

Configuring IP

2

This chapter describes how to configure Internet Protocol (IP) routing on your ERX system.

Topic	Page
Overview	2-2
References	2-4
IP Features	2-5
IP Addressing	2-6
Before You Configure IP	2-11
Creating a Profile	2-12
Address Resolution Protocol (ARP)	2-14
Broadcast Addressing	2-18
Fragmentation	2-20
IP Routing	2-20
Shared IP Interfaces	2-31
Subscriber Interfaces	2-35
Internet Control Message Protocol	2-42
Reachability Commands	2-44
Response Time Reporter	2-46
Monitoring IP	2-59

Overview

TCP/IP is a suite of data communications protocols. Two of the more important protocols in the suite are the Transmission Control Protocol (TCP) and the Internet Protocol (IP).

IP provides the basic packet delivery service for all TCP/IP networks. IP is a *connectionless* protocol, which means that it does not exchange control information to establish an end-to-end connection before transmitting data. A *connection-oriented* protocol exchanges control information with the remote computer to verify that it is ready to receive data before sending it.

IP relies on protocols in other layers to establish the connection if connection-oriented services are required and to provide error detection and error recovery. IP is sometimes called an unreliable protocol, because it contains no error detection or recovery code.

IP Packets

A *packet* is a block of data that carries with it the information necessary to deliver it to a destination address. A *packet-switching network* uses the addressing information in the packets to switch packets from one physical network to another, moving them toward their final destination. Each packet travels the network independently of any other packet. The *datagram* is the packet format defined by IP.

IP Functions

Some of the functions IP performs include:

- Moving data between the network access layer and the host-to-host transport layer
- Routing datagrams to remote hosts
- Fragmenting and reassembling datagrams

Moving Data Between Layers

When IP receives a datagram that is addressed to the local host, it must pass the data portion of the datagram to the correct host-to-host transport layer protocol. IP uses the *protocol number* in the datagram header to select the transport layer protocol. Each host-to-host transport layer protocol has a unique protocol number that identifies it to IP.

Routing Datagrams to Remote Hosts

Internet gateways are commonly referred to as IP routers because they use IP to route packets between networks. In traditional TCP/IP terms, there are only two types of network devices: gateways and hosts. Gateways forward packets between networks, and hosts do not. However, if a host is connected to more than one network (called a *multihomed host*), it can forward packets between the networks. When a multihomed host forwards packets, it acts like any other gateway and is considered to be a gateway.

Fragmenting and Reassembling Datagrams

As a datagram is routed through different networks, it may be necessary for the IP module in a gateway to divide the datagram into smaller pieces. A datagram received from one network may be too large to be transmitted in a single packet on a different network. This condition occurs only when a gateway interconnects dissimilar physical networks.

Each type of network has a maximum transmission unit (MTU) that determines the largest packet it can transfer. If the datagram received from one network is longer than the other network's MTU, it is necessary to divide the datagram into smaller fragments for transmission in a process called *fragmentation*. See *Fragmentation* later in this chapter.

IP Layering

TCP/IP is organized into four conceptual layers (as shown in Figure 2-1).

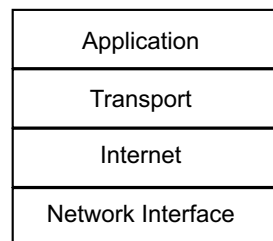


Figure 2-1 TCP/IP conceptual layers

Network Interface Layer

The network interface layer is the lowest level of the TCP/IP protocol stack. It is responsible for transmitting datagrams over the physical medium to their final destinations.

Internet Layer

The internet layer is the second level of the TCP/IP protocol stack. It provides host-to-host communication. In this layer, packets are encapsulated into datagrams, routing algorithms are run, and the datagram is passed to the network interface layer for transmission on the attached network.

Transport Layer

The transport layer is the third level of the TCP/IP protocol stack. It is responsible for providing communication between applications residing in different hosts. By placing identifying information in the data datagram (such as socket information), the transport layer enables process-to-process communication.

The transport layer provides either a reliable transport service (TCP) or unreliable service (User Data Protocol). In a reliable delivery service, the destination station acknowledges the receipt of a datagram.

Application Layer

The application layer is the fourth and highest level of the TCP/IP protocol stack. Some applications that run in this layer are:

- Telnet
- File Transfer Protocol (FTP)
- Simple Mail Transfer Protocol (SMTP)
- Simple Network Management Protocol (SNMP)
- Domain Name System (DNS)

References

For more information about IP, consult the following resources:

- RFC 768 – User Datagram Protocol (August 1980)
- RFC 791 – Internet Protocol DARPA Internet Program Protocol Specification (September 1981)
- RFC 792 – Internet Control Message Protocol (September 1981)
- RFC 793 – Transmission Control Protocol (September 1981)
- RFC 854 – Telnet Protocol Specification (May 1983)
- RFC 919 – Broadcasting Internet Datagrams (October 1984)

- RFC 922 – Broadcasting Internet Datagrams in the Presence of Subnets (October 1984)
- RFC 950 – Internet Standard Subnetting Procedure (August 1985)
- RFC 1112 – Host Extensions for IP Multicasting (August 1989)
- RFC 1122 – Requirements for Internet Hosts—Communication Layers (October 1989)
- RFC 1812 – Requirements for IP Version 4 Routers (June 1995)
- *ERX Release Notes, Appendix A, System Maximums* – Refer to the Release Notes corresponding to your software release for information on maximum values.

IP Features

The ERX system supports the following IP features:

- Internet Control Message Protocol (ICMP)
- Traceroute
- User Datagram Protocol (UDP)
- Transmission Control Protocol (TCP)
- Classless interdomain routing (CIDR)
- Maximum transmission unit (MTU)
- Support for simultaneous multiple logical IP stacks on the same system.
- Flexible IP address assignment to support any portion of a physical interface (for example, a channel or circuit), exactly one physical interface, or multilink PPP interfaces
- Packet segmentation and reassembly
- Loose source routing to specify the IP route
- Strict source routing to specify the IP route for each hop
- Record route to track the route taken
- Internet timestamp
- Broadcast addressing, both limited broadcast and directed broadcast

- Support for 32,000 discrete, simultaneous IP interfaces per system to support thousands of logical connections
- Capability of detecting and reporting changes in the up or down state of any IP interface

IP Addressing

This section provides an overview of IP addressing in general and includes a discussion of CIDR, which your system fully supports.

Physical and Logical Addresses

Physical node addresses are used at the network access layer to identify physical devices in a network. For example, each Ethernet controller comes from the manufacturer with a physical address, called a MAC address.

IP implements a system of logical host addresses called IP addresses. The IP addresses are used by the internetwork and higher layers to identify devices and to perform internetwork routing. The Address Resolution Protocol (ARP) enables IP to identify the physical (MAC) address that matches a given IP address. You can use ARP only on Ethernet and bridged IP 1483 interfaces.

IP is used by all protocols in the layers above and below it to deliver data. This means that all TCP/IP data flows through IP when it is sent and received, regardless of its final destination.

Internet Addresses

Internet addressing uses a 32-bit address field. The bits in this address field are numbered 0 to 31. The 32-bit address field consists of two parts: a network number and a host number whose boundaries are defined based on the class of IP address. Hosts attached to the same network must share a common prefix designating their network number.

Four types of IP classes lend themselves to different network configurations, depending on the desired ratio of networks to hosts. Figure 2-2 shows the format of IP address classes.

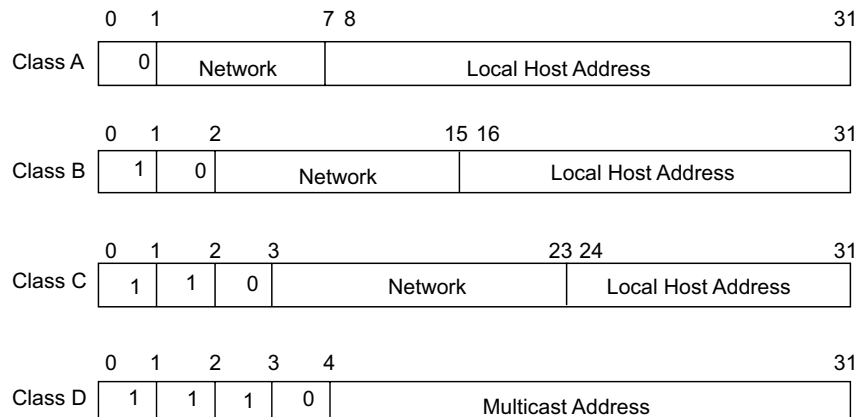


Figure 2-2 IP address classes

- Class A – The leading bit is set to 0, a 7-bit number, and a 24-bit local host address. Up to 125 class A networks can be defined, with up to 16,777,214 hosts per network.
- Class B – The two highest-order bits are set to 1 and 0, a 14-bit network number, and a 16-bit local host address. Up to 16,382 class B networks can be defined, with up to 65,534 hosts per network.
- Class C – The three leading bits are set to 1, 1, and 0, a 21-bit network number, and an 8-bit local host address. Up to 2,097,152 class C networks can be defined, with up to 254 hosts per network.
- Class D – The four highest-order bits are set to 1, 1, 1, and 0. Class D is used as a multicast address.

Subnet Addressing

A subnet is a subset of a class A, B, or C network. Subnets cannot be used with class D (multicast) addresses.

A network mask is used to separate the network information from the host information on an IP address. Figure 2-3 shows the network mask 255.0.0.0 applied to network 10.0.0.0. The mask in binary notation is a series of 1s followed by a series of contiguous 0s. The 1s represent the network number; the 0s represent the host number. The example address splits the IP address 10.0.0.1 into a network portion of 10 and a host portion of 0.0.1.



Note: The system supports a 31-bit mask on point-to-point links. This means that the IP address 1.2.3.4 255.255.255.254 is valid. A point-to-point link in which only one end supports the use of 31-bit prefixes may not operate correctly.

	Decimal			Binary		
IP Address	40.	0.0.1	00101000	00000000	00000000	00000001
Mask	255.	0.0.0	11111111	00000000	00000000	00000000
			Network portion			Host portion

Figure 2-3 Basic network masking

Classes A, B, and C have the following *natural masks*, which define the network and host portions of each class:

- Class A natural mask 255.0.0.
- Class B natural mask 255.255.0.0
- Class C natural mask 255.255.255.0

The use of masks can divide networks into subnetworks by extending the network portion of the address into the host portion. Subnetting increases the number of subnetworks and reduces the number of hosts.

For example, a network of the form 10.0.0.0 accommodates one physical segment with about 16 million hosts on it. Figure 2-4 shows how the mask 255.255.0.0 is applied to network 10.0.0.0. The mask divides the IP address 10.0.0.1 into a network portion of 10, a subnet portion of 0, and a host portion of 0.1. The mask has borrowed a portion of the host space and has applied it to the network space. The network space of the class 10 has increased from a single network 10.0.0.0 to 256 subnetworks, ranging from 10.0.0.0 to 10.255.0.0. This process decreases the number of hosts per subnet from 16,777,216 to 65,536.

	Decimal			Binary			
IP Address	40.	0	.0.1	00101000	00000000	00000000	00000001
Mask	255.	255	.0.0	11111111	00000000	00000000	00000000
				Network portion	Subnet portion		Host portion

Figure 2-4 Subnetting

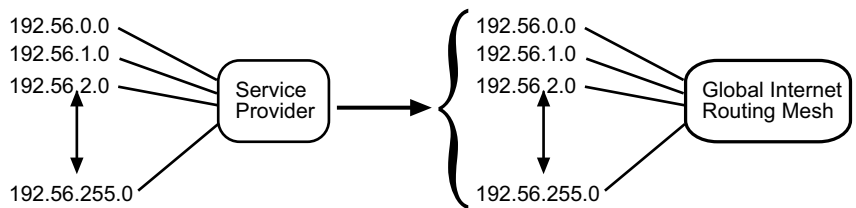
Classless Addressing with CIDR

Classless interdomain routing (CIDR) is a system of addressing that improves the scaling factor of routing in the Internet. CIDR does not use an implicit mask based on the class of network. In CIDR, an IP network is represented by a prefix, which is an IP address and an indication of the leftmost contiguous significant bits within this address.

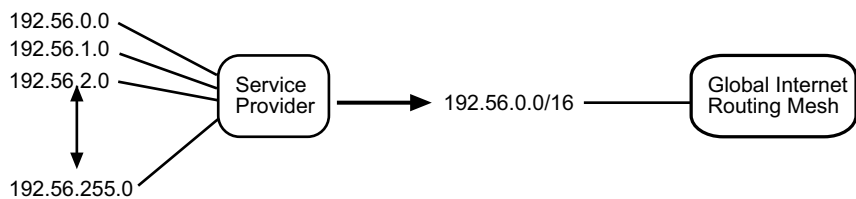
For example, without CIDR, the class C network address 192.56.0.0 would be an illegal address. With CIDR, the address becomes valid with the notation: 192.56.0.0/16. The /16 indicates that 16 bits of mask are being used (counting from the far left). This would be similar to an address 198.32.0.0. with a mask of 255.255.0.0.

A network is called a *supernet* when the prefix boundary contains fewer bits than the network’s natural mask. For example, a class C network 192.56.10.0 has a natural mask of 255.255.255.0. The representation 192.56.0.0/16 has a shorter mask than the natural mask (16 is less than 24), so it is a supernet.

Figure 2-5 shows how CIDR can reduce the number of entries globally in Internet routing tables. A service provider has a group of customers with class C addresses that begin with 192.56. Despite this relationship, the service provider announces each of the networks individually into the global Internet routing mesh.



Interdomain routing without CIDR



Interdomain routing with CIDR

Figure 2-5 Routing with and without CIDR

Adding and Deleting Addresses

This section provides information on adding or deleting IP addresses.

Multinetting is adding more than one IP address to an IP interface—that is, a primary address and one or more secondary addresses.

To make an interface unnumbered, see *Setting Up an Unnumbered Interface* later in this chapter.

Adding a Primary Address

- The primary address must be the first address added to the interface.
- Adding a new primary address overwrites the existing primary address.
- You can change a secondary address to be the primary address on an interface only via SNMP.
- An unnumbered address is always the primary address; adding an unnumbered address, therefore, overwrites any other numbered address.

Deleting a Primary Address

- You must always remove the primary address from an interface last.
- You cannot delete the primary address if the interface still has assigned secondary addresses.

Adding a Secondary (Multinet) Address

- You cannot add a secondary address until you add the primary address.
- You cannot add a secondary address to bridged Ethernet interfaces.
- You cannot change a primary address to a secondary address.
- An interface can have multiple secondary addresses.

Deleting a Secondary Address

- You must delete secondary addresses before deleting the primary address.

ip address Command

Use the following command to add addresses to or delete addresses from an interface:

ip address

- Use to add a primary address or to add secondary addresses to an interface.
- To add multiple addresses to a single IP interface, use the **secondary** keyword. (Remember, if you add an address using the **ip address** command and do not include the **secondary** keyword, the new address becomes the primary address.)
- Example – this example adds a primary address (192.168.2.77) and two secondary addresses (172.31.7.22 and 10.8.7.22); the Fast Ethernet interface now has addresses in three networks

```
host1(config)#interface fastEthernet 0/0
host1(config-if)#ip address 192.168.2.77 255.255.255.0
host1(config-if)#ip address 172.31.7.22 255.255.255.0
                secondary
host1(config-if)#ip address 10.8.7.22 255.255.255.0
                secondary
```



Note: You can use this command in Interface Configuration, Subinterface Configuration, or Profile Configuration mode.

- Use the **no** version to remove an IP address. If you remove a primary IP address, IP processing is disabled on the interface.

Before You Configure IP

Before you configure IP, you should have created lower-layer interfaces over which IP traffic flows.

For example, to configure an ATM interface:

```
host1(config)#interface atm 1/0
host1(config-if)#atm sonet stm-1
host1(config-if)#no loopback
host1(config-if)#atm clock internal chassis
host1(config-if)#interface atm 1/0.10
host1(config-if)#atm pvc 10 0 20 aal5snap
```

Refer to the appropriate chapters for information on configuring a specific type of interface.



Note: If you choose to configure VRRP, it is recommended that you complete all IP address configurations before you configure VRRP. See Chapter 9, Configuring VRRP, for additional information.

Creating a Profile

You can configure an IP interface dynamically by creating a profile. A profile is a set of characteristics that acts as a pattern that can be dynamically assigned to an IP interface. You can manage a large number of IP interfaces efficiently by creating a profile with a specific set of characteristics. In addition, you can create a profile to assign an IP interface to a virtual router.

A profile can contain one or more of the following characteristics:

- `access-route` – Enables the creation of host access routes on an interface
- `address` – Configures an IP address on an interface
- `directed-broadcast` – Enables directed broadcast forwarding
- `mtu` – Configures the mtu for a network
- `redirects` – Enables transmission of ICMP redirect messages
- `unnumbered` – Configures IP on this interface without a specific address
- `virtual-router` – Specifies a virtual router to which interfaces created by this profile will be attached

Use the **profile** command from Global Configuration mode to create or edit a profile. See *ERX Physical and Link Layers Configuration Guide, Chapter 21, Configuring Dynamic Interfaces* for information on creating profiles and on other characteristics that can be applied to the profile.

```
host1(config)#profile acton
host1(config-profile)#ip virtual-router warf
host1(config-profile)#ip unnumbered atm 3/0
```

ip access-routes

- Use to enable an access route in a profile.
- Example

```
host1(config)#profile foo
host1(config-profile)#ip access-routes
```

- Use the **no** version to remove the access route.

ip address

- Use to assign an IP address to a profile.
- You must first specify the layer 2 encapsulation before you can set the IP address for serial interfaces.
- Use the **no** version to remove the IP address assigned to the profile.

ip directed-broadcast

- Use to enable a directed broadcast address in a profile.
- Use the **no** version to remove the directed broadcast address from the profile.

ip mtu

- Use to assign the MTU size sent on an IP interface.
- Use the **no** version to remove the assignment from the profile.

ip redirects

- Use to enable the sending of redirect messages if the software is forced to resend a packet through the same interface on which it was received.
- Use the **no** version to remove the assignment from the profile.

ip unnumbered

- Use to specify the numbered interface with which dynamic unnumbered interfaces created with the profile are associated.
- You can specify an unnumbered interface using RADIUS instead of using the **ip unnumbered** command in a profile.
- Example

```
host1(config-profile)#ip unnumbered fastEthernet 0/0
```
- Use the **no** version to remove the assignment from the profile.

ip virtual-router

- Use to assign a virtual router to a profile.
- You can configure a virtual router using RADIUS instead of adding one to the profile by using the **ip virtual-router** command.
- Use the **no** version to remove the virtual router assignment.

profile

- Use to create a profile.
- You specify a profile name with up to 80 characters.
- Example

```
host1(config)#profile foo
```
- Use the **no** version to remove a profile.

Assigning a Profile

To assign a profile to an interface, use the **profile** command from Interface mode.

profile

- Use to assign a profile to a PPP interface. The profile configuration is used to dynamically create an upper IP interface.

- Example

```
host1(config-if)#interface serial 2/1
host1(config-if)#encapsulation ppp
host1(config-if)#profile acton
```

- Use the **no** version to remove the assignment from the interface.

Address Resolution Protocol (ARP)

Sending IP packets on a multiaccess network requires mapping from an IP address to a MAC address (the physical or hardware address).

In an Ethernet environment, ARP is used to map a MAC address to an IP address. ARP dynamically binds the IP address (the logical address) to the correct MAC address. Before IP unicast packets can be sent, ARP discovers the MAC address used by the Ethernet interface where the IP address is configured.

Hosts that use ARP maintain a cache of discovered Internet-to-Ethernet address mappings to minimize the number of ARP broadcast messages. To keep the cache from growing too large, an entry is removed if it is not used within a certain period of time. Before sending a packet, the host looks in its cache for Internet-to-Ethernet address mapping. If the mapping is not found, the host sends an ARP request.



Note: For information on MAC address validation, see the section *MAC Address Validation*.

How ARP Works

The way ARP works can best be explained in a simple example. As shown in Figure 2-6, host 1 wants to send an IP packet to host 2 on a different subnet. To complete this transmission, host 1 needs the MAC address of router 1, to be used as the forwarding gateway.

A typical scenario is:

- 1 Host 1 broadcasts an ARP request to all devices on subnet 1, composed by a query with the IP address of router 1. The IP address of router 1 is needed because host 2 is on a different subnet.
- 2 All devices on subnet 1 compare their IP address with the enclosed IP address sent by host 1.

- Having the matching IP address, router 1 sends an ARP response, which includes its MAC address, to host 1.

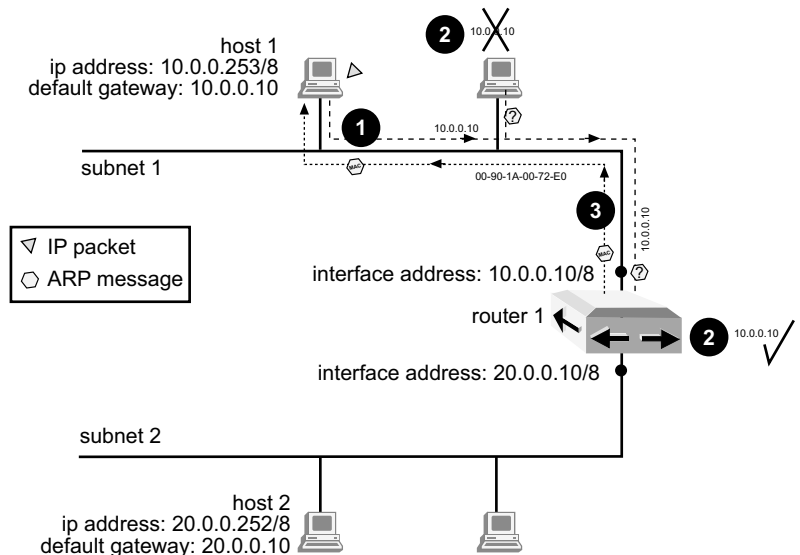


Figure 2-6 How ARP works, steps 1, 2, and 3

- Host 1 proceeds with its intended transmission of IP packet to layer 3 DA (host 2) using router 1's MAC address.
- Router 1 forwards IP packet to host 2. Router 1 may send an ARP request to identify the MAC of host 2.

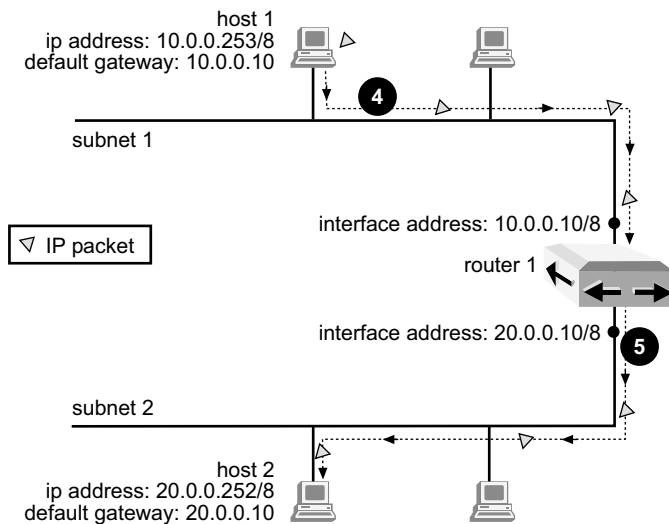


Figure 2-7 How ARP works, steps 4 and 5

ARP forces all receiving hosts to compare their IP addresses with the IP address of the ARP request. Keeping the above example in mind, if host 1 sent another IP packet to host 2, host 1 would first check its ARP table for the router 1's MAC address.

If the default router/gateway becomes unavailable, then all the routing/packet forwarding to remote destinations ceases. Usually, it requires manual intervention to restore connectivity, even though there may be alternative paths available. Alternatively, Virtual Router Redundancy Protocol (VRRP) may be used to prevent loss of connectivity. See *Chapter 9, Configuring VRRP*.

arp

- Use to add a static (permanent) entry in the ARP cache.
- You specify one of the following:
 - › *ipAddress* and *macAddress* of the interface
 - › *ipAddress*, *interfaceType* and *interfaceSpecifier*, and an optional MAC address
- You can issue this command only for an Ethernet or bridged IP 1483 interface.
- Example

```
host1(config)#arp 192.56.20.1 0090.1a00.0170
```
- Use the **no** version to remove an entry from the ARP cache.

arp timeout

- Use to specify how long an entry remains in the ARP cache.
- On the FE-2 module, you can set the ARP timeout only on bridged IP 1483 and Fast Ethernet interfaces. You cannot set the timeout on the SRP module.
- The default value is 21,600 seconds (6 hours). Use the **show config** command to display the current value.
- If you specify a timeout of 0 seconds, entries are never cleared from the ARP cache.
- Example

```
host1(config-if)#arp timeout 8000
```
- Use the **no** version to restore the default value.

clear arp

- Use to clear dynamic entries from the ARP cache.
- To clear a particular entry, specify all of the following:
 - › *ipAddress* – IP address in four-part dotted-decimal format corresponding to the local data link address
 - › *interfaceType* – encapsulation type (for example, ATM)

- › *interfaceSpecifier* – number of the interface specified in the appropriate format. Refer to the specific type of interface for details on the format required.
- To clear all dynamic ARP entries, specify an asterisk (*).
- Example


```
host1#clear arp
```
- There is no **no** version.

ip proxy-arp

- Use the **ip proxy-arp** command to enable proxy ARP on an Ethernet or bridge1483 interface.
- Proxy ARP is enabled by default.
- Example


```
host1(config-if)#ip proxy-arp unrestricted
```
- Use the **no** version to disable proxy ARP on the interface.

MAC Address Validation

MAC address validation is a verification process performed on each incoming packet to prevent spoofing on IP Ethernet-based interfaces, including bridged Ethernet interfaces. When an incoming packet arrives on a layer 2 interface, the validation table is used to compare the packet's source IP address to its MAC address. If the MAC address and IP address match, the packet is forwarded; if it does not match, the packet is dropped.



Note: MAC address validation for bridged Ethernet interfaces is supported only on OC12a line modules.

MAC address validation on the ERX system can be accomplished in two ways:

- You can statically configure it on a physical interface via the **arp validate** command
- You can allow DHCP to perform the function independently and dynamically. See *DHCP* in *ERX Physical and Link Layers Configuration Guide, Chapter 17, Configuring Bridged IP*.

The **arp validate** command adds the IP-MAC address pair to the validation table maintained on the physical interface.

If the validation is added statically via the CLI, the IP address–MAC address pairs are stored in NVS. The entries are used for MAC validation only if MAC validation is enabled on the interface via the **ip mac-validate** command.



Caution: When you configure an interface using the **arp validate** command, you cannot overwrite the ARP values that were added by DHCP.

You can enable or disable MAC address validation on a per interface basis by issuing the **ip mac-validate** command. See *ERX Physical and Link Layers Configuration Guide, Chapter 6, Configuring Ethernet Interfaces* or *ERX Physical and Link Layers Configuration Guide, Chapter 18, Configuring Bridged Ethernet* for information.

arp validate

- Use to add IP address–MAC address validation pairs. When validation is enabled, all packets with the source IP address received on this IP interface are validated against the IP-MAC entries.
- You specify one of the following:
 - › *ipAddress* and *macAddress* of the interface
 - › *ipAddress*, *interfaceType* and *interfaceSpecifier*, and an optional MAC address
- You can issue this command only for an IP Ethernet-based interface.
- For subscriber interface configurations, the IP address–MAC address pair must have a matching source prefix that already exists on the subscriber interface. If the matching source prefix does not exist, the IP–MAC address pair is rejected.
- The following example shows packets originating from host 192.56.20.1 and validated at Gigabit Ethernet interface with the MAC address 0090.1a00.0170.


```
host1(config)#arp 192.56.20.1 gig 2/0 0090.1a00.0170
                validate
```
- The following example shows the subscriber interface MAC address validation enabled.


```
host1(config)#arp 192.168.32.0 ip subsc 1 000.0001.8100
```
- Use the **no** version to remove an entry from the ARP cache.

Broadcast Addressing

A broadcast is a data packet destined for all hosts on a particular physical network. Network hosts recognize broadcasts by special addresses.

The system supports the following kinds of broadcast types:

- Limited broadcast – A packet is sent to a specific network or series of networks. A limited broadcast address includes the network or subnet fields. In a limited broadcast packet destined for a local network, the network identifier portion and host identifier portion of the destination address is either all ones (255.255.255.255) or all zeros (0.0.0.0).
- Flooded broadcast – A packet is sent to every network.

- Directed broadcast – A packet is sent to a specific destination address where only the host portion of the IP address is either all ones or all zeros (such as 192.20.255.255 or 190.20.0.0).

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

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

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

Broadcast Tasks

There are two broadcast-related IP commands you can use to perform broadcast-related tasks.

ip broadcast-address

- Use to broadcast to all addresses in the host portion of an IP address.
- You specify an IP address to set the broadcast address.
- Example

```
host1(config-if)#ip broadcast-address 255.255.255.255
```
- Use the **no** version to restore the default IP broadcast address.

ip directed-broadcast

- Use to enable translation of directed broadcasts to physical broadcasts.
- Example

```
host1(config-if)#ip directed-broadcast
```
- Use the **no** version to disable the function.

Fragmentation

Fragmentation is the process of segmenting a large IP datagram into several smaller pieces. Fragmentation is required when IP must transmit a large packet through a network that transmits smaller packets, or when the MTU size of the other network is smaller.

By default, the system does not fragment the packet if the don't-fragment bit (DF bit) is set in the IP header. You can specify that the system not consider the DF bit before determining whether to fragment a packet.



Note: *It is possible that lower-layer protocols may also set the MTU value. If MTU values set in lower layers differ from the one set at the IP layer, the system always uses the MTU lower-layer's value.*

ip ignore-df-bit

- Use to force the system to ignore the DF bit if it is set in the IP packet header for packets on an interface.
- Example

```
host1(config-if)#ip ignore-df-bit
```
- Use the **no** version to restore the default behavior, which is to consider the DF bit before fragmentation.

ip mtu

- Use to set the MTU size of IP packets sent on an interface.
- The range is 128–10240.
- Example

```
host1(config-if)#ip mtu 1000
```
- Use the **no** version to restore the default MTU size.

IP Routing

The Internet is a large collection of hosts that communicate with each other and use routers as intermediate packet switches.

Routers forward a packet through the interconnected system of networks and routers until the packet reaches a router that is attached to the same network as the destination host. The router delivers the packet to the specified host on its local network.

Routing Tables

A router makes forwarding decisions based on the information that is contained in the router's routing table. Routers announce and receive

route information to and from other routers. They build routing tables based on the collected information on all the best paths to all the destinations they know how to reach.

Figure 2-8 shows an example of a small network composed of four networks and three routers. The hosts that are attached to each network are not displayed, because each router makes its forwarding decisions based on the network number and not the address of each individual host. A router uses ARP to find the physical address that corresponds to the Internet address for any host or router on networks directly connected to it.

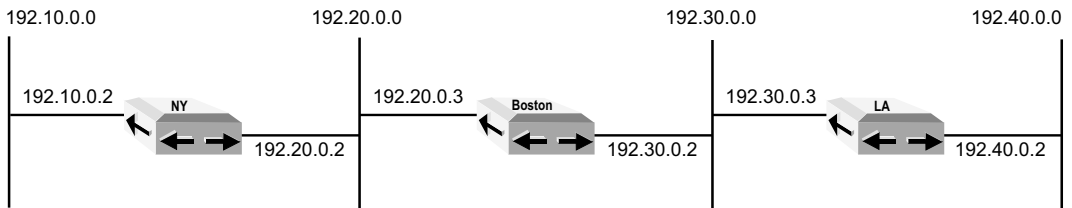


Figure 2-8 Routers in a small internet

The routing tables are shown in the following tables for routers Boston, NY, and LA. The routing tables contain one entry for each route. The columns of the routing table include:

- The destination IP network address
- The IP address of the next-hop router
- An administrative distance that is used to select the least-cost route if multiple routes to the destination network have the same metric.
- A metric that is used to select the least-cost route if more than one route exists for the destination network.

Table 2-1 Routing table for router NY

Destination Network	Next-Hop Router	Administrative Distance	Metric
192.10.0.0	Router NY	0	0
192.20.0.0	Router NY	1	0
192.30.0.0	192.20.0.3	110	1
192.40.0.0	192.20.0.3	120	2

Table 2-2 Routing table for router Boston

Destination Network	Next-Hop Router	Administrative Distance	Metric
192.10.0.0	192.20.0.2	110	1
192.20.0.0	Router Boston	0	0
192.30.0.0	Router Boston	1	0
192.40.0.0	192.30.0.3	120	1

Table 2-3 Routing table for router LA

Destination Network	Next-Hop Router	Administrative Distance	Metric
192.10.0.0	192.30.0.2	110	2
192.20.0.0	192.30.0.2	110	1
192.30.0.0	Router LA	0	0
192.40.0.0	Router LA	1	0

Setting the Administrative Distance for a Route

The administrative distance is an integer that is associated with each route known to a router. The distance represents how reliable the source of the route is considered to be. A lower value is preferred over a higher value. An administrative distance of 255 indicates no confidence in the source; routes with this distance are not installed in the routing table.

Table 2-4 shows the default distance for each type of source from which a route can be learned.

Table 2-4 Default administrative distances for route sources

Route Source	Default Distance
Connected interface	0
Static route	1
Internal access route	2
Access route	3
External BGP	20
OSPF	110
IS-IS	115
RIP	120
Internal BGP	200
Unknown	255

If the IP routing table contains several routes to the same prefix—for example, an OSPF route and a RIP route—the route with the lowest administrative distance is used for forwarding.

To set the administrative distance for BGP routes, see *Setting the Administrative Distance for a Route* in *ERX Routing Protocols Configuration Guide, Vol. 2, Chapter 1, Configuring BGP Routing*.

To set the administrative distance for RIP, IS-IS, and OSPF, use the following **distance** commands in Router Configuration mode.

distance

- Use to set an administrative distance for RIP or OSPF routes in the range 0–255.
- For RIP routes, the default value is 120.
- For OSPF routes, the default value is 110.
- Example

```
host1(config)#router rip
host1(config-router)#distance 100
```
- Use the **no** version to restore the default value.

distance ip

- Use to set the administrative distance for IS-IS routes in the range 1–255.
- Example

```
host1(config)#router isis
host1(config-router)#distance 80 ip
```
- Use the **no** version to restore the default value of 115.

Routing Operations

Routers keep track of next-hop information that enables a data packet to reach its destination through the network. A router that does not have a direct physical connection to the destination checks its routing table and forwards packets to another next-hop router that is closer to that destination. This process continues until the packet reaches its final destination.

Identifying a Router Within an Autonomous System

The router ID is commonly one of the router's defined IP addresses. Although the router ID is, by convention, formatted as an IP address, it is not required to be a configured address of the router. If you do not use the **ip router-id** command to assign a router ID, the system uses one of its configured IP addresses as the router-id.

ip router-id

- Use to assign a router ID—a unique identifier that IP routing protocols use to identify the router within an autonomous system.
- Example

```
host1(config)#ip router-id 192.32.15.23
```
- Use the **no** version to remove the router ID assignment.

Establishing a Static Route

You can set a destination to receive and send traffic by a specific route through the network.

ip route

- Use to establish a static route.
- Example

```
host1(config)#ip route 192.56.15.23 255.255.255.0 192.66.0.1
```
- Use the **no** version to remove a static route from the routing table.

Setting Up Default Routes

A router examines its routing table to find a path for each packet. If the router cannot locate a route, it must discard the packet. You can set up a default route using the special address: 0.0.0.0. If the router cannot locate a path to a destination network and a default route is defined, the system forwards the packet to the default router. For example:

```
host1(config)#ip route 0.0.0.0 0.0.0.0 192.56.21.3
```

Default routes are typically used to reduce the size of the routing table. Routing is simplified because the router can test for a few local networks or use the default route. However, a disadvantage of default routes is the possible creation of multiple paths and routing loops.

Setting Up an Unnumbered Interface

An unnumbered interface does not have an IP address assigned to it. Unnumbered interfaces are often used in point-to-point connections where an IP address is not required.

ip unnumbered

- Use to set up an unnumbered interface.
- This command enables IP processing on an interface without assigning an explicit IP address to the interface.

- You supply an interface location, which is the type and number of another interface on which the router has an assigned IP address. This interface cannot be another unnumbered interface.
- Example

```
host1(config-if)#ip unnumbered fastEthernet 0/0
```
- Use the **no** version to disable IP processing on an interface.

Adding a Host Route to a Peer on a PPP Interface

You can enable the ability to create host access routes on a PPP interface. This feature is useful in B-RAS applications.

ip access-routes

- Use to enable the ability to create host access routes on a PPP interface.
- Example

```
host1(config-if)#ip access-routes
```
- Use the **no** version to disable this feature.

Enabling Source Address Validation

Source address validation verifies that a packet has been sent from a valid source address. When a packet arrives on an interface, the system performs a routing table lookup using the source address. The result from the routing table lookup is an interface to which packets destined for that address are routed. This interface must match the interface on which the packet arrived. If it does not match, the system drops the packet.

ip sa-validate

- Use to enable source address validation.
- Example

```
host1(config-if)#ip sa-validate
```
- Use the **no** version to disable source address validation.

Shutting Down an IP Interface

The system lets you disable an interface at the IP level without removing it.

ip shutdown

- Use to shut down an IP interface.
- Example

```
host1(config-if)#ip shutdown
```
- Use the **no** version to restart the interface.

Removing the IP Configuration

You can remove the IP configuration from an interface or subinterface.

no ip interface

- Use to remove the IP configuration from an interface or subinterface and disable IP processing on the interface.
- Example

```
host1(config-if)#no ip interface
```

Clearing IP Routes

The system enables you to clear the specified routing entries from the routing table. You must specify the IP address prefix and the mask of the IP address prefix to clear specific routes. You can later reinstall the routes you have cleared.

clear ip routes

- Use to clear specified IP routes according to an IP prefix or a VPN routing and forwarding (VRF) table.
- Use an asterisk (*) to clear all dynamic routes from the routing table.
- Example

```
host1#clear ip routes *
```
- There is no **no** version.

ip refresh-route

- Use to enable the owning protocols (BGP, IS-IS, OSPF) to reinstall routes removed from the IP routing table by the **clear ip routes** command.
- Example

```
host1#ip refresh-route
```
- There is no **no** version.

Clearing IP Interfaces

The system enables you to clear the counters on the specified IP interface(s).

clear ip interface

- Use to clear a specified IP interface.
- Example

```
host1#clear ip interface pos 2/0
```
- There is no **no** version.

Setting a Baseline

The system enables you to set a baseline for statistics on an IP interface.

baseline ip interface

- Use to set a baseline for a specified IP interface.
- Example

```
host1#baseline ip interface pos 2/0
```
- There is no **no** version.

Disabling Forwarding of Packets

The system allows you to disable forwarding of packets on an SRP Ethernet interface.

ip disable-forwarding

- Use to disable forwarding of packets on the SRP Ethernet interface.
- The purpose of this command is to maintain system performance by maximizing the CPU time available for routing protocols. Although you can allow data forwarding on the SRP Ethernet interface, system performance will be affected.
- You see an error message if you try to set this command for interfaces other than the SRP Ethernet interface.
- Example

```
host1(config-if)#ip disable-forwarding
```
- Use the **no** version to enable forwarding of packets on the interface.

Enabling Forwarding of Source-Routed Packets

IP packets are normally routed according to the destination address they contain based on the routing table at each hop through a path. The originator or source of the source-routed packets specifies the path (the series of hops) that the packets must traverse; the source makes the

routing decisions. The source can specify either of the following types of source routing:

- *Strict-source* routing specifies every hop that the packet must traverse. The specified path consists of adjacent hops. The source generates an ICMP error if the exact path cannot be followed. For example, for a path going from source router A to router B to router C to router D, router A would specify a strict-source route as B, C, D.
- *Loose-source* routing specifies a set of hops that the packet must traverse, but not necessarily every hop in the path. That is, the specified hops do not have to be adjacent. For example, for a path going from source router A to router B to router C to router D, router A would specify a loose-source route as B, D or C, D, or B, C, D.

ip source-route

- Use to enable forwarding of source-routed packets.
- Forwarding is enabled by default.
- Example

```
host1#ip source-route
```
- Use the **no** version to disable forwarding of source-routed packets on the interface.

Forcing an Interface to Appear Up

The system enables you to force an IP interface to appear as if it is up, regardless of the state of the lower layers.

ip alwaysup

- Use to force an IP interface to appear as up regardless of the state of lower layers.
- This command reduces route topology changes when the network attached to this link is single-homed.
- Example

```
host1(config-if)#ip alwaysup
```
- Use the **no** version to make the interface appear in the current state.

Specifying a Debounce Time

You can set a debounce time that requires an IP interface to be in a given state—for example, up or down—for the specified time before the state is reported. This feature prevents a link that briefly goes up or down from causing an unnecessary topology change, for example by causing an interface down condition.

ip debounce-time

- Use to set the interval for which an interface must maintain a given state before the state change is reported.
- Example

```
host1(config)#ip debounce-time 5000
```
- Use the **no** version to remove the debounce time requirement.

Adding a Description

The system enables you to add a text description to an IP interface.

description

- Use to add a text description to an IP interface.
- The description must not exceed 256 characters.
- Example

```
host1(config-if)#description serial interface FR-T1 contact
Bob Brown at 999-9991 x456
```
- Use the **no** version to delete the description.

Enabling Link Status Traps

The system allows you to enable link status traps on an interface.

snmp trap ip link-status

- Use to enable link status traps on an interface.
- Example

```
host1(config-if)#snmp trap ip link-status
```
- Use the **no** version to disable link status traps on an interface.

Configuring the Speed

The system enables you to set the speed of an IP interface.

ip speed

- Use to set the speed of the interface in bits per second.
- By default, the speed is determined from a lower-layer interface.
- Example

```
host1(config-if)#ip speed 1000
```
- Use the **no** version to set the speed to the default, 0.

Configuring Equal-Cost Multipath Load Sharing

Equal-cost multipath (ECMP) sets are formed when the system finds routing table entries for the same destination with equal cost. You can add routing table entries manually (as static routes), or they are formed as routers discover their neighbors and exchange routing tables (via OSPF, BGP, and other routing protocols). The system then balances traffic across these sets of equal-cost paths using one of the following ECMP modes:

- Hashed – uses hashing of source and destination addresses to determine which of the available paths in the ECMP set to use
- Round-robin – distributes packets equally among the available paths in the ECMP set

ip multipath round-robin

- Use to specify round-robin as the mode for ECMP load sharing on an interface.
- ECMP uses the round-robin mode when you have configured *all* interfaces in the set to round-robin. Otherwise, ECMP defaults to hashed mode because round-robin mode could cause reordering of packets. You must explicitly ensure that the possible reordering is acceptable on all the member interfaces by setting them to round-robin mode.
- Use the **no** version to set the ECMP mode to the default, hashed.
- Example

```

host1(config)#virtual-router router_0
host1:router_0(config)#interface serial 4/0:1/22.22
host1:router_0(config-subif)#ip multipath round-robin
host1:router_0(config-subif)#exit
host1:router_0(config)#exit
host1:router_0#show ip interface serial 4/0:1/22.22
serial4/0:1/22.22 is up, line protocol is up
  Network Protocols: IP
  Internet address is 190.121.1.1/255.255.0.0
  Broadcast address is 255.255.255.255
  Operational MTU = 1600  Administrative MTU = 0
  Operational speed = 64000  Administrative speed = 0
  Router advertisement = disabled
  Administrative debounce-time = disabled
  Operational debounce-time = disabled
  Access routing = disabled
  Multipath mode = round-robin

  In Received Packets 0, Bytes 0
  In Policed Packets 0, Bytes 0
  In Error Packets 0
  In Invalid Source Address Packets 0
  Out Forwarded Packets 0, Bytes 0

```

```
Out Scheduler Drops Packets 0, Bytes 0
```

maximum-paths

- Use to control the maximum number of parallel routes that the routing protocol (BGP, IS-IS, OSPF, or RIP) can support.
- The maximum number of routes can range from 1–6 for BGP and from 1–16 for IS-IS, OSPF, or RIP.
- Example

```
host1(config-router)#maximum-paths 2
```
- Use the **no** version to restore the default value, 1 for BGP or 4 for IS-IS, OSPF, or RIP.

Setting a TTL Value

You can use the **ip ttl** command to set the TTL (time-to-live) field in the IP header for all IP operations. The TTL specifies a hop count. This configured TTL value can be overridden by other commands that specify a TTL.

ip ttl

- Use to set a default value for the IP header TTL field for all IP operations.
- Example

```
host1(config)#ip ttl 255
```
- Use the **no** version to restore the default value, 127.

Shared IP Interfaces

You can create multiple *shared* IP interfaces over the same layer 2 logical interface—for example, atm 5/3.101—enabling more than one IP interface to share the same logical resources. This capability is useful, for example, when data received in one VRF needs to be forwarded out an interface in another VRF, such as for BGP/MPLS VPNs (see *ERX Routing Protocols Configuration Guide, Vol. 2, Chapter 3, Configuring BGP/MPLS VPNs*, for more information). You can configure one or more shared IP interfaces. Data sent over shared interfaces use the same layer 2 interface. You can configure shared interfaces as you would unshared IP interfaces. Each shared interface has its own statistics.

Some layer 2 interfaces require a primary IP interface to negotiate certain IP parameters—for example, IPCP for PPP, ARP for Ethernet, and Inverse ARP for Frame Relay. If you do not configure a primary IP interface in such cases, the layer 2 interface cannot become operationally up.

A primary IP interface is the default interface for receiving data that arrives on the layer 2 interface. If you configure shared IP interfaces for the same layer 2 interface as your primary IP interface, by default data received on the layer 2 interface is received on the virtual router corresponding to the primary IP interface. A primary IP interface and all of its shared IP interfaces have the same interface location. You cannot configure a shared IP interface to receive data on the same layer 2 interface as a primary IP interface. You can delete primary and shared IP interfaces independently of each other.

You can create a primary IP interface as you do any other IP interface, as shown in the following example:

```
host1(config)#virtual-router vr-a:vrf-2
host1:vr-a:vrf-2:(config)#interface atm 5/3.101
host1:vr-a:vrf-2:(config-if)#ip address 10.1.1.1
255.255.255.255
host1:vr-a:vrf-2:(config-if)#exit
```

You do not have to configure a primary IP interface if you do not need one as described above. In the absence of a primary interface, you can still configure shared IP interfaces; however, in this scenario, data received on the layer 2 interface is discarded.

You cannot create shared IP interfaces for the following kinds of interface:

- IP floating interfaces (IP interfaces that stack over MPLS stacked tunnels)
- Loopback interfaces
- Null interfaces

For information about configuring shared IP interfaces to receive data on the same layer 2 interface as a primary IP interface, see *Subscriber Interfaces*, later in this chapter.

Configuring Prefixes for Dynamic Shared IP Interfaces

For the typical BGP/MPLS VPN scenario, shared interfaces are created dynamically based on VPN membership in support of overlapping VPNs. In screen output, dynamic interfaces are distinguished from those you configure statically by a prefix (default prefix is dyn). You can define the prefix with the **ip dynamic-interface-prefix** command.

See *ERX Routing Protocols Configuration Guide, Vol. 2, Chapter 3, Configuring BGP/MPLS VPNs*, for more information on VPNs and shared IP interfaces.

Configuring Shared IP Interfaces

To share IP interfaces:

- 1 Create a layer 2 interface.

```
host1(config)#interface atm 5/3
host1(config-if)#interface atm 5/3.101
```

- 2 (Optional) Create a primary IP interface.

```
host1(config-if)#ip address 10.1.1.1 255.255.255.255
host1(config-if)#exit
```

- 3 Create the shared IP interface.

```
host1(config)#interface ip si0
```

- 4 Associate the shared IP interface with the layer 2 interface by one of the following methods:

- Statically

```
host1(config-if)#ip share-interface atm5/3.101
```

- Dynamically

```
host1:vr-a:vrf-1(config-if)#ip share-nexthop 10.0.0.1
```

- 5 To fully configure the shared interface, assign an address (or make it unnumbered).

```
host1(config-if)#ip address 2.2.2.2 255.0.0.0
```

- 6 (Optional) For BGP/MPLS VPNs, configure a prefix for the shared IP interfaces.

```
host1(config-if)#ip dynamic-interface-prefix vrf2 rt66
```

interface ip

- Use to create an IP interface for interface sharing.
- Use the specified name to refer to the shared IP interface; you cannot use the layer 2 interface to refer to them, because the shared interface can be moved.

- Example

```
host1(config)#interface ip si0
```

- Use the **no** version to delete the IP interface.

ip dynamic-interface-prefix

- Use to specify the prefix that distinguishes dynamic shared IP interfaces (created for overlapping BGP/MPLS VPNs) from static shared IP interfaces.
- You can specify the VRF in which the shared IP interface is created.
- Dynamic shared IP interfaces are represented in screen output as

```
ipprefixvrfName-number
```

where *prefix* is the prefix you specified, *vrfName* is the VRF where the primary interface is located, and *number* is incremented as these interfaces are created.

- **Example**

```
host1:pe1(config)#ip dynamic-interface-prefix vrf pe12 vpnsh
host1:pe1(config)#exit
host1:pe1#show ip vrf
VRF Name          Default RD          Interfaces
  pe1              1:1                null0
                  atm4/0.2
                  ipdynpe12-6
  pe2              1:2                null0
                  ipvpnshpe11-5
                  atm4/0.3
```

- The **no** version restores the default prefix value, dyn.

ip share-interface

- Use to specify the layer 2 interface used by a shared IP interface. The command fails if the layer 2 interface does not yet exist.
- If you issue this command on a shared IP interface, you cannot issue the **ip share-nexthop** command for the interface.
- After creating the shared IP interface, you can configure it as you would any other IP interface.
- The shared interface is operationally up when the layer 2 interface is operationally up.
- You can create operational shared IP interfaces in the absence of a primary IP interface.
- **Example**

```
host1(config-if)#ip share-interface atm5/3.101
```

- Use the **no** version to remove the association between the layer 2 interface and the shared IP interface. You can delete shared and primary IP interfaces independently.

ip share-nexthop

- Use to specify that the shared IP interface dynamically tracks a next hop. If the next hop changes, the shared IP interface moves to the new layer 2 interface associated with the IP interface toward the new next hop.
- If you issue this command on a shared IP interface, you cannot issue the **ip share-interface** command for the interface.
- If you specify a virtual router, the command fails if the VR does not already exist. If you do not specify a VR, the current VR is assumed.
- After creating the shared IP interface, you can configure it as you would any other IP interface.

- The shared interface is operationally up when the layer 2 interface associated with the specified next hop is operationally up.
- Use the **no** version to halt tracking of the next hop.

Moving IP Interfaces

You cannot directly move a primary IP interface. However, primary IP interfaces created dynamically by MPLS move when the underlying MPLS minor interface moves or changes.

You can move an IP shared interface from one layer 2 interface to another by issuing the **ip share-interface** command to specify a different layer 2 interface. Moving an IP interface does not affect interface statistics, packets forwarded through the interface, or policies attached to the IP interface.

Example The following commands create shared interface si0 on the layer 2 interface atm5/3.101:

```
host1(config)#virtual-router vr-a:vrf-1
host1:vr-a:vrf-1(config)#interface ip si0
host1:vr-a:vrf-1(config-if)#ip share-interface atm5/3.101
host1:vr-a:vrf-1(config-if)#exit
```

The following commands move shared interface si0 to the layer 2 interface atm5/3.201:

```
host1:vr-a:vrf-1(config)#interface ip si0
host1:vr-a:vrf-1(config-if)#ip share-interface atm5/3.201
```

IP Shared Interface Statistics

Each shared interface has its own statistics. Packets transmitted on a shared IP interface are always counted only in the shared IP interface.

Subscriber Interfaces

The ERX system supports subscriber interfaces on a particular type of layer 2 interface, Ethernet. In the absence of VLANs, Ethernet does not have a demultiplexing layer. A subscriber interface adds a demultiplexing layer for an Ethernet interface that is configured without VLANs. Using subscriber interfaces, the system can demultiplex or separate the traffic associated with different subscribers. For information about which Ethernet modules support subscriber interfaces, see the *Release Notes*.

You can configure subscriber interfaces with VLANs. If you do so, the ERX system demultiplexes packets using first the VLAN and then the subscriber interface.

Overview

A subscriber interface is an extension of a *shared IP interface* (see *Shared IP Interfaces* earlier in this chapter). A shared IP interface is one of a group of IP interfaces that use the same layer 2 interface. Shared IP interfaces are unidirectional—they can transmit but not receive traffic. Subscriber interfaces are bidirectional—they can both receive and transmit traffic.

A subscriber interface operates only with a *primary IP interface*—a normal IP interface on an Ethernet interface. You create a primary interface by assigning an IP address to the Ethernet interface. Although you can configure a subscriber interface directly on an Ethernet interface, the subscriber interface will not operate until you assign an IP address to the Ethernet interface.

To configure a subscriber interface you must associate either a source address or a destination address with the interface. The system will receive packets on a subscriber interface after demultiplexing the packet according to the specified source address or destination address. You can associate multiple source addresses or multiple destination addresses with a subscriber interface. However, you cannot associate both source addresses and destination addresses with one subscriber interface.

Figure 2-9 illustrates the relationship between subscriber interfaces, the associated primary IP interface, and the associated Ethernet interface.

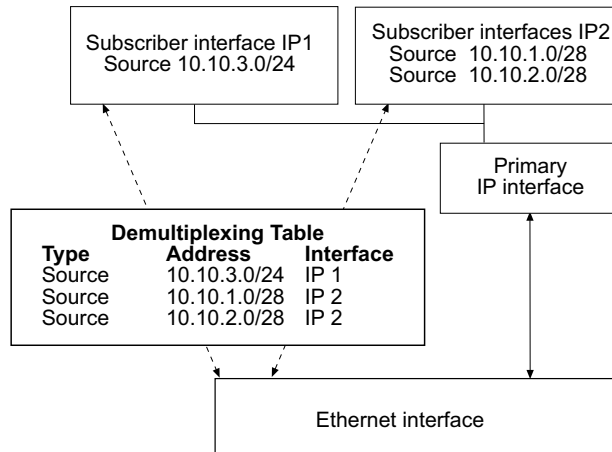


Figure 2-9 Subscriber interfaces

When the system receives traffic on a subscriber interface, it forwards the traffic as it would forward traffic received on the associated primary interface.

Moving Interfaces

Subscriber interfaces move with the shared IP interface with which they are associated. If the shared IP interface dynamically tracks a next hop and the next hop changes, the shared IP interface and the subscriber interface move to the new layer 2 interface associated with the IP interface toward the new next hop.

IP Spoofing

You can prevent IP spoofing on subscriber interfaces by using MAC address validation. See *MAC Address Validation*, earlier in this chapter.

Routing Protocols

You configure unicast routing protocols on subscriber interfaces in the same way that you configure routing protocols on primary IP interfaces. Subscriber interfaces do not support multicast routing protocols.

Policies and QoS

You can configure policies, such as rate limiting and filtering, and QoS for subscriber interfaces in the same way that you would for primary IP interfaces. For more information, see *ERX Policy and QoS Configuration Guide*.

Applications

In a cable modem network, service providers can use subscriber interfaces to allocate bandwidth to subscribers based on the level of service purchased. Figure 2-10 shows an example of a cable modem network. Multiple cable modem termination systems (CMTSs) connect to multiple shared media access LANs. Many subscribers connect to each LAN.

Service providers can assign a subscriber interface to the subscribers who purchase the same level of service. Rate limits and policies on the subscriber interface will customize the level of service for the associated subscribers. In this application, the ERX system is the first-hop router for the subscribers, and subscriber interfaces demultiplex traffic based on the source address.

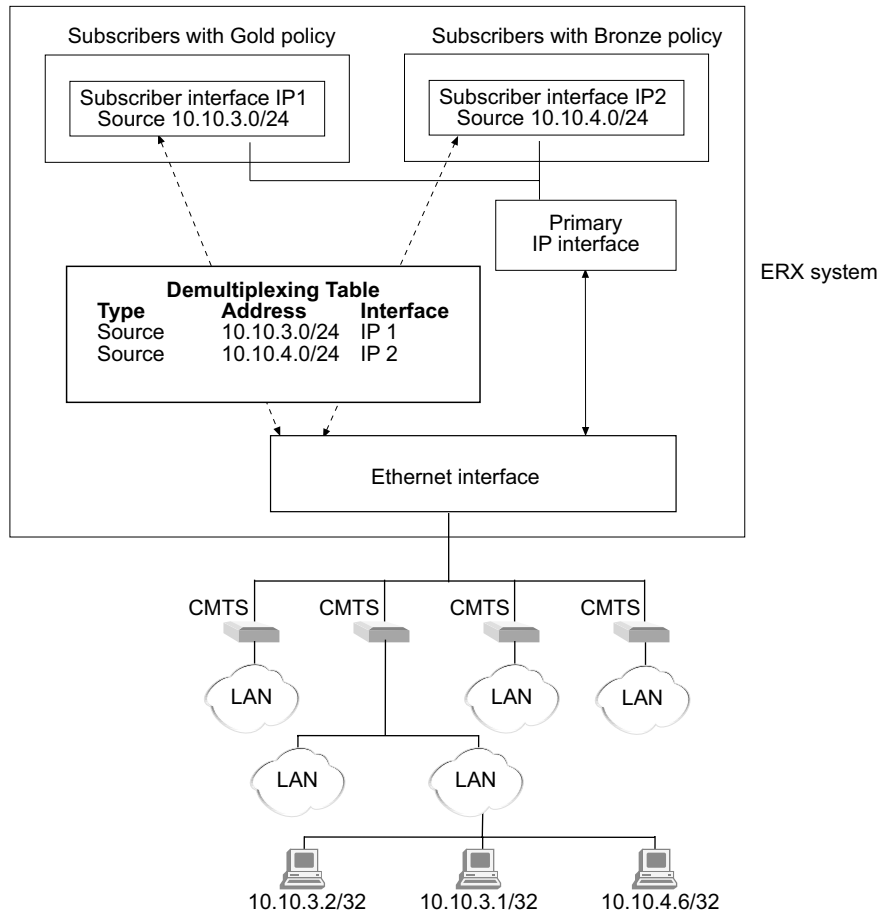


Figure 2-10 Subscriber interfaces in cable modem network

Similarly, service providers can use subscriber interfaces to differentiate traffic for VPNs. Figure 2-11 shows an example of this application. Customers on subnet A need to connect to VPN B, and customers on subnet B need to connect to VPN A. The ERX system connects to VPN A through virtual router A and to VPN B through virtual router B. Using two subscriber interfaces on the same primary interface (one on virtual router B and one on virtual router A) the ERX system can separate the traffic from subnets A and B. Because the ERX system is forwarding traffic in this application, the shared ip interface should demultiplex the traffic using a destination address.

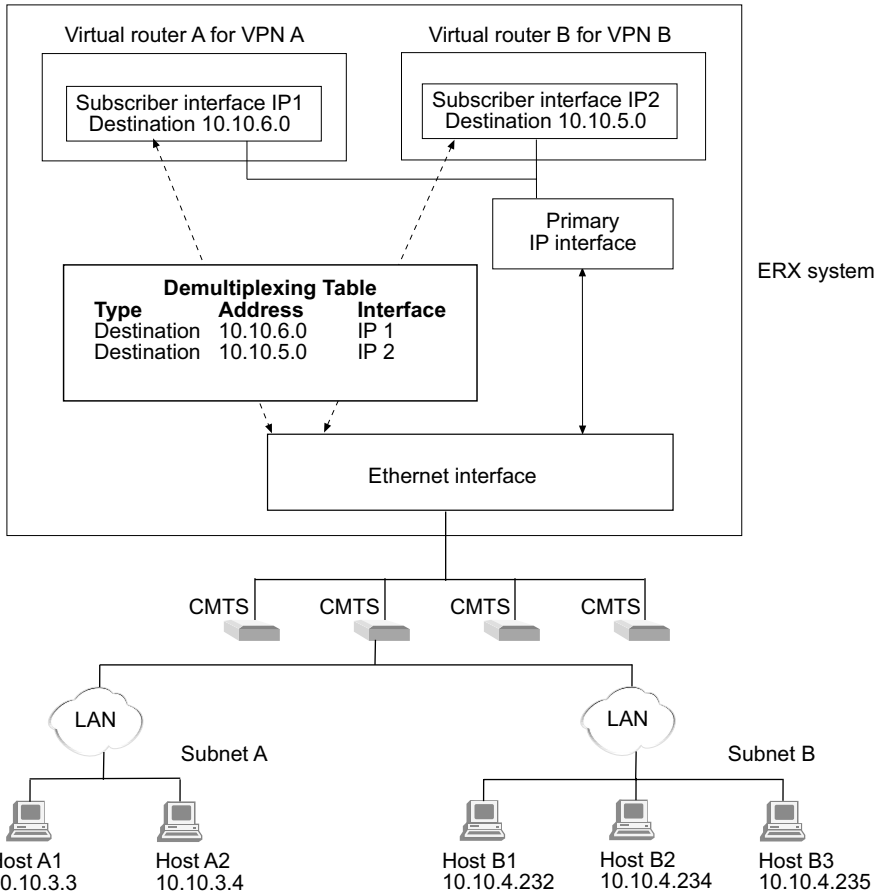


Figure 2-11 Associating subnets with a VPN using subscriber interfaces

Configuring Subscriber Interfaces

To configure a subscriber interface:

- 1 Configure a shared IP interface on an Ethernet interface. (See *Configuring Shared IP Interfaces*, steps 1 to 5, earlier in this chapter.)
- 2 (Optional) Configure the subscriber interface to demultiplex traffic using a destination address.

```
host1(config-if)#ip demux-type da-prefix
```

By default, subscriber interfaces demultiplex traffic using source addresses.

3 Specify the addresses that the subscriber interface will demultiplex:

- To specify the source addresses, use the **ip source-prefix** command.

```
host1(config-if)#ip source-prefix 2.2.2.2 255.0.0.0
```

- To specify the destination addresses, use the **ip destination-prefix** command.

```
host1(config-if)#ip destination-prefix 2.2.2.2 255.0.0.0
```

```
host1(config-if)#ip destination-prefix 2.2.2.3 255.0.0.0
```

ip demux-type da-prefix

- Use to specify that the system should use a destination address to demultiplex traffic for the subscriber interface.
- Examples

```
host1(config-if)#ip demux-type da-prefix
```
- Use the **no** version to restore the default situation in which the system uses a source address to demultiplex traffic.

ip destination-prefix

- Use to specify a destination address for a subscriber interface.
- Example

```
host1(config-if)#ip destination-prefix 2.2.2.2 255.0.0.0
```
- Use the **no** version to delete the subscriber interface.

ip source-prefix

- Use to specify a source address for a subscriber interface.
- Example

```
host1(config-if)#ip source-address 2.2.2.2 255.0.0.0
```
- Use the **no** version to delete the subscriber interface.

Preventing IP Spoofing

To configure subscriber interfaces to prevent IP spoofing, see *MAC Address Validation*, earlier in this chapter.

Monitoring Subscriber Interfaces

No counters or statistics are associated with a subscriber interface. The state of the subscriber interface is determined by state of the Ethernet interface and the existence of the primary IP interface, which you can monitor with the **show ip interface** command.

You can use the **show ip demux interface** command to monitor the configuration of subscriber interfaces.

For more information about these commands, see *IP show Commands*, later in this chapter.

Internet Control Message Protocol

IP was not designed to provide reliable delivery service. The higher-layer protocols that operate as clients of IP implement their own reliability procedures if reliable communications are required.

Internet Control Message Protocol (ICMP) provides a mechanism that enables a router or destination host to report an error in data traffic processing to the original source of the packet. ICMP messages provide feedback about problems that occur in the communication environment.

ICMP messages are sent only when errors occur in either the processing of an unfragmented data packet or the first fragment of a fragmented data packet.

ICMP messages are encapsulated as part of the data portion of an IP data packet and are routed like any other IP data packets. Thus, there is no guarantee to the sender of an ICMP message that it will be delivered to its destination.

The system supports ICMP redirects. When a packet enters an IP interface and exits the same interface, the system may send an ICMP message to the originator of the packet. This message notifies the originator that there is a better gateway for the assigned destination address.

The **ip redirects** command (used in Interface Configuration mode) lets you enable or disable ICMP redirects. This attribute is enabled by default. If it is enabled on the IP interface, and if the internal ICMP redirect queue is not full, the system sends an ICMP redirect packet to the originator. When the originator receives the ICMP redirect notification, it is up to the originator to start using the better gateway.

ICMP Tasks

You can enable the following ICMP features:

- ICMP Router Discovery Protocol (IRDP)
- ICMP netmask reply
- Sending of IP redirects
- Generation of ICMP unreachable messages

ip irdp

- Use to enable IRDP processing on an interface.
- Example

```
host1(config-if)#ip irdp
```
- Use the **no** version to disable the function.

ip mask-reply

- Use to enable ICMP netmask reply.
- Example

```
host1(config-if)#ip mask-reply
```
- Use the **no** version to disable the function.

ip redirects

- Use to enable the sending of redirect messages if software is forced to resend a packet through the same interface on which it was received.
- Example

```
host1(config-if)#ip redirects
```
- Use the **no** version to disable the sending of redirect messages.

ip unreachable

- Use to enable the generation of an ICMP unreachable message when a packet is received that the router cannot deliver.
- Example

```
host1(config-if)#ip unreachable
```
- Use the **no** version to disable function.

Reachability Commands

Use the **ping** and **traceroute** commands to determine reachability of destinations in the network.

- Use the **ping** command to send an ICMP echo request packet. In the following example, the request packet is sent to address 192.35.42.1 with a packet count of 10 and a timeout value of 10 seconds:

```
host1#ping 192.35.42.1 10 timeout 10
```

- Use the **traceroute** command to discover routes that router packets follow when traveling to their destination. In the following example, the trace destination address is 192.56.20.1, the maximum number of hops of the trace is 20, and the timeout value is 10 seconds:

```
host1#traceroute 192.56.20.1 20 timeout 10
```

ping

- Use to send an ICMP echo request packet to the IP address that you specify.
- You can specify a VRF context.
- Use the **source interface** keywords to specify a source interface other than the one from which the probe originates.
- Use the **source address** keywords to specify a source IP address other than the one from which the probe originates.
- You can specify the following options:
 - › **packetCount** – number of packets to send to the destination IP address. If you specify a zero, echo requests packets are sent indefinitely.
 - › **data-pattern** – sets the type of bits contained in the packet to all ones, all zeros, a random mixture of ones and zeros, or a specific hexadecimal data pattern that can range from 0x0–0xFFFFFFFF. The default is all zeros.
 - › **data-size** – sets the number of bytes comprising the IP packet and reflected in the IP header in the range 0–64000; the default is 100 bytes
 - › **extended** header attributes – set the following:
 - A value to be set in the type of service (ToS) byte, in the range 0–255, to support quality of service (QoS) offerings
 - Don't-fragment bit to prevent IP from fragmenting the packet if it is too long for the MTU of a given link; if the nonfragmented packet cannot be delivered, it is discarded
 - Strict-source or loose-source routing, including the IP address of the hops the packets must traverse. For loose-source-route, you specify some or all of the hops, but they do not have to be adjacent. For strict-source-route, you must specify every adjacent hop through which the packet must traverse.
 - The IP addresses to be recorded for a specified number of routers that the packets traverse
 - The time that a packet traverses a router to be recorded for a specified number of routers

- An interface type and specifier of a destination address on the system that is configured for external loopback; the command succeeds only if the specified interface is configured for external loopback
- › **sweep-interval** – specifies the change in the size of subsequent ping packets while sweeping across a range of sizes. For example, you can configure the sweep interval to sweep across the range of packets from 100 bytes to 1000 bytes in increments equal to the sweep interval. By default the system increments packets by one byte; for example, it sends 100, 101, 102, 103, ... 1000. If the sweep interval is 5, the system sends 100, 105, 110, 115, ... 1000.
- › **sweep-sizes** – enables you to vary the sizes of the echo packets being sent. This capability is useful for determining the minimum sizes of the MTUs configured on the nodes along the path to the destination address. This reduces packet fragmentation, which contributes to performance problems. The default is not to sweep (all packets are the same size).
- › **timeout** – sets the number of seconds to wait for an ICMP echo reply packet before the connection attempt times out
- › **ttl** – sets the time-to-live hop count in the range 1–255; the default is 32
- The following characters can appear in the display after issuing the **ping** command:
 - › ! – reply received
 - › . – timed out while waiting for a reply
 - › ? – unknown packet type
 - › A – address mask request message
 - › a – address mask reply message
 - › D – router discovery advertisement message
 - › d – router discovery request message
 - › H – host unreachable
 - › I – information request message
 - › i – information reply message
 - › L – TTL expired message
 - › M – could not fragment, DF bit set
 - › m – parameter problem message
 - › N – network unreachable
 - › P – protocol unreachable
 - › Q – source quench
 - › r – redirect message
 - › T – timestamp request message
 - › t – timestamp reply message
 - › U – destination unreachable

- Example


```
host1(config)#interface serial 5/2:1/1
host1(config-if)#ip address 172.16.1.1 255.255.255.0
host1(config-if)#exit
host1#ping 172.16.1.1 extended interface serial 5/2:1/1
```
- There is no **no** version.

traceroute

- Use to discover the routes that router packets follow when traveling to their destination.
- You can specify:
 - › A VRF context
 - › Destination IP address
 - › Source interface for each of the transmitted packets
 - › Source IP address for each of the transmitted packets
 - › Maximum number of hops of the trace and a timeout value
 - › Size of the IP packets (not the ICMP payload) in the range 0–64000 bytes sent with the **traceroute** command. Including a size might help locate any MTU problems that exist between your system and a particular device.
 - › Extended IP header attributes, including the ToS byte and whether to set the DF bit for the transmitted packets.
- You can also force transmission of the packets on a specified interface regardless of what the IP address lookup indicates.
- Example


```
host1#traceroute 172.20.13.1 20 timeout 10
```
- There is no **no** version.

Response Time Reporter

The Response Time Reporter (RTR) feature enables you to monitor your network's performance and its resources. It does this by measuring response times and the availability of your network devices.

Configuration Tasks

To configure RTR:

- 1 Configure the probe type—an echo probe or a path echo probe.
- 2 (Optional) Configure probe characteristics:
 - frequency
 - hops-of-statistics-kept (path echo)

- max-response-failure (path echo)
- operations-per-hop (path echo)
- owner
- request-data-size
- samples-of-history-kept
- tag
- timeout (echo)
- tos



Note: You cannot set any of these characteristics until you have set the probe type. The default values of these characteristics depend on the type of the entry.

- 3 (Optional) Set reaction conditions.
- 4 Schedule the probe.
- 5 (Optional) Capture statistics and collect error information.
- 6 (Optional) Collect history.



Note: RTR configuration is associated with a specific virtual router, distinct from any other virtual router.

Configuring the Probe Type

To begin configuring RTR, enter RTR Configuration mode and configure the probe type—either an *echo* probe or a *path echo* probe.

rtr

- Use to configure an RTR probe and to enter RTR Configuration mode.
- Example

```
host1(config)#rtr 1
```
- Use the **no** version to delete all configuration information for an RTR probe.

type

- Use to set an echo or path echo probe:
 - › **echo** – limited to end-to-end RTR operations; corresponds to SNMP ping
 - › **pathEcho** – finds a path to the destination and echoes each device in the path; corresponds to SNMP traceroute
- You must specify this value before any other.
- If you change the type for an existing RTR entry, all values are reset, including the administrative status. There is no default value.

- More than one RTR entry can become active, provided each entry's target address is unique.
- If you use a target address already configured for another RTR entry that is active, the test will not run if both entries are in the same virtual router. If they are in distinct virtual routers, however, there is no restriction.
- Example


```
host1(config-rtr)#type echo protocol ipIcmpEcho 10.10.0.9
```
- Use the **no** version to remove the type configured for the probe.

Configuring Optional Characteristics

In addition to configuring the probe's type, you can configure the probe characteristics presented in Table 2-5.

Table 2-5 Probe characteristics

Characteristic	Description
frequency	Time between tests (in seconds)
hops-of-statistics-kept	Hops per path for which statistics are gathered
max-response-failure	Maximum number of consecutive failures
operations-per-hop	Number of probes per hop
owner	Owner of the probe
request-data-size	Request's payload size
samples-of-history-kept	Maximum number of history samples
tag	User-defined tag
timeout	Probe timeout (in milliseconds)
tos	A value for the TOS byte

frequency

- Use to set the rate (in seconds) the RTR probe uses to start a response time operation.
- Example


```
host1(config-rtr)#frequency 90
```
- Use the **no** version to return to the default value, 60 seconds.

operations-per-hop

- Use to set the number of RTR probe operations sent to a given hop.
- You can apply this option only to a pathEcho type.
- Example


```
host1(config-rtr)#operations-per-hop 5
```
- Use the **no** version to return to the default, 3.

owner

- Use to identify the owner of the probe.
- If the SNMP agent is the owner of the probe, the owner's name may begin with *agent*.
- Example

```
host1(config-rtr)#owner 192.10.27.6 rtc.boston.com 555.1212
```
- Use the **no** version to return to the default, no owner.

request-data-size

- Use to set the protocol data size in the request packet.
- Example

```
host1(config-rtr)#request-data-size 20
```
- Use the **no** version to return to the default value, 1 byte.

tag

- Use to set an identifier for the probe.
- Example

```
host1(config-rtr)#tag westford
```
- Use the **no** version to return to the default, no tag.

timeout

- Use to set the time (in milliseconds) that the probe waits for a response.
- You can apply this option only to an echo type.
- Do not set the value for timeout to more than the value set for frequency. If you do, it will be ignored.
- Example

```
host1(config-rtr)#timeout 3000
```
- If you set the timeout to 0, no timeout is set.
- Use the **no** version to return to the default value, 5000 milliseconds.

tos

- Use to set the type of service (ToS) byte in the probe's IP header.
- Example

```
host1(config-rtr)#tos 16
```
- Use the **no** version to return to the default value, 0. The default applies to both the echo and pathEcho types.

Capturing Statistics

The primary objective of RTR is to collect statistics and information on network performance. You can control the number and type of statistics collected.

hops-of-statistics-kept

- Use to set the number of hops per path for which statistics are collected.
- When the number of hops reaches the specified number (that is, *size*), no additional statistical information on the path is stored.
- This option applies only to pathEcho entries.

- Example

```
host1(config-rtr)#hops-of-statistics-kept 5
```

- To turn off this feature, set the value to 0.
- Use the **no** version to set the default, 16 hops.

max-response-failure

- Use to set the maximum number of consecutive failures to respond to a probe's request.
- When the maximum number is reached, the test stops.
- This option applies only to pathEcho entries.

- Example

```
host1(config-rtr)#max-response-failure 2
```

- To turn off this feature, set the value to 0.
- Use the **no** version to set the default, 5 consecutive failures.

Collecting History

RTR can collect data samples for a given probe. These samples are referred to as history data. When RTR collects history, it refers to tests. A test is the lifetime of a probe operation.

samples-of-history-kept

- Use to set the maximum number of entries in the history table for each RTR probe.
- This command enables you to control the number of samples saved in the history table.
- Because collecting history increases memory usage, you should do so only when there is a problem in your network.

- Example

```
host1(config-rtr)#samples-of-history-kept 5
```

- If you set the number of samples to 0, no samples will be kept.
- Use the **no** version to set the default, 16 hops for pathEcho type; 1 hop for echo type

Setting Reaction Conditions

You can set the RTR probe to react to events that take place and to send notifications about these events.



Note: The only **no** version for all the *rtr reaction-configuration* commands is **no rtr reaction-configuration rtrIndex**. Use the **no** version to clear all traps. This works for all the options.

rtr reaction-configuration action-type

- Use to specify the type of actions to occur depending on the events controlled by RTR.
- The default is to take the traps of enabled events.
- Example

```
host1(config)#rtr reaction-configuration 1 action-type
trapOnly
```
- There is no **no** version.

rtr reaction-configuration operation-failure

- Use to enable the operation-failure reaction.
- The operation-failure event is triggered when a number of consecutive probe operations are not received or when they are received after a timeout.
- Example

```
host1(config)#rtr reaction-configuration 1
operation-failure 3
```
- There is no **no** version.

rtr reaction-configuration path-change

- Use to enable the path-change reaction.
- The path-change event is triggered when a change is detected in the hop table. At most, there can be one such event per test.
- Example

```
host1(config)#rtr reaction-configuration 1 path-change
```
- There is no **no** version.

rtr reaction-configuration test-completion

- Use to enable test-completion reaction.
- The test-completion event is triggered when a test is completed successfully.
 - › For echo, a successful test means that all probes were sent.
 - › For pathEcho, a successful test means that the destination was reached at least once.
- At most, there can be one such event per test.
- Example

```
host1(config)#rtr reaction-configuration 1 test-completion
```
- There is no **no** version.

rtr reaction-configuration test-failure

- Use to enable test-failure reaction.
- The test-failure event is triggered when a test fails. Failure is determined in the following ways:
 - › If Echo, this event is triggered after testFailureValue probes are either not received or are received after a timeout.
 - › If PathEcho, this event is triggered when the test ends and no responses are received from the destination.
- At most, there can be one such event per test.
- Example

```
host1(config)#rtr reaction-configuration 1 test-failure
```
- There is no **no** version.

Scheduling the Probe

When you have configured the RTR probe, you must schedule the operation to begin collecting statistics and other information on problems that may arise.

rtr schedule

- Use to create an RTR schedule.
- Example

```
host1(config)#rtr schedule 5
```
- Use the **no** version to stop the test. The **no** version stops the probe operation by putting it in the pending state. The **no** version also resets the restart-time attribute and the life attribute.

rtr schedule life

- Use to schedule the test's length.
- Life is a value that depends on the type of the RTR entry; it is not a length of time.

- › If the type is echo, life relates to the number of probes sent until a test finishes.
- › If the type is pathEcho, life relates to the maximum number of hops used by the traceRoute trap.
- Example

```
host1(config)#rtr schedule 5 life 1800
```
- Use the **no** version to stop the test. The **no** version stops the probe operation by putting it in the pending state. The **no** version also resets the life attribute.

rtr schedule restart-time

- Use to restart a test.
- Example

```
host1(config)#rtr schedule 5 restart-time 15
```
- Use the **no** version to stop the test. The **no** version stops the probe operation by putting it in the pending state. The **no** version also resets the restart-time attribute.

rtr schedule start-time

- Use to schedule a test's starting time.
- Example

```
host1(config)#rtr schedule 5 start-time now
```
- Use the **no** version to stop the test. The **no** version stops the probe operation by putting it in the pending state.

Shutting Down the Probe

You can shut down the RTR probe operation.

rtr reset

- Use to shut down the RTR, stop all probe operations, and clear the RTR configuration for the given virtual router.
- We recommend that you use this command only in extremely serious situations, such as problems with the configurations of a number of probe operations.
- Example

```
host1(config)#rtr reset
```

Monitoring RTR

You can monitor RTR by displaying status and configuration information.

show rtr application

- Use to display global information about RTR.
- Field descriptions
 - › numberOfEntries – number of RTR entries according to type
 - › entriesEnabled – RTR entries with administrative status enabled
 - › entriesActive – RTR entries with operational status enabled
- Example

```
host1#show rtr application
```

	numberOfEntries	entriesEnabled	entriesActive
	-----	-----	-----
echo	1	1	1
pathEcho	1	1	1
total	2	2	2

show rtr collection-statistics

- Use to display statistical information for a particular probe operation or for all operations.
- Field descriptions
 - › rtrIndex – index number of the RTR probe
 - › operationsSent – number of probe operations sent
 - › operationsRecvd – number of probe operations received
 - › lastGoodResponse – time when last valid probe operation was received
 - › operStatus – operational status of the probe: enabled, disabled
 - › minRtt – minimum round trip time in milliseconds
 - › maxRtt – maximum round trip time in milliseconds
 - › avgRtt – average round trip time in milliseconds
 - › rttSumSqr – sum of the square of all round trip times in milliseconds
 - › testAttempts – number of times the test ran
 - › testSuccesses – number of times the test ran successfully
 - › currentHop – current hop (TTL) used in the test
 - › currentOperation – current probe operation index sent to the hop

- Example

```
host1#show rtr collection-statistics
```

```
Echo Entries:
```

rtrIndex	operationsSent	operationsRcvd	lastGoodResponse
1	5208	5187	08/30/2000 05:09

rtrIndex	operStatus	minRtt	maxRtt	avgRtt	rttSumSqr
1	enabled	0	1785	3	7109208

```
PathEcho Entries:
```

rtrIndex	testAttempts	testSuccesses	lastGoodResponse
2	156	156	08/30/2000 05:09

rtrIndex	operStatus	currentHop	currentOperation
2	enabled	2	4

show rtr configuration

- Use to display the configuration for a particular probe or for all probes.
- Field descriptions
 - › rtrIndex – index number of the RTR probe
 - › type – probe type: echo, pathEcho
 - › targetAddress – address of the probe's target
 - › reqSize – protocol data size in the request packet
 - › freq – rate in seconds that the RTR probe uses to start a response time operation
 - › life – length of the test
 - › source – interface from which the probe is sent
 - › restartTime – restart time of the test in seconds
 - › owner – owner of the probe
 - › samples – maximum number of entries saved in the history table for this RTR probe
 - › admin – administrative status of the probe: enabled, disabled
 - › tos – setting of the type of service (ToS) byte in the probe's IP header
 - › reactionConfiguration – RTR reactions that are configured for the probe
 - › operFail – operation failure event is triggered when this number of consecutive probe operations is not received or when the operations are received after a timeout

- › testFail – test failure event is triggered when this number of probe operations is not received or when the operations are received after a timeout
- › timeout – time in milliseconds that the probe waits for a response
- › tag – identifier configured for the probe
- › operPerHop – number of RTR probe operations sent to a given hop
- › maxFail – maximum number of consecutive failures to respond to a probe's request. When the maximum number is reached, the test stops. Applies only to pathEcho entries.
- › hopKpt – number of hops per path for which statistics are collected. When this number is reached, no additional statistical information on the path is stored. Applies only to pathEcho entries.

- Example

host1#show rtr configuration

```

rtrIndex      type      targetAddress reqSize freq  life
-----
      1      echo      10.5.0.200    1    1    20
      2  pathEcho  10.5.0.11    1    1    30

rtrIndex      source                restartTime  owner
-----
1              fastEthernet0/0          10
2
rtrIndex      samples  admin   tos  reactionConfiguration
-----
      1         5  enabled  0
      2         5  enabled  0

rtrIndex      operFail  testFail  timeout  tag
-----
      1         1         1      10000

rtrIndex      operPerHop  maxFail  hopKpt  tag
-----
2              5           3       16

```

show rtr history

- Use to display history (data samples) for a particular probe or for all probes.
- Field descriptions
 - › rtrIndex – index number of the RTR probe
 - › operation – index number of the probe operation
 - › rtt – round trip time in milliseconds
 - › statusDescription
 - concurrentLimitFail – target already being used by another rtrIndex
 - ifInactiveToTarget – interface used to reach target is not operational

- invalidHostAddress – target address is not supported
- noRouteToTarget – target address is not reachable
- responseReceived – probe operation replied by target
- requestTimedOut – probe operation not replied to by target or reply received after timeout
- unknownDestAddress – target address is invalid
- unableToResolveName – target address could not be looked up
- › timeStamp – date and time when the RTR entry was created
- › test – index number of the pathEcho test
- › hop – index number of the hop count
- › operation – index number of the probe operation
- › rtt – round trip time in milliseconds
- › timeStamp – date and time the entry was created
- › address – address of router at the hop
- Example

host1#show rtr history

Echo Entries:

rtrIndex	operation	rtt	statusDescription	timeStamp
1	5476	0	responseReceived	08/30/2000 05:17
1	5477	0	responseReceived	08/30/2000 05:17
1	5478	0	responseReceived	08/30/2000 05:17
1	5479	0	responseReceived	08/30/2000 05:17
1	5480	0	responseReceived	08/30/2000 05:17

PathEcho Entries:

rtrIndex	test	hop	operation	rtt	statusDescription
2	165	3	5	0	responseReceived
2	165	3	1	0	responseReceived
2	165	3	2	0	responseReceived
2	165	3	3	0	responseReceived
2	165	3	4	0	responseReceived

rtrIndex	test	hop	operation	timeStamp	address
2	165	3	5	08/30/2000 20:39	10.5.0.11
2	165	3	1	08/30/2000 20:40	10.5.0.11
2	165	3	2	08/30/2000 20:40	10.5.0.11
2	165	3	3	08/30/2000 20:40	10.5.0.11
2	165	3	4	08/30/2000 20:40	10.5.0.11

show rtr hops

- Use to display RTR hops information.
- Field descriptions
 - › rtrIndex – index number of the RTR probe
 - › hop – index number of the hop count
 - › address – address of the router at the hop
 - › minRtt – minimum round trip time in milliseconds
 - › maxRtt – maximum round trip time in milliseconds
 - › avgRtt – average round trip time in milliseconds
 - › rttSumSqr – sum of the square of all round trip times in milliseconds
 - › operationsSent – number of probe operations sent
 - › operationsRcvd – number of probe operations received
 - › lastGoodResponse – time when last valid probe operation was received
- Example

host1#**show rtr hops**

rtrIndex	hop	address	minRtt	maxRtt	avgRtt	rttSumSqr
2	1	192.168.1.1	1	276	1	955363
2	2	10.2.0.3	0	1109	2	10094451

rtrIndex	hop	operationsSent	operationsRcvd	lastGoodResponse
2	1	36985	36838	09/18/2000 20:20
2	2	30717	21494	09/18/2000 20:20

show rtr operational-state

- Use to display RTR operational information.
- Field descriptions
 - › rtrIndex – index number of the RTR probe
 - › type – type of RTR probe: echo, pathEcho
 - › entryStatus – if the entry was created via the SNMP DISMAN MIB, the row may be partially constructed; if that is the case, the CLI displays notReady as the entry's status
 - › adminStatus – derived from the **rtr schedule start-time** command; if the option is **now**, the status is enabled; if the option is **pending**, the status is disabled
 - › operStatus – enabled only if entryStatus and adminStatus are enabled and the test is running; operStatus remains enabled if the test finishes and restart time is not 0

- Example

```
host1#show rtr operational-state
rtrIndex      type      entryStatus  adminStatus  operStatus
-----      -
1      echo      active       enabled      enabled
2      pathEcho  active       enabled      enabled
```

Monitoring IP

This section shows how to set a statistics baseline and use the **show** commands to view your IP configuration and monitor IP interfaces and statistics.

Establishing a Baseline

IP statistics are stored in system counters. The only way to reset the system counters is to reboot the system. You can, however, establish a baseline for IP statistics by setting a group of reference counters to zero.

baseline ip

- Sets a statistics baseline for IP statistics.
- The system implements the baseline by reading and storing the statistics at the time the baseline is set and then subtracting this baseline whenever baseline-relative statistics are retrieved.
- Use the **delta** keyword with IP **show** commands to specify that baselined statistics are to be shown.
- Example

```
host1#baseline ip
```

- There is no **no** version.

baseline ip tcp

- Sets a statistics baseline for TCP statistics.
- The system implements the baseline by reading and storing the statistics at the time the baseline is set and then subtracting this baseline whenever baseline-relative statistics are retrieved.
- Use the **delta** keyword with IP **show** commands to specify that baselined statistics are to be shown.
- Example

```
host1#baseline ip tcp
```

- There is no **no** version.

baseline ip udp

- Sets a statistics baseline for UDP statistics.
- The system implements the baseline by reading and storing the statistics at the time the baseline is set and then subtracting this baseline whenever baseline-relative statistics are retrieved.
- Use the **delta** keyword with IP **show** commands to specify that baselined statistics are to be shown.
- Example

```
host1#baseline ip udp
```
- There is no **no** version.

IP show Commands

You can monitor the following aspects of IP using **show ip** commands:

To Display	Command
Access lists	show access-list show ip as-path-access-list
ARP	show arp
General IP information	show ip
IP addresses	show ip address
Community lists	show ip community-list
Prefix for the names of dynamic IP shared interfaces	show ip dynamic-interface-prefix
Routing table	show ip forwarding-table slot
Interfaces	show ip interface
Shared IP interfaces	show ip interface shares
Subscriber interfaces	show ip demux interface
Protocols	show ip protocols
Redistribution policies	show ip redistribute
Routes	show ip route
Interfaces and next hops	show ip route slot
Static routes	show ip static
TCP statistics	show ip tcp statistics
Traffic	show ip traffic
UDP statistics	show ip udp statistics
Profiles	show ip profile
Route maps	show route-map

To set a statistics baseline for IP interfaces, use the **baseline ip tcp** and **baseline ip udp** commands. Use the **delta** keyword with **IP show** commands to specify that baselined statistics are to be shown.

You can use the output filtering feature of the **show** command to include or exclude lines of output based on a text string that you specify. See *ERX System Basics Configuration Guide, Chapter 2, Command Line Interface*, for details.

show access-list

- Use to display information about access lists, including the instances of each access list.
- Example

```
host1#show access-list
IP Access List 1:
    permit ip 172.31.192.217 0.0.0.0 0.0.0.0 255.255.255.255
    permit ip 12.40.0.0 0.0.0.3 0.0.0.0 255.255.255.255
    deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
IP Access List 2:
    permit ip 172.19.0.0 0.0.255.255 0.0.0.0 255.255.255.255
    deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
IP Access List 10:
    permit ip 0.0.0.0 255.255.255.255 0.0.0.0
    255.255.255.255
IP Access List 11:
    deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
```

show arp

- Use to display information about ARP.
- Field descriptions
 - › Address – IP address of the entry
 - › Age – time to live for this entry in seconds
 - › Hardware Addr – physical (MAC) address of the entry
 - › Interface – interface-specifier of the entry (for example, FastEthernet6/0 is an Ethernet interface on slot 6, port 0)
 - › * – indicates that an ARP entry was added because of an **arp-validate** command, rather than just an **arp** command.
- Example

```
host1#show arp
      Address      Age      Hardware Addr  Interface
172.31.192.217    21340    00d0.58f2.67e0  loopback1
      192.168.1.0    20730    00e0.09ed.5312  fastEthernet6/0 *
      192.168.1.1    12550    00e0.b06a.4c75  fastEthernet6/0 *
      192.168.1.217  21600    0090.1a00.0230  fastEthernet6/0 *
      192.168.1.255  21600    00f0.c2d1.1200  fastEthernet6/0 *
```

```

12.40.0.2    24320  0020.6393.4233 atm5/0.1
172.18.2.1   21600  0020.bed2.8738 atm5/1.1
172.18.2.2   21600  0020.5b91.60f2 atm5/1.1
172.31.192.206 21600  00d0.43b5.1032 atm5/1.1

```

show ip

- Use to display general information on IP.
- Example

```

host1#show ip
IP Router Id: 192.168.1.155
Router Name: default
Default TTL: 60
Reassemble Timeout: 30

```

show ip as-path-access-list

- Use to display information about AS-path access lists.
- Example

```

host1#show ip as-path-access-list
AS Path Access List 1:
  permit .*
AS Path Access List 2:
  deny .*
AS Path Access List 3:
  permit _109_
  deny .*
AS Path Access List 4:
  permit _109$
  deny .*
AS Path Access List 10:
  deny _109$
  permit ^108_
  deny .*

```

show ip address

- Use to display detailed or summary information on a particular IP interface.
- Field descriptions
 - › Network Protocols – network protocols configured on this interface
 - › Internet address – IP address and subnet mask of this interface
 - › Broadcast address – broadcast address of this interface
 - › Operational MTU – MTU of this interface
 - › Administrative MTU – value of the MTU if it has been administratively overridden using the configuration
 - › Operational speed – speed of the interface

- › Administrative speed – value of the speed if it has been administratively overridden using the configuration
- › Discontinuity Time – value of the SysUpTime when the interface statistics last started being valid
- › Router advertisement – status of router discovery advertisement: enabled, disabled
- › Proxy Arp – status of the feature: enabled, disabled
- › Administrative debounce-time – configured time in milliseconds that an interface event has to be in the same state before being reported
- › Operational debounce-time – time in milliseconds that an interface event has to be in the same state before being reported
- › Access routing – access route addition: enabled, disabled
- › Multipath mode – equal cost multipath mode method: hashed, round robin
- › In Received Packets, Bytes – total number of packets and bytes received on this interface
 - Unicast Packets, Bytes – unicast packets and bytes received on the IP interface; link-local received multicast packets (non-multicast-routed frames) are counted as unicast packets
 - Multicast Packets, Bytes – multicast packets and bytes received on the IP interface which are then multicast-routed are counted as multicast packets
- › In Policed Packets, Bytes – packets and bytes that were received and dropped because of rate limits
- › In Error Packets – number of packets received with errors
- › In Invalid Source Address Packets – packets received with invalid source address (for example, spoofed packets)
- › In Discarded Packets – packets received that were discarded for reasons other than rate limits, errors, and invalid source address
- › Out Forwarded Packets, Bytes – total number of packets and bytes that were sent from this interface
 - Unicast Packets, Bytes – unicast packets and bytes that were sent from this interface
 - Multicast Routed Packets, Bytes – multicast packets and bytes that were sent from this interface
- › Out Scheduler Drops Committed Packets, Bytes – outgoing packets and bytes dropped by the scheduler even though they had a committed traffic contract
- › Out Scheduler Drops Conformed Packets, Bytes – outgoing packets and bytes dropped by the scheduler even though they conformed to the traffic contract
- › Out Scheduler Drops Exceeded Packets, Bytes – outgoing packets and bytes that were dropped by the scheduler because they exceeded the contract
- › Out Policed Packets, Bytes – outgoing packets and bytes dropped because of rate limiters

- › Out Discarded Packets – outgoing packets that were discarded for reasons other than those dropped by the scheduler and those dropped because of rate limits
- Example

```

host1#show ip address 10.6.136.73
fastEthernet0/0 is up, line protocol is up
  Network Protocols: IP
  Internet address is 10.6.136.73/255.255.128.0
  Broadcast address is 255.255.255.255
  Operational MTU = 0   Administrative MTU = 0
  Operational speed = 1   Administrative speed = 0
  Discontinuity Time = 5766
  Router advertisement = disabled
  Proxy Arp = disabled
  Administrative debounce-time = 10 mSecs
  Operational debounce-time   = disabled
  Access routing = disabled
  Multipath mode = hashed

  In Received Packets 2849, Bytes 759428
    Unicast Packets 2849, Bytes 759428
    Multicast Packets 0, Bytes 0
  In Policed Packets 0, Bytes 0
  In Error Packets 0
  In Invalid Source Address Packets 0
  In Discarded Packets 0
  Out Forwarded Packets 1866, Bytes 84650
    Unicast Packets 1866, Bytes 84650
    Multicast Routed Packets 0, Bytes 0
  Out Scheduler Drops Committed Packets 0, Bytes 0
  Out Scheduler Drops Conformed Packets 0, Bytes 0
  Out Scheduler Drops Exceeded Packets 0, Bytes 0
  Out Policed Packets 0, Bytes 0
  Out Discarded Packets 0

```

show ip community-list

- Use to display routes that are permitted by a BGP community list.
- Example

```

host1#show ip community-list
Community List 1:
  permit 752877569 (11488:1)
  permit 752877570 (11488:2)
  permit 752877571 (11488:3)
  permit 752877572 (11488:4)
Community List 2:
  permit 4294967043 (local-as)

```

show ip demux interface

- Use to display information about subscriber interfaces.
- Field descriptions
 - › Prefix/Length – source or destination addresses that the subscriber interface demultiplexes
 - › SA/DA – demultiplexing method for subscriber interface
 - SA – source address
 - DA – destination address
 - › Subscriber-Intf – name of shared interface on which subscriber interface is configured
 - › VR/VRF – name of VR or VRF on which the subscriber interface is configured
 - › Description – text description for the IP interface on which subscriber interface is configured (added with the **description** command)
- Example

```
host1#show ip demux interface fastEthernet 2/0
Prefix/Length      SA/DA  Subscriber-Intf  VR/VRF      Description
132.2.2.2/32      SA     ip subsc1       3            subsc1@test
132.2.2.3/32      SA     ip subsc2       3            subsc2@test
132.2.2.4/32      SA     ip subsc3       3            subsc3@test
132.2.2.5/32      SA     ip subsc4       3            subsc4@test
```

show ip dynamic-interface-prefix

- Use to display the prefix for the names of dynamic IP shared interfaces.
- Example

```
host1#show ip dynamic-interface-prefix
Dynamic Interface Prefix: dyn
```

show ip forwarding-table slot

- Use to display details on the routing table for a specific line module, including the memory used by each virtual router configured on the line module and free memory available on the module.
- The Load Errors field records any failed routing table distribution attempt as an error. Attempts can fail for many reasons during normal operation; a failed attempt does not necessarily indicate a problem. It is normal to see many Load Errors per day.
- If the Status field does not indicate Valid, then the routing table distribution has failed constantly for that VR. It is normal and appropriate behavior for the Status field to indicate Valid while the Load Error field increases daily.

- Field descriptions
 - › Free Memory – amount of routing table memory free on the line module, in kilobytes
 - › Virtual Router – name of the virtual router(s) configured on the line module
 - › Memory (KB) – amount of routing table memory consumed by the virtual router, in kilobytes
 - › Load Errors – count of errors made while loading the routing table on the line module
 - › Status – indicates whether the routing table for the virtual router is valid
- Example

```
host1#show ip forwarding-table slot 9
```

```
Free Memory = 3,166KB
```

Virtual Router	Memory (KB)	Load Errors	Status
vr1	4128	0	Valid
vr2	3136	0	Valid
vr3	2256	0	Valid
vr4	1512	0	Valid
default	1024	0	Valid

show ip interface

- Use to display the current state of all IP interfaces or the IP interfaces you specify.
- The default is all interface types and all interfaces.
- Field descriptions
 - › interface – interface type and interface specifier
 - › interface status – status of the interface
 - › line protocol – status of the line protocol
 - › Link up/down trap – status of SNMP link up/down traps on the interface
 - › Internet address – IP address of the interface
 - › IP Statistics Rcvd:
 - local destination – frames with this router as their destinations
 - hdr errors – number of packets containing header errors
 - addr errors – number of packets containing addressing errors
 - unkn proto – number of packets received containing unknown protocols
 - discards – number of discarded packets
 - › IP Statistics Frags:
 - reasm ok – number of reassembled packets
 - reasm req – number of requests for reassembly
 - reasm fails – number of reassembly failures
 - frag ok – number of packets fragmented successfully

- frag req – number of frames requiring fragmentation
- frag fails – number of packets unsuccessfully fragmented
- › IP Statistics Sent:
 - generated – number of packets generated
 - no routes – number of packets that could not be routed
 - discards – number of packets that could not be routed that were discarded
- › ICMP Statistics Rcvd:
 - errors – error packets received
 - dst unreach – packets received with destination unreachable
 - time exceed – packets received with time-to-live exceeded
 - param probs – packets received with parameter errors
 - src quench – source quench packets received
 - redirect – receive packet redirects
 - echo req – echo request (PING) packets
 - echo rpy – echo replies received
 - timestamp req – requests for a timestamp
 - timestamp rpy – replies of timestamp requests
 - addr mask req – mask requests sent
 - addr mask rpy – mask replies sent
- › ICMP Statistics Sent:
 - errors – error packets sent
 - dst unreach – packets sent with destination unreachable
 - time excd – packets sent with time-to-live exceeded
 - param probs – packets sent with parameter errors
 - src quench – source quench packets sent
 - redirect – send packet redirects
 - timestamp req – requests for a timestamp
 - timestamp rpy – replies to timestamp requests
 - addr mask req – address mask requests
 - addr mask rpy – address mask replies
- › In Received Packets, Bytes – total number of packets and bytes received on the IP interface
 - Unicast Packets, Bytes – unicast packets and bytes received on the IP interface; link-local received multicast packets (non-multicast-routed frames) are counted as unicast packets
 - Multicast Packets, Bytes – multicast packets and bytes received on the IP interface which are then multicast-routed are counted as multicast packets
- › In Forwarded Packets, Bytes – packets and bytes forwarded into an output IP interface

- › In Total Dropped Packets, Bytes – total number of packets and bytes that were dropped on the interface; sum of all the drop reasons indented below this field
 - In Policed Packets – packets discarded on a receive IP interface due to token bucket limiting
 - In Invalid Source Address Packets – packets discarded on a receive IP interface due to invalid IP source address (sa-validate enabled)
 - In Error Packets – packets discarded on a receive IP interface due to IP header errors
 - In Discarded Packets – packets discarded on the ingress interface due to a configuration problem rather than a problem with the packet itself
 - In Fabric Dropped Packets – packets discarded on a receive IP interface due to internal fabric congestion
- › Out Forwarded Packets, Bytes – total number of packets and bytes forwarded out the IP interface
 - Unicast Packets, Bytes – unicast packets and bytes forwarded out the IP interface
 - Multicast Routed Packets, Bytes – multicast packets and bytes forwarded out the IP interface
- › Out Requested Packets, Bytes – packets and bytes requested to be forwarded out an IP interface
- › Out Total Dropped Packets, Bytes – total number of packets and bytes that were discarded on the egress interface; sum of all the drop reasons indented below this field
 - Out Scheduler Drops Committed Packets, Bytes – packets and bytes dropped by the scheduler even though they had a committed traffic contract
 - Out Scheduler Drops Conformed Packets, Bytes – packets and bytes dropped by the scheduler even though they conformed to the traffic contract
 - Out Scheduler Drops Exceeded Packets, Bytes – packets and bytes dropped by the scheduler because they exceeded the contract
 - Out Policed Packets – packets discarded on the egress interface due to rate limiting
 - Out Discarded Packets – packets discarded on the egress interface due to a configuration problem rather than a problem with the packet itself
 - Out Fabric Dropped Packets – packets dropped due to internal fabric congestion
- Example 1

```
host1#show ip interface detail FastEthernet 0/0
FastEthernet0/0 is up, line protocol is up
Link up/down trap is disabled
```

```
Internet address is 1.1.1.2/255.255.255.0
IP statistics:
  Rcvd:  0 local destination
        0 hdr errors, 0 addr errors
        0 unkn proto, 0 discards
  Frags: 0 reasm ok, 0 reasm req, 0 reasm fails
        0 frag ok, 0 frag creates, 0 frag fails
  Sent:  31656835 generated, 0 no routes, 0 discards
ICMP statistics:
  Rcvd:  0 errors, 0 dst unreachable, 0 time exceed
        0 param probs, 0 src quench, 0 redirect,
        0 echo req, 31656816 echo rpy
        0 timestmp req, 0 timestmp rpy
        0 addr mask req, 0 addr mask rpy
  Sent:  0 errors, 0 dst unreachable, 0 time excd
        0 param probs, 0 src qnch, 0 redirect
        0 timestamp req, 0 timestamp rpy
        0 addr mask req, 0 addr mask rpy

In Received Packets 246220, Bytes 344624800
  Unicast Packets 246162, Bytes 344621410
  Multicast Packets 58, Bytes 3390
In Forwarded Packets 245464, Bytes 343566400
In Total Dropped Packets 756, Bytes 1058400
  In Policed Packets 756
  In Invalid Source Address Packets 0
  In Error Packets 0
  In Discarded Packets 0
  In Fabric Dropped Packets 0

Out Forwarded Packets 117, Bytes 87297
  Unicast Packets 117, Bytes 87297
  Multicast Routed Packets 0, Bytes 0
Out Requested Packets 117, Bytes 87297
Out Total Dropped Packets 0, Bytes 0
  Out Scheduler Drops Committed Packets 0, Bytes 0
  Out Scheduler Drops Conformed Packets 0, Bytes 0
  Out Scheduler Drops Exceeded Packets 0, Bytes 0
  Out Policed Packets 0
  Out Discarded Packets 0
  Out Fabric Dropped Packets 0
```

If you are losing packets because of fabric congestion, you can use the *In Fabric Dropped Packets* and *Out Fabric Dropped Packets* statistics to help determine the location of the bottleneck. Both statistics count the same

thing—the same packets dropped because of fabric congestion—but in different directions.

At any given time, the total number of packets dropped in the fabric for all interfaces in the chassis is equal to the sum of all *In Fabric Dropped Packets* for all interfaces in the chassis, which equals the sum of all *Out Fabric Dropped Packets* for all interfaces in the chassis.

Packets not dropped for another listed reason are considered to have been dropped in the fabric. The system calculates *In Fabric Dropped Packets* by subtracting the total number of inbound packets dropped for all other reasons from the *In Total Dropped Packets* number. The system calculates *Out Fabric Dropped Packets* by subtracting the total number of outbound packets dropped for all other reasons from the *Out Total Dropped Packets* number.

The system calculates *In Total Dropped Packets* by subtracting *In Forwarded Packets* from *In Received Packets*. The system calculates *Out Total Dropped Packets* by subtracting *Out Forwarded Packets* from *Out Received Packets*. These statistics are reported as traffic is moving through the system. It is possible to get false statistics based on packets being forwarded and/or received after polling and based on which of the statistics is reported first. For example, *In Forwarded Packets* could be reported as greater than *In Received Packets*. Rather than displaying *In Total Dropped Packets* as a negative value, the command displays it as the sum of all drop reasons other than fabric drops; fabric drops are reported as 0, but might actually be nonzero. If you halt traffic, the *In Total Dropped Packets* and *Out Total Dropped Packets* values are always correct.

show ip interface shares

- Use to display information about shared IP interfaces.
- If you specify an IP interface specifier, the command displays information only for that interface and any shared IP interfaces associated with it.
- Field descriptions
 - › Interface – interface specifier or name of the interface
 - › IP-Address – IP address associated with the interface
 - › Status – operational state of the interface
 - › Protocol – state of the protocol running on the interface
 - › Virtual Router – virtual router in which the interface is configured

- Example 1

```
host1#show ip interface shares brief
```

Interface	IP-Address	Status	Protocol	Virtual Router
null0	255.255.255.255/32	up	up	
fastEthernet0/0	10.13.5.17/24	up	up	
loopback100	202.1.1.1/24	up	up	
atm4/0.1	10.1.1.1/24	up	up	
ip si0	Unnumbered	up	up	vr-a
ip si1	Unnumbered	up	up	vr-b:vrf-1

- Example 2

```
host1#show ip interface shares brief atm 4/0.1
```

Interface	IP-Address	Status	Protocol	Virtual Router
atm4/0.1	10.1.1.1/24	up	up	
ip si0	Unnumbered	up	up	vr-a
ip si1	Unnumbered	up	up	vr-b:vrf-1

- Example 3 – for a description of the following fields, see the **show ip address** command

```
host1#show ip interface shares atm 4/0.1
```

```
atm4/0.1 is up, line protocol is up
```

```
Network Protocols: IP
```

```
Unnumbered Interface on loopback100
```

```
( IP address 202.1.1.1 )
```

```
Operational MTU = 1500 Administrative MTU = 0
```

```
Operational speed = 155520000 Administrative speed = 0
```

```
Discontinuity Time = 0
```

```
Router advertisement = disabled
```

```
Administrative debounce-time = disabled
```

```
Operational debounce-time = disabled
```

```
Access routing = disabled
```

```
Multipath mode = hashed
```

```
In Received Packets 120, Bytes 12000
```

```
Unicast Packets 60, Bytes 6000
```

```
Multicast Packets 60, Bytes 6000
```

```
In Policed Packets 0, Bytes 0
```

```
In Error Packets 0
```

```
In Invalid Source Address Packets 0
```

```
Out Forwarded Packets 101, Bytes 5252
```

```
Unicast Packets 101, Bytes 5252
```

```
Multicast Routed Packets 0, Bytes 0
```

```
Out Scheduler Drops Committed Packets 0, Bytes 0
```

```
Out Scheduler Drops Conformed Packets 0, Bytes 0
```

```
Out Scheduler Drops Exceeded Packets 0, Bytes 0
Out Policed Packets 0, Bytes 0
```

```
ip si0 is up, line protocol is up
Network Protocols: IP
Virtual Router vr-a
Layer 2 interface atm4/0.1
Unnumbered Interface on loopback100
( IP address 202.1.1.1 )
Operational MTU = 1500 Administrative MTU = 0
Operational speed = 155520000 Administrative speed = 0
Discontinuity Time = 0
Router advertisement = disabled
Administrative debounce-time = disabled
Operational debounce-time = disabled
Access routing = disabled
Multipath mode = hashed
```

```
In Received Packets 0, Bytes 0
  Unicast Packets 0, Bytes 0
  Multicast Packets 0, Bytes 0
In Policed Packets 0, Bytes 0
In Error Packets 0
In Invalid Source Address Packets 0
Out Forwarded Packets 101, Bytes 5252
  Unicast Packets 101, Bytes 5252
  Multicast Routed Packets 0, Bytes 0
Out Scheduler Drops Committed Packets 0, Bytes 0
Out Scheduler Drops Conformed Packets 0, Bytes 0
Out Scheduler Drops Exceeded Packets 0, Bytes 0
Out Policed Packets 0, Bytes 0
```

```
ip si1 is up, line protocol is up
Network Protocols: IP
Virtual Router vr-b:vrf-1
Layer 2 interface atm4/0.1
.
.
.
Out Policed Packets 0, Bytes 0
```

- **Example** – You can also display statistics for shared IP interfaces with the **show ip interface** command:

```
host1#show ip interface ip si0
ip0 is up, line protocol is up
Network Protocols: IP
Layer 2 interface atm4/0.1
```

```
Unnumbered Interface on loopback100
( IP address 202.1.1.1 )
Operational MTU = 1500 Administrative MTU = 0
Operational speed = 155520000 Administrative speed = 0
Discontinuity Time = 0
Router advertisement = disabled
Administrative debounce-time = disabled
Operational debounce-time = disabled
Access routing = disabled
Multipath mode = hashed

In Received Packets 0, Bytes 0
  Unicast Packets 0, Bytes 0
  Multicast Packets 0, Bytes 0
In Policed Packets 0, Bytes 0
In Error Packets 0
In Invalid Source Address Packets 0
Out Forwarded Packets 101, Bytes 5252
  Unicast Packets 101, Bytes 5252
  Multicast Routed Packets 0, Bytes 0
Out Scheduler Drops Committed Packets 0, Bytes 0
Out Scheduler Drops Conformed Packets 0, Bytes 0
Out Scheduler Drops Exceeded Packets 0, Bytes 0
Out Policed Packets 0, Bytes 0
```

show ip profile

- Use to display information about a specific IP profile.
- Field descriptions
 - › IP profile – profile name
 - › IP address – IP address and subnet mask of the interface or none if the interface is unnumbered
 - › Unnumbered interface – specifier for the unnumbered interface or none if the interface is numbered
 - › Router – router name
 - › Directed Broadcast – enabled or disabled
 - › ICMP Redirects – enabled or disabled
 - › Access Route Addition – enabled or disabled
 - › Source-Address Validation – enabled or disabled
 - › Ignore DF Bit – enabled or disabled
 - › Administrative MTU – MTU size

- Example

```

host1#show ip profile foo
IP profile : foo
IP address : none
Unnumbered interface : none
Router :
Directed Broadcast : Enabled
ICMP Redirects : Disabled
Access Route Addition : Enabled
Source-Address Validation : Enabled
Ignore DF Bit : Disabled
Administrative MTU : 0

```

show ip protocols

- Use to display configured protocols.
- Field descriptions
 - › For BGP:
 - Redistributing – protocol to which BGP is redistributing routes
 - Default local preference – local preference value
 - IGP synchronization – status of IGP synchronization: enabled, disabled
 - Always compare MED – status of multiexit discrimination: enabled, disabled
 - Router flap damping – status of route dampening: enabled, disabled
 - Administrative Distance – external, internal, and local administrative distances
 - Neighbor Address – IP address of the BGP neighbor
 - Neighbor Incoming/Outgoing update distribute list – number of the access list for outgoing routes
 - Neighbor Incoming/Outgoing update prefix list – number of the prefix list for incoming or outgoing routes
 - Neighbor Incoming/Outgoing update prefix tree – number of the prefix tree for incoming or outgoing routes
 - Neighbor Incoming/Outgoing update filter list – number of filter list for incoming routes
 - Routing for Networks – network for which BGP is currently injecting routes
 - › For IS-IS:
 - System Id – 6-byte value of the system
 - IS-Type – routing type of the router: Level 1, Level 2
 - Distance – administrative distance for IS-IS learned routes
 - Address Summarization – aggregate addresses defined in the routing table for multiple groups of addresses at a given level or routes learned from other routing protocols
 - Routing for Networks – network for which IS-IS is currently injecting routes

- › For OSPF:
 - Router ID – OSPF process ID for the router
 - Distance – administrative distance for OSPF learned routes
 - Redistributing – protocol to which OSPF is redistributing routes
 - Address Summarization – aggregate addresses defined in the routing table for multiple