **all** is specified, all entries in the filter are deleted. Existing patterns with the same or higher entry numbers are pushed up the filter to occupy the vacant entry.

Examples

To delete entry 3 from filter 2, use the command:

```
del ip fil=2 ent=3
```

Related Commands

add ip filter
add ip trusted
delete ip trusted
set ip filter
show ip filter
show ip trusted

delete ip helper

Syntax

```
DELete IP HElper DEStination=ipadd INTerface=interface
   POrt=port-number
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *port-number* is a UDP port number from 1 to 65535, or one of the predefined UDP port names DNS (port 53), NT or NETBIOS (ports 137 and 138), TACACS (port 49), TIME (port 37) or TFTP (port 69).

Description This command deletes either a port from the list of UDP ports to be forwarded or a destination IP address to which UDP broadcasts are being forwarded.

The **destination** parameter specifies the IP address to which the UDP broadcast traffic is forwarded.

The **interface** parameter specifies the interface to which the UDP port list is assigned. UDP broadcasts are forwarded that are received for the specified interface for one of the UDP ports in the UDP port list. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

The **port** parameter specifies the UDP port, as a decimal number from 1 to 65535, or the recognised name of a UDP port set. All broadcast traffic received by the router on the specified port or set of ports is redirected to the IP host at the destination address.

Examples To stop forwarding all NETBIOS broadcasts received via interface eth0 to IP address 192.168.202.3, use the command:

```
del ip he po=netbios des=192.168.202.3 int=eth0
```

To stop forwarding all broadcasts to UDP port 3001 received via interface eth0 to IP address 192.168.100.2, use the command:

```
del ip he po=3001 int=eth0 des=192.168.100.2
```

Related Commands

add ip helper
disable ip helper
enable ip helper
show ip helper

delete ip host

Syntax `DELete IP HOst=`*`name`*

where *name* is a character string up to 60 characters long. If the string contains spaces, it must be in double quotes.

Description This command deletes a user-defined name for an IP host from the host name table. The host name table makes it easier to Telnet to commonly accessed hosts by enabling the user to enter a shorter, easier to remember name for the host rather than the host's full IP address or domain name.

The **host** parameter specifies the user-defined name to be deleted. The specified host name must exist in the host name table.

Examples To delete the host name "zaphod" from the host name table, use:

```
del ip ho=Zaphod
```

Related Commands

add ip host
set ip host
set ip nameserver
set ip secondarynameserver
show ip host

delete ip interface

Syntax `DELete IP INTerface=`*`interface`*

where *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description This command deletes a logical interface from the IP module so that the logical interface is no longer used by the IP routing module.

The **interface** parameter specifies the name of the logical interface to be deleted. The interface must already be assigned to the IP routing module. At least two interfaces must be assigned to the IP module for the router to route IP packets, but only one interface (usually Ethernet) needs to be assigned when the router is acting as a server. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

When an IP interface is deleted, static routes and ARP entries related to the interface are also deleted.

Examples To delete PPP interface 2, use:

```
del ip int=ppp2
```

To delete the third logical interface attached to PPP0, use:

```
del ip int=ppp0-2
```

Related Commands

add ip interface
disable ip interface
enable ip interface
reset ip interface
set ip interface
show ip interface

delete ip local

Syntax `DELete IP LOCal=1..15`

Description This command deletes a local interface from the IP module. The selected local interface will no longer be used by the IP routing module.

When an IP interface is deleted, any static routes and ARP entries specific to the interface will also be deleted

Examples To delete local interface 5, use:

```
del ip local=5
```

Related Commands

add ip local
set ip local
show ip interface

delete ip nat

Syntax

```
DELete IP NAT IP=ipadd MASK=ipadd GBLINterface=interface
    [GBLMask=ipadd] [GBLPort=port] [POrt=port]
    [PROTocol={protocol|ALL|EGP|GRE|ICmp|OSPF|SA|TCp|UDp}]

DELete IP NAT IP=ipadd GLBIPaddress=ipadd [MASK=ipadd]
    [GBLMask=ipadd] [GBLPort=port] [POrt=port]
    [PROTocol={protocol|ALL|EGP|GRE|ICmp|OSPF|SA|TCp|UDp}]
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *port* is an IP port number or the predefined name for an IP service.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *protocol* is an IP protocol number.

Description

This command deletes a NAT or ENAT mapping. The first variant deletes an interface NAT. The **gblinterface** parameter is required and the **gblipaddress** parameter is not valid. The second variant deletes any other type of NAT. The **gblipaddress** parameter is required and the **gblinterface** parameter is not valid.

The **ip** parameter specifies either a host or network IP address for the private network. This parameter can be used with the **mask** parameter to specify a range of IP addresses for a private network.

The **protocol** parameter specifies the IP protocol number or the name of a predefined protocol type to be used with the static ENAT entry. If TCP or UDP is specified, then the **port** parameter must also be specified.

The **port** parameter specifies the port number or service name (Table 14-15 on page 14-84) for the port used on the private IP host when specifying a static ENAT entry.

The **gblipaddress** parameter specifies either an officially assigned global IP address or the start of a range of addresses. This parameter can be used with the **gblmask** parameter to specify the range of global IP addresses for an entry.

The **gblport** parameter specifies the port number or service name (Table 14-15 on page 14-84) for the port available to global Internet access when creating a static ENAT.

The **gblinterface** parameter specifies the interface that has or will dynamically obtain an officially assigned global IP address. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

Examples To delete a static NAT mapping between the private IP address 10.1.1.2 and the officially assigned global IP address 202.1.1.1, use the command:

```
del ip nat IP=10.1.1.2 gblip=202.1.1.1
```

To delete a static ENAT entry to allow access to a WWW server at IP address 192.168.100.54 on the private network, use the command:

```
del ip nat ip=192.168.100.54 prot=tcp po=80 gblip=202.1.1.1
   gblp=80
```

Related Commands

add ip nat
disable ip nat
enable ip nat
show ip nat

delete ip rip

Syntax

```
DELete IP RIP INTerface=interface [CIRCuit=miox-circuit]
   [DLCi=dlci] [IP=ipadd]
```

where:

- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit) from 0 to 1023.
- *ipadd* is an IP address in dotted decimal notation.

Description This command deletes a RIP neighbour. Use this command to stop sending and/or receiving RIP to and/or from a RIP neighbour.

The **interface** parameter specifies the interface via which RIP packets are received from the RIP neighbour. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

The **circuit** parameter specifies the X.25 circuit on which to send or receive RIP packets. It is a required parameter for X25T interfaces and is valid when the interface is an X25T interface.

The **dlci** parameter specifies the Frame Relay DLC on which to send or receive RIP packets. It is a required parameter for Frame Relay interfaces and is valid when the interface is a Frame Relay interface.

The **ip** parameter specifies the IP address of the neighbour to delete.

Examples To delete a neighbour that is broadcasting RIP on an Ethernet interface (eth0), use the command:

```
del ip rip int=eth0
```

To delete a neighbour that is sending to a specific IP address on a PPP interface, use the command:

```
del ip rip int=ppp0 IP=172.16.248.33
```

Related Commands

add ip egp
add ip rip
delete ip egp
disable ip egp
disable ip exportrip
enable ip egp
enable ip exportrip
set ip rip
show ip
show ip rip

delete ip route

Syntax

```
DELete IP ROUte=ipadd MASK=ipadd INTerface=interface
   NEXThop=ipadd
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description This command deletes an existing static route from the IP route table. However, if the route exists as a dynamic route (such as RIP-derived), the static route may not be deleted. A maximum of 300 static routes can be defined.

The **route** parameter specifies the IP address of the static route.

The **interface** parameter specifies the IP interface with which the route is associated. The interface must already exist and be assigned to the IP module. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

The **nexthop** parameter specifies the IP address of the next hop (router) for the route. The default is the IP address of the interface specified by the **interface** parameter. For a PPP link, **nexthop** should be the IP address of the remote end of the PPP link.

The **mask** parameter specifies the subnet mask for the route. A check is performed on the route and mask to verify that the route is the same before and after masking. This ensures that a static route is not specified to more than its subnet mask.

Examples To delete a default route that points to a router at the remote end of a PPP link attached to interface ppp0, with the IP address 172.16.8.82, use the command:

```
del ip rou=0.0.0.0 mask=0.0.0.0 int=pp0 next=172.16.8.82
```

Related Commands

add ip route
set ip route
show ip route

delete ip route filter

Syntax

```
DELete IP ROUte FILter=1..100
```

Description This command deletes a route filter. A route filter controls which routes are sent and received by the routing protocols.

The **filter** parameter specifies the index in the filter list of the filter to delete. The specified entry must exist.

Examples To delete route filter 3, use the command:

```
del rou fil=3
```

Related Commands

add ip route filter
set ip route filter
show ip route filter

delete ip route template

Syntax `DELete IP ROUte TEMPlate=`*`name`*

where *name* is a character string 1 to 31 characters long, and is not case-sensitive. Valid characters are any printable character. If *name* contains spaces, it must be in double quotes.

Description This command deletes the specified IP route template.

Examples To delete an IP route template named "branch_office", use the command:

```
del ip rou temp=branch_office
```

Related Commands

add ip route template
create ipsec policy
set ip route template
show ip route template

delete ip sa

Syntax `DELete IP SA=`*`sa-id`* `INTerface=`*`interface`*

where:

- *sa-id* is a number from 0 to 100.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description This command deletes a security association from the list of security associations for an IP interface.

The **ip sa** commands provide support for RFCs 1825, 1827, and 1829, which have been superseded by IP Security. See Chapter 45, IP Security (IPsec) and RFCs 2401–2412 for more information about IPsec.

The **sa** parameter specifies the identifier of the security association. The security association must have been created previously using the **create sa** command on page 45-58 of Chapter 45, IP Security (IPsec), and must currently be assigned to the interface.

The **interface** parameter specifies the name of the interface. The interface must already be assigned to the IP routing module. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

Examples To delete security association 3 from the IP interface ppp0, use the command:

```
del ip sa=3 int=ppp0
```

Related Commands

add ip sa
create sa
set ip interface
show ip interface
show ip sa

delete ip trusted

Syntax

```
DELete IP TRusted=ipadd
```

where *ipadd* is an IP address in dotted decimal notation

Description This command deletes an entry from the trusted router table. This table acts as a filter that determines which sources of routing information (RIP) are to be accepted. For example, it would be used in the situation where the router is connected to a LAN to which several other routers are connected. It may be desirable for the router to route packets from networks known to the other routers (the usual case). In which case, the other routers broadcast routing information onto the LAN (such as RIP, EGP or OSPF), which is then picked up by the router and used to develop the internal routing table.

In the default case where no trusted routers have been specified, the router accepts all routing information unless the source has been filtered in some way. For example, it could be filtered using the **add ip filter** command on page 14-68. However, this blocks all information from being processed, including routing information. The related **add ip route filter** command on page 14-90 should be used for filtering routing information.

The trusted table ensures that the router’s routing table is updated by *trusted* sources of routing information. Other routers are not filtered, but their routing information is not used until they are added to the table.

The **trusted** parameter specifies the IP address of a host from which RIP information is no longer accepted. Deleting all trusted routers automatically disables the trusted router option.

Examples To delete the host with an IP address of 172.16.8.33 as a trusted source of RIP information, use:

```
del ip tr=172.16.8.33
```

Related Commands

add ip filter
add ip trusted
delete ip filter
set ip filter
show ip filter
show ip trusted

delete tcp

Syntax

```
DELete TCP=tcb
```

where *tcb* is the index of a TCP connection in the TCP connection table

Description This command deletes an active TCP session. The TCP parameter specifies the index in the TCP connection table of the TCP connection to be deleted. The index can be obtained from the output of the **show tcp** command. TCP sessions in the listen state cannot be deleted.

Examples To delete TCP session number 6, use the command:

```
del tcp=6
```

Related Commands **show tcp**

destroy ip pool

Syntax

```
DESTroy IP POOL=pool-name
```

where *pool-name* is a character string 1 to 15 characters long. Valid characters are any printable characters. If *pool-name* contains spaces, it must be in double quotes.

Description This command destroys an existing pool of IP addresses. An IP address pool can be destroyed when there are no IP addresses in use. The **pool** parameter specifies the name of the IP address pool.

Examples To destroy the IP pool named “dialin”, use the command:

```
dest ip pool=dialin
```

Related Commands **create ip pool**
show ip pool

disable bootp relay

Syntax

```
DISable BOOTp RELAy
```

Description This command disables the BOOTP Relay Agent. The BOOTP Relay Agent relays BOOTREQUEST messages originating from any of the router’s interfaces to a user-defined destination, and relays BOOTREPLY messages addressed to BOOTP clients on networks directly connected to the router. BOOTREPLY messages addressed to clients on networks not directly connected to the router are ignored by the relay agent and treated as ordinary IP packets for forwarding. The BOOTP Relay Agent is disabled by default.

Examples To disable the BOOTP relay agent, use the command:

```
dis boot rela
```

Related Commands

add bootp relay
delete bootp relay
enable bootp relay
purge bootp relay
set bootp maxhops
show bootp relay

disable ip

Syntax

```
DISable IP
```

Description This command disables the IP routing module when it is enabled. The router no longer routes IP packets, responds to SNMP requests, uses TFTP to download software upgrades, or provides Telnet services. By default the IP module is disabled. The current operational mode of the IP module is retained so it can be restored when the IP module is enabled again.

The IP module operates in server mode or forwarding mode. In server mode, the router does not route IP packets, but provides Telnet services, responds to SNMP requests, and uses TFTP to download software upgrades. In forwarding mode, the router routes IP packets, as well as performing all the functions of server mode. The default is forwarding.

Related Commands

disable ip forwarding
disable ip srcroute
enable ip
enable ip forwarding
enable ip srcroute
show ip

disable ip advertise

Syntax

```
DISable IP ADVertise
```

Description This command globally disables ICMP Router Discovery advertisements on the device. All transmitting and processing of Router Discovery messages ceases immediately on all interfaces.

Examples To disable Router Discovery advertisements, use the command:

```
dis ip adv
```

Related Commands

delete ip advertise interface
enable ip advertise
set ip advertise interface
show ip advertise

disable ip arp log

Syntax `DISable IP ARP LOG`

Description This command disables logging of all MAC and IP addresses of all equipment connected to the router LAN interfaces accessing the WAN interface.

Related Commands **enable ip arp log**

disable ip debug

Syntax `DISable IP DEBug`

Description This command disables the IP debugging facility. Incorrectly formatted IP packets are buffered for later diagnosis. Up to 40 packets can be stored in the buffer, with subsequent packets replacing the oldest packets. The packet headers can be displayed with the **show ip debug** command on page 14-182. The debugging facility is disabled by default.

Related Commands **enable ip debug**
show ip debug
show ip

disable ip dnsrelay

Syntax `DISable IP DNSRelay`

Description This command disables the DNS relay agent. The router stops forwarding DNS requests from hosts to the router's own configured DNS server. The DNS relay agent is disabled by default.

Related Commands **enable ip dnsrelay**
set ip dnsrelay
show ip

disable ip echoreply

Syntax

```
DISable IP ECHoreply
```

Description This command disables the generation of ICMP echo reply messages in response to ICMP echo request messages. Echo reply messages are enabled by default.

Related Commands **enable ip echoreply**

disable ip egp

Syntax

```
DISable IP EGP
```

Description This command disables the EGP routing module and disconnects all connections to EGP neighbours. The EGP module must already be enabled. The EGP module is disabled by default.

Related Commands

add ip egp
add ip rip
delete ip egp
delete ip rip
disable ip exportrip
enable ip egp
enable ip exportrip
show ip
show ip egp
show ip rip

disable ip exportrip

Syntax

```
DISable IP EXPortrip
```

Description This command disables the transfer of RIP routing information to outgoing EGP messages, preventing interior routing information from being transmitted to exterior routers. This option is disabled by default.

Related Commands

add ip egp
add ip rip
delete ip egp
delete ip rip
disable ip egp
enable ip egp
enable ip exportrip
set ip autonomous in Chapter 49, Border Gateway Protocol version 4 (BGP-4)
show ip egp
show ip rip

disable ip fofilter

Syntax

```
DISable IP FOFilter
```

Description

This command disables the filtering (discarding) of IP packets with a fragment offset of 1, and is intended for use in secure environments to prevent attacks by intruders using *tiny fragments* or *overlapping fragments* (see RFC 1858 for a detailed description). By default, the filter is enabled.

In the *tiny fragment* attack, the attacker transmits IP packets of the minimum fragment size. The first packet contains the IP header and 8 octets of data, which is insufficient to hold a complete TCP header. The TCP flags field is forced into the second fragment. Filters that attempt to discard connection requests (TCP datagrams with the SYN bit set and the ACK bit clear) are unable to test these flags in the first fragment and typically ignore them in subsequent fragments. As a result, the IP packet is not discarded. The fragment offset filter discards all fragments with a fragment offset of one, preventing reassembly of the packet at the receiving host.

In the *overlapping fragment* attack, the attacker transmits IP packets in fragments that overlap in an attempt to circumvent filters that discard connection requests. The first fragment contains a complete TCP header (so it avoids filters that discard fragments with a fragment offset of one) with the SYN bit clear and the ACK bit set (so it passes filters that discard connection requests). The second fragment has an offset of eight octets and contains another set of TCP flags, this time with the SYN bit set and the ACK bit clear. Typically, this fragment is passed by the filter, and at the receiving host the reassembly process results in the second fragment partially overwriting the first fragment.

The fragment offset filter discards all fragments with a fragment offset of one, preventing reassembly of the packet at the receiving host and effectively preventing both tiny fragment and overlapping fragment attacks.

If IP traffic filters have been created to drop connection requests (with the **session=start** parameter of the **add ip filter** command on page 14-68 or the **set ip filter** command on page 14-140), the fragment offset filter should be enabled to prevent tiny fragment and offset fragment attacks from circumventing the IP traffic filters.

Related Commands

add ip filter
delete ip filter
enable ip fofilter
set ip filter
show ip filter

disable ip forwarding

Syntax

```
DISable IP FORwarding
```

Description

This command sets the IP module's operational mode to server, which disables the routing function. This flushes all dynamic routes, ARPs, and L3 table entries so that forwarding stops. The IP module must already be enabled and in forwarding mode.

The IP module operates in one of two modes: server or forwarding. In server mode, the router does not route IP packets, but provides Telnet services, responds to SNMP requests, and uses TFTP to download software upgrades. In forwarding mode, the router routes IP packets as well as performing all functions of the server mode. The default is forwarding.

Related Commands

disable ip
disable ip srcroute
enable ip
enable ip forwarding
enable ip srcroute
show ip

disable ip helper

Syntax

```
DISable IP HElper
```

Description

This command disables the forwarding of broadcast UDP traffic on specific UDP ports to specific destination IP addresses.

Examples

To disable broadcast forwarding, use the command:

```
dis ip he
```

Related Commands

add ip helper
delete ip helper
enable ip helper
show ip helper

disable ip icmpreply

Syntax

```
DISable IP ICMPreply[={ALL|NETunreach|HOSTunreach|
    REDirect}]
```

Description

This command disables ICMP reply messages.

If **all** is specified, all configurable ICMP message replies are disabled.

If **netunreach** is specified, all network unreachable message replies are disabled (RFC792 Type 3 Code 0).

If **hostunreach** is specified, all host unreachable message replies are disabled (RFC792 Type 3 Code 1).

If **redirect** is specified, all ICMP redirect message replies are disabled (RFC792 Type 5 Code 0, 1, 2, 3).

Example

To disable all configurable ICMP messages, use the command:

```
dis ip icmp=all
```

Related Commands

enable ip icmpreply
show ip icmpreply

disable ip interface

Syntax

```
DISable IP INTerface=interface
```

where *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description

This command temporarily disables a logical IP interface. The logical interface is not used by the IP routing module. The effect is equivalent to physically disconnecting the router from the attached IP network. Routes associated with a disabled interface are not explicitly deleted. However, routes learned via a routing protocol such as RIP are eventually deleted by the routing protocol's aging mechanism. Static routes are retained until explicitly removed by deleting the specific static route entry or by deleting the IP interface.

The **interface** parameter specifies the name of the logical interface to be disabled. The interface must be assigned to the IP routing module and currently enabled. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)

- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

Examples To disable the first logical IP interface attached to PPP0, use the command:

```
dis ip int=ppp0-0
```

Related Commands

add ip interface
delete ip interface
enable ip interface
reset ip interface
set ip interface
show ip interface

disable ip nat

Syntax

```
DISable IP NAT [FRAgment={ICMP|UDP|OTHER}[,...]]
    [LOG={ALL|FAILS|INTCP|INUDP|OUTTCP|OUTUDP}[,...]]
```

Description This command disables NAT, disables enhanced packet fragment handling when NAT is in use, or disables the logging of a specific class of NAT events.

The **fragment** parameter specifies that IP packets for a specific protocol be dropped when NAT is used when they have been fragmented into more than 8 fragments, or have a total payload of more than 1780 bytes of protocol data. This restores the default functionality where fragmented packets are permitted when there are no more than 8 fragments and the combined payload consists of a maximum of 1780 bytes. There is no default.

The **log** parameter specifies the class of NAT event to log. The **fails** option logs IP traffic received by the global interface of the router that could not be delivered because a service had not been specified on the private network. The **intcp** option logs TCP session opens to servers on the private network. The **inudp** option logs UDP flows initiated to a server on the private network. The **outtcp** option logs TCP sessions originating on the private network destined for the Internet. The **outudp** option logs UDP flows originating on the private network destined for the Internet. Messages are logged by the router's Logging facility. Logging multiple classes of events can be disabled by entering a comma-separated list of event classes.

NAT is automatically disabled when the firewall is enabled because the firewall provides NAT services. However, the NAT configuration is retained so that it can be manually enabled again if the firewall is disabled.

Examples To disable NAT, use the command:

```
dis ip nat
```

To disable the logging of inward and outward TCP connections to a server, use the command:

```
dis ip nat log=intcp,outtcp
```

Related Commands add ip nat
delete ip nat
enable ip nat
show ip nat

disable ip remoteassign

Syntax

```
DISable IP REMoteassign
```

Description This command disables the remote assignment of IP addresses for unnumbered PPP interfaces. The router does not allow a remote PPP peer to set the IP address of the local PPP interface

Examples To disable remote IP address assignment, use the command:

```
dis ip rem
```

Related Commands enable ip remoteassign
show ip

disable ip route

Syntax

```
DISable IP ROUte [CAChe|COUnt|MULtipath|DEBug]
```

Description This command disables route caching, route counters, or equal cost multipath routing.

The **cache** parameter disables route caching. The cache is enabled by default.

The **count** parameter disables counting of octets sent and received to and from a network. It is disabled by default.

The **multipath** parameter disables equal cost multipath routing (ECMP). The router still learns multiple routes, but only forwards packets over the best route to the destination. ECMP is enabled by default.

The **debug** parameter disables the IP debugging facility.

Examples To disable equal cost multipath routing, use the command:

```
dis ip rou mul
```

Related Commands enable ip route
show ip route

disable ip srcroute

Syntax

```
DISable IP SRCRoute[={LOOSE|STrict|ALL}]
```

Description

This command disables the forwarding of source-routed IP packets. If a specific type of source-routed IP packet is specified, forwarding of that type will be enabled. Otherwise, forwarding of all source-routed IP packets will be disabled.

When forwarding is enabled, source-routed IP packets are forwarded by the router as normal. When forwarding is disabled, source-routed packets are discarded by the router. The default is to disable forwarding.

Source routing is rarely used for legitimate purposes, and is a common method used to circumvent packet-filtering firewalls and to masquerade as a trusted host inside the destination network. This command is therefore an extra security feature, which is why source routed packets are discarded by default.

When forwarding IP source routed datagrams is disabled, all source routed packets are logged by the Logging facility with a message type/subtype of IPFIL/SRCRT.

Related Commands

disable ip
enable ip
enable ip forwarding
enable ip srcroute
show ip

disable telnet server

Syntax

```
DISable TELnet SErver
```

Description

This command blocks telnet access to the router. Telnet access is enabled by default. For security reasons, you may want to disable telnet access to the router.

Example

To disable telnet access to the router, use the command:

```
dis tel se
```

. **Related Commands**

enable telnet server
set telnet
show telnet
telnet

enable bootp relay

Syntax

```
ENAble BOOTp RELAy
```

Description

This command enables the BOOTP Relay Agent. The BOOTP Relay Agent relays BOOTREQUEST messages originating from any of the router’s interfaces to a user-defined destination, and relays BOOTREPLY messages addressed to BOOTP clients on networks directly connected to the router. BOOTREPLY messages addressed to clients on networks not directly connected to the router are ignored by the relay agent and treated as ordinary IP packets for forwarding. The BOOTP Relay Agent is disabled by default.

Related Commands

add bootp relay
delete bootp relay
disable bootp relay
purge bootp relay
set bootp maxhops
show bootp relay

enable ip

Syntax

```
ENAble IP
```

Description

This command enables the IP routing module when it has been disabled. The IP module is disabled by default.

The IP module operates in server mode or forwarding mode, and the operational mode is restored from when the IP module was last disabled. In server mode, the router does not route IP packets, but provides Telnet services, responds to SNMP requests, and uses TFTP to download software upgrades. In forwarding mode, the router routes IP packets, as well as performing all the functions of server mode. The default is forwarding.

Related Commands

disable ip
disable ip forwarding
disable ip srcroute
enable ip forwarding
enable ip srcroute
show ip

enable ip advertise

Syntax

```
ENAble IP ADVertise
```

Description

This command globally enables ICMP Router Discovery advertisements on the router. However, the device does not send or process Router Discover messages until at least one IP interface is configured with the **add ip advertise interface** command on page 14-63.

Examples

To enable Router Discovery advertisements, use the command:

```
ena ip adv
```

Related Commands

add ip advertise interface
add ip interface
disable ip advertise
set ip advertise interface
set ip interface
show ip advertise

enable ip arp log

Syntax

```
ENAble IP ARP LOG
```

Description

This command enables logging of all MAC and IP addresses of all equipment connected to the router LAN interfaces accessing the WAN interface. This occurs at the time these addresses are added to or deleted from the router's ARP table.

Related Commands

disable ip arp log

enable ip debug

Syntax

```
ENAble IP DEBug[=PACket]
```

Description

This command enables the IP debugging facility, which is disabled by default. The **packet** option prints all packet headers coming in and going out of all IP interfaces.

Without the **packet** option, the command logs incorrectly formatted IP packets for later diagnosis. Up to 40 packets can be stored in the buffer, with subsequent packets replacing the oldest packets. The contents of each packet and the reason for its rejection are stored. Packet headers can be displayed with the **show ip debug** command on page 14-182.

Related Commands

disable ip debug
show ip debug
show ip

enable ip dnsrelay

Syntax

```
ENAble IP DNSRelay
```

Description This command enables the DNS relay agent so that the router forwards DNS requests from hosts to the router's own configured DNS server. The DNS relay agent is disabled by default.

Related Commands

disable ip dnsrelay
set ip dnsrelay
show ip

enable ip echoreply

Syntax

```
ENAble IP ECHoreply
```

Description This command enables the generation of ICMP echo reply messages in response to ICMP echo request messages. Echo reply messages are enabled by default.

Related Commands

disable ip echoreply

enable ip egp

Syntax

```
ENAble IP EGP
```

Description This command enables the EGP routing module and attempts to start a connection to all statically defined EGP neighbours. The EGP module must not already be enabled. The EGP module is disabled by default.

Related Commands

add ip egp
add ip rip
delete ip egp
delete ip rip
disable ip egp
disable ip exportrip
enable ip exportrip
set ip autonomous in Chapter 49, Border Gateway Protocol version 4 (BGP-4)
show ip
show ip egp
show ip rip

enable ip exportrip

Syntax

```
ENAble IP EXPortrip
```

Description

This command enables the transfer of RIP routing information to outgoing EGP messages, enabling interior routing information to be transmitted to exterior routers. This option is disabled by default.

Related Commands

add ip egp
add ip rip
delete ip egp
delete ip rip
disable ip egp
disable ip exportrip
enable ip egp
set ip autonomous in Chapter 49, Border Gateway Protocol version 4 (BGP-4)
show ip egp
show ip rip

enable ip fofilter

Syntax

```
ENAble IP FOFilter
```

Description

This command enables the filtering (discarding) of IP packets with a fragment offset of 1, and is intended for use in secure environments to prevent attacks by intruders using *tiny fragments* or *overlapping fragments* (see RFC 1858 for a detailed description). By default, the filter is enabled.

In the *tiny fragment* attack, the attacker transmits IP packets of the minimum fragment size. The first packet contains the IP header and 8 octets of data, which is insufficient to hold a complete TCP header. The TCP flags field is forced into the second fragment. Filters that attempt to discard connection requests (TCP datagrams with the SYN bit set and the ACK bit clear) are unable to test these flags in the first fragment and typically ignore them in subsequent fragments. As a result, the IP packet is not discarded. The fragment offset filter discards all fragments with a fragment offset of one, preventing reassembly of the packet at the receiving host.

In the *overlapping fragment* attack, the attacker transmits IP packets in fragments that overlap in an attempt to circumvent filters that discard connection requests. The first fragment contains a complete TCP header (so it avoids filters that discard fragments with a fragment offset of one) with the SYN bit clear and the ACK bit set (so it passes filters that discard connection requests). The second fragment has an offset of eight octets and contains another set of TCP flags, this time with the SYN bit set and the ACK bit clear. Typically, this fragment is passed by the filter, and at the receiving host the reassembly process results in the second fragment partially overwriting the first fragment.

The fragment offset filter discards all fragments with a fragment offset of one, preventing reassembly of the packet at the receiving host and effectively preventing both tiny fragment and overlapping fragment attacks.

IP datagrams discarded by the fragment offset filter are logged by the Logging facility with a message type/subtype of IPFIL/FRAG.

If IP traffic filters have been created to drop connection requests (using the **session=start** parameter of the **add ip filter** command on page 14-68 or the **set ip filter** command on page 14-140), the fragment offset filter should be enabled to prevent tiny fragment and offset fragment attacks from circumventing the IP traffic filters.

Related Commands

add ip filter
delete ip filter
disable ip fofilter
set ip filter
show ip filter

enable ip forwarding

Syntax

```
ENAble IP FORwarding
```

Description

This command sets the IP module's operational mode to a routing function. The IP module must already be enabled and in server mode.

The IP module operates in one of two modes: server or forwarding. In server mode, the router does not route IP packets, but provides Telnet services, responds to SNMP requests, and uses TFTP to download software upgrades. In forwarding mode, the router routes IP packets as well as performing all functions of the server mode. The default is forwarding.

Related Commands

disable ip
disable ip forwarding
disable ip srcroute
enable ip
enable ip srcroute
show ip

enable ip helper

Syntax

```
ENAble IP HElper
```

Description

This command enables the forwarding of broadcast UDP traffic on specified UDP ports to specified destination IP addresses.

Examples

To enable broadcast forwarding, use the command:

```
ena ip he
```

Related Commands

add ip helper
delete ip helper
disable ip helper
show ip helper

enable ip icmpreply

Syntax

```
ENAble IP ICMPreply[={ALL|NETunreach|HOSTunreach|
   REDirect}]
```

Description

This command enables ICMP reply messages.

If **all** is specified, all configurable ICMP message replies are enabled. If **netunreach** is specified, all network unreachable message replies are enabled (RFC 792 Type 3 Code 0). If **hostunreach** is specified, all host unreachable message replies are enabled (RFC 792 Type 3 Code 1). If **redirect** is specified, all ICMP redirect message replies are enabled (RFC 792 Type 5 Code 0, 1, 2, 3).

Example

To enable all configurable ICMP messages, use the command:

```
ena ip icmp=all
```

Related Commands

disable ip icmpreply
show ip icmpreply

enable ip interface

Syntax

```
ENAble IP INTerface=interface
```

where *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description

This command enables a logical IP interface for use by the IP routing module.

The **interface** parameter specifies the name of the logical interface to be enabled. The interface must be assigned to the IP routing module and currently disabled. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

Examples

To enable the first logical IP interface attached to PPP0, use the command:

```
ena ip int=ppp0-0
```

Related Commands

add ip interface
delete ip interface
disable ip interface
reset ip interface
set ip interface
show ip interface

enable ip nat

Syntax

```
ENAble IP NAT [FRAgment={ICMP|UDP|OTHER}[,...]] [LOG={ALL|
    FAILS|INTCP|INUDP|OUTTCP|OUTUDP}[,...]]
```

Description

This command enables NAT, enables enhanced packet fragment handling when NAT is in use, or enables the logging of a class of specific NAT events.

The **fragment** parameter specifies that IP packets for a specific protocol be permitted when NAT is used when they have been fragmented into more than 8 fragments. There is no limit on total data within this number of fragments other than the MTU restrictions of the interfaces involved in forwarding the packets. If **icmp** is specified, IP NAT permits ICMP ping (echo) requests and replies that have been fragmented into more than 8 fragments. The default behaviour is that fragmented packets are permitted when there are no more than 8 fragments and the combined protocol data consists of a maximum of 1780 bytes. The number of fragments that can be handled is configured by the **set ip nat maxfragments** command on page 14-152.

The **log** parameter specifies the class of NAT event to log. The **fails** option logs IP traffic received by the global interface of the router that could not be delivered because a service had not been specified on the private network. The **intcp** option logs TCP session opens to servers on the private network. The **inudp** option logs all UDP flows initiated to a server on the private network. The **outtcp** option logs TCP sessions originating on the private network destined for the Internet. The **outudp** option logs UDP flows originating on the private network destined for the Internet. Messages are logged by the router's Logging facility. Logging multiple classes of events can be disabled by entering a comma-separated list of event classes.

NAT is automatically disabled when the firewall is enabled because the firewall provides NAT services. However, the NAT configuration is retained so that it can be manually enabled again if the firewall is disabled.

Examples

To enable NAT, use the command:

```
ena ip nat
```

To enable the logging of inward and outward TCP connections to a server, use the command:

```
ena ip nat log=intcp,outtcp
```

Related Commands

add ip nat
delete ip nat
disable ip nat
show ip nat

enable ip remoteassign

Syntax

```
ENAble IP REMoteassign
```

Description

This command enables the remote assignment of IP addresses for unnumbered PPP interfaces.

If a PPP interface is created with an IP address of 0.0.0.0, and remote IP address assignment is enabled, then during the IP control protocol (IPCP) negotiation process the router allows the remote PPP peer to set the IP address of the local PPP interface. If the local PPP interface has an IP number other than 0.0.0.0, or if remote IP address assignment is disabled, the router does not allow the remote PPP peer to set the IP address of the local PPP interface.

Examples

To enable remote IP addresses to be assigned, use the command:

```
ena ip rem
```

Related Commands

disable ip remoteassign
show ip

enable ip route

Syntax

```
ENAble IP ROUte [CAChe|COUnt|MULtipath|DEBug]
```

Description

This command enables route caching, route counters, or equal cost multipath routing. Route caching is enabled by default.

The **cache** parameter enables route caching. The cache is enabled by default.

The **count** parameter enables counting of octets sent and received to and from a network. It is disabled by default.

The **multipath** parameter enables equal cost multipath routing (ECMP). If the router learns multiple routes of the same cost to a destination, it distributes the packets across all the routes. You can have up to 16 individual routes to a destination. For more information see "Equal Cost Multipath Routing" on page 14-21. ECMP routing is enabled by default.

The **debug** parameter enables the IP debugging facility. Incorrectly formatted IP packet headers are captured for later analysis.

Examples

To enable byte counting for routes, use the command:

```
ena ip rou cou
```

Related Commands

disable ip route
show ip route

enable ip srcroute

Syntax

```
ENAble IP SRCRoute[={LOOSE|STrict|ALL}]
```

Description

This command enables the forwarding of source-routed IP packets. If a specific type of source-routed IP packet is specified, forwarding of that type will be enabled. Otherwise, forwarding of all source-routed IP packets will be enabled.

When forwarding is enabled, source-routed IP packets are forwarded by the router as normal. When forwarding is disabled, source-routed packets are discarded. The default is to disable forwarding.

Source routing is rarely used for legitimate purposes, and is a common method used to circumvent packet-filtering firewalls and to masquerade as a trusted host inside the destination network. This command is therefore an extra security feature, which is why source routed packets are discarded by default.

Related Commands

disable ip
disable ip srcroute
enable ip
enable ip forwarding
show ip

enable telnet server

Syntax

```
ENAble TELnet SErver
```

Description

This command allows remote users to telnet to the router. Telnet access is enabled by default. For security reasons, you may want to disable telnet access to the router.

Example

To enable telnet access to the router, use the command:

```
ena tel se
```

Related Commands

disable telnet server
set telnet
show telnet
telnet

finger

Syntax

FINGer [*username*]@*host*[@*host*]... [DETail={HIgh|LOW}]

where:

- *username* is the account name from 1 to 20 characters long to be fingered. Valid characters are uppercase and lowercase letters, digits (0–9), and the underscore (“_”). Wildcards are not allowed.
- *host* is the finger server to be queried, an IP address in dotted decimal notation or a host name from the host name table.

Description

This commands sends a finger query to the finger server on the specified host or hosts. The response from the finger server is sent to the terminal or telnet session from which the command was entered. When more than one host is given, finger forwarding is attempted, and each host refers to each step in the chain of finger servers.

The **detail** parameter specifies the level or details to request from the finger server. The finger server either interprets the command or ignores it, depending on whether the server supports it. The format of the information varies from server to server. The default is **low**.

Figure 14-12 shows a typical response from a finger query to a remote host requesting information about a specific user. A plan is a type of file that is appended to a finger response, and acts in a similar manner as signature files in email messages.

Figure 14-12: Example output of the response to a finger query.

```
> finger root@admin
Login: root                                  Name: root
Directory: /root                           Shell: /bin/tcsh
Last login Wed Jul 14 12:12 (NZDT) on ttyp0 from 192.168.11.17
New mail received Wed Jul 14 01:02 1999 (NZDT)
     Unread since Tue Jun  1 12:23 1999 (NZDT)
No Plan.
```

Examples

To send a finger query to the host `admin` requesting detailed information about user “root”, use the command:

```
fing root@admin det=hi
```

To send a finger query to the host `admin` (at IP address 192.168.11.5) requesting a list of all logged in users, use either of the commands:

```
fing @admin
```

```
fing @192.168.11.5
```

ping

Syntax

```
PING [IPaddress=]{ipadd|ipv6add[%interface]|host}
   [DElay=seconds] [Length=number] [NUMber={number|
   CONTinuous}] [PATTern=hexnum] [SIPAddress={ipadd|
   ipv6add}] [SCReenoutput={OFf|ON|NO|YES}]
   [TIMEOut=1..65535] [TOS=0..255]

PING [IPXAddress=]network:station [DElay=seconds]
   [LENGTH=number] [NUMBER={number|CONTinuous}]
   [PATTern=hexnum] [SIPXaddress=network:station]
   [SCReenoutput={OFf|ON|NO|YES}] [TIMEOut=1..65535]

PING [APPLEAddress=]network.node [DElay=seconds]
   [Length=number] [NUMber={number|CONTinuous}]
   [PATTern=hexnum] [SAPpleaddress=network.node]
   [SCReenoutput={OFf|ON|NO|YES}] [TIMEOut=1..65535]

PING [OSIAddress=]nsap [DElay=seconds] [Length=number]
   [NUMber={number|CONTinuous}] [PATTern=hexnum]
   [SOSIaddress=nsap] [SCReenoutput={OFf|ON|NO|YES}]
   [TIMEOut=1..65535]
```

where:

- *ipadd* is an IPv4 address in dotted decimal notation.
- *ipv6add* is a valid IPv6 address.
- *interface* is the interface the ping request is sent for a request to ping an IPv6 link-local address, e.g. vlan1eth0.
- *host* is a host name from the host name table.
- *network:station* is a valid Novell network number and station MAC address, expressed as hexadecimal numbers. Leading zeros may be omitted.
- *network.node* is an AppleTalk network number from 0 to 65279 or an AppleTalk network number range in the format “nnnnn-nnnnn”, and an AppleTalk node number from 0 to 127.
- *nsap* is a valid OSI NSAP address in dotted hexadecimal notation.
- *seconds* is a decimal number from 0 to 4294967295.
- *hexnum* is an 8-digit hexadecimal number, optionally proceeded by the characters “0x”.

Description

This command can be used to test that a valid path (route) exists to a destination. Packets are sent to the address specified; if the destination host replies, the time taken for the response to be received is recorded, and optionally displayed. The parameters of this command override the defaults set with the **set ping** command on page 14-163. The extended **ping** command supports IPv4, IPv6, IPX, OSI, and AppleTalk addresses.

Pinging an IPv6 link-local address requires interface information as well as the address because a single link-local address can belong to several interfaces. To ping a link-local address, specify the interface out which the ping request is sent, as well as the address. This interface is the interface, on the router from which the ping request originates, that is connected to the required destination interface (Figure 15-2 on page 15-19 in Chapter 15, Internet Protocol Version 6

(IPv6)). For example:

```
ping fe80::7c27%eth0
```

The **ping** command does not perform domain name server (DNS) lookups. A valid IP address or a host name defined in the host name table must be specified. Hosts can be added to the host name table with the **add ip host** command on page 14-76.

The **ipaddress**, **ipxaddress**, **osiaddress**, and **appleaddress** parameters specify the destination address for ping packets for IP, IPX, OSI, and AppleTalk networks, respectively.

The **delay** parameter specifies the time interval, in seconds, between ping packets. The default is 1 second.

The **length** parameter specifies the number of data bytes of the specified pattern to include in the data portion of the ping packet. If this parameter is not specified, the default is used.

The **number** parameter specifies the number of ping packets to transmit. If this parameter is not specified, the default is used. If **continuous** is specified, the **timeout** parameter must be set to a value greater than 0, and packets are sent continuously until the **stop ping** command on page 14-224 is issued.

The **pattern** parameter specifies the data to use to fill the data portion of the ping packet. If this parameter is not specified, the default is used.

The **sipaddress**, **sipxaddress**, **sosiaddress**, and **sappleaddress** parameters specify the source address to use in ping packets for IP, IPX, OSI, and AppleTalk networks, respectively. If the source address is not specified, and has not been set with the **set ping** command on page 14-163, the default is to use the address of the interface from which the ping packets are transmitted. In the special case of IP addresses, the router's local interface IP address is used, if set. Otherwise, the IP address of the interface from which the ping packets are transmitted is used. If the ping request is to an IPv6 link-local address, **sipaddress** must be on the outgoing interface and cannot be a link-local address.

The **screenoutput** parameter specifies whether the output is sent to the terminal. If **yes** is specified, the response time for each echo reply packet is displayed to the terminal as the reply is received from the destination host (Figure 14-13 on page 14-130).

Figure 14-13: Example output from the **ping** command when **screenoutput** is **yes**

```
Echo reply 1 from 172.16.8.2 time delay 20 ms

Echo reply 2 from 172.16.8.2 time delay 40 ms

Echo reply 3 from 172.16.8.2 time delay 0 ms

Echo reply 4 from 172.16.8.2 time delay 0 ms

Echo reply 5 from 172.16.8.2 time delay 60 ms
```

If **no** is specified, the results are stored and not displayed. To view the results, use the **show ping** command on page 14-216. If **screenoutput** is not specified, the default is used.

The **timeout** parameter specifies the length of time in seconds to wait for a response to a ping packet, and cannot be zero. If this parameter is not specified, the default is used.

The **tos** parameter specifies the value of the TOS (Type Of Service) field in the IP header of the ping packet. This parameter is valid for IP addresses and is ignored for IPv6 addresses. If this parameter is not specified, the default is used.

Related Commands

add ip host
set ping
show ping
stop ping

purge bootp relay

Syntax

```
PURge BOOTp RELAy
```

Description

This command purges the BOOTP configuration. The BOOTP module is disabled, all configuration data (including non-volatile storage) is purged, and then BOOTP is re-enabled with the default settings.

Related Commands

add bootp relay
delete bootp relay
disable bootp relay
enable bootp relay
set bootp maxhops
show bootp relay

purge ip

Syntax

```
PURge IP
```

Description

This command purges all configuration information relating to the IP routing module, and reinitialises the data structures used by the IP module. It should be used when first setting up the IP module or when a major change is required.

All current configuration information will be lost. Use with extreme caution!

Minor changes, such as changing the IP address of an interface, can be done without using the PURGE IP command. The configuration information is kept in non-volatile storage so that information is retained after a power down.

Related Commands

reset ip

reset ip

Syntax

```
RESET IP
```

Description

This command reinitialises dynamic IP data structures. It does not make the router operational if incorrect or incomplete information (for example, no IP address assigned to an interface) has been entered. It restarts routing timers and clears the ARP cache and the route table. IP only is affected; protocols that co-operate with and use IP (such as OSPF and BGP-4) are not affected by this command. The command also sends a message to the Logging facility if one has been configured. Note that all dial-in SLIP/PPP connections will be disconnected when this command is executed.

This command is not typically necessary during normal operation of the router. However, some occasions require the IP module to be restarted. One example is when a change is made to one of the interfaces assigned to the IP module with the **add ip interface** command on page 14-77, the **delete ip interface** command on page 14-101 or the **set ip interface** command on page 14-145. In this case, the relevant command automatically restarts the IP module, and a manual restart with the **reset ip** command is not necessary. Another example is when an underlying interface (such as a PPP interface) has changed, and the IP module must be restarted to discover changes.

Related Commands

purge ip
reset ip counter
reset ip interface

reset ip counter

Syntax

```
RESET IP COUnter={ALL|ARP|EGP|ICmp|INTerface|IP|MULticast|
    ROUte|SNmp|UDP}
```

Description

This command resets a specific group of IP counters to zero (0). The **counter** parameter specifies the group of counters to be reset. If **all** is specified, all IP counters are reset.

Examples

To reset the IP route counters to zero, use the command:

```
reset ip cou=rou
```

Related Commands

reset ip
reset ip interface
show ip counter

reset ip interface

Syntax

```
RESET IP INTerface=interface
```

where *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description

This command resets a logical IP interface. All routes associated with the interface, except interface and static routes, are purged from the routing table. All ARPs associated with this interface are purged from the ARP cache, and all counters for this interface are reset to zero (0).

The **interface** parameter specifies the name of the logical interface to be reset. The interface must currently be assigned to the IP routing module. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

Examples

To reset the first logical IP interface attached to vlan1, use the command:

```
reset ip int=vlan1
```

Related Commands

add ip interface
delete ip interface
disable ip interface
enable ip interface
reset ip
reset ip counter
set ip interface
show ip interface

set bootp maxhops

Syntax

```
SET BOOTp MAXHops=1..16
```

Description

This command sets the hop count threshold for discarding BOOTP messages. When the hops field in a BOOTP message exceeds the threshold, the BOOTP message is discarded. The hop count in a BOOTP message is incremented each time a router forwards the message. The default is 4.

Related Commands

add bootp relay
delete bootp relay
disable bootp relay
enable bootp relay
purge bootp relay
show bootp relay

set ip advertise interface

Syntax

```
SET IP ADVertise INTerface=interface
    [ADVertisementaddress=ALL|LIMited]
    [MAXadvertisementinterval=4..1800]
    [MINadvertisementinterval=3..MAXadvertisementinterval]
    [LIFetime=MAXadvertisementinterval..9000]
```

where *interface* is an interface name formed by concatenating an interface type and an interface instance (e.g. vlan1)

Description

This command modifies the Router Discovery advertisement settings on a single IP interface.

The **advertisementaddress** parameter specifies the IP destination address to be used for multicast advertisements sent from the interface. If **all** is specified, the destination is the all-systems multicast address, 224.0.0.1. If **limited** is specified, the destination is the limited-broadcast address, 255.255.255.255. The default is **all**.

The **maxadvertisementinterval** parameter specifies the maximum time allowed between sending multicast advertisements from the interface. The default is 600 seconds.

The **minadvertisementinterval** parameter specifies the minimum time allowed between sending multicast advertisements from the interface. The default is 450 seconds.

The **lifetime** parameter specifies the maximum length of time that the advertised addresses are to be considered as valid router addresses by hosts. The default is 1800 seconds.

If you change the advertising intervals, keep these proportions:
lifetime=3 x maxadvertisementinterval
minadvertisementinterval=0.75 x maxadvertisementinterval

Examples

To modify the advertisement address to the more limited broadcast address 255.255.255.255 and set the maximum advertisement interval to 1000 seconds on vlan3:

```
set ip adv int=vlan3 adv=lim max=1000 min=750 lif=3000
```

Related Commands

add ip advertise interface
delete ip advertise interface
disable ip advertise
enable ip advertise

set ip arp

Syntax

```
SET IP ARP=ipadd INTerface=interface
    {CIRCuit=miox-circuit|DLCi=dlci|ETHernet=macadd|
    POrt=port-number}
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit).
- *macadd* is the physical Ethernet (MAC) address of a host.
- *port-number* is the physical switch port number. Port numbers start at 1 and end at m, where m is the highest numbered Ethernet switch port, including uplink ports.

Description

This command modifies a static ARP entry in the ARP cache. The ARP entry must already exist.

The **arp** parameter specifies the IP address of the host.

The **interface** parameter specifies an existing interface over which the host can be reached. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

The **circuit** parameter specifies the MIOX circuit on an X.25 interface. If **circuit** is specified, **dlci** and **ethernet** cannot be specified.

The **dlci** parameter specifies the physical address for the host on a Frame Relay interface. If **dlci** is specified, **circuit** and **ethernet** cannot be specified.

The **ethernet** parameter specifies the physical (MAC) address for the host on an Eth or VLAN interface. If **ethernet** is specified, **dlci** and **circuit** cannot be specified.

The **port** parameter specifies the physical switch port number in a VLAN. If **interface** is a VLAN, the **port** parameter is valid; otherwise it is invalid.

Examples

To change the ARP entry for host 172.16.9.197 on interface eth0 (because, for example, the Ethernet interface on the host has been replaced and the host now has an Ethernet address of 00-BC-00-03-2F-9B), use:

```
set ip arp=172.16.9.197 int=eth0 eth=00-bc-00-03-2f-9b
```

To change the ARP entry for host 192.168.4.101 on interface vlan4 (because, for example, the Ethernet interface on the host has been replaced and the host now has an Ethernet address of 00-BC-00-03-2F-9B), use:

```
set ip art=192.168.4.101 int=vlan4 po=3 eth=00-bc-00-03-2f-9b
```

Related Commands

add ip arp
delete ip arp
show ip arp

set ip arp timeout

Syntax

`SET IP ARP TIMEOut=`*`multiplier`*

where *multiplier* is an integer number

Description

This command sets a multiplier value used to increase the ARP timeout by set increments. ARP timeouts vary between 256 and 512 seconds, depending on when ARP replies are received. Specifying a value of 4 increases the default timeout from 256-512 seconds to 1024-2048 seconds. The default multiplier value is 4.

Related Commands

add ip arp
delete ip arp
set ip arp
show ip arp

set ip dns

Syntax

```
SET IP DNS [DOMain={ANY|domain-name}]
   {INTerface=interface|[PRIMary=ipadd] [SECOndary=ipadd]}
```

where:

- *domain-name* is a character string of up to 255 characters. Valid characters are uppercase and lowercase letters, digits (0-9), and the underscore character (“_”).
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed (that is, eth0 is equivalent to eth0-0).
- *ipadd* is an IP address in dotted decimal notation.

Description

This command sets configuration information for the DNS servers to be used to resolve hosts in a particular domain into IP addresses. DNS servers for this domain must have already been configured with the **add ip dns** command on page 14-65.

The **domain** parameter specifies the domain for which this DNS server is to be used to resolve host names. DNS requests for hosts in this domain are sent to this server. If **any** is specified, the name server is used for domains not otherwise matched by another DNS entry. The default is **any**.

The **interface** parameter specifies the interface over which the router learns the address of a primary and/or a secondary name server. The primary and secondary name server’s addresses can be either statically configured using the **primary** and **secondary** parameters, or learned dynamically over an interface. Name servers can be learned via DHCP over an Ethernet or VLAN interface or via IPCP over a PPP interface. If the **interface** parameter is specified, the **primary** and **secondary** parameters are not required. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

The **primary** parameter specifies the IP address of the name server to be used as the primary name server for resolving hosts in the specified domain. If the **primary** parameter is specified, the **interface** parameter must not be specified.

The **secondary** parameter specifies the IP address of the name server to be used as the secondary name server for resolving hosts in the specified domain. If the **secondary** parameter is specified, the **interface** parameter must not be specified.

If the router was originally configured to learn name servers dynamically over a particular interface for use in resolving host names in the specified domain, this configuration can be overridden by specifying values for one or both of the static name server parameters (**primary** and **secondary**). Similarly, if static name server addresses were originally configured, using the **interface** parameter causes name server information learned dynamically to overwrite the static name server configuration; static name server addresses are lost.

Examples To set the IP addresses of the default primary and secondary name servers to 192.168.20.1 and 192.168.20.2 respectively, use the command:

```
set ip dns prim=192.168.20.1 seco=192.168.20.2
```

To set the IP addresses of the primary and secondary name servers for use when resolving host names in the domain "oranges.com" to 192.168.20.1 and 192.168.20.2 respectively, use the command:

```
set ip dns dom=oranges.com prim=192.168.20.1
   seco=192.168.20.2
```

Related Commands

add ip dns
delete ip dns
show ip dns

set ip dns cache

Syntax

```
SET IP DNS CAChe [SIze=cache-entries]
   [TIMeout=cache-max-age]
```

where:

- *cache-entries* is a number from 0 to 1000.
- *cache-max-age* is a time from 1 to 60 minutes.

Description This command sets the parameters for the IP DNS cache. The DNS cache stores the responses to DNS requests that the router receives.

The **size** parameter specifies the maximum number of entries allowed in the cache at any time. If the maximum number of entries has been reached when a new DNS request is made, the oldest entry is deleted to make space. If 0 is specified, the route does not cache responses to DNS requests. A DNS cache containing 100 entries uses approximately 30 kilobytes of RAM. The default is 0.

The **timeout** parameter specifies the maximum time an entry remains in the DNS cache. Cache entries are deleted when they reach the age specified by this parameter. Cache entries that reach the expiry time indicated by the DNS server they came from before the cache timeout period has passed are deleted as they expire. The default is 30 minutes.

Examples To configure the DNS cache so it has a maximum size of 250 entries that are kept for no more than 15 minutes, use the command:

```
set ip dns cac si=250 tim=15
```

Related Commands

add ip dns
delete ip dns
set ip dns
show ip dns

set ip dnsrelay

Syntax

```
SET IP DNSRelay INTerface={interface|NONE}
```

where *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description

This command has been made obsolete by the **add ip dns** command on page 14-65 and **set ip dns** command on page 14-137, and is described for backwards compatibility. It no longer appears in dynamically generated configuration scripts, and router-generated configuration scripts replace **set ip nameserver** commands with **add ip dns** commands.

This command specifies the interface that the router uses to learn remote DNS server addresses. The interface is typically a dial-up connection to an ISP that provides the DNS name server PPP option. The router learns DNS addresses that have not been set using the **set ip nameserver** command. If **none** is specified, the router does not learn DNS server addresses via IPCP addresses. By default, DNS relay is not used to learn DNS server addresses.

Related Commands

disable ip dnsrelay
enable ip dnsrelay
set ip nameserver

set ip filter

Syntax

```
SET IP FILter=0..399 SOurce=ipadd [SMask=ipadd]
    [SPort={port-name|port-id}] [DEStination=ipadd
    [DMask=ipadd]] [DPort={port-name|port-id}]
    [ICMPCode={icmp-code-name|icmp-code-id}]
    [ICmptype={icmp-type-name|icmp-type-id}] [LOG={4..1600|
    Dump|Header|None}] [OPtions={False|OFF|ON|NO|True|YES}]
    [PROTocol={protocol|Any|Egp|Icmp|Ospf|Tcp|Udp}]
    [SEssion={Any|Established|Start}] [SIze=size]
    [ENTry=entry-number] {ACtion={INCLude|EXCLude}|
    POLIcy=0..15|PRIOrity=P0..P7}
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *port-name* is the predefined name for an IP port.
- *port-id* is an IP port number, or a range of ports in the form *low*:*high*.
- *icmp-code-name* is the predefined name for an ICMP reason code.
- *icmp-code-id* is the number of an ICMP reason code.
- *icmp-type-name* is the predefined name of an ICMP message type.
- *icmp-type-id* is the number of an ICMP message type.
- *protocol* is an IP protocol number.
- *size* is a number from 64 to 65535.
- *entry-number* is the position of this entry in the filter.

Description

This command changes a pattern in an IP traffic filter, policy filter, priority filter or routing filter.

The **filter** parameter specifies the number of the filter where the pattern is to be changed.

- Filters with numbers from 0 to 99 are treated as traffic filters, and use the **action** parameter to specify the action to take with a packet that matches the pattern.
- Filters with numbers from 100 to 199 are treated as policy filters, and use the **policy** parameter to specify the policy to use when routing a packet that matches the pattern.
- Filters with numbers from 200 to 299 are treated as priority filters, and use the **priority** parameter to specify the priority to assign to a packet that matches the pattern.
- Filters with numbers from 300 to 399 are treated as routing filters, and use the **action** parameter to specify the action to take with a route that matches the pattern.

An interface may have a maximum of one traffic filter, one policy filter, one priority filter and one routing filter, but the same traffic, policy or priority filter can be assigned to more than one interface. Traffic and routing policy filters are applied to packets received via the interface, whereas policy and priority filters are applied to packets as they are transmitted. Routing filters are used in commands that manipulate the passing of IP routing information in and out of the router.

All parameters are valid for traffic (0-99), policy (100-199) and priority (200-299) filters. The **source**, **smask**, **entry**, and **action** parameters are valid for routing filters (300-399).

The **source** parameter specifies the source IP address, in dotted decimal notation, for the pattern.

The **smask** parameter specifies the mask, in dotted decimal notation to apply to source addresses for this pattern. The mask is used to determine the portion of the source IP address in the IP packet that is significant for comparison with this pattern.

The values of **source** and **smask** must be compatible. For each bit in **smask** that is set to zero (0), the equivalent bit in **source** must also be zero (0). If either **source** or **smask** is 0.0.0.0, then both must be 0.0.0.0. The default is 255.255.255.255.

The **sport** parameter specifies the source port to check against for this pattern as the recognised name of a well-known UDP or TCP port (Table 14-11 on page 14-70), a decimal value from 0 to 65535, or a range of numbers formatted *low:high*. If *low* is omitted, 0 is assumed; if *high* is omitted, the maximum port number is assumed. If a port other than **any** is specified, the **protocol** parameter is required and must be TCP or UDP. The default is **any**.

The **destination** parameter specifies the destination IP address for the pattern in dotted decimal notation. The default is 0.0.0.0.

The **dmask** parameter specifies the mask in dotted decimal notation to apply to the destination address for this pattern. The mask determines the portion of the destination IP address in the IP packet that is significant for comparison with this pattern. If **dmask** is specified, **destination** must also be specified.

The values of **destination** and **dmask** must be compatible. For each bit in **dmask** that is set to zero (0), the equivalent bit in **destination** must also be zero (0). If either **destination** or **dmask** is 0.0.0.0, then both must be 0.0.0.0. If **destination** is specified, the default for **dmask** is 255.255.255.255. If **destination** is not specified, the default for **dmask** is 0.0.0.0.

The **dport** parameter specifies the destination port to check against for this pattern as the recognised name of a well-known UDP or TCP port (Table 14-11 on page 14-70), a decimal value from 0 to 65535, or a range of numbers formatted *low:high*. If *low* is omitted, 0 is assumed; if *high* is omitted, the maximum port number is assumed. If a port other than **any** is specified, the **protocol** parameter is required and must be TCP or UDP. The default is **any**.

The **icmptype** and **icmpcode** parameters specify the ICMP message type and ICMP message reason code to match against the ICMP type and code fields in an ICMP packet. The **icmptype** parameter specifies the ICMP message type to match as a decimal value from 0 to 255, or the recognised name of an ICMP type (Table 14-12 on page 14-71). The **icmpcode** parameter specifies the ICMP message reason code to match as a decimal value from 0 to 255, or the recognised name of an ICMP reason code (Table 14-13 on page 14-71). Both parameters are valid when the **protocol** parameter is set to **icmp**.

The **log** parameter specifies whether matches to a filter entry result in a message being sent to the router’s Logging facility, and the content of the log messages. This parameter enables logging of the IP packet filtering process down to the level of an individual filter entry.

If a number from 4 to 1600 is specified, the filter number, entry number, and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/PASS (for patterns with an **include** action) or IPFIL/FAIL (for patterns with an **exclude** action). In addition, the first 4 to 1600 octets of the data portion of TCP, UDP, and ICMP packets or the first 4 to 1600 octets after the IP header of other protocol packets are logged with a message type/subtype of IPFIL/DUMP.

If **dump** is specified, the filter number, entry number, and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/PASS (for patterns with an **include** action) or IPFIL/FAIL (for patterns with an **exclude** action). In addition, the first 32 octets of the data portion of TCP, UDP, and ICMP packets or the first 32 octets after the IP header of other protocol packets are logged with a message type/subtype of IPFIL/DUMP.

If **header** is specified, the filter number, entry number, and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/PASS (for patterns with an **include** action) or IPFIL/FAIL (for patterns with an **exclude** action). If **none** is specified, matches to the filter entry are not logged. The default is **none**.

The **options** parameter specifies the IP options field is used to check against the pattern. If **yes**, the pattern matches IP packets with options set; if **no**, the pattern matches packets without options set. The default is to match IP packets with or without IP options set.

The **protocol** parameter specifies the protocol to check against for this pattern as a decimal value from 0 to 65534. Valid protocol names are:

- Exterior Gateway Protocol (EGP)
- Internet Control Message Protocol (ICMP)
- Open Shortest Path First Protocol (OSPF)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

If either **sport** or **dport** are specified, **protocol** must be defined as TCP or UDP. Specifying TCP or UDP filters packets from companion protocols, for example ICMP, RIP, and OSPF, that do not use TCP or UDP as a transport mechanism. The default is **any**.

The **session** parameter specifies the type of TCP packet to match, and can be used when the **protocol** parameter specifies TCP. If **start** is specified, the pattern matches TCP packets with the SYN bit set and the ACK bit clear. If **established** is specified, the pattern matches TCP packets with either the SYN bit clear or the ACK bit set. If **any** is specified, the pattern matches any TCP packet. The default is **any**.

The **size** parameter specifies the maximum reassembled size to match against for each IP fragment. If the fragment's offset plus size is greater than the value specified, the fragment is discarded.

The **entry** parameter specifies the entry number in the filter to be changed. Existing patterns with the same or higher entry numbers are pushed down the filter. The default is to add the new pattern to the end of the filter.

The **action** parameter specifies, for traffic filters, the action to take when the pattern is matched. The **action**, **policy**, and **priority** parameters are mutually exclusive so only one may be specified. If **include** is specified, the IP packet is processed and forwarded for traffic filters, or the IP route is selected for routing filters. If **exclude** is specified, the IP packet is discarded for traffic filters, or the IP route is excluded for routing filters.

The **policy** parameter is used for policy-based routing and specifies the policy to use when the pattern is matched. The policy number is assigned to incoming packets but employed during forwarding (transmission). The **action**, **policy**, and **priority** parameters are mutually exclusive so only one may be specified.

- For policy numbers from 0 to 7, routes with a matching policy are considered first.
- For policy numbers from 8 to 15, routes with a policy of *n*-8 (where *n* is the filter policy) are considered first, and the policy value *n*-8 is written into the TOS field of the packet.

The **priority** parameter is used for priority routing and specifies the priority when the pattern is matched. The priority number is assigned to incoming packets but employed during forwarding (transmission). Packets can be assigned a priority from p3 (highest) to p7 (lowest). The default is p5. Priority levels p0, p1, and p2 should not be used because they may conflict with router system activities. The **action**, **policy**, and **priority** parameters are mutually exclusive so only one may be specified.

Examples To set the session to be matched by entry 3 of filter 2 to established, use the command:

```
set ip fil=2 ent=3 se=e
```

Related Commands

add ip filter
add ip route filter
delete ip filter
delete ip route filter
show ip filter
show ip route filter

set ip host

Syntax

```
SET IP HOst=name IPaddress=ipadd
```

where:

- *name* is a character string up to 60 characters long. If the string contains spaces, it must be in double quotes.
- *ipadd* is an IP address in dotted decimal notation.

Description

This command modifies the IP address associated with a user-defined name for an IP host in the host name table. The host name table makes it easier to Telnet to commonly accessed hosts by enabling the user to enter a shorter, easier to remember name for the host rather than the host’s full IP address or domain name.

The **host** parameter specifies the user-defined name for the IP host. A host with the same name must already exist in the host name table. When a host name is specified in the Telnet command, the entire name is used to match a name in the host name table. All characters are used in the comparison, including nonalphabetic characters when present.

The **ipaddress** parameter specifies the IP address for the host.

Examples

To change the IP address for host name “zaphod” in the host name table from 172.16.1.5 to 172.16.9.8, use:

```
set ip ho=Zaphod ip=172.16.9.8
```

To Telnet to the host, use any of the following commands:

```
telnet zaphod
telnet zaphod.company.com
telnet 172.16.9.8
```

Related Commands

add ip host
delete ip host
set ip nameserver
set ip secondarynameserver
show ip host

set ip interface

Syntax

```
SET IP INTerface=interface [ADVertise={YES|NO}]
   [PREferencelevel={-2147483648..2147483647|NOTDEFAULT}]
   [BROadcast={0|1}] [DIRectedbroadcast={False|NO|OFF|ON|
   True|YES}] [FILter={0..99|NONE}] [FRAgment={NO|OFF|ON|
   YES}] [GRAtuitousarp={ON|OFF}] [GRE={0..100|NONE}]
   [IGMPProxy={OFF|UPstream|DOWNstream}] [INVersearp={ON|
   OFF}] [IPaddress=ipadd|DHCP] [MASK=ipadd]
   [METric=1..16] [MULticast={BOTH|OFF|ON|RECeive|SENd}]
   [OSPFmetric=1..65534|DEFAULT] [POLicyfilter={100..199|
   NONE}] [PRIorityfilter={200..299|NONE}]
   [PROxyarp={False|NO|OFF|ON|True|YES|STrict|DEFRoute}]
   [RIPMetric=1..16] [SAMode={Block|Passthrough}]
   [VJC={False|NO|OFF|ON|True|YES}] [VLANTAG={1..4094|
   NONE}]
```

where:

- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *ipadd* is an IP address in dotted decimal notation.

Description

This command modifies the configuration of a logical interface used by the IP module. When the router is in security mode, this command can be issued only by a user with security officer privilege. Note that all dial-in SLIP/PPP connections will be disconnected when this command is executed.

The IP configuration of an interface cannot be changed while DVMRP or PIM is attached to the interface. The DVMRP or PIM interface must first be deleted, and then re-added after the IP changes have been made. See Chapter 17, IP Multicasting for information about DVMRP and PIM.

The **interface** parameter specifies the name of the logical interface, and implicitly, the attached Layer 2 interface. The interface must currently be assigned to the IP module. At least two interfaces must be defined before the router can route IP packets, but only one interface (usually eth0) needs to be defined when the router is acting as a server. A maximum of 640 interfaces can be added. When an interface is added, it is automatically enabled. Only one logical interface may be configured to the same IP network or subnet. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

The **advertise** parameter specifies whether the logical interface is to send Router Discovery advertisements. The default is YES.

The **broadcast** parameter specifies whether a broadcast address with all 1s or all 0s is used. The default is 1. This parameter should not be set to 0 without careful consideration of the consequences. It is provided to allow compatibility with older host implementations that do not meet the current standard.

The **directedbroadcast** parameter specifies whether the router allows network or subnet broadcasts to be forwarded to the network directly attached to the logical interface. The default is **no**.

The **filter** parameter specifies the traffic filter to apply to IP packets transmitted or received over the logical interface. The filter must already have been defined with the **add ip filter** command on page 14-68. A logical interface may have a maximum of one traffic filter, one policy filter and one priority filter, but the same traffic, policy or priority filter can be assigned to more than one interface. Traffic filters are applied to packets received via the logical interface. The default is not to apply a filter.

The **fragment** parameter specifies whether the "*Do not fragment*" bit is obeyed for outgoing IP packets that are larger than the MTU of the interface. If **yes**, the "*Do not fragment*" bit is ignored and outgoing IP packets larger than the MTU of the interface are fragmented. This is particularly useful for interfaces configured with GRE, SA and/or IPsec encapsulation that can potentially increase packet sizes beyond the MTU of the interface. If **no**, the "*Do not fragment*" bit is obeyed and IP packets larger than the MTU of the interface are discarded. This is the normal behaviour for IP. The **fragment** parameter has no effect on processing packets smaller than the interface MTU. The default is **no**.

The **gratuitousarp** parameter enables or disables the acceptance of gratuitous ARP request or gratuitous ARP reply. The default is **on**.

The **gre** parameter specifies the GRE (Generic Routing Encapsulation) entity associated with the logical interface. The GRE entity must have been created previously with the **add gre** command on page 29-10 of Chapter 29, Generic Routing Encapsulation (GRE). The default is **none**.

The **igmpproxy** parameter specifies the status of IGMP proxying for the specified interface. If **off**, the interface does not do IGMP Proxy. If **upstream**, the interface passes IGMP messages in the upstream direction. A router can have one interface with the IGMP proxy direction equal to **upstream**. If **downstream**, the interface can receive IGMP messages from the downstream direction. The default is **off**. To display information about IGMP and multicast group membership for each IP interface, use the **show ip igmp** command on page 17-73 of Chapter 17, IP Multicasting.

The **inversearp** parameter enables or disables the operation of the Inverse Address Resolution Protocol (INVARP) on ATM interfaces. The **inversearp** parameter must be set to **on** for IPoA configurations, and to **off** for RFC 1483 Routed configurations. The default is **off**. (Inverse ARP is always on for Frame Relay interfaces.)

The **ipaddress** parameter specifies the IP address of the logical interface. If **dhcp** is specified, the router acts as a DHCP client and obtains the configuration of the IP interface via DHCP. Table 14-14 on page 14-79 lists the parameters from the DHCP reply that the router uses. If an IP interface is configured to use DHCP to obtain its IP address and subnet mask, the interface does not take part in IP routing until the IP address and subnet mask have been set by DHCP. Remote address assignment must be enabled with the **enable ip remoteassign** command on page 14-126 before IP interfaces accept addresses dynamically assigned by DHCP.

The **mask** parameter specifies the subnet mask for the logical interface. The value must be consistent with the value specified for the **ipaddress** parameter. The default is the network mask for the address class of the IP address (for example, 255.255.0.0 for a Class B address, 255.255.255.0 for a Class C address). If **ipaddress** is set to **dhcp**, the **mask** parameter should not be set because the subnet mask received from the DHCP server is used.

The **multicast** parameter specifies whether the interface receives and forwards multicast packets, when neither DVMRP nor PIM are enabled. If **both** or **on** is specified, the router sends and receives multicast packets. If **off**, the router does not send or receive multicast packets. If **receive** is specified, the router receives but does not send packets. If **send** is specified, the router sends but does not receive them. Note that this parameter applies to the entire IP interface, not an individual logical interface so setting this parameter on one logical interface sets it for all associated with the same IP interface. This parameter determines the interface's static behaviour for multicast packets. When DVMRP or PIM-SM is enabled, it determines the forwarding behaviour of interfaces dynamically, and this parameter has no effect (Chapter 17, IP Multicasting). The default is **receive**.

The **ospfmetric** parameter specifies the cost of crossing the logical interface, for OSPF. If DEFAULT is specified the interface is restored to the default metric value. The setting of the OSPF metric to a value other than DEFAULT provides a mechanism to provide a metric for an interface that is preferred over the OSPF automatic metric setting (if enabled via SET OSPF AUTOCOST=ON). If the OSPFMETRIC has been set to a numerical value it must be set to DEFAULT before SET OSPF AUTOCOST can take effect for this interface. The default is 1.

The **policyfilter** parameter specifies the policy filter to apply to IP packets received over the logical interface. The filter must already have been defined with the **add ip filter** command on page 14-68. A logical interface may have a maximum of one traffic filter, one policy filter, and one priority filter. However, the same traffic, policy or priority filter can be assigned to more than one interface. Policy filters are applied to packets when they are transmitted. The default is not to apply a filter.

The **preferencelevel** parameter specifies the preference of the address as a default router address relative to other router addresses on the same subnet, as a decimal integer. If the minimum value (-2147483648) or **notdefault** is specified, the address is not used by neighbouring hosts as a default address, even though it may be advertised. The default is the mid range 0.

The **priorityfilter** parameter specifies the priority filter to apply to IP packets transmitted over the logical interface. The filter must already have been defined with the **add ip filter** command on page 14-68. A logical interface may have a maximum of one traffic filter, one policy filter, and one priority filter. However, the same traffic, policy or priority filter can be assigned to more than one interface. Priority filters are applied to packets as they are transmitted. The default is not to apply a filter.

The **proxyarp** parameter enables or disables proxy ARP responses to ARP requests. This parameter is valid for Eth and VLAN interfaces. The default is **on**.

If the **on/true/yes** option is specified, the device will respond to proxy ARP Requests using specific routes if they exist. If the **off/false/no** option is specified, the device will not respond to ARP requests. If the **defroute** option is specified, the device will respond to proxy ARP Requests using specific routes if they exist or a default route (0.0.0.0) if it exists. If the **strict** option is specified, the router will only respond to ARP requests using specific routes if they exist.

*If the **defroute** option is currently enabled, any other **proxyarp** option selected will disable the **defroute** mode of operation.*

When the device is operating in defroute mode, it is non-compliant with RFC 1027.

The **ripmetric** parameter specifies the cost of crossing the logical interface for RIP. The default is 1. The **metric** parameter is also accepted for backwards compatibility.

The **samode** parameter specifies how the logical interface handles IP packets that do not belong to one of the security associations assigned to the logical interface. If **block** is specified, IP packets that do not belong to a security association assigned to the logical interface are blocked from transiting the interface and are discarded. If **passthrough** is specified, IP packets that do not belong to a security association assigned to the logical interface are allowed to transit the interface and are forwarded normally by the IP routing software. The default is **block**. This parameter takes affect when one or more security associations have been assigned to the logical interface with the **add ip sa** command on page 14-94.

The **vjc** parameter is valid for Point-to-Point Protocol (PPP) and X25T interfaces, and specifies whether Van Jacobson header compression is to be used on the Layer 2 interface. The **vjc** parameter applies to all logical interfaces attached to the same Layer 2 interface. Changing the setting on one logical interface alters the setting on the others attached to the Layer 2 interface. Compression provides the most advantage on slower link speeds (up to 48 kbps). At speeds of 64 kbps and higher, compression actually reduces efficiency and should be disabled. Van Jacobson's TCP/IP header compression should not be enabled on a multilink PPP interface.The default is **off**.

The **vlantag** parameter specifies the VID (VLAN Identifier) to be included in the header of each frame that is transmitted over the logical interface. This parameter is valid for Eth interfaces only. Multiple logical interfaces on the same physical interface can share the same VLAN tag. The default is **none**, which mean no VID is included. For more information, see "VLAN Tagging on Eth Interfaces" on page 14-30.

Examples

To set the first IP interface attached to PPP2 with an IP address of 172.16.248.33, a subnet mask of 255.255.255.0, and a metric of 5, use the command:

```
set ip int=ppp2-0 ip=172.16.248.33 mask=255.255.255.0 ripm=5
```

To associate the second IP interface attached to Eth2 with GRE entity 3, use:

```
set ip int=eth2-1 gre=3
```

Related Commands

add ip advertise interface
add ip interface
delete ip advertise interface

delete ip interface
disable ip advertise
disable ip interface
enable ip advertise
enable ip interface
reset ip interface
set ip advertise interface
show ip advertise
show ip igmp in Chapter 17, IP Multicasting
show ip interface

set ip local

Syntax

```
SET IP LOCal[={DEFAULT|1..15}] [FILter={0..299|NONE}]
   [GRE={0..100|NONE}] [IPaddress=ipadd]
   [POLicyfilter={0..299|None}] [PRIorityfilter={0..299|
   None}]
```

where *ipadd* is an IP address in dotted decimal notation

Description

This command modifies the parameters of one the router's local interfaces. If the **local** parameter is either not specified or **default**, then the router's default local interface is modified.

Each of the local IP interfaces are virtual and are able to represent the IP routing module itself. Each of the local interface interfaces can be assigned IP addresses that can then be used as the source address of IP packets generated internally by IP protocols such as RIP, OSPF, PING and NTP. Higher layer protocols such as RIP, OSPF, PING and NTP must assign a source IP address to packets passed to IP for forwarding. Use the following rules to determine which IP address to use as the source address:

1. If the higher layer protocol's configuration specifies a source IP address to use, then the configured address is used as the packet's source IP address.

   For example, the **sipaddress** parameter in the **ping** command on page 14-129 specifies the source IP address to use in ping packets.

2. If a local IP interface has been assigned an IP address, then the IP address of that local interface is used as the packet's source IP address.
3. Otherwise, the IP routing module determines the interface over which the packet is to be transmitted, and assigns the IP address of the interface as the packet's source IP address.

The **filter** parameter specifies the filter to apply to IP packets transmitted or received over the interface. The filter must already have been defined with the **add ip filter** command on page 14-68. An interface may have a maximum of one traffic filter, one policy filter and one priority filter, but the same traffic, policy or priority filter can be assigned to more than one interface. Traffic filters are applied to packets received via the interface. The default is not to apply a filter.

The **gre** parameter specifies the GRE (Generic Routing Encapsulation) entity associated with the interface. The specified GRE entity must have been created previously using the **add gre** command on page 29-10 of Chapter 29, Generic Routing Encapsulation (GRE). The default is NONE.

The **ipaddress** parameter specifies the IP address of the interface. The IP address must be the IP address of one of the router's active IP interfaces. Specifying an IP address of 0.0.0.0 effectively 'unsets' the IP address of the local interface.

The **policyfilter** parameter specifies the policy filter to apply to IP packets received over the interface. The filter must already have been defined with the **add ip filter** command on page 14-68. An interface may have a maximum of one traffic filter, one policy filter, and one priority filter, but the same traffic, policy, or priority filter can be assigned to more than one interface. Policy filters are applied to packets as they are transmitted. The default is not to apply a filter.

The **priorityfilter** parameter specifies the priority filter to apply to IP packets transmitted over the interface. The filter must already have been defined with the **add ip filter** command on page 14-68. An interface may have a maximum of one traffic filter, one policy filter, and one priority filter, but the same traffic, policy, or priority filter can be assigned to more than one interface. Priority filters are applied to packets as they are transmitted. The default is not to apply a filter.

Examples

To set the IP address of the local IP interface to 192.168.33.11, use:

```
set ip loc ip=192.168.33.11
```

`To set the local interface 3 to 192.168.33.11, use:`

```
set ip local=3 ip=192.168.33.1
```

To remove the IP address of the local IP interface, use:

```
set ip loc ip=0.0.0.0
```

Related Commands

add ip interface
delete ip interface
add ip local
delete ip local
add ip local
set ip interface
show ip interface

set ip nameserver

Syntax

```
SET IP NAMEserver=ipadd
```

where *ipadd* is an IP address in dotted decimal notation

Description

This command has been made obsolete by the **add ip dns** command on page 14-65 and **set ip dns** command on page 14-137, and is described for backwards compatibility. It no longer appears in dynamically generated configuration scripts, and router-generated configuration scripts replace **set ip nameserver** commands with **add ip dns** commands.

This command specifies the IP address of a host able to act as the primary name server for the router. Name servers are used to resolve Telnet requests to host names that are not in the host name table. If the host is entered into the host table, then no access to a name server is required. This may suit installations that have no name server.

A secondary name server can also be specified with the **set ip secondarynameserver** command on page 14-162, another obsolete command. When the router performs a DNS lookup, it firsts sends the request to the primary name server. If a response is not received within 20 seconds the request is sent to the secondary name server.

Examples

To specify the host with IP address 172.16.1.5 as a name server, use:

```
set ip name=172.16.1.5
```

The equivalent command to the example given above, using the ADD IP DNS command, is:

```
add ip dns prim=172.16.1.5
```

This command would be used if the default primary name server had not previously been configured. If the primary name server had previously been configured the IP address may be changed with the command:

```
set ip dns prim=172.16.1.5
```

Related Commands

add ip host
delete ip host
set ip dnsrelay
set ip host
set ip secondarynameserver
show ip
show ip host

set ip nat maxfragments

Syntax

```
SET IP NAT MAXFragments=8..50
```

Description

This command sets the maximum number of fragments that a fragmented IP packet may consist of when enhanced fragment handling is enabled for IP NAT.

The **maxfragments** parameter specifies the maximum number of fragments that an IP packet may consist of. The default is 20.

Enhanced fragment handling for IP NAT is disabled by default. When disabled, fragmented IP packets can only be processed by IP NAT if the packet consists of no more than 8 fragments, and the total data contained in all the fragments is 1780 bytes or less. Enhanced fragment handling for IP NAT is enabled with the command **enable ip nat** command on page 14-125.

Examples

To set the maximum number of fragments in a packet to be processed by IP NAT to 25, use the command:

```
set ip nat maxf=25
```

Related Commands

disable ip nat
enable ip nat
show ip nat

set ip rip

Syntax

```
SET IP RIP INTerface=interface [CIRCuit=miox-circuit]
    [DLCi=dlci] [IP=ipadd] [NEXThop=ipadd] [SENd={NOne|
    RIP1|RIP2|COmpatible}] [RECeive={NOne|RIP1|RIP2|BOth}]
    [DEMand={False|NO|OFF|ON|True|YES}] [AUth={NOne|
    PASSword|MD5}] [PASSword=password] [STATicesport={YES|
    NO}]
```

where:

- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit) from 0 to 1023.
- *ipadd* is an IP address in dotted decimal notation.
- *password* is a character string 1 to 63 characters long. It may contain uppercase and lowercase letters, digits (0-9), the hyphen (-), and the underscore character (“_”).

Description

This command sets attributes of the RIP neighbour. The IP address and the interface identify which RIP neighbour to change.

The **interface** parameter specifies an existing interface that the RIP neighbour is on. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

The **circuit** parameter specifies the X.25 circuit on which to send or receive RIP packets. It is a required parameter for X25T interfaces and is valid only when the interface is an X25T interface.

The **dlci** parameter specifies the Frame Relay DLCI on which to send or receive RIP packets. It is a required parameter for Frame Relay interfaces and is valid for Frame Relay only.

The **ip** parameter specifies the IP address of the RIP neighbour. If an IP address is specified, then RIP packets received on the interface are accepted from this address. If no IP address is specified, then the source address of RIP packets is not checked. RIP updates generated by the device being configured are sent to the specific IP address. If no **ip** parameter is specified, RIP packets are sent to the RIP multicast address 224.0.0.9 (if the **send** parameter is **rip2** or **compatible**), or the broadcast address (if the **send** parameter is **rip1**).

The **nexthop** parameter is carried in RIP v2 packets to inform the destination of the next hop address returning to the device being configured. The default is 0.0.0.0, indicating the (local) source of the RIP route update. If **nexthop** is specified, **ip** must also be specified, and **send** must not indicate that RIPv1 packets are to be sent.

The **send** parameter specifies the version of RIP packet to send. If **none** is specified, then no RIP packets are sent. If **rip1** is specified, RIP version 1 packets are sent; if **rip2**, version 2 packets are sent. If **compatible** is specified, RIP version 2 packets are sent without routes that a router receiving only RIP version 1 treats as host routes. The default is **rip1**.

The **receive** parameter specifies the version of RIP packets to receive. If **none**, then no RIP packets are accepted from the IP address on the interface. If **rip1** is specified, RIP version 1 packets are accepted; if **rip2**, version 2 packets are accepted. If **both** is specified, then either RIP version 1 or RIP version 2 packets are accepted (as long as a version compatibility rule is not violated). The default is **both**.

The **demand** parameter specifies whether to use RIP demand procedures when send and receiving RIP, and for routes received from this neighbour. If **no**, demand procedures are not used; if **yes**, they are used. The default is **no**.

The **authentication** parameter specifies the method used to authenticate RIP packets. This must be **none** unless using RIP version 2. If **none**, no authentication is used. If **password** is specified, a plaintext password is used to authenticate RIP packets; if **md5**, an encrypted password is used. The default is **none**.

The **password** parameter specifies the password to use if the **authentication** parameter is set to **password** or **md5**. This parameter is required when authentication is used. Although 63 characters are allowed as a password, only the first 16 are used. A warning to this effect is generated when the command is entered.

The **staticexport** parameter specifies whether static routing information is propagated from this interface. If **yes**, static routes are included in routing exports; if **no**, they are omitted. The default is **yes**.

Examples

To change the password for a RIP neighbour using authentication, use the command:

```
set ip rip int=ppp0 ip=172.16.248.33 pass=supersecret
```

To change a RIP neighbour from on-demand using RIP version 2 to not on-demand sending RIP version 1 compatible packets, and receiving RIP version 1 and 2, use the command:

```
set ip rip int=ppp0 ip=172.16.248.33 dem=no sen=co rec=bo
```

Related Commands

add ip rip
delete ip rip
set ip riptimer
show ip rip

set ip riptimer

Syntax

```
SET IP RIPTimer [FLush=1..4294967295]
   [HOlddown=1..4294967295] [INvalid=1..4294967295]
   [UPdate=1..4294967295]
```

Description

This command sets the values of the global RIP timers in seconds. This command does not change **flush**, **holddown**, or **invalid** time intervals for existing IP RIP routes. Existing routes continue to be invalidated by time intervals previously set.

The **update** parameter sets the time between RIP updates for all interfaces not using RIP on demand. The default is 30 seconds.

The **invalid** parameter sets the time after which the router deems a route to be invalid because no update has been received. The default is 180 seconds.

The **holddown** parameter sets the time after a route has become invalid during which the router ignores updates for the route that would normally make the route valid again. The default is 120 seconds.

The **flush** parameter sets the time for when the route is last updated until it is flushed from the route table. This time must equal or exceed the sum of the **invalid** and **holddown** times. The default is 300 seconds.

After a valid update, the **flush** and **invalid** timers are restarted. When the **invalid** timer expires, the route is invalidated and the **holddown** timer started. The **flush** timer continues to run. When the **holddown** timer expires, valid updates for the route result in the router being reinstated. When the **flush** timer expires, the route is deleted from the route table.

Examples

To force RIP routes to be invalidated and flushed as soon as a single update is missed, use the command:

```
set ip ript in=35 ho=0 fl=35
```

Related Commands

set ip rip
show ip rip
show ip riptimer

set ip route

Syntax

```
SET IP ROUte=ipadd INTerface=interface MASK=ipadd
   NEXThop=ipadd [CIRCuit=miox-circuit] [DLCi=dlci]
   [METRIC=1..16] [METric1=1..16] [METRIC2=1..65535]
   [POLIcy=0..7] [PREFerence=0..65535] [TAG=1..65535]
```

where:

- *ipadd* is an IP address in dotted decimal notation.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit).

Description

This command modifies a static route in the IP route table. Static routes can be used to define default routes to external routers or networks. A default route is one with a network address of 0.0.0.0. When the router receives data and cannot find a route for it, it sends the data to the default route. To define a default route, **ipaddress** is set to 0.0.0.0 and **nexthop** points to the network (router) to which default packets are to be directed. The static route must not already exist. However, if the route exists as a dynamic route (such as RIP-derived), the static route can still be added. A maximum of 300 static routes can be defined.

This command also defines subnets. Multiple routes can be defined for a single interface (usually a LAN). This is useful for configuring more than one network or subnet on a particular interface. A common problem is when hosts exceed the capacity of a single subnet. Additional subnets can be assigned by adding static routes. In this case **ipaddress** is set to the new subnet, **nexthop** is set to 0.0.0.0, and **metric** set to 1.

The **route** parameter specifies the IP address of the static route.

The **interface** parameter specifies the IP interface with which the route is associated. The interface must already exist and be assigned to the IP module. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

If the interface is a Frame Relay interface, the **dlci** parameter is required and specifies the DLC to use on the Frame Relay interface. If the interface is an X.25 DTE interface, the **circuit** parameter is required and specifies the name of a MIOX circuit already defined for the X.25 DTE interface.

The **nexthop** parameter specifies the IP address of the next hop (router) for the route. The default is the IP address of the interface specified by the **interface** parameter. For a PPP link, **nexthop** should be the IP address of the remote end of the PPP link.

The **mask** parameter specifies the subnet mask for the route. A check is performed on the route and mask to verify that the route is the same before and after masking. This ensures that a static route is not specified to more than its subnet mask.

The **metric1** parameter specifies the cost of traversing the route for RIP. The default is 1. The normal range is from 2 to 16. A metric of 1 should be used when adding a subnet to an interface. The **metric** parameter is also accepted for backwards compatibility.

The **metric2** parameter specifies the cost of traversing the route for OSPF. The default is 1.

The **policy** parameter specifies the type of service for the route. The default is 0.

The **preference** parameter specifies the preference for the route. When more than one route in the route table matches the destination address in an IP packet, the route with the lowest preference value is used to route the packet. If two or more routes have the same preference, the route with the longest subnet mask is used. Interface routes have a preference of 0 and RIP routes have a preference of 100. The default preference for static routes other than 0.0.0.0 is 60. The default for the default static route 0.0.0.0 is 360.

The **tag** parameter specifies an integer to tag the route with. You can then match against this number in a route map and only import the appropriately-tagged routes into BGP.

Examples To set the subnet 172.16.9.0 on interface eth0 to have a RIP metric of 2, use the command:

```
set ip rou=172.16.9.0 mask=255.255.255.0 int=eth0
   next=0.0.0.0 met=2
```

Related Commands

add ip route
delete ip route
show ip route

set ip route filter

Syntax

```
SET IP ROUte FILter=filter-id [IP=ipadd] [MASK=ipadd]
   [ACtion={INCLude|EXCLude}] [DIrection={RECeive|SENd|
   BOTH}] [INTerface=interface] [NEXThop=ipadd]
   [POLIcy=0..7] [PROTocol={ANY|EGP|OSPF|RIP}]
```

where:

- *filter-id* is a number from 1 to 100.
- *ipadd* is an IP address in dotted decimal notation.
- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description

This command modifies a route filter. A route filter controls which routes are sent and received by the routing protocols. Note that there are some filtering limitations. For more information, see “Routing Information Filters” on page 14-22. Route filters do not apply to static or interface routes.

When a route is received or transmitted by a routing protocol, the list of route filters is searched for a match to the route. Processing stops when a match is found or the end of the list is reached. If at least one route filter is defined, then the filter list has an implicit “exclude all” after the last entry in the list. Therefore, it may be necessary to add an “include all” filter at the end of the list to allow all other routes that do not match.

The **filter** parameter specifies an existing index of the filter to modify.

The **ip** parameter specifies the network address to match. The wildcard character (”*”) can be used to match a network range. For example, 192.168.*.* matches all destination networks that start with 192.168. The wildcard character can replace a complete number; for example, 192.168.*.* is valid but 192.16*.*.* is not.

The **mask** parameter specifies the network mask of the network to match. The wildcard character (”*”) can be used to match a network mask range. For example, 255.255.*.* matches all destination network masks that start with 255.255. The wildcard character can replace a complete number; for example, 255.255.*.* is valid but 255.25*.*.* is not.

The **action** parameter specifies what to do with routes that match the filter. If **include** is specified, the route is included; if **exclude** is specified, it is excluded.

The **direction** parameter specifies whether to filter the route when receiving or sending it. If **receive** is specified, the **protocol** parameter specifies the routing protocol that receives the route information; if it is **send**, the **protocol** parameter specifies the routing protocol that advertises the routes.

The **interface** parameter specifies the interface to which the filter applies. If specified, the route is filtered when the route is sent or received on the interface. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)

- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

The **nexthop** parameter specifies the IP address of the next hop router to match. If specified, the route is filtered when the route is sent or received to or from the next hop.

The **policy** parameter specifies the type of service to filter. If not specified, all types of service are filtered.

The **protocol** parameter specifies the routing protocol to which the filter applies. The default is **any**. When **direction** is **receive**, then **protocol** specifies the routing protocol that receives the route information. If **direction** is **send**, **protocol** specifies the routing protocol that advertises the routes.

The way that the OSPF protocol works affects how the route filter operation on OSPF Link State Advertisement (LSA) works. A route filter with **direction=send** filters only matching routes regarded as Autonomous System (AS) external routes by OSPF. Also, the **interface** parameter is ignored, meaning all interfaces are treated indifferently.

Examples

To modify route filter 1 to include only OSPF-derived routes, use the command:

```
set ip rou fil=1 prot=ospf
```

Related Commands

add ip route filter
delete ip route filter
show ip route filter

set ip route preference

Syntax

```
SET IP ROUte PREFerence={DEFault|1..65535}
   PROTocol={BGP-ext|BGP-int|OSPF-EXT1|OSPF-EXT2|
   OSPF-INTEr|OSPF-INTRa|OSPF-Other|RIP}
```

Description

This command sets the IP route table preference for routes learned via a specific protocol. When more than one route in the route table matches the destination address in an IP packet, the route with the lowest preference value is used to route the packet. If two or more routes have the same preference, the one with the longest prefix is used.

Existing dynamically learned routes in the routing table and new routes added later are updated with the specified preference value. Packet processing times may be affected for a short time while the routing table is updated with new preference values.

The **preference** parameter sets the preference for routes learned via the specified protocol. A route with a low preference value has priority over a route with a high preference value. If **default** is specified, the preference reverts to the default for this protocol type (Table 14-16). The preference values for OSPF protocol types must be set in such a way that **ospf-intra < ospf-inter < ospf-ext1 < ospf-ext2 < ospf-other.**

Table 14-16: Default preference values for each protocol type

Protocol type	Default preference
rip	100
ospf-intra	10
ospf-inter	11
ospf-ext1	150
ospf-ext2	151
ospf-other	152
bgp-int	170
bgp-ext	170

The **protocol** parameter specifies which protocol's routing table is updated with the new preference value.

Examples

To set the router to use OSPF routes in preference to equally-specific RIP routes, give RIP routes a higher preference value than OSPF routes by using the command:

```
set ip rou pref=160 prot=rip
```

Related Commands

show ip route preference

set ip route template

Syntax

```
SET IP ROUte TEMPlate=name [NEXThop=ipadd]
   [CIRCuit=miox-circuit] [DLCi=dlci] [METric=1..16]
   [METRIC1=1..16] [METRIC2=1..65535] [POLIcy=0..7]
   [PREFerence=0..65535]
```

where:

- *name* is a character string 1 to 31 characters long, and is not case-sensitive. Valid characters are any printable character. If *name* contains spaces, it must be in double quotes.
- *ipadd* is an IP address in dotted decimal notation.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit).

Description

This command modifies an existing IP route template. IP route templates are used by the router to add IP routes to IP subnetworks discovered during normal operation by other protocols such as IPsec. This is required if IP traffic to the discovered IP subnetwork needs to be routed via a route other than the default route.

The **dlci** parameter specifies the DLC to use on the Frame Relay interface. This parameter is valid when the template was added with a Frame Relay interface.

The **circuit** parameter specifies the name of a MIOX circuit to use on the X.25 DTE interface. This parameter is valid when the template was added with a X.25 DTE interface.

The **nexthop** parameter specifies the IP address of the next hop (router) for any route added using this template. The default is the IP address of the interface specified by the **interface** parameter. For a PPP link, **nexthop** should be the IP address of the remote end of the PPP link.

The **metric1** parameter specifies the cost of traversing any route added using this template for RIP. The default is 1. The normal range is from 2 to 16. A metric of 1 should be used when adding a subnet to an interface.The **metric** parameter is also accepted for backwards compatibility.

The **metric2** parameter specifies the cost of traversing any route added using this template for OSPF. The default is 1.

The **policy** parameter specifies the type of service for any route added using this template. The default is 0.

The **preference** parameter specifies the preference for routes added with this template. When more than one route in the route table matches the destination address in an IP packet, the route with the lowest preference value is used to route the packet. If two or more routes have the same preference, the route with the longest subnet mask is used. Interface routes have a preference of 0 and RIP routes have a preference of 100. The default preference for static routes other than 0.0.0.0 is 60. The default for the default static route 0.0.0.0 is 360.

Examples To set the preference of routes created with the IP route template named "branch_office" to 90, use the command:

```
set ip rou temp=branch_office pref=90
```

Related Commands

add ip route template
create ipsec policy
delete ip route template
show ip route template

set ip secondarynameserver

Syntax `SET IP SECOndarynameserver=`*`ipadd`*

where *ipadd* is an IP address in dotted decimal notation

Description This command has been made obsolete by the **add ip dns** command on page 14-65 and **set ip dns** command on page 14-137, and is described for backwards compatibility. It no longer appears in dynamically generated configuration scripts, and router-generated configuration scripts replace **set ip secondarynameserver** commands with **add ip dns** commands.

This command sets the IP address of the secondary name server. Name servers are used to resolve Telnet requests to host names that are not in the host name table. If the host is entered into the host table, then no access to a name server is required. This may suit installations that have no name server. When the router performs a DNS lookup, it firsts sends the request to the primary name server. If a response is not received within 20 seconds, the request is sent to the secondary name server.

Example To set the router's secondary name server address to 192.168.2.1, use the command:

```
set ip seco=192.168.2.1
```

Related Commands

add ip host
delete ip host
set ip host
set ip nameserver
show ip
show ip host

set ping

Syntax

```
SET PING [[IPaddress=]{ipadd|ipv6add[%interface]|host}]
   [DElay=seconds] [Length=number] [NUMber={number|
   CONTinuous}] [PATTern=hexnum] [SIPAddress={ipadd|
   ipv6add}] [SCReenoutput={YES|NO}] [TIMEOut=1..65535]
   [TOS=0..255]

SET PING [[IPXAddress=]network:station] [DElay=seconds]
   [Length=number] [NUMber={number|CONtinuous}]
   [PATTern=hexnum] [SIPXaddress=network:station]
   [SCReenoutput={YES|NO}] [TIMEOut=1..65535]

SET PING [[APPLEAddress=]network.node] [DElay=seconds]
   [Length=number] [NUMBER={number|CONtinuous}]
   [PATTern=hexnum] [SAPpleaddress=network.node]
   [SCReenoutput={YES|NO}] [TIMEOut=1..65535]

SET PING [[OSIAddress=]nsap] [DElay=seconds]
   [Length=number] [NUMber={number|CONtinuous}]
   [PATTern=hexnum] [SOSIaddress=nsap] [SCReenoutput={YES|
   NO}] [TIMEOut=1..65535]
```

where:

- *ipadd* is an IPv4 address in dotted decimal notation.
- *ipv6add* is a valid IPv6 address.
- *interface* is the interface the ping request is sent for a request to ping an IPv6 link-local address, e.g. eth0.
- *host* is a host name from the host name table.
- *network:station* is a valid Novell network number and station MAC address, expressed as hexadecimal numbers. Leading zeros may be omitted.
- *network.node* is an AppleTalk network number from 0 to 65279 or an AppleTalk network number in the format “nnnnn-nnnnn”, and an AppleTalk node number from 0 to 127.
- *nsap* is a valid OSI NSAP address in dotted hexadecimal notation.
- *seconds* is a decimal number from 0 to 4294967295.
- *hexnum* is an 8-digit hexadecimal number, optionally proceeded by the characters “0x”.

Description

This command sets the defaults for the **ping** command on page 14-129. The extended **ping** command on page 14-129 supports IPv4, IPv6, IPX, OSI, and AppleTalk addresses.

If there is no default destination and a destination is not specified on the **ping** command on page 14-129, a ping is not generated and an error message is displayed.

The **ipaddress**, **ipxaddress**, **osiaddress**, and **appleaddress** parameters specify the destination address for ping packets for IP, IPX, OSI and AppleTalk networks, respectively. If theses parameters have already been set, they can be restored to their default “not set” state by specifying values of 0.0.0.0 (for IPv4 addresses), :: (for IPv6 addresses), 0:0 (for IPX addresses) or 0.0 (for AppleTalk addresses).

Pinging an IPv6 link-local address requires interface information as well as the address because a single link-local address can belong to several interfaces. To ping a link-local address, specify the interface out which the ping request is to be sent, as well as the address. This interface is the interface, on the router from which the ping request originates, that is, connected to the required destination interface (Figure 15-2 on page 15-19 in Chapter 15, Internet Protocol Version 6 (IPv6)). For example:

```
ping fe80::7c27%eth0
```

The **delay** parameter specifies the time interval in seconds between ping packets. The default is 1 second.

The **length** parameter specifies the number of data bytes of the specified pattern to include in the data portion of the ping packet. If this parameter is not specified, the default is used.

The **number** parameter specifies the number of ping packets to transmit. If this parameter is not specified, the default is used. If **continuous** is specified, the **timeout** parameter must be set to a value greater than 0, and packets are sent continuously until the **stop ping** command on page 14-224 is issued.

The **pattern** parameter specifies the data to use to fill the data portion of the ping packet. If this parameter is not specified, the default is used.

The **sipaddress**, **sipxaddress**, **sosiaddress**, and **sappleaddress** parameters specify the source address to use in ping packets for IP, IPX, OSI, and AppleTalk networks, respectively. If the source address is not specified, and has not been set using the **set ping** command on page 14-163, the default is to use the address of the interface from which the ping packets are transmitted. In the special case of IP addresses, the router's local interface IP address, if set, is used. Otherwise, the IP address of the interface from which the ping packets are transmitted is used. If the ping request is to an IPv6 link-local address, the **sipaddress** must be on the outgoing interface and cannot be a link-local address. If the **sipaddress**, **sipxaddress**, **sosiaddress**, or **sappleaddress** parameters have already been set, they can be restored to their default "not set" state by specifying values of 0.0.0.0 (for IPv4 addresses), :: (for IPv6 addresses), 0:0 (for IPX addresses) or 0.0 (for AppleTalk addresses).

The **screenoutput** parameter specifies whether the output is sent to the terminal. If **yes** is specified, the response time for each echo reply packet is displayed to the terminal as the reply is received from the destination host (Figure 14-13 on page 14-130). If **no** is specified, the results are stored and not displayed. To view the results, use the **show ping** command on page 14-216. If this parameter is not specified, the default is used.

The **timeout** parameter specifies how many seconds to wait for a response to a ping packet, and cannot be zero. If this parameter is not specified, the default is used.

The **tos** parameter specifies the value of the TOS (Type Of Service) field in the IP header of the ping packet. The **tos** parameter is valid for IP addresses, and is ignored for IPv6 addresses. If this parameter is not specified, the default is used.

Related Commands

add ip host
ping
show ping
stop ping

set trace

Syntax

```
SET TRAce [[IPaddress=]ipadd] [ADDROnly={No|OFf|ON|Yes}]
   [MAXTtl=number] [MINTtl=number] [NUMber=number]
   [POrt= 1..65535] [SCReenoutput={No|OFf|ON|Yes}]
   [SOurce=ipadd] [TIMEOut=number] [TOS=0..25]
```

where:

- *ipadd* is an IPv4 address in dotted decimal notation, a valid IPv6 address or a host name from the host name table.
- *number* is a decimal number.

Description

This command sets default options for the trace route command.

If there is no default destination and a destination is not specified with the **trace** command on page 14-225, then a trace is not performed and an error message is displayed.

The **ipaddress** parameter specifies the destination IP address. The command traces the route to this IP address.

The **addronly** parameter specifies whether trace output is presented as IP addresses only, as opposed to IP addresses and their DNS name equivalent. If **on**, output is presented as IP addresses. The default is **on**.

The **maxttl** parameter specifies the maximum value for the TTL (Time To Live) field in the IP packet, and is used to limit the trace route to a maximum number of hops. If this parameter is not specified, the default is used.

The **minttl** parameter specifies the initial value of the TTL (Time To Live) field in the IP packet, and can be used to skip hops at the start of the route. If this parameter is not specified, the default is used.

The **number** parameter specifies the number of packets to send to each hop. If this parameter is not specified, the default is used. A maximum of 100 packets may be transmitted.

The **port** parameter specifies the UDP destination port number for the packets being transmitted. It also detects whether there is an IP device listening on the specified port. If a device is listening, the ICMP “unreachable” message is not returned.

The **screenoutput** parameter specifies whether the output is sent to the terminal. If this parameter is not specified, the default is used.

The **source** parameter specifies the IP address to use as a source address in the packets. If this parameter is not set, the default IPv4 address of the interface is set as the source address. Because this IPv4 address causes a conflict when an

IPv6 address is specified in the **ipaddress** parameter, this parameter is required when tracing a route to an IPv6 address.

The **timeout** parameter specifies how long to wait for a response before sending packets to the next hop. If this parameter is not specified, the default is used. If ICMP "unreachable" messages are received within the timeout period, packets are transmitted to the next hop immediately.

The **tos** parameter specifies the value of the TOS (Type Of Service) field in the IP header of the packets being transmitted. If this parameter is not specified, the default is used.

Related Commands

add ip host
show trace
stop trace
trace

show bootp relay

Syntax

`SHow BOOTp RELAy`

Description

This command displays the current configuration of the BOOTP Relay Agent (Figure 14-14 on page 14-166, Table 14-17 on page 14-167).

Figure 14-14: Example output from the **show bootp relay** command

```
BOOTP Relaying Agent Configuration.

Status       : ENABLED
Maximum Hops : 4

BOOTP Relay Destinations
------------------------
192.231.35.29
------------------------


BOOTP Counter
---------------
InPackets   OutPackets   InRejects   InRequests   InReplies
0000000000  0000000000   0000000000  0000000000   0000000000
```

Table 14-17: Parameters in the output of the **show bootp relay** command

Parameter	Meaning
Status	Whether the BOOTP Relay Agent is enabled.
Maximum Hops	Maximum value allowed for the hops field in a BOOTP message before the message is discarded.
BOOTP Relay Destinations	List of IP addresses where BOOTREQUEST messages are forwarded.
InPackets	Total number of BOOTP packets received.
OutPackets	Total number of BOOTP packets transmitted.
InRejects	Number of incoming BOOTP packets rejected because of an error in the packet.
InRequests	Number of BOOTP requests received.
InReplies	Number of BOOTP replies received.

Related Commands

add bootp relay
delete bootp relay
disable bootp relay
enable bootp relay
purge bootp relay
set bootp maxhops

show ip

Syntax `SHow IP`

Description This command displays general configuration information regarding the router (Figure 14-15 on page 14-168, Table 14-18 on page 14-169).

Figure 14-15: Example output from the **show ip** command

```
IP Module Configuration
------------------------------------------------------------

Module Status .................. ENABLED
IP Packet Forwarding ........... ENABLED
IP Echo Reply .................. ENABLED
Debugging ...................... DISABLED
IP Fragment Offset Filtering ... ENABLED
Default Name Servers
  Primary Name Server .......... 192.168.1.1 (ppp0)
  Secondary Name Server ........ Not Set
Name Server .................... 192.168.1.1 (ppp0)
Secondary Name Server .......... Not Set
Source-Routed Packets .......... Discarded
Remote IP address assignment ... DISABLED
DNS Relay ...................... DISABLED
IP ARP LOG ..................... ENABLED

Routing Protocols

RIP Neighbours ................. 1
EGP Status ..................... DISABLED
Autonomous System Number ....... Not Set
Transfer RIP to EGP ............ DISABLED
ARP aging timer multiplier...... 4 (1024-2048 secs)
OSPF Status .................... DISABLED
IGMP Status .................... ENABLED
DVMRP Status ................... ENABLED
PIM Status ..................... DISABLED
BGP Status ..................... ENABLED

Active Routes

Static ......................... 0
Interface ...................... 1
RIP ............................ 4
EGP ............................ 0
OSPF ........................... 0
BGP ............................ 0
Other .......................... 0
Multicast ...................... 5

IP Filter Configuration

Total filters .................. 0

Dynamic Interfaces ............. 0
```

Table 14-18: Parameters in the output of the **show ip** command

Parameter	Meaning
Module Status	Whether the IP module is enabled.
IP Packet Forwarding	Whether the IP forwarding function is enabled and forwarding or disabled and in server mode.
IP Echo Reply	Whether replies to echo request messages are enabled.
Debugging	Whether the IP debugging facility is enabled.
IP Fragment Offset Filter	Whether the IP fragment offset filtering is enabled.
Default Name Servers	Configuration of default name servers. More detailed information about the default and domain specific name servers can be displayed with the SHOW IP DNS command.
Primary Name Server	IP address of the default primary name server if assigned. If the address was learned using IPCP negotiation, the name of the interface used for the IPCP negotiation is also displayed.
Secondary Name Server	IP address of the default secondary name server if assigned. If the address was learned using IPCP negotiation, the name of the interface used for the IPCP negotiation is also displayed.
Source-Routed Packets	Whether source-routed packets are forwarded or discarded.
Remote IP address assignment	Whether remote IP address assignment is enabled.
DNS Relay	Whether the DNS relay agent is enabled.
IP ARP LOG	Whether ARP logging is enabled.
RIP Neighbours	Number of RIP neighbours defined.
EGP Status	Whether the EGP routing module is enabled.
Autonomous System Number	The autonomous system number used by the EGP and BGP module, or “Not Set” if an autonomous system number is not assigned.
Transfer RIP to EGP	Whether RIP information is transferred to EGP broadcasts.
ARP aging timer multiplier	The multiplier value applied to ARP aging timers, and the resulting current range of ARP aging timer values.
OSPF Status	Whether the OSPF routing module is enabled.
IGMP Status	Whether the IGMP protocol module is enabled.
DVMRP Status	Whether the DVMRP routing module is enabled.
PIM Status	Whether the PIM-SM multicast routing module is enabled.
BGP Status	Whether the BGP-4 (Border Gateway Protocol version 4) module is enabled.
Static	Number of static routes in use.
Interface	Number of interface-related routes in use.
RIP	Number of RIP-derived routes in use.
EGP	Number of EGP-derived routes in use.
OSPF	Number of OSPF-derived routes in use.

Table 14-18: Parameters in the output of the **show ip** command (continued)

Parameter	Meaning
BGP	Number of BGP-derived routes in use.
Other	Number of other routes in use.
Multicast	Number of multicast forwarding table entries.
Filter *n*	Number of patterns in filter *n*.
Total Filters	Total defined IP filters.
Dynamic Interfaces	Number of dynamic interfaces created by the Asynchronous Call Control (ACC) module when a dial-in user initiates a SLIP or asynchronous PPP connection.

Related Commands

disable ip
disable ip debug
disable ip dnsrelay
disable ip egp
disable ip exportrip
disable ip forwarding
disable ip srcroute
disable snmp
enable ip
enable ip debug
enable ip dnsrelay
enable ip egp
enable ip exportrip
enable ip forwarding
enable ip srcroute
enable snmp
set ip nameserver
set ip secondarynameserver

show ip advertise

Syntax `SHow IP ADVertise`

Description This command displays the Router Discovery advertising configuration for all IP interfaces.

Figure 14-16: Example output from the **show ip advertise** command

```
Router Advertisement ................ Enabled

Interface ........................... vlan2
    Advertisement Address ........... 224.0.0.1 (all)
    Max Advertisement Interval ...... 600
    Min Advertisement Interval ...... 450
    Lifetime ........................ 1800
    Advertisements sent ............. 1
    Solicitations received .......... 0

    Logical Interface   IP Address      Advertise   Preference Level
    ----------------------------------------------------------------
    vlan2-0             192.168.1.1     Yes        -1
    vlan2-1             192.168.2.1     Yes         1
```

Table 14-19: Parameters in the output of the **show ip advertise** command

Parameter	Meaning
Router Advertisement	Whether the ICMP Router Discovery advertisements feature is enabled.
Interface	IP physical interface.
Advertisement Address	Either the all-systems multicast address (224.0.0.1) or the limited-broadcast address (255.255.255.255).
Max Advertisement Interval	Maximum time allowed between sending multicast router advertisements.
Min Advertisement Interval	Minimum time allowed between sending multicast router advertisements.
Lifetime	Maximum time that the advertised address should be treated as valid.
Advertisements sent	How many router advertisements the interface has sent since advertising was enabled.
Solicitations received	How many router solicitations the interface has received since advertising was enabled.
Logical Interface	IP logical interface on this physical interface.
IP Address	IP address assigned to the interface.
Advertise	Whether the address for this logical interface should be advertised.
Preference Level	Preferability of the address as a default router address relative to other router addresses on the same subnet.

Related Commands

add ip advertise interface
delete ip advertise interface
disable ip advertise
enable ip advertise
set ip advertise interface

show ip arp

Syntax `SHow IP ARP`

Description This command displays the contents of the ARP cache. The ARP cache contains mappings of IP addresses to physical addresses for hosts to which the router has recently routed packets. To have an entry in the ARP cache, a host must have attempted to access another host, and it must have found the physical address by using the ARP protocol (Figure 14-17 on page 14-172, Table 14-20 on page 14-172).

Figure 14-17: Example output from the **show ip arp** command

```
 Interface     IP Address       Physical Address      ARP Type    Status
----------------------------------------------------------------------------
 eth0          172.16.8.1       AA-00-04-00-2D-08     Dynamic     Active
 eth0          172.16.8.2       AA-00-04-00-28-08     Dynamic     Active
 eth0          172.16.8.34      00-00-0C-02-5A-0A     Dynamic     Active
 eth0          172.16.9.185     08-03-50-37-00-00     Dynamic     Active
 fr0           172.16.240.2     20                    Static      Active
 x25t0         172.16.198.1     Remote1               Static      Active
 eth0          192.168.163.47   FF-FF-FF-FF-FF-FF     Other       Active
 eth0          255.255.255.255  FF-FF-FF-FF-FF-FF     Other       Active
----------------------------------------------------------------------------
```

Table 14-20: Parameters in the output of the **show ip arp** command

Field	Meaning	
Interface	Interface over which the network device is accessed. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen ("-") and the logical interface number.	
IP Address	IP address of the network device.	
Physical Address	Physical address of the network device. For an Ethernet, this is the Ethernet address; for a Frame Relay DLC it is the DLCI; for a MIOX (X.25) circuit it is the circuit name.	
ARP Type	Type of entry:	
	Static	Added with the **add ip arp** command on page 14-64
	Dynamic	Learned from ARP request/reply message exchanges
	Invalid	The interface may not exist
	Other	Automatically generated by the system, for example, general IP broadcast and IP subnet/network broadcast addresses are added when the IP module is configured
Status	Whether the ARP entry is active or inactive.	

Related Commands

add ip arp
delete ip arp
set ip arp

show ip counter

Syntax

```
SHow IP COUnter[={ALL|ARP|EGp|ICmp|INTerface|IP|MUlticast|
    ROutes|SNmp|UDp}]
```

Description

This command displays all or selected parts of the IP MIB. If **all** is specified or no option, then all IP counters are displayed. The MIB can be selectively displayed by specifying one of the following options:

- ARP (Figure 14-18 on page 14-173, Table 14-21 on page 14-173)
- EGP (Figure 14-19 on page 14-174, Table 14-22 on page 14-174)
- ICMP (Figure 14-20 on page 14-174, Table 14-23 on page 14-175)
- INTERFACE (Figure 14-21 on page 14-176, Table 14-24 on page 14-176)
- IP (Figure 14-22 on page 14-177, Table 14-25 on page 14-177)
- MULTICAST (Figure 14-23 on page 14-178, Table 14-26 on page 14-178)
- ROUTE (Figure 14-24 on page 14-179, Table 14-27 on page 14-179)
- SNMP (Figure 14-25 on page 14-180, Table 14-28 on page 14-180)
- UDP (Figure 14-26 on page 14-181, Table 14-29 on page 14-181)

Figure 14-18: Example output from the **show ip counter=arp** command

```
ARP counter

  arpRxPkts ............... 0   arpTxPkts .............. 0
  arpRxReqPkts ............ 0   arpTxReqPkts ........... 0
  arpRxRespPkts ........... 0   arpTxRespPkts .......... 0
  arpRxDiscPkts ........... 0   arpTxDiscPkts .......... 0
```

Table 14-21: Parameters in the output of the **show ip counter=arp** command

Parameter	Meaning
arpRxPkts	Number of ARP packets received.
arpRxReqPkts	Number of ARP request packets received.
arpRxRespPkts	Number of ARP Response packets received.
arpRxDiscPkts	Number of inbound ARP packets discarded.
arpTxPkts	Number of ARP packets transmitted.
arpTxReqPkts	Number of ARP request packets transmitted.
arpTxRespPkts	Number of ARP Response packets transmitted.
arpTxDiscPkts	Number of outbound ARP packets discarded.

Figure 14-19: Example output from the **show ip counter=egp** command

```
EGP counter

  inMsgs .............. 0       outMsgs ............. 0
  inErrors ............ 0       outErrors ........... 0
```

Table 14-22: Parameters in the output of the **show ip counter=egp** command

Parameter	**Meaning**
inMsgs	Number of EGP packets received by the router.
inErrors	Number of EGP packets received discarded because of errors.
outMsgs	Number of EGP packets transmitted by the router.
outErrors	Number of locally generated EGP packets not transmitted due to resource limitations.

Figure 14-20: Example output from the **show ip counter=icmp** command

```
ICMP counters

  inMsgs ................ 0       outMsgs ............... 0
  inErrors .............. 0       outErrors ............. 0
  inDestUnreachs ........ 0       outDestUnreachs ....... 0
  inTimeExcds ........... 0       outTimeExcds .......... 0
  inParamProbs .......... 0       outParamProbs ......... 0
  inSrcQuenchs .......... 0       outSrcQuenchs ......... 0
  inRedirects ........... 0       outRedirects .......... 0
  inEchos ............... 0       outEchos .............. 0
  inEchoReps ............ 0       outEchoReps ........... 0
  inTimestamps .......... 0       outTimestamps ......... 0
  inTimestampReps ....... 0       outTimestampReps ...... 0
  inAddrMasks ........... 0       outAddrMasks .......... 0
  inAddrMaskReps ........ 0       outAddrMaskReps ....... 0
```

Table 14-23: Parameters in the output of the **show ip counter=icmp** command

Parameter	**Meaning**
inMsgs	Number of ICMP packets received.
inErrors	Number of ICMP packets received that had ICMP-specific errors (bad ICMP checksums, bad length, etc).
inDestUnreachs	Number of ICMP Destination Unreachable packets received.
inTimeExcds	Number of ICMP Time Exceeded packets received.
inParamProbs	Number of ICMP Parameter Problem packets received.
inSrcQuenchs	Number of ICMP Source Quench request packets received.
inRedirects	Number of ICMP Redirect request packets received.
inEchos	Number of ICMP echo request (ping) packets received.
inEchoReps	Number of ICMP echo reply packets received.
inTimestamps	Number of ICMP Timestamp request packets received.
inTimestampReps	Number of ICMP Timestamp reply packets received.
inAddrMasks	Number of ICMP Address Mask request packets received.
inAddrMaskReps	Number ICMP Address Mask reply packets received.
outMsgs	Number of ICMP packets transmitted.
outErrors	Number of ICMP packets that should have been transmitted but were not.
outDestUnreachs	Number of ICMP Destination Unreachable packets transmitted.
outTimeExcds	Number of ICMP Time Exceeded packets transmitted.
outParamProbs	Number of ICMP Parameter Problem packets transmitted.
outSrcQuenchs	Number of ICMP Source Quench reply packets transmitted.
outRedirects	Number of ICMP Redirect packets transmitted.
outEchos	Number of ICMP echo request (ping) packets transmitted.
outEchoReps	Number of ICMP echo reply packets transmitted.
outTimestamps	Number of ICMP Timestamp request packets transmitted.
outTimestampReps	Number of ICMP Timestamp reply packets transmitted.
outAddrMasks	Number of ICMP Address Mask request packets transmitted.
outAddrMaskReps	Number of ICMP Address Mask reply packets transmitted.

Figure 14-21: Example output from the **show ip counter=interface** command

```
IP Interface Counters
--------------------------------------------------------------------------------
Interface       ifInPkts    ifInBcastPkts    ifInUcastPkts     ifInDiscards
Type           ifOutPkts   ifOutBcastPkts   ifOutUcastPkts    ifOutDiscards
--------------------------------------------------------------------------------
eth0               23531            23224              307                0
Static               230                0              230                0

eth1                   0                0                0                0
Static             63289            63289                0                0

ppp0                   0                0                0                0
Static                 0                0                0                0
--------------------------------------------------------------------------------
```

Table 14-24: Parameters in the output of the **show ip counter=interface** command

<table>
<tr><th>Parameter</th><th colspan="2">Meaning</th></tr>
<tr><td>Interface</td><td colspan="2">Name of the interface (such as ppp0), or "local" for the local IP interface. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen ("-") and the logical interface number.</td></tr>
<tr><td>Type</td><td colspan="2">Type of interface:</td></tr>
<tr><td></td><td>Static</td><td>Permanent interface that is active and in use</td></tr>
<tr><td></td><td>Dynamic</td><td>Non-permanent interface created, for example by Asynchronous Call Control (ACC), when a dial-in user initiates a SLIP or PPP connection. The interface disappears when the user logs off, when the router is restarted, or when the IP module is reset with the <b>reset ip</b> command on page 14-132.</td></tr>
<tr><td></td><td>Inactive</td><td>An inactive interface is a permanent interface that could not attach to the lower-layer (FR ETH) interface for some reason. The interface is not in use but remains configured and becomes active when the lower-layer attachment succeeds on the next <b>reset ip</b> or <b>restart</b> command. The most common cause of inactive interfaces is the deletion of the lower-layer interface. Inactive interfaces may be deleted by the manager but cannot be modified.</td></tr>
<tr><td>ifInPkts</td><td colspan="2">Number of packets received over the interface.</td></tr>
<tr><td>ifOutPkts</td><td colspan="2">Number of packets transmitted over the interface.</td></tr>
<tr><td>ifInBcastPkts</td><td colspan="2">Number of multicast packets received over the interface.</td></tr>
<tr><td>ifOutBcastPkts</td><td colspan="2">Number of multicast packets transmitted over the interface.</td></tr>
<tr><td>ifInUcastPkts</td><td colspan="2">Number of unicast packets received over the interface.</td></tr>
<tr><td>ifOutUcastPkts</td><td colspan="2">Number of unicast packets transmitted over the interface.</td></tr>
<tr><td>ifInDiscards</td><td colspan="2">Number of packets received over the interface that were discarded.</td></tr>
<tr><td>ifOutDiscards</td><td colspan="2">Number of packets to be transmitted over the interface that were discarded.</td></tr>
</table>

Figure 14-22: Example output from the **show ip counter=ip** command

```
IP counters

  inReceives ......... 1005      outRequests ........... 0
  inHdrErrors ........... 0      outDiscards ........... 0
  inAddrErrors .......... 0      outNoRoutes ........... 0
  inUnknownProtos ....... 0      forwDatagrams ........ 33
  inDiscards ............ 0      routingDiscards ....... 0
  inDelivers .......... 972
  reasmReqds ............ 0      fragCreates ........... 0
  reasmOKs .............. 0      fragOKs ............... 0
  reasmFails ............ 0      fragFails ............. 0

IP Gateway Discards
  tinyFragments ......... 0      spoofedPkts .......... 12
  invalHdrOption ........ 0      dirBroadcasts ......... 0
  saSpoofedPkts ......... 0      saBlockedPkts ......... 0
  saEncodeFails ......... 0
```

Table 14-25: Parameters in the output of the **show ip counter=ip** command

Parameter	Meaning
inReceives	Number of IP packets the router received.
inHdrErrors	Number of IP packets received with header errors.
inAddrErrors	Number of IP packets received with address errors.
inUnKnownProtos	Number of IP packets received with unsupported protocols.
inDiscards	Number of IP packets received but discarded due to resource limitations at the IP level.
inDelivers	Number of IP packets received and passed on by the IP software to other modules.
reasmReqds	Number of IP packets received that needed reassembly.
reasmOKs	Number of IP packets successfully reassembled.
reasmFails	Number of reassembly failures.
outRequests	Number of IP packets requested to be transmitted by higher layers.
outDiscards	Number of output IP packets discarded due to resource limitations at the IP level.
outNoRoutes	Number of output IP packets discarded because no route existed to the destination.
forwDatagrams	Number of IP packets forwarded.
routingDiscards	Number of routing entries discarded even though they were valid (possibly to free up buffer space).
fragCreates	Number of fragments created.
fragOKs	Number of IP packets successfully fragmented.
fragFails	Number of IP packets that needed fragmenting but the IP flags field indicated not to fragment.
tinyFragments	Number of packets discarded because they were part of a tiny fragment attack.
invalHdrOption	Number of packets discarded because they contained an invalid header option.

Table 14-25: Parameters in the output of the **show ip counter=ip** command

Parameter	Meaning
saSpoofedPkts	Number of packets discarded because they claimed to be from a Security Association partner but were not encoded correctly.
saEncodeFails	Number of packets discarded because the Security Association encoding failed.
spoofedPkts	Number of packets discarded because they were spoofed packets.
dirBroadcasts	Number of packets discarded because directed broadcasts are not allowed.
saBlockedPkts	Number of packets a Security Association discards because they originated from addresses that do not belong to the Security Association.

Figure 14-23: Example output from the **show ip counter=multicast** command

```
IP Multicast Counters
------------------------------------------------------------------------
Interface  ifInMultPkts ifInMultDiscard ifOutMultPkts  ifOutMultDiscards
------------------------------------------------------------------------
eth0                123               2           321                 1
eth1               1234               2         12321                 3
-------------------------------------------------------------------------
```

Table 14-26: Parameters in the output of the **show ip counter=multicast** command

Parameter	Meaning
Interface	Name of the interface (such as PPP0), or "local" for the local IP interface. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen ("-") and the logical interface number.
ifInMultPkts	Number of multicast packets received over the interface.
ifInMultDiscard	Number of multicast packets received over the interface that were discarded.
ifOutMultPkts	Number of multicast packets transmitted over the interface.
ifOutMultDiscards	Number of multicast packets to be transmitted over the interface that were discarded.

Figure 14-24: Example output from the **show ip counter=route** command

```
Route Counters
 IP address       NextHop         Interface  Metric  Octets rcvd   Octets sent
--------------------------------------------------------------------------------
 0.0.0.0          202.36.163.21   eth0          4           984             0
 192.168.19.0     202.36.163.21   eth0          3             0             0
 192.168.39.0     202.36.163.21   eth0          4             0             0
 192.168.42.0     202.36.163.21   eth0          4             0             0
 192.168.119.0    202.36.163.21   eth0          4             0             0
 192.168.255.0    202.36.163.21   eth0          3             0             0
 202.36.163.0     0.0.0.0         eth0          1         81504          1468
 202.49.72.0      202.36.163.21   eth0          2             0             0
 202.49.74.0      202.36.163.21   eth0          3             0             0
 203.97.191.0     202.36.163.21   eth0          3             0             0
--------------------------------------------------------------------------------
```

Table 14-27: Parameters in the output of the **show ip counter=route** command

Parameter	Meaning
IP address	IP address of the remote network pointed to by this route – could be any IP address.
NextHop	IP address of the next router on the path to the remote network. Always an address on one of the router's interfaces.
Interface	Interface over which the next hop is reached. This field is blank when the next hop is over an addressless PPP interface. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen ("-") and the logical interface number.
Metric	Cost to reach the remote network. If there are two paths to the same network, the one with the lowest metric is used. If both have the same metric, then the first occurrence is taken.
Octets rcvd	Number of octets of data received over this route.
Octets sent	Number of octets of data transmitted over this route.

Figure 14-25: Example output from the **show ip counter=snmp** command

```
SNMP counters:
  inPkts ................ 0       outPkts .............. 0
  inBadVersions ......... 0       outTooBigs ........... 0
  inBadCommunityNames ... 0       outNoSuchNames ....... 0
  inBadCommunityUses .... 0       outBadValues ......... 0
  inASNParseErrs ........ 0       outGenErrs ........... 0
  inTooBigs ............. 0       outGetRequests ....... 0
  inNoSuchNames ......... 0       outGetNexts .......... 0
  inBadValues ........... 0       outSetRequests ........0
  inReadOnlys ........... 0       outGetResponses ...... 0
  inGenErrs ............. 0       outTraps ............. 0
  inTotalReqVars ........ 0
  inTotalSetVars ........ 0
  inGetRequests ......... 0
  inGetNexts ............ 0
  inSetRequests ......... 0
  inGetResponses ........ 0
  inTraps ............... 0
```

Table 14-28: Parameters in the output of the **show ip counter=snmp** command

Parameter	Meaning
inPkts	Number of SNMP packets the router received.
inBadVersions	Number of SNMP packets with a bad version field the router received.
inBadCommunityNames	Total number of SNMP PDUs delivered to the SNMP agent that used an unknown SNMP community name.
inBadCommunityUses	Total number of SNMP PDUs delivered to the SNMP agent that represented an SNMP operation not allowed by the SNMP community name in the PDU.
inASNParseErrs	Total number of ASN.1 parsing errors, either in encoding or syntax, encountered by the SNMP agent when decoding received SNMP PDUs.
inTooBigs	Total number of valid SNMP PDUs delivered to the SNMP agent for which the value of the errorStatus component was tooBig.
inNoSuchNames	Number of SNMP packets received with an error status of nosuchname.
inBadValues	Number of SNMP packets received with an error status of badvalue.
inReadOnlys	Number of SNMP packets received with an error status of readonly.
inGenErrs	Number of SNMP packets received with an error status of generr.
inTotalReqVars	Total number of SNMP MIB objects requested.
inTotalSetVars	Total number of SNMP MIB objects that were changed.
inGetRequests	Number of SNMP get request packets the router received.
inGetNexts	Number of SNMP get Next request packets the router received.
inSetRequests	Number of SNMP set request packets the router received.

Table 14-28: Parameters in the output of the **show ip counter=snmp** command (continued)

Parameter	Meaning
inGetResponses	Number of SNMP get Response packets the router received.
inTraps	Number of SNMP trap message packets the router received.
outPkts	Number of SNMP packets the router transmitted.
outTooBigs	Number of SNMP packets transmitted with an error status of toobig.
outNoSuchNames	Number of SNMP packets transmitted with an error status of nosuchname.
outBadValues	Number of SNMP packets transmitted with an error status of badvalue.
outGenErrs	Number of SNMP packets transmitted with an error status of generror.
outGetRequests	Number of SNMP get request response packets transmitted by the router.
outGetNexts	Number of get Next response packets the router transmitted.
outSetRequests	Number of set request packets the router transmitted.
outGetResponses	Number of SNMP get response packets transmitted.
outTraps	Number of SNMP Traps the router transmitted.

Figure 14-26: Example output from the **show ip counter=udp** command

```
UDP counters

  inDatagrams ....... 307      outDatagrams ........ 0
  inErrors ............ 0      noPorts ............. 6
```

Table 14-29: Parameters in the output of the **show ip counter=udp** command

Parameter	Meaning
inDatagrams	Number of UDP packets the router received.
inErrors	Number of UDP packets dropped because they contained an error at the UDP layer.
outDatagrams	Number of UDP packets the router transmitted.
noPorts	Number of UDP packets dropped because their destination port was not known.

Related Commands

show ip egp
show ip interface
show ip route
show snmp community
show tcp

show ip debug

Syntax

```
SHow IP DEBug[=1..40]
```

Description

This command displays selected entries from the IP debug queue. The debug queue is enabled with **enable ip debug** command on page 14-120. Incorrectly formatted IP packet headers are captured for later analysis. The queue can have up to 40 entries, each entry consists of the first 64 bytes from the packet in question.

If no packet number is specified, the command returns the number of packets in the queue or that no packets have been found.

The following are possible responses to the **show ip debug** command:

```
No packets are currently stored in the debug queue.
<value> packets are currently stored in the debug queue.
```

Some limited analysis of the captured packets is done. The following are possible responses to the **show ip debug=*n*** command, where *n* is a number between 1 and 40, or the maximum number of packets captured so far.

```
Error = Bad destination or source address
Error = Packet length exceeds interface mtu
Error = Bad IP header checksum
Error = Unknown
Error = Packet IP header length too short
Error = Bad IP version
```

An explanation of the possible cause of these problems is beyond the scope of this document.

Related Commands

disable ip debug
enable ip debug
show ip

show ip dns

Syntax `SHow IP DNS`

Description This command displays information about the DNS name servers used by the router (Figure 14-27 on page 14-183, Table 14-30 on page 14-183).

Figure 14-27: Example output from the **show ip dns** command

```
DNS Server Configuration
--------------------------------------------------------------------------------
Domain                         Int/Status   Primary         Secondary     Requests
--------------------------------------------------------------------------------
ANY                            No           192.168.20.1    192.168.10.2       327
mycorp.local.pc                ppp0-1/Up    10.8.0.1        10.8.0.2            29
--------------------------------------------------------------------------------
Cache:
  Maximum entries ................... 250
  Current entries ................... 172 (50912 bytes)
  Timeout (minutes) ................. 4
  Cache hits ........................ 94
```

Table 14-30: Parameters in the output of the **show ip dns** command

Parameter	Meaning
Domain	Shows either the domain for which the following DNS server configuration applies or the string “ANY” if the configuration applies to all domains not covered specifically by another set of servers.
Int/Status	Interface over which DNS server information is learned - via IPCP (over a PPP interface) or DHCP (over an Ethernet interface). “No” is displayed when the DNS servers are statically configured. Also shows the status of the interface when one is given. “Up” indicates that the interface is operational and “Down” indicates that it is not operational.
Status	Whether the PPP interface over which DNS server information is learned is active. This parameter is present when a PPP interface is displayed for the Interface parameter.
Primary	IP address of the primary DNS server used in resolving domain names that match the Domain parameter. When DNS server information is learned over a PPP interface, this parameter shows an IP address when the interface is active; otherwise it shows “Not set”. When DNS server information is statically configured but no IP has been specified for the primary DNS server, “Not set” is displayed.
Secondary	IP address of the secondary DNS server used in resolving domain names that match the Domain parameter. When DNS server information is learned over a PPP interface, this parameter shows an IP address when the interface is active; otherwise it shows “Not set”. When DNS server information is statically configured but no IP has been specified for the secondary DNS server, “Not set” is displayed.
Requests	Number of requests for which these name servers have been used.
Cache	Configuration of the DNS cache.
Maximum Entries	Maximum number of entries allowed in the DNS at any time.

Table 14-30: Parameters in the output of the **show ip dns** command

Parameter	Meaning
Current Entries	Number of entries currently in the DNS cache. Also shows the amount of RAM being used by the cache.
Timeout	Minutes that an entry can remain in the DNS cache.
Cache Hits	Number of DNS requests that have been successfully matched to an entry in the cache.

Examples To display the router's DNS server settings, use the command:

```
sh ip dns
```

Related Commands

add ip dns
delete ip dns
set ip dns
set ip dns cache
show ip dns cache

show ip dns cache

Syntax

```
SHow IP DNS CAChe
```

Description This command displays the contents of the router's DNS cache (Table 14-28, Table 14-31 on page 14-184).

Figure 14-28: Example output from the **show ip dns cache** command

```
DNS Cache                Entries ... 4 (1184 bytes)
---------------------------------------------------------
Domain Name             IP Address       TTL    Matches
---------------------------------------------------------
www.apples.com          207.145.123.198  213         27
ftp.oranges.co.uk       234.222.145.156   40         12
www.yahoo.co.jp         112.30.241.12    357         32
grex.cyberspace.org     157.29.82.173     12          5
---------------------------------------------------------
```

Table 14-31: Parameters in the output of the **show ip dns cache** command

Parameter	Meaning
Entries	Number of domain names that currently have a record in the DNS cache. Also shows the amount of RAM the cache is using.
Domain Name	Domain related to the cache entry.
IP Address	IP address to which traffic for the domain is to be sent.
TTL	Maximum amount of time in seconds that the entry remains in the cache.
Matches	Number of times the cache entry has been used to respond to a DNS request.

Examples To display information about the contents of the DNS cache, use the command:

```
sh ip dns cac
```

Related Commands

add ip dns
delete ip dns
set ip dns
show ip dns

show ip egp

Syntax

```
SHow IP EGP
```

Description This command displays the current state of the links to known EGP neighbours (Figure 14-29, Table 14-32 on page 14-185).

Figure 14-29: Example output from the **show ip egp** command

```
Neighbour address     State   Mode     Remote AS   Hello   Poll   Reaches
--------------------------------------------------------------------------
130.216.0.0           up     active      64          10      10      2
130.217.0.0           up     active      63          10      10      2
172.16.0.0            down   active      56          10      10      2
```

Table 14-32: Parameters in the output of the **show ip egp** command

Parameter	Meaning
Neighbour address	IP address of the neighbour.
State	Whether the link is in idle, acquisition, down, up, or cease mode.
Mode	Whether the link is active.
Remote AS	Remote autonomous system number of the EGP neighbour.
Hello	The hello timer period in seconds.
Poll	Poll timer in seconds.
Reaches	Number of reachability indicators received. The maximum is 4.

Related Commands

add ip egp
delete ip egp
disable ip egp
disable ip exportrip
enable ip egp
enable ip exportrip
set ip autonomous in Chapter 49, Border Gateway Protocol version 4 (BGP-4)
show ip counter

show ip filter

Syntax `SHow IP FILter[=0..399]`

Description This commands displays information about filters. If a filter is specified, the patterns in the filter are displayed. If a filter is not specified, the patterns in all filters are displayed (Figure 14-30 on page 14-186, Table 14-33 on page 14-187).

Figure 14-30: Example output from the **show ip filter** command

```
IP Filters
--------------------------------------------------------------------------------
No. Ent. Source Port   Source Address    Source Mask      Session           Size
         Dest. Port    Dest. Address     Dest. Mask       Prot.(C/T)     Options
         Type          Act/Pol/Pri       Logging                         Matches
--------------------------------------------------------------------------------
 1    1  Any           192.168.163.23    255.255.255.255  Any                 No
         Any           192.168.163.39    255.255.255.255  Any                 No
         General       Exclude           Off                                   0
      2  Any           192.168.163.24    255.255.255.255  Any                 No
         23            192.168.163.39    255.255.255.255  Any                 No
         General       Exclude           Off                                   0
      3  Any           192.168.163.22    255.255.255.255  Any                 No
         23            192.168.163.39    255.255.255.255  Any                 No
         General       Exclude           Off                                   0
      4  Any           192.168.163.21    255.255.255.255  Any                 No
         23            192.168.163.39    255.255.255.255  TCP                 No
         General       Exclude           Off                                   0

     Requests: 636          Passes: 0            Fails: 636
--------------------------------------------------------------------------------
 2    1  Any           192.168.166.2     255.255.255.255  Any                Yes
         Any           192.168.163.39    255.255.255.255  Any                 No
         General       Include           Off                                   0
      2  Any           192.168.163.21    255.255.255.255  Any                Yes
         23            192.168.163.39    255.255.255.255  TCP                 No
         General       Exclude           Off                                   0

     Requests: 0            Passes: 0            Fails: 0
--------------------------------------------------------------------------------
 3    1  2:34          192.168.163.0     255.255.255.0    Start              Yes
         Any           Any               Any              TCP                 No
         General       Include           Off                                   0

     Requests: 0            Passes: 0            Fails: 0
--------------------------------------------------------------------------------
```

Table 14-33: Parameters in the output of the **show ip filter** command

<table>
<tr><th>Parameter</th><th colspan="2">Meaning</th></tr>
<tr><td>No.</td><td colspan="2">Number of the filter.</td></tr>
<tr><td>Ent.</td><td colspan="2">Entry number in this filter for the pattern.</td></tr>
<tr><td>Source Port</td><td colspan="2">Source IP port for this pattern.</td></tr>
<tr><td>Source Address</td><td colspan="2">Source IP address for this pattern.</td></tr>
<tr><td>Source Mask</td><td colspan="2">Source IP address mask for this pattern.</td></tr>
<tr><td rowspan="4">Session</td><td colspan="2">The type of TCP packet to match when the Pro field contains TCP:</td></tr>
<tr><td>Start</td><td>Matches TCP packets with the SYN bit set and the ACK bit clear</td></tr>
<tr><td>Established</td><td>Matches TCP packets with either the SYN bit clear or the ACK bit set</td></tr>
<tr><td>Any</td><td>Matches any TCP packet</td></tr>
<tr><td>Size</td><td colspan="2">Maximum reassembly size for IP fragments, or “Any” if no maximum size has been set.</td></tr>
<tr><td>Dest. Port</td><td colspan="2">Destination IP port for this pattern.</td></tr>
<tr><td>Dest. Address</td><td colspan="2">Destination IP address for this pattern.</td></tr>
<tr><td>Dest. Mask</td><td colspan="2">Destination IP address mask for this pattern.</td></tr>
<tr><td>Prot. (C/T)</td><td colspan="2">Protocol for this pattern; either ANY, EGP, ICMP, OSPF, TCP, or UDP. For the ICMP protocol, the ICMP code and type are also listed.</td></tr>
<tr><td>Options</td><td colspan="2">IP options field for this pattern; either Any, Yes, or No.</td></tr>
<tr><td>Type</td><td colspan="2">Whether the pattern type is general or specific.</td></tr>
<tr><td>Act/Pol/Pri</td><td colspan="2">Filter action for traffic filters (either Exclude or Include), the policy number for policy filters, or the priority of priority filters.</td></tr>
<tr><td>Logging</td><td colspan="2">Whether matches to this entry generate messages to the router’s Logging facility, and the content of log messages; either Off, Head, Dump, or a number from 4 to 1600.</td></tr>
<tr><td>Matches</td><td colspan="2">Number of IP packets that have matched this pattern.</td></tr>
<tr><td>Requests</td><td colspan="2">Number of IP packets checked against this filter.</td></tr>
<tr><td>Passes</td><td colspan="2">Number of IP packets included by this filter.</td></tr>
<tr><td>Fails</td><td colspan="2">Number of IP packets excluded by this filter.</td></tr>
</table>

Related Commands

add ip filter
add ip trusted
delete ip filter
delete ip trusted
set ip filter
show ip trusted

show ip helper

Syntax `SHow IP HElper [COUnter]`

Description This command displays information about the state of broadcast forwarding on the router. If no optional parameters are specified, the current configuration is displayed (Figure 14-31 on page 14-188, Table 14-34 on page 14-188).

If the **counter** parameter is specified, counters for forwarded traffic are displayed (Figure 14-32 on page 14-188, Table 14-35 on page 14-189).

Figure 14-31: Example output from the **show ip helper** command

```
IP HELPER Configuration

Status : Enabled
--------------------------------------------
Interface : eth0
  UDP port : 137
     Destination(s) ...... 192.168.2.2
  UDP port : 138
     Destination(s) ...... 192.168.2.2
--------------------------------------------
```

Table 14-34: Parameters in the output of the **show ip helper** command

Parameter	Meaning
Status	Whether broadcast forwarding is enabled.
Interface	Interface where broadcast UDP packets are received. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), interface names include a hyphen and the logical interface number.
UDP port	UDP port number to be matched against UDP broadcast packets that are received. If the port number of a UDP packet matches one on the list, then the packet is forwarded to each of the destination IP addresses.
Destination	Destination IP address where matching broadcast UDP packets are forwarded.

Figure 14-32: Example output from the **show ip helper counter** command

```
IP HELPER Counters

----------------------------------------
Interface : eth0
  InPackets ........... 1
  InNoDestination ...... 0
  Port : 137
    OutPackets ......... 0
  Port : 138
    OutPackets ......... 1
----------------------------------------
```

Table 14-35: Parameters in the output of the **show ip helper counter** command

Parameter	Meaning
Interface	Interface where broadcast UDP packets are received. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen and the logical interface number.
InPackets	Number of broadcast UDP packets received on the interface. Opening a UDP listen port means that all matching UDP packets received on any interface are processed.
InNoDestination	Number of broadcast UDP packets received on the interface that did not match a requested port.
Port	UDP port number to be matched against UDP broadcast packets that are received.
OutPackets	Number of packets forwarded to the destinations listed for the UDP port.

Examples To display the current status of broadcast forwarding, use the command:

```
sh ip he
```

Related Commands

add ip helper
delete ip helper
disable ip helper
enable ip helper

show ip host

Syntax

```
SHow IP HOst
```

Description This command displays the IP host name table and the IP address of the nameserver, if defined. (Figure 14-33 on page 14-190, Table 14-36 on page 14-190). A host name can be any arbitrary string and need not be the full domain name. The host name table makes it easier to Telnet to commonly accessed hosts by enabling the user to enter a shorter, easier to remember name for the host rather than the host's full IP address or domain name.

When a host name is specified in the **telnet** command on page 21-31 of Chapter 21, Terminal Server, the entire name is used to match a name in the host name table. All characters are used in the comparison, including nonalphabetic characters if they are present. The comparison is not case-sensitive.

Figure 14-33: Example output from the **show ip host** command

```
  IP Address         Host Name
------------------------------------------------------------
  172.16.8.2         ip4
  172.16.8.3         Zaphod
  172.29.2.8         Admin
------------------------------------------------------------
```

Table 14-36: Parameters in the output of the **show ip host** command

Parameter	Meaning
IP Address	IP address of an IP host.
Host name	Nickname of the IP host that can be used in the **telnet** command on page 21-31 of Chapter 21, Terminal Server (for example, telnet zaphod).

Related Commands **add ip host**
delete ip host
set ip host
set ip nameserver
set ip secondarynameserver

show ip icmpreply

Syntax `SHow IP ICMPreply`

Description This command displays the status of configurable ICMP messages (Figure 14-34 on page 14-190, Table 14-37 on page 14-190).

Figure 14-34: Example output from the **show ip icmpreply** command

```
SHOW IP ICMP REPLY MESSAGES
---------------------------------------------------
ICMP REPLY MESSAGES:
  Network Unreachable ................ disabled
  Host Unreachable ................... disabled
  Redirect ........................... enabled
---------------------------------------------------
```

Table 14-37: Parameters in the output of the **show ip icmpreply** command

Parameter	Meaning
ICMP Reply Messages	List of ICMP-configurable reply messages and whether they are enabled.

Related Commands **enable ip icmpreply**
disable ip icmpreply

show ip interface

Syntax `SHow IP INTerface[=`*`interface`*`] [COUnter[=MULticast]]`

where *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description This command displays interface configuration information for interfaces assigned to the IP module with the **add ip interface** command on page 14-77. If an interface is specified, details for the interface is displayed; otherwise, information for all IP interfaces is displayed (See Figure 14-35 on page 14-191, Table 14-38 on page 14-192).

A hash symbol (#) after the interface name indicates that the interface has an operational status of “down”. Note that interface routes are propagated by RIP when their status at a physical level is “up”.

The **counter** parameter displays counters for all interfaces or a specific one (Figure 14-35 on page 14-191, Figure 14-36 on page 14-193, and Table 14-39 on page 14-194).

Figure 14-35: Example output from the **show ip interface** command

```
Interface     Type     IP Address       Bc Fr PArp  Filt RIP Met.   SAMode  IPSc
Pri. Filt     Pol.Filt Network Mask     MTU   VJC   GRE  OSPF Met.  DBcast  Mul.
VLAN Tag
--------------------------------------------------------------------------------
Local         ---      Not set          -  -  -     ---  --         Pass    --
---           ---      Not set          1500  -     ---  --         ---     ---
Loopback      192.168.10.100            -  n  -     --   -          -       --
---           ---      -                -     -     --   -          -       ---
eth0-0        Static   192.168.2.1      1  n  On    ---  01         Pass    No
---           ---      255.255.255.0    1500  -     ---  0000000001 No      Rec
1
eth0-1        Static   192.101.2.1      1  n  On    ---  01         Pass    No
---           ---      255.255.255.0    1500  -     ---  0000000001 No      Rec
3
eth0-2        Static   192.168.23.3     1  n  On    ---  01         Pass    No
---           ---      255.255.255.0    1500  -     ---  0000000001 No      Rec
4
--------------------------------------------------------------------------------
```

Table 14-38: Parameters in the output of the **show ip interface** command

<table>
<tr><th>Parameter</th><th colspan="2">Meaning</th></tr>
<tr><td>Interface</td><td colspan="2">Name of the interface (such as PPP0), or “local” for the local IP interface. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen (“-”) and the logical interface number.</td></tr>
<tr><td rowspan="5">Type</td><td colspan="2">Type of interface:</td></tr>
<tr><td>Static</td><td>Active and in use</td></tr>
<tr><td>Dynamic</td><td>Non-permanent interface created, for example by Asynchronous Call Control (ACC), when a dial-in user initiates a SLIP or PPP connection. This interface disappears when the user logs off, when the router is restarted, or when the IP module is reset with the <b>reset ip</b> command on page 14-132.</td></tr>
<tr><td>Inactive</td><td>Permanent interface that could not attach to the lower-layer (FR, PPP, ETH, etc) interface for some reason. This interface is not in use but remains configured and becomes active when the lower-layer attachment succeeds on the next <b>reset ip</b> or <b>restart</b> command. The most common cause of inactive interfaces is the deletion of the lower-layer interface. Inactive interfaces can be deleted by the manager but cannot be modified.</td></tr>
<tr><td>Loopback</td><td>Virtual interface that is not attached to a physical interface. Only local interfaces 1-15 can be configured as local interfaces.</td></tr>
<tr><td>IP Address</td><td colspan="2">IP address assigned to this interface. For an interface configured with DHCP, this field shows the value assigned by DHCP, or 0.0.0.0 if a DHCP reply has not yet been received.</td></tr>
<tr><td>Bc</td><td colspan="2">This parameter is set to 0 if an all ‘0’ broadcast is required and ‘1’ otherwise. It defaults to ‘1’.</td></tr>
<tr><td>PArp</td><td colspan="2">Whether this interface supports proxy ARP and if ARP responses will be generated if a default route exists; one of “On” (respond to ARP Requests only if a specific route exists), “Off” or “Def” (respond to ARP Requests if a specific route or a default route exists). This option is valid for Eth or VLAN interfaces.</td></tr>
<tr><td>Fr</td><td colspan="2">Whether packets larger than the interface MTU are fragmented, which overrides the “Do not fragment” bit.</td></tr>
<tr><td>Filt</td><td colspan="2">Number of the traffic filter applied to the interface, if any assigned.</td></tr>
<tr><td>RIP Met.</td><td colspan="2">RIP metric associated with transmitting packets over this interface.</td></tr>
<tr><td>SAMode</td><td colspan="2">Whether packets that do not belong to a security association assigned to the interface are forwarded or discarded.</td></tr>
<tr><td>IPSc</td><td colspan="2">Whether an IPSec policy is attached to the interface.</td></tr>
<tr><td>Pri. Filt</td><td colspan="2">Number of the priority filter applied to the interface, if any.</td></tr>
<tr><td>Pol.Filt</td><td colspan="2">Number of the policy filter applied to the interface, if any.</td></tr>
<tr><td>Network Mask</td><td colspan="2">Subnet mask assigned to the IP address of this interface. For an interface configured with DHCP, this field shows the value assigned by DHCP, or 0.0.0.0 if a DHCP reply has not been received.</td></tr>
<tr><td>MTU</td><td colspan="2">Maximum packet size that can be transmitted over this interface.</td></tr>
<tr><td>VJC</td><td colspan="2">Whether Van Jacobson’s header compression is active on the interface. This option is valid for PPP interfaces.</td></tr>
<tr><td>GRE</td><td colspan="2">Number of the GRE entity associated with the interface, if any.</td></tr>
</table>

Table 14-38: Parameters in the output of the **show ip interface** command (continued)

Parameter	**Meaning**	
OSPF Met.	OSPF metric associated with transmitting packets over this interface.	
DBcast	Whether network and subnet broadcasts are forwarded to the network attached to the interface.	
Mul.	How multicast packets are handled on the interface:	
	On	Sent and received
	Rec	Received but not sent
	Snd	Sent but not received
	Off	Neither sent nor received
VLAN Tag	VID (VLAN Identifier) included in the header of each frame transmitted over the Eth interface.	

Figure 14-36: Example output from the **show ip interface counter** command

```
IP Interface Counters
--------------------------------------------------------------------------------
Interface       ifInPkts    ifInBcastPkts    ifInUcastPkts     ifInDiscards
Type           ifOutPkts   ifOutBcastPkts   ifOutUcastPkts    ifOutDiscards
--------------------------------------------------------------------------------
eth0               23531            23224              307                0
Static               230                0              230                0

eth1                   0                0                0                0
Static             63289            63289                0                0

--------------------------------------------------------------------------------
```

Table 14-39: Parameters in the output of the **show ip interface counter** command

<table>
<tr><th>Parameter</th><th colspan="2">Meaning</th></tr>
<tr><td>Interface</td><td colspan="2">The name of the interface (such as PPP0), or “local” for the local IP interface. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen (“-”) and the logical interface number.</td></tr>
<tr><td rowspan="4">Type</td><td colspan="2">Type of interface:</td></tr>
<tr><td>Static</td><td>Active and in use</td></tr>
<tr><td>Dynamic</td><td>Non-permanent interface created, for example by Asynchronous Call Control (ACC), when a dial-in user initiates a SLIP or PPP connection. This interface disappears when the user logs off, when the router is restarted, or when the IP module is reset with the **reset ip** command on page 14-132.</td></tr>
<tr><td>Inactive</td><td>Permanent interface that could not attach to the lower-layer (FR, PPP, ETH, etc) interface for some reason. This interface is not in use but remains configured and becomes active when the lower-layer attachment succeeds on the next **reset ip** or **restart** command. The most common cause of inactive interfaces is the deletion of the lower-layer interface. Inactive interfaces can be deleted by the manager but cannot be modified.</td></tr>
<tr><td>ifInPkts</td><td colspan="2">Number of packets received over the interface.</td></tr>
<tr><td>ifOutPkts</td><td colspan="2">Number of packets transmitted over the interface.</td></tr>
<tr><td>ifInBcastPkts</td><td colspan="2">Number of multicast packets received over the interface.</td></tr>
<tr><td>ifOutBcastPkts</td><td colspan="2">Number of multicast packets transmitted over the interface.</td></tr>
<tr><td>ifInUcastPkts</td><td colspan="2">Number of unicast packets received over the interface.</td></tr>
<tr><td>ifOutUcastPkts</td><td colspan="2">Number of unicast packets transmitted over the interface.</td></tr>
<tr><td>ifInDiscards</td><td colspan="2">Number of packets received via the interface that were discarded.</td></tr>
<tr><td>ifOutDiscards</td><td colspan="2">Number of packets to be transmitted over the interface that were discarded.</td></tr>
</table>

Related Commands

add ip interface
delete ip interface
disable ip interface
enable ip interface
reset ip interface
set ip interface
show ip counter

show ip nat

Syntax `SHow IP NAT [COUnter] [SUMmary]`

Description This command displays information about the configuration of NAT on the router, or counters for traffic handled by NAT. If no optional parameters are specified, information about the configuration of NAT is displayed (Figure 14-37 on page 14-195, Table 14-40 on page 14-196).

If **counter** is specified, traffic counters for packets processed by NAT are displayed (Figure 14-38 on page 14-197, Table 14-41 on page 14-198). If **summary** is specified, only summary information is displayed.

Figure 14-37: Example output from the **show ip nat** command

```
IP NAT Configuration

Status  : Enabled
Logging : Fails
Enhanced Fragment Handling: udp
Maximum Packet Fragments : 20
-------------------------------------------------------------------------------
Private IP : 10.20.20.0 - 10.20.20.255
Global IP  : 192.168.34.96
  Method .................. Dynamic ENAT
  Number of entries ....... 5
  Current port ............ 5062
  ENAT static configurations:
    Protocol  PrivateIP:Port        GlobalIP:Port
    TCP       10.20.20.5:23         192.168.34.97:23
    TCP       10.20.20.4:23         192.168.34.96:23
  Current entries:
    Protocol  PrivateIP:Port        GlobalIP:Port         DestinationIP:Port
    TCP       10.20.20.4:53330      192.168.34.96:5027    202.1.1.20:23
      Start time .............. 23:01:11 04-Mar-1997
      TCP state ............... established
      Minutes to deletion ..... 1438
    TCP       10.20.20.4:23         192.168.34.96:23      202.1.1.56:1025
      Start time .............. 22:58:29 04-Mar-1997
      TCP state ............... established
      Minutes to deletion ..... 1435
    TCP       10.20.20.10:1024      192.168.34.96:5013    202.1.1.20:21
      Start time .............. 23:00:18 04-Mar-1997
      TCP state ............... established
      Minutes to deletion ..... 1437
    ICMP      10.20.20.4:19         192.168.34.96:5039    202.1.1.42
      Start time .............. 23:02:27 04-Mar-1997
      ICMP type ............... Echo request
      Minutes to deletion ..... 2
    UDP       10.20.20.4:2051       192.168.34.96:5020    202.1.1.20:53
      Start time .............. 23:01:11 04-Mar-1997
      Minutes to deletion ..... 3
-------------------------------------------------------------------------------
```

Table 14-40: Parameters in the output of the **show ip nat** command

Parameter	**Meaning**
Status	Whether NAT is enabled.
Logging	NAT events being logged.
Enhanced Fragment Handling	A list of the protocol types for which large fragmented packets can be handled when IP NAT is enabled. If "other" is listed, protocols that are not ICMP or UDP (or TCP) are permitted to send large fragmented packets. If "none" is listed, no protocols are allowed to send large fragmented packets. If a protocol is not listed, or "none" is listed, then the default fragment constraints apply, which means that the IP packet must consist of no more than 8 fragments with a total of 1780 bytes of data.
Maximum Packet Fragments	Maximum number of fragments that a packet may consist of when enhanced fragment handling is enabled.
Private IP	IP address of a host or a range of them on the private network.
Global IP	Officially assigned global IP address or a range of them.
Global interface	Interface connected to the Internet with an officially assigned global IP address. This is displayed for interface ENAT entries. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen ("-") and the logical interface number.
Method	Method of translation; either Static NAT, Dynamic NAT, Static ENAT, Dynamic ENAT, or Interface ENAT.
Number of entries	Number of current TCP sessions, UDP flows, or ICMP requests currently mapped by NAT for an interface.
Current port	Last assigned unique port or sequence number NAT uses for the interface. This is displayed for ENAT entries.
Protocol	Protocol or IP protocol number. of the port.
PrivateIP:Port	IP address of a host on the private network and the local port on that host of the session or flow.
GlobalIP:Port	IP address and port where the private IP address and port are translated before being forwarded.
DestinationIP:Port	Destination IP address and port where the session or flow is forwarded.
Start time	Time the session or flow started; not displayed for summary information.
TCP state	For TCP sessions, the state of the TCP session; not displayed for summary information.
ICMP type	For ICMP flows, this is the type of the ICMP frame that started the flow; not displayed for summary information.
Minutes to deletion	Time before the session or flow when information is terminated in the absence of any further traffic; not displayed for summary information.

Figure 14-38: Example output from the **show ip counter** command

```
IP NAT Counters

-------------------------------------------------------------------------------
Private IP : 10.20.20.0 - 10.20.20.255
Global IP  : 192.168.34.96
  Total packets received from private address(es) ....... 256
  Total packets received by global address(es) .......... 290
  Number of cache hits from private address(es) ......... 247
  Number of cache hits from global address(es) .......... 284
  Number of entries created for configuration ........... 10
  Number of dropped packets due to no match ............. 0
  Number of unknown IP protocols packets dropped ........ 0
  Number of unknown ICMP type packets dropped ........... 0
  Number of dropped ICMP packets ........................ 0
  Number of spoofing packets for private address(es) .... 0
  Number of dropped packets as global address zero ...... 0
  Number of dropped packets due to no spare entries ..... 0
  Number of FTP port commands processed ................. 1
  Number of FTP port commands dropped ................... 0
  ENAT static configurations:
    Protocol  PrivateIP:Port        GlobalIP:Port         Number hits
    TCP       10.20.20.5:23         192.168.34.97:23      0
    TCP       10.20.20.4:23         192.168.34.96:23      1
  Current entries:
    Protocol  PrivateIP:Port        GlobalIP:Port         DestinationIP:Port
    TCP       10.20.20.4:53330      192.168.34.96:5027    202.1.1.20:23
      Packets from private IP ............. 92
      Octets from private IP .............. 3776
      Packets to private IP ............... 96
      Octets to private IP ................ 4234
    TCP       10.20.20.4:23         192.168.34.96:23      202.1.1.56:1025
      Packets from private IP ............. 79
      Octets from private IP .............. 3455
      Packets to private IP ............... 102
      Octets to private IP ................ 4165
    TCP       10.20.20.10:1024      192.168.34.96:5013    202.1.1.20:21
      Packets from private IP ............. 16
      Octets from private IP .............. 712
      Packets to private IP ............... 16
      Octets to private IP ................ 999
    ICMP      10.20.20.4:19         192.168.34.96:5039    202.1.1.42
      Packets from private IP ............. 1
      Octets from private IP .............. 56
      Packets to private IP ............... 0
      Octets to private IP ................ 0
    UDP       10.20.20.4:2051       192.168.34.96:5020    202.1.1.20:53
      Packets from private IP ............. 1
      Octets from private IP .............. 67
      Packets to private IP ............... 1
      Octets to private IP ................ 134
-------------------------------------------------------------------------------
```

Table 14-41: Parameters in the output of the **show ip nat counter** command

Parameter	Meaning
Private IP	IP address of a host or a range of them on the private network.
Global IP	Officially assigned global IP address or range of them.
Global interface	Interface connected to the Internet with an officially assigned global IP address. This is displayed for interface ENAT entries. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen ("-") and the logical interface number.
Interface	Interface associated with the private IP network
Total packets received from private address(es)	Number of packets received from hosts with the private IP address or range of addresses.
Total packets received by global address(es)	Number of packets received from external Internet hosts addressed to the global IP address or range of addresses.
Number of cache hits from private address(es)	Number of IP packets received from the private network that matched a known session or flow.
Number of cache hits from global address(es)	Number of IP packets received from the global Internet that matched a known session or flow.
Number of entries created for configuration	Number of sessions or flow entries created for this private network address or range.
Number of dropped packets due to no match	Number of IP packets dropped because there was not a current flow or session started to map the packet to its destination.
Number of unknown IP protocols packets dropped	Number of IP packets dropped because they contained an IP protocol other than the protocols listed on the *Protocols* field.
Number of unknown ICMP type packets dropped	Number of ICMP packets dropped because the ICMP type field was set to an unknown value.
Number of dropped ICMP packets	Number of ICMP packets dropped because there was no known destination for the packet.
Number of spoofing packets for private address(es)	Number of packets dropped because they were directly addressed to the private network.
Number of dropped packets as global address zero	Number of packets dropped because the global IP address for the interface was set to zero. This may happen on a interface ENAT where the IP address has still to be determined. The router buffers a limited number of packets while trying to bring up the link but drops packets and increment this counter.
Number of dropped packets due to no spare entries	Number of IP packets that failed to create a new session or flow entry and were dropped because the router reached the maximum number of session or flow entries.
Number of FTP port commands processed	Number of FTP port commands processed. Each FTP port command requires special processing by NAT because the IP address of the source is embedded as ASCII text.
Number of FTP port commands dropped	Number of FTP port commands that could not be translated successfully.
Protocol	Protocol or IP protocol number of the port.

Table 14-41: Parameters in the output of the **show ip nat counter** command (continued)

Parameter	**Meaning**
PrivateIP:Port	IP address of a host on the private network and the local port on that host of the session or flow.
GlobalIP:Port	IP address and port to which the private IP address and port are translated before forwarding.
Number Hits	IP protocol number of sessions or flows that have started to the specified entry.
DestinationIP:Port	Destination IP address and port where the session or flow is forwarded.
Packets from private IP	Number of packets NAT received for the entry from the host on the private network.
Octets from private IP	Number of octets NAT received for the entry from the host on the private network.
Packets to private IP	Number of packets NAT sent to the host on the private network for this session or flow.
Octets to private IP	Number of octets NAT sent to the host on the private network for this session or flow.

Examples To show the NAT configuration, use the command:

```
sh ip nat
```

Related Commands

add ip nat
delete ip nat
disable ip nat
enable ip nat

show ip pool

Syntax

```
SHow IP POOL[=pool-name] [IPaddress=ipadd[-ipadd]]
    [SUMmary]
```

where:

- *pool-name* is a character string 1 to 15 characters long. Valid characters are any printable characters. If *pool-name* contains spaces, it must be in double quotes.
- *ipadd* is an IP address in dotted decimal notation.

Description This command displays information about a single IP address pool or all IP address pools.

The **pool** parameter specifies the name of an existing pool to display. If a value is not specified, information for all defined IP pools is displayed (Figure 14-39 on page 14-200, Table 14-42 on page 14-200).

The **ip** parameter limits the display to a specific IP address or range of IP addresses from the pool.

If **summary** is specified, summary information is displayed (Figure 14-40 on page 14-200).

Figure 14-39: Example output from the **show ip pool** command.

```
IP Pool
--------------------------------------------------------------------------------
Pool Name: dialin ( 192.168.1.1 - 192.168.1.8 )
Number of requests ........................ 102
Request successes ......................... 101
Request failures .......................... 1
Number in use ............................. 5
IP Address Interface Status Start Time End time
192.168.1.1 PPP0 inuse 24-Jun-1999 15:21:58
192.168.1.2 PPP1 free  24-Jun-1999 10:02:04 24-Jun-1999 16:23:50
192.168.1.3 PPP2 inuse 24-Jun-1999 15:32:17
192.168.1.4 PPP3 inuse 24-Jun-1999 15:36:01
192.168.1.5 PPP4 inuse 24-Jun-1999 15:37:46
192.168.1.6 PPP5 inuse 24-Jun-1999 15:51:06
192.168.1.7 PPP6 free  24-Jun-1999 15:59:51 24-Jun-1999 16:03:11
192.168.1.8      free never used
--------------------------------------------------------------------------------
```

Table 14-42: Parameters in the output of the **show ip pool** command

Parameter	Meaning
Pool Name	Name of the IP address pool and the IP addresses assigned to the pool.
Number of requests	Total number of requests to allocate an IP address from the specified pool.
Request successes	Number of successful requests to allocate an IP address from the specified pool.
Request failures	Number of failed requests to allocate an IP address from the specified pool.
Number in use	Number of IP addresses currently in use for the specified pool.
IP Address	IP address in the specified pool.
Interface	Interface that last requested the IP address.
Status	Whether the IP address is in use or free.
Start Time	Date and time the IP address was allocated from the pool.
End Time	Data and time the IP address was released back to the pool.

Figure 14-40: Example output from the **show ip pool summary** command

```
IP Pool
--------------------------------------------------
Pool Name: dialin ( 192.168.1.1 - 192.168.1.16 )
Number of requests ........................ 102
Request successes ......................... 101
Request failures .......................... 1
Number in use ............................. 5
--------------------------------------------------
```

Examples To display detailed information about the IP address pool named “dialin”, use the command:

```
sh ip pool=dialin
```

Related Commands create ip pool
destroy ip pool

show ip rip

Syntax `SHow IP RIP [INTerface=`*`interface`*`] [CIRCuit=`*`miox-circuit`*`] [DLCi=`*`dlci`*`] [IP=`*`ipadd`*`]`

where:

- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number in the range. If a logical interface is not specified, 0 is assumed.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is **not** case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit) from 0 to 1023.
- *ipadd* is an IP address in dotted decimal notation.

Description This command displays information about the RIP configuration for IP (Figure 14-41 on page 14-201,Figure 14-42 on page 14-202, and Table 14-43 on page 14-202). The **interface**, **circuit**, **dlci** and **ip** parameters can be used to restrict the display to RIP neighbours on specific interfaces, MIOX circuits, Frame Relay DLCs or with specific IP addresses. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

Figure 14-41: Example output from the **show ip rip** command

```
Interface  IP Address     Send     Receive    Demand     Static   NextHop     Auth
           Password
-------------------------------------------------------------------------------------
eth0       -              COMP     BOTH       NO         YES        -         NO
           NO
ppp0       172.16.249.34  RIP1     RIP2       YES        NO         -         PASS
           ********
ppp1       172.16.250.2   RIP2     NONE       YES        YES        -         PASS
           NOT SET
-------------------------------------------------------------------------------------
```

Figure 14-42: Example output from the **show ip rip** command (X.25, Frame Relay interface)

```
Interface  Circuit/DLCI   IP Address     Send       Receive  Dmd    Stc    Nexthop
           Auth           Password
----------------------------------------------------------------------------------
eth0         -             -             COMP       BOTH     YES    NO        -
             NO           NO
ppp0         -            172.16.249.34  RIP1       RIP2     NO     YES       -
             PASS         ********
ppp1         -            172.16.250.2   -          RIP2     NONE   YES       -
             PASS         NOT SET
----------------------------------------------------------------------------------
```

Table 14-43: Parameters in the output of the **show ip rip** command

Parameter	Meaning
Interface	Interface over which RIP packets are exchanged with the RIP neighbour. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen and the logical interface number.
Circuit/DLCI	Circuit name or DLCI number if this is an X.25 or Frame Relay interface.
IP Address	IP address of the RIP neighbour.
Send	Whether the type of RIP packets is none, RIP1, RIP2, or comp.
Receive	Whether to receive RIP1, RIP2, or both types of RIP packets, or none.
Dmd (demand)	Whether to use the demand RIP procedures.
Stc (static)	Whether static routes are exported.
NextHop	IP address destination of the RIP update of the next hop back to the configured device. Valid when using RIPv2.
Auth	Whether to use password, MD5, or no authentication with the RIP neighbour.
Password	Whether a password is set.

It is possible to transfer RIP derived routes to or from other IP routing protocols such as EGP or OSPF.

Examples

To show the RIP configuration for the eth0 interface, use the command:

```
sh ip rip int=eth0
```

Related Commands

add ip rip
delete ip rip
disable ip exportrip
enable ip exportrip
set ip rip
show ip
show ip counter

show ip rip counter

Syntax

```
SHow IP RIP COUnter[={Detail|Summary}]
    [INTerface=interface] [CIRCuit=miox-circuit]
    [DLCi=dlci] [IP=ipadd]
```

where:

- *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.
- *miox-circuit* is the name of a MIOX circuit defined for an X.25 interface 1 to 15 characters long. The name is not case-sensitive.
- *dlci* is the Data Link Connection Identifier (DLCI) of a Frame Relay DLC (circuit) from 0 to 1023.
- *ipadd* is an IP address in dotted decimal notation.

Description

This command displays counters for RIP (Figure 14-43 on page 14-204, Table 14-44 on page 14-204).

The **counter** parameter specifies whether to display summary or detailed information. If **detail** is specified, counters for each RIP neighbour and total counts for all RIP neighbours are displayed. Otherwise, the total counts for all RIP neighbours are displayed.

The **interface**, **circuit**, **dlci** and **ip** parameters restrict the display to RIP neighbours on specific interfaces, MIOX circuits, Frame Relay DLCs or with specific IP addresses. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- ATM (e.g. atm0.1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

Figure 14-43: Example output from the **show ip rip counter=detail** command

```
IP RIP Counters:
Interface: eth0
  Input:                      Output:
    inResponses ...... 2568     outResponses ..... 2567
    inTrigRequests ...... 0     outTrigRequests ..... 0
    inTrigResponses ..... 0     outTrigResponses .... 0
    inTrigAcks .......... 0     outTrigAcks ......... 0
    inDiscards .......... 0

Interface: fr0       Dlci: 9  IP Address: 172.16.249.34
  Input:                     Output:
    inResponses ..... 2567     outResponses ...... 2567
    inTrigRequests ....  0     outTrigRequests ...... 0
    inTrigResponses .... 0     outTrigResponses ..... 0
    inTrigAcks ......... 0     outTrigAcks .......... 0
    inDiscards ......... 0

IP RIP Counter Summary:
  Input:                     Output:
    inResponses ..... 5135     outResponses ...... 5134
    inTrigRequests ..... 0     outTrigRequests ...... 0
    inTrigResponses .... 0     outTrigResponses ..... 0
    inTrigAcks ......... 0     outTrigAcks .......... 0
    inDiscards ......... 0
```

Table 14-44: Parameters in the output of the **show ip rip counter** command

Parameter	Meaning
Interface	Interface of the RIP neighbour. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen (“-”) and the logical interface number.
Circuit/DLCI	Circuit name or DLCI number if this is an X.25 or Frame Relay interface.
IP Address	IP address of the RIP neighbour.
inResponses	Number of response packets received.
inTrigRequests	Number of triggered request packets received.
inTrigResponses	Number of triggered response packets received.
inTrigAcks	Number of triggered acknowledge packets received.
inDiscards	Number of packets discarded. Packets may be discarded due to authentication failure, packets received when receive is disabled, or mismatched sequence number of a triggered acknowledgement.
outResponses	Number of response packets transmitted.
outTrigRequests	Number of triggered request packets transmitted.
outTrigResponses	Number of triggered response packets transmitted.
outTrigAcks	Number of triggered acknowledge packets transmitted.

Related Commands show ip counter
show ip rip

show ip riptimer

Syntax `SHow IP RIPTimer`

Description This command displays the current settings of the global RIP timers (Figure 14-44 on page 14-205, Table 14-45 on page 14-205).

Figure 14-44: Example output from the **show ip riptimer** command

```
IP RIP timers
Timer name     Default     Current
------------------------------------
Update         30          5
Invalid        180         15
Holddown       120         60
Flush          300         75
------------------------------------
```

Table 14-45: Parameters in the output of the **show ip riptimer** command

Parameter	Meaning
Timer name	Timer name.
Default	Default in seconds for the timer.
Current	Current value in seconds for the timer.
Update	Time in seconds between RIP updates for all interfaces not using RIP on demand.
Invalid	Time in seconds after which the router deems a route to be invalid when no update has been received for the route.
Holddown	Time in seconds after a route has become invalid during which the router ignores updates for the route that would normally make the route valid again.
Flush	Time in seconds from the last update of a route until the route is flushed from the route table.

Examples To display the current settings of the global RIP timers, use the command:

```
sh ip ript
```

Related Commands set ip riptimer

show ip route

Syntax `SHow IP ROUte[=ipadd] [{GENeral|CAChe|COUnt|FULl}]`

where *ipadd* is an IP address in dotted decimal notation

Description This command displays information about the IP route table. If no optional parameters are specified, the contents of the route table is displayed (Figure 14-45 on page 14-207, Table 14-46 on page 14-207). If **route** is specified with an IP address that does not contain the wildcard character (“*”), the display lists all routes that can be used to reach the specified destination address, including the default route 0.0.0.0. If **route** is specified with an IP address that ends with the wildcard character (“*”), the display lists all routes beginning with the specified address. The wildcard character can be used to replace a complete number in the address, but not part of a number. For example, 192.168.*.* is valid and displays all routes in the route table that start with 192.168, but 192.168.12*.* is not valid.

This command shows the “best” routes—routes whose outgoing Layer 2 interface is up. The exceptions are interface and static routes, which are always displayed. If the outgoing Layer 2 interface for the route is down, a “#” character is displayed after the Layer 2 interface name. Note that interface routes are only propagated by RIP when their status at a physical level is up.

If **general** is specified, summary information is displayed (See Figure 14-46 on page 14-208, Table 14-47 on page 14-208).

If **cache** is specified, the contents of the route cache is displayed (See Figure 14-47 on page 14-208, Table 14-48 on page 14-208). If **route** is also specified with an IP address, routes in the route cache are displayed that were used to forward packets to the destination specified by the IP address.

If **count** is specified, summary information about the numbers of octets received and transmitted via each route is displayed (See Figure 14-48 on page 14-209, Table 14-49 on page 14-209).

If **full** is specified, all routes in the route table regardless of whether the outgoing Layer 2 interface is up or down are displayed (See Figure 14-49 on page 14-209 and Table 14-46 on page 14-207). Routes whose outgoing Layer 2 interface is down are marked with the “#” character after the layer two interface name.

Figure 14-45: Example output from the **show ip route** command

```
IP Routes
--------------------------------------------------------------------------------
Destination       Mask              NextHop             Interface            Age
DLCI/Circ.        Type     Policy   Protocol   Tag      Metrics       Preference
--------------------------------------------------------------------------------
0.0.0.0           0.0.0.0           202.36.163.21       eth0                   1
-                 remote   0        rip        -        5                    100
192.168.69.0      255.255.255.0     202.36.163.35       eth0                   0
-                 remote   0        rip        -        2                    100
192.168.201.0     255.255.255.0     202.36.163.21       eth0                   1
-                 remote   0        rip        -        5                    100
192.168.202.0     255.255.255.0     202.36.163.21       eth0                   1
-                 remote   0        rip        -        6                    100
192.168.203.0     255.255.255.0     202.36.163.21       eth0                   1
-                 remote   0        rip        -        5                    100
192.168.204.0     255.255.255.0     202.36.163.21       eth0                   1
-                 remote   0        rip        -        3                    100
192.168.206.0     255.255.255.0     202.36.163.21       eth0                   1
-                 remote   0        rip        -        4                    100
10.0.0.0          255.0.0.0         0.0.0.0             eth0                   4
-                 direct   0        interface  -        1                      0
11.0.1.0          255.255.255.0     10.42.0.22          eth0                   4
-                 direct   0        static     -        1                     60
11.0.2.0          255.255.255.0     10.42.0.22          eth0                   4
-                 direct   0        static     45535    1                     60
--------------------------------------------------------------------------------
```

Table 14-46: Parameters in the output of the **show ip route** and the **show ip route full** command

Parameter	Meaning
Destination	IP address of the destination network.
Mask	Subnet mask for the route.
NextHop	IP address of the next router on the route to the destination, or the *ifIndex* of an addressless PPP interfaces in dotted decimal notation.
Interface	Interface over which the destination network can be reached. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen ("-") and the logical interface number.
Age	Time in seconds that the route has been known.
Circuit/DLCI	Circuit name or DLCI number if this is an X.25 or Frame Relay interface.
Type	Whether the route is remote, direct, or another kind.
Policy	Policy number of this route.
Protocol	Whether the protocol that determines the route is interface (automatically created when the interface is created), static (manually created), RIP, EGP, or OSPF.
Tag	A number to identify the route. You can match against this number in a route map and only import the appropriately-tagged routes into BGP.
Metrics	Routing metric (cost) to reach the destination network.
Preference	Routing preference value. Routes with a high preference (low value) are used before routes with a low preference (high value).

Figure 14-46: Example output from the **show ip route general** command

```
IP Route General Information
----------------------------
Number of routes ............... 12
Cache size ..................... 1024
Source route byte counting ..... no
Route debugging ................ no
Multipath routing .............. yes
```

Table 14-47: Parameters in the output of the **show ip route general** command

Parameter	Meaning
Number of routes	Number of routes in the route table.
Cache size	Size of the route cache (the number of entries).
Source route byte counting	Whether source route byte counting is enabled.
Route debugging	Whether route debugging is enabled.
Multipath routing	Whether multipath route is enabled.

Figure 14-47: Example output from the **show ip route cache** command

```
IP Route Cache
--------------------------------------------------------------------------------
Destination      Route            Route mask       Nexthop          Interface
--------------------------------------------------------------------------------
202.36.163.4     202.36.163.0     255.255.255.192  0.0.0.0          eth0
202.36.163.5     202.36.163.0     255.255.255.192  0.0.0.0          eth0
202.36.163.6     202.36.163.0     255.255.255.192  0.0.0.0          eth0
202.36.163.11    202.36.163.0     255.255.255.192  0.0.0.0          eth0
202.36.163.21    202.36.163.0     255.255.255.192  0.0.0.0          eth0
202.36.163.31    202.36.163.0     255.255.255.192  0.0.0.0          eth0
202.36.163.36    202.36.163.0     255.255.255.192  0.0.0.0          eth0
202.36.163.51    202.36.163.0     255.255.255.192  0.0.0.0          eth0
202.36.163.61    202.36.163.0     255.255.255.192  0.0.0.0          eth0
202.36.163.5     202.36.163.0     255.255.255.192  0.0.0.0          eth0
            hits:        875          misses:          11
--------------------------------------------------------------------------------
```

Table 14-48: Parameters in the output of the **show ip route cache** command

Parameter	Meaning
Destination	Destination IP address.
Route	Route used to forward packets to the destination IP address.
Route mask	Network mask for the route.
NextHop	Next hop on the route.
Interface	Interface over which the destination network can be reached. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen (”-”) and the logical interface number.

Figure 14-48: Example output from the **show ip route count** command

```
Route Counters

 IP address        NextHop          Interface  Metric  Octets rcvd   Octets sent
--------------------------------------------------------------------------------
 192.168.1.0       202.36.163.21    eth1          1            0             0
 192.168.1.0       202.36.163.21    eth1          1            0             0
 192.168.1.64      202.36.163.21    eth1          1            0             0
 192.168.1.128     202.36.163.21    eth1          1            0             0
 192.168.1.192     202.36.163.21    eth1          1            0             0
 192.168.1.208     202.36.163.21    eth1          1            0             0
--------------------------------------------------------------------------------
```

Table 14-49: Parameters in the output of the **show ip route count** command

Parameter	**Meaning**
IP address	IP address of the destination to which packets were transmitted using this route.
NextHop	IP address of the next router on the route to the destination.
Interface	Interface over which the destination network can be reached. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen ("-") and the logical interface number.
Metric	Routing metric (cost) to reach the destination network.
Octets rcvd	Number of octets received through this route.
Octets sent	Number of octets transmitted through this route.

Figure 14-49: Example output from the **show ip route full** command

```
IP Routes
--------------------------------------------------------------------------------
Destination       Mask               NextHop             Interface           Age
DLCI/Circ.        Type      Policy   Protocol            Metrics      Preference
--------------------------------------------------------------------------------
192.168.1.0       255.255.255.0      0.0.0.0             eth0#               166
-                 direct    0        interface           1                     0
192.168.2.0       255.255.255.0      0.0.0.0             eth0                166
-                 direct    0        interface           1                     0
192.175.176.0     255.255.255.0      192.168.1.1         eth1#               137
-                 remote    0        rip                 16                  100
--------------------------------------------------------------------------------
```

Related Commands add ip route
delete ip route
set ip route

show ip route filter

Syntax `SHow IP ROUte FILter`

Description This command displays information about configured IP route filters (Figure 14-50 on page 14-210, Table 14-50 on page 14-210).

Figure 14-50: Example output from the **show ip route filter** command

```
IP Route Filters
--------------------------------------------------------------------------------
Ent.   IP Address        Mask              Nexthop           Policy      Matched
       Protocol          Direction         Interface         Action
--------------------------------------------------------------------------------
  1    0.0.0.0           0.0.0.0           Any               0                 0
       RIP               Both              -                 Include

       Request: 1                Passes: 1                Fails: 0
--------------------------------------------------------------------------------
```

Table 14-50: Parameters in the output of the **show ip route filter** command

Parameter	Meaning
Ent.	Filter number.
IP Address	IP address of the network to be filtered.
Mask	Network mask for the network address.
Nexthop	Next hop to which the filter applies.
Policy	Policy or type of service to which the filter applies.
Matched	Number of times this pattern has been matched.
Protocol	Routing protocol to which the filter applies.
Direction	Whether the direction to which the filter applies is receive, send, or both.
Interface	Interface to which the filter applies. When multihoming is enabled (two or more logical interfaces have been assigned to a single Layer 2 interface), all interface names include a hyphen ("-") and the logical interface number.
Action	Whether the action is to include or exclude when a route matches the pattern.

Related Commands

add ip route filter
delete ip route filter
set ip route filter

show ip route multicast

Syntax `SHow IP ROUte MULticast`

Description This command displays information about the IP multicast forwarding table (Figure 14-51 on page 14-211, Table 14-51 on page 14-211).

Figure 14-51: Example output from the **show ip route multicast** command

```
Source           Group          Prot   Uptime   InPort
   Outports
------------------------------------------------------
192.168.196.1    224.10.10.10   DVMRP      17   eth0
   bay1.eth0
202.36.163.197   224.10.10.10   DVMRP      16   bay1.eth0
   eth0
```

Table 14-51: Parameters in the output of the **show ip route multicast** command

Parameter	Meaning
Source	Host that sources multicast datagrams addressed to the specified groups.
Group	Class D IP address to which multicast datagrams are addressed. Note that a given Source may send packets to many different Multicast Groups.
Prot	Multicast routing protocol that contributes this forwarding entry.
InPort	Parent port for the (source, group) pair.
OutPorts	Child ports over which multicast datagrams for the (source, group) pair are forwarded.

Examples To display the forwarding information for multicast groups, use the command:

```
sh ip rou mul
```

Related Commands show dvmrp
show pim

show ip route preference

Syntax `SHow IP ROUte PREFerence`

Description This command displays information about the current IP route table preferences for each of the routing protocols. See Figure 14-52 on page 14-212 and Table 14-52 on page 14-212.

Figure 14-52: Example output from the **show ip route preference** command

```
IP Route Preference
------------------------------------------------------------
 Protocol                            Preference
 -----------------------------------------------------------
 RIP ............................... 100 (default)
 OSPF-INTRA ........................ 10 (default)
 OSPF-INTER ........................ 11 (default)
 OSPF-EXT1 ......................... 97
 OSPF-EXT2 ......................... 98
 OSPF-OTHER ........................ 99
 BGP-INT ........................... 170 (default)
 BGP-EXT ........................... 170 (default)
------------------------------------------------------------
```

Table 14-52: Parameters in the output of the **show ip route preference** command

Parameter	Meaning
Protocol	Available routing protocols.
Preference	Preference value for the routing protocol - a larger preference value indicates a less desirable routing protocol.

Related Commands **set ip route preference**

show ip route template

Syntax `SHow IP ROUte TEMPlate[=name]`

where *name* is a character string 1 to 31 characters long, and is not case-sensitive. Valid characters are any printable character. If *name* contains spaces, it must be in double quotes.

Description This command displays information about the specified or all IP route templates. If a template is not specified, summary information about all IP route templates is displayed (Figure 14-53 on page 14-213, Table 14-53 on page 14-213).

If a name is specified, details are displayed for it (Figure 14-54 on page 14-213, Table 14-54 on page 14-213).

Figure 14-53: Example output from the **show ip route template** command

```
Template                          Interface
------------------------------------------------------------
branch_office                     vlan1
home                              vlan1
------------------------------------------------------------
```

Table 14-53: Parameters in the output of the **show ip route template** command

Parameter	Meaning
Template	Name of the IP route template.
Interface	IP interface specified by the IP route template.

Figure 14-54: Example output from the **show ip route template** command for a specific template

```
IP route template ................... branch_office
Interface ........................... 0
Next hop ............................ 192.168.23.3
Rip metric .......................... DEFAULT (1)
Ospf metric ......................... DEFAULT (FFFFFFFF)
Policy .............................. DEFAULT (0)
Preference .......................... 90
Dlci ................................ 67
```

Table 14-54: Parameters in the output of the **show ip route template** command for a specific template

Parameter	Meaning
IP route template	Name of the IP route template.
Interface	IP interface specified by the IP route template.
Next hop	Next hop specified by the IP route template.
Rip metric	RIP metric specified by the IP route template.
Ospf metric	OSPF metric specified by the IP route template.
Policy	Policy specified by the IP route template.
Preference	Preference specified by the IP route template.
Dlci	DLCI specified by the IP route template.

Examples To display detailed information about the IP route template named "branch_office", use the command:

```
sh ip rou temp=branch_office
```

Related Commands

add ip route template
create ipsec policy
delete ip route template
set ip route template

show ip sa

Syntax

```
SHow IP SA INTerface=interface
```

where *interface* is an interface name formed by concatenating a Layer 2 interface type, an interface instance, and optionally a hyphen followed by a logical interface number from 0 to 15. If a logical interface is not specified, 0 is assumed.

Description

This command displays the list of security associations assigned to an IP interface.

The IP SA commands provide support for RFCs 1825, 1827, and 1829, which have been superseded by IP Security. See Chapter 45, IP Security (IPsec) and RFCs 2401–2412 for more information about IPsec.

The **interface** parameter specifies the name of the interface. The interface must already be assigned to the IP routing module. Valid interfaces are:

- eth (e.g. eth0, eth0-1)
- PPP (e.g. ppp0, ppp1-1)
- VLAN (e.g. vlan1, vlan1-1)
- FR (e.g. fr0, fr0-1)
- X.25 DTE (e.g. x25t0, x25t0-1)

To see a list of interfaces currently available, use the **show interface** command on page 7-66 of Chapter 7, Interfaces, or the **show ip interface** command on page 14-191.

Examples

To display the list of security associations assigned to interface ppp0, use the command:

```
sh ip sa int=ppp0
```

Related Commands

add ip sa
create sa
delete ip sa
set ip interface
show ip interface

show ip trusted

Syntax `SHow IP TRusted`

Description This command displays the contents of the trusted router table and the state of the enable flag (Figure 14-55 on page 14-215). The trusted router table ensures that the router's routing table is updated only by *trusted* sources of routing information. Other routers are not filtered but their routing information is not used until they are added to the table.

Figure 14-55: Example output from the **show ip trusted** command

```
   Host address
------------------
   172.16.8.33
------------------
```

Related Commands

add ip filter
add ip trusted
delete ip filter
delete ip trusted
set ip filter
show ip filter

show ip udp

Syntax `SHow IP UDP`

Description This command displays the state of current UDP sessions (Figure 14-56, Table 14-55 on page 14-215). UDP listens for SNMP packets and RIP. It also shows a connection when the TFTP download is initiated as part of loading new software.

Figure 14-56: Example output from the **show ip udp** command

```
   Local port     Local address     Remote port
------------------------------------------------
   00520          0.0.0.0           00000
   00161          0.0.0.0           00000
------------------------------------------------
```

Table 14-55: Parameters in the output of the **show ip udp** command

Parameter	Meaning
Local port	Port number for the UDP connection on this router. See Table 14-11 on page 14-70 for a list of commonly used, assigned UDP port numbers.
Local address	IP address for the UDP connection on this router.
Remote port	Port number for the UDP connection on the remote host. See Table 14-11 on page 14-70 for a list of commonly used, assigned UDP port numbers.

Related Commands show ip counter
show tcp

show ping

Syntax `SHow PING`

Description This command displays information about the ping configuration and the results of the current or previous **ping** command (Figure 14-57 on page 14-216, Table 14-56 on page 14-217).

Figure 14-57: Example output from the **show ping** command

```
Ping Information
----------------------------------------------------------
Defaults:
  Type .......................... IP
  Source ........................ 0.0.0.0
  Destination ................... 192.168.2.1
  Number of packets ............. 10
  Size of packets (bytes) ....... 24
  Timeout (seconds) ............. 1
  Delay (seconds) ............... 1
  Data pattern .................. Not set
  Type of service ............... 0
  Direct output to screen ....... Yes

Current:
  Type .......................... IP
  Source ........................ 0.0.0.0
  Destination ................... 192.168.2.1
  Number of packets ............. 10
  Size of packets (bytes) ....... 24
  Timeout (seconds) ............. 1
  Delay (seconds) ............... 1
  Data pattern .................. 0x00000000
  Type of service ............... 0
  Direct output to screen ....... Yes


Results:
  Ping in progress .............. No
  Packets sent .................. 10
  Packets received .............. 10
  Round trip time minimum (ms) .. 20
  Round trip time average (ms) .. 22
  Round trip time maximum (ms) .. 40
  Last message .................. Finished succesfully
----------------------------------------------------------
```

Table 14-56: Parameters in the output of the **show ping** command

Parameter	Meaning
Type	Whether the network protocol type is IP, IPX, OSI-CLNS, or AppleTalk.
Source	Source IP address used in the ping packet.
Destination	IP address or host name to ping.
Number of packets	Number of ping packets to send.
Size of packets (bytes)	Number of data pattern bytes to include in the packet.
Timeout (seconds)	Seconds to wait for a reply before sending the next packet.
Delay (seconds)	Seconds to wait before sending the next packet.
Data pattern	Data bytes to be used in the data portion of packets transmitted.
Type of service	Value of the TOS (Type Of Service) field in the IP header of IP ping packets transmitted.
Direct output to screen	Whether the output is sent to the terminal.
Ping in progress	Whether a ping is in progress.
Packets sent	Number of packets sent.
Packets received	Number of packets received.
Round trip time minimum (ms)	Quickest round trip time in milliseconds.
Round trip time average (ms)	Average round trip time in milliseconds.
Round trip time maximum (ms)	Slowest round trip time in milliseconds.
Last message	Last message from the **ping** command on page 14-129.

Examples To display the current ping configuration, use the command:

```
sh ping
```

Related Commands

ping
set ping
stop ping

show tcp

Syntax `SHow TCP[=tcb]`

where *tcb* is the index of a TCP connection in the TCP connection table

Description This command displays the state of current TCP connections. If a TCP connection is specified, details are displayed for it (Figure 14-58 on page 14-218, Table 14-57 on page 14-219).

If a TCP connection is not specified, the TCP portion of the MIB-II MIB is displayed along with summary information about all current TCP connections (Figure 14-59 on page 14-220, Table 14-58 on page 14-221). Index numbers 3 and 4 indicate locally sourced Telnet sessions. These sessions are from the asynchronous ports attached to the router. Index numbers 5 and 6 indicate *remotely* sourced Telnet sessions.

This command is useful to show if Telnet or other TCP sessions are active, and whether they are running over IPv4 or IPv6. Port 23 is typically reserved for Telnet. Other typical listen ports are reserved for X.25 over TCP and for permanent assignments. When a Telnet session is active, the IP address of the source and destination allows the particular session to be identified.

See Table 14-11 on page 14-70 for a list of commonly assigned TCP port numbers.

Figure 14-58: Example output from the **show tcp** command for a specific TCP connection

```
TCB: 05  Local: 192.168.35.45,00023  Remote: 192.168.35.61,01032
State: ESTAB  O/P State: IDLE
SND.UNA: 0047376265  SND.NXT: 0047376265  SND.WND: 04096
Last Seq: 0641204304  Last Ack: 0047376265
SendCon: 06022  DataCount: 0000000000
RCV.NXT: 0641204305  RCV.WND: 00000
Round Trip Time
SendSrt: 00218  Deviation: 00013  SendReXmit: 00033
Timers:
Event       Time (cs)
No events in timer queue
Fragment list:
Sequence     Length    End sequence
No fragments in fragment list
```

Table 14-57: Parameters in the output of the **show tcp** command for a specific TCP connection

Parameter	**Meaning**	
TCB	Index into the TCP connection table for this connection.	
Local	Local IP address and port for the connection. See Table 14-11 on page 14-70 for a list of commonly assigned TCP port numbers.	
Remote	Remote IP address and port for the connection. See Table 14-11 on page 14-70 for a list of commonly assigned TCP port numbers.	
State	State of the connection (Table 14-59 on page 14-222).	
O/P State	Output queue state:	
	IDLE	
	PERST	Remote host has closed its receive window and router is transmitting data one character at a time to aid the process of re-opening the window
	TRANS	There is data to transmit
	RETRN	The router is retransmitting data
SND.UNA	Sequence number of the last unacknowledged octet transmitted over the connection.	
SND.NXT	Sequence number of the next octet to be transmitted over the connection.	
SND.WND	Transmit window for the connection.	
Last Seq	Packet received from the connection.	
Last Ack	Last acknowledgement received from the connection.	
SendCon	Internal congestion parameter.	
DataCount	Number of data octets transmitted over this connection.	
RCV.NXT	Next octet expected from the connection.	
RCV.WND	Receive window for the connection.	
SendSrt, Deviation, SendReXmit	Round trip time parameters used to implement Van Jacobson's retransmit time algorithm.	
Event	An event on the timer queue:	
	None	No data
	Send	Transmit data
	Persist	Transmit data one character at a time if in persist state
	Transmit	Retransmit data
	Delete	Clear TCP connection
Time (cs)	Time to this event in centiseconds.	
Sequence	First sequence number of a fragment waiting for defragmentation.	
Length	Length of the fragment.	
End sequence	Last sequence number of the fragment.	

Figure 14-59: Example output from the **show tcp** command

```
 TCP MIB parameters, counters and connections
------------------------------------------------------------
 RTO Algorithm:           vanj
 RTO Min (ms):      0000000500   RTO Max (ms):    0000020000

 Maximum connections:    00040

 Active Opens:           00004   Passive Opens:        00005
 Attempt Fails:          00000   Established Resets:   00000
 Current Established:    00004

 In Segs:           0000000070   In Segs Error:   0000000000
 Out Segs:          0000000104   Out Segs Retran: 0000000000
 Out Segs With RST: 0000000000

 Connection Table:
 Index   Proto  State
         Local port and address
         Remote port and address
 ----------------------------------------------------------
     0   IPv4   listen
         00023  0.0.0.0
         00000  0.0.0.0
----------------------------------------------------------
     1   IPv6   listen
         00023  ::
         00000  ::
------------------------------------------------------------
     2   IPv4   listen
         00080  0.0.0.0
         00000  0.0.0.0
----------------------------------------------------------
     3   IPv4   established
         00127  172.16.253.2
         00023  172.16.8.5
----------------------------------------------------------
     4   IPv4   established
         00133  172.16.253.2
         00023  172.16.8.5
----------------------------------------------------------
     5   IPv4   established
         00023  172.16.40.254
         00002  172.16.248.51
----------------------------------------------------------
     6   IPv4   established
         00023  172.16.40.254
         02123  172.16.9.190
----------------------------------------------------------
     7   IPv6   established
         00023  3001:0001::0022
         01046  3001:0001::0001
```

Table 14-58: Parameters in the output of the **show tcp** command

Parameter	**Meaning**
RTO Algorithm	Retransmit time algorithm.
RTO Min (ms), RTO Max (ms)	Retransmit time algorithm parameters (milliseconds)
Maximum connections	Maximum number of TCP connections allowed.
Active Opens	Number of active TCP opens. Active opens initiate connections.
Passive Opens	Number of TCP passive opens. Passive opens are issued to wait for a connection from another host.
Attempt Fails	Number of failed connection attempts.
Established Resets	Number of connections established but have been reset.
Current Established	Number of current connections.
In Segs	Number of segments received.
In Segs Error	Number of segments received with an error.
Out Segs	Number of segments transmitted.
Out Segs Retran	Number of segments retransmitted.
Out Segs With RST	Number of segments transmitted with the RST bit set.
Index	Entry number in the table.
Proto	Protocol type of the session; IPv4 or IPv6.
State	State of the session (Table 14-59 on page 14-222). These are the names of the various states in the TCP state diagram. For more detailed information, refer to the RFC or a text on TCP/IP.
Local port and address	The router’s TCP port number and IP address. See Table 14-11 on page 14-70 for a list of commonly used, assigned UDP port numbers.
Remote Port and address	TCP port number and IP address of the remote host. See Table 14-11 on page 14-70 for a list of commonly assigned UDP port numbers.

Table 14-59: TCP states

State	Meaning
Closed	The starting state and should not be present at any time since the server module should immediately go into the listen state.
Listen	This is a passive open and is entered when the server module is waiting for external connections to be made.
Synsent	The server enters this state when a connection is being initiated from a local session and also when a remotely initiated session is being set up just prior to entering the established state.
Synreceived	This state is entered when a SYN packet is received indicating that a remote system is attempting to establish a session.
Established	This state indicates that a connection has been made and is currently active. Data packets can now flow in both directions.
Finwait1	This state indicates the first step of a locally initiated termination of a session. The closewait state indicates a remote station is initiating the termination.
Finwait2	This state is also part of the local termination process and is required to ensure that no data in transit in lost.
Closewait	This state is entered when the remote entity has sent a FIN packet to terminate this link. The server entity sends an ACK packet.
Lastack	The ACK packet from above causes the remote system to send a close packet and the server enters this state and sends a FIN packet thereby terminating this link.
Closing	This state is entered when the established local session has initiated a termination (gone to FINWAIT1) and received a FIN packet from the remote entity indicating that it can now terminate also. This is an alternate path to FINWAIT2.
Timewait	This state may be entered as part of the termination process while waiting for a remote entity to respond to the final ACK packet. The session is then closed.

Related Commands

show ip counter
show ip udp

show trace

Syntax `SHow TRAce`

Description This command displays information about the current trace route configuration and the result of the current or previous trace route operation (Figure 14-60 on page 14-223, Table 14-60 on page 14-224).

Figure 14-60: Example output from the **show trace** command

```
Trace information
------------------------------------------------------------
Defaults:
  Destination ................... 121.23.5.4
  Source ........................ 202.36.163.31
  Number of packets per hop ..... 3
  Timeout (seconds) ............. 1
  Type of service ............... 8
  Port .......................... 33434
  Minimum time to live .......... 1
  Maximum time to live .......... 20
  Addresses only output ......... Yes
  Direct output to screen ....... Yes

Current:
  Destination ................... 206.123.21.3
  Source ........................ 202.36.163.31
  Number of packets per hop ..... 3
  Timeout (seconds) ............. 1
  Type of service ............... 8
  Port .......................... 33434
  Minimum time to live .......... 1
  Maximum time to live .......... 12
  Addresses only output ......... Yes
  Direct output to screen ....... Yes

Results:
  Trace route in progress ....... No

 1. 202.36.163.21       20     20     20 (ms)
 2. 202.49.72.62         0      0      0 (ms)
 3. 203.97.191.65        0      0      0 (ms)
 4. 203.97.191.22       80     93    100 (ms)
 5. 140.200.128.2       40     46     60 (ms)
 6. 131.119.17.205     460    473    480 (ms)
 7. 131.119.0.129      540    553    560 (ms)
 8. 4.0.1.90           800    800    800 (ms)
 9. 4.0.1.14           440    440    440 (ms)
10. 198.32.136.39      480    480    480 (ms)
11. 140.223.9.21       520    520    520 (ms)
12. 140.223.9.18       560    560    560 (ms)

  Last message .................. Target unreached
------------------------------------------------------------
```

Table 14-60: Parameters in the output of the **show trace** command

Parameter	**Meaning**
Destination	Destination IP address or host name.
Source	Source IP address to use in the packets transmitted.
Number of packets per hop	Number of packets to transmit to each hop on the route.
Timeout	Seconds to wait for a reply before sending the next packet.
Type of service	Value of the TOS field in the IP header of packets transmitted.
Port	Destination UDP port number.
Minimum time to live	Minimum TTL (Time To Live) used to skip some hops at the start of the route.
Maximum time to live	Maximum hops to which packets are transmitted.
Addresses only output	Whether address-to-name translation is performed for the output.
Direct output to screen	Whether output is sent to the terminal.
Trace route in progress	Whether a trace route is in progress.
1-*n*	Hop number, IP address, and the maximum, minimum and average round trip time in milliseconds to each hop on the route.
Last message	Last message from the **ping** command on page 14-129.

Examples To show the current trace route configuration, use the command:

```
sh tra
```

Related Commands **set trace**
stop trace
trace

stop ping

Syntax `STop PING`

Description This command stops a ping in progress.

Examples To stop a ping in progress, use the command:

```
st ping
```

Related Commands **ping**
set ping
show ping

stop trace

Syntax

```
STop TRAce
```

Description This command stops a trace route in progress.

Examples To stop a trace route that is in progress, use the command:

```
st tra
```

Related Commands

show trace
stop trace
trace

trace

Syntax

```
TRAce [[IPaddress=]ipadd] [ADDROnly={No|OFf|ON|Yes}]
    [MAXTtl=number] [MINTtl=number] [NUMber=number]
    [POrt=1..65535] [SCReenoutput={No|OFf|ON|Yes}]
    [SOurce=ipadd] [TIMEOut=number] [TOS=0..255]
```

where:

- *ipadd* is an IPv4 address in dotted decimal notation, a valid IPv6 address, or a host name from the host name table.
- *number* is a decimal number.

Description This command performs a trace route. The parameters in this command override defaults set with the **set trace** command on page 14-165.

This command can be used to view the path to a node running IPv6.

The **ipaddress** parameter specifies the destination IP address; this command traces the route to this IP address. If you do not specify an IP address here or in the **set trace** command on page 14-165 then a trace is not performed and an error message is displayed.

The **addronly** parameter specifies whether trace output is presented as IP addresses only, as opposed to IP addresses and their DNS name equivalent. If **on**, output is presented as IP addresses. The default is **on**.

The **maxttl** parameter specifies the maximum value for the TTL (Time To Live) field in the IP packet, and is used to limit the trace route to a maximum number of hops. If this parameter is not specified, the default is used.

The **minttl** parameter specifies the initial value of the TTL (Time To Live) field in the IP packet, and can be used to skip hops at the start of the route. If this parameter is not specified, the default is used.

The **number** parameter specifies the number of packets to send to each hop. If this parameter is not specified, the default is used. A maximum of 100 packets may be transmitted.

The **port** parameter specifies the UDP destination port number for the packets being transmitted. It also detects whether there is an IP device listening on the specified port. If a device is listening, the ICMP “unreachable” message is not returned.

The **screenoutput** parameter specifies whether the output is sent to the terminal. If this parameter is not specified, the default is used.

The **source** parameter specifies the IP address to use as a source address in the packets. If this parameter is not set, the default IPv4 address of the interface is set as the source address. Because this IPv4 address causes a conflict when an IPv6 address is specified in the **ipaddress** parameter, this parameter is required when tracing a route to an IPv6 address.

The **timeout** parameter specifies how long to wait for a response before sending packets to the next hop. If this parameter is not specified, the default is used. If ICMP “unreachable” messages are received within the timeout period, packets are transmitted to the next hop immediately.

The **tos** parameter specifies the value of the TOS (Type Of Service) field in the IP header of the packets being transmitted. If this parameter is not specified, the default is used.

Related Commands

set trace
show trace
stop trace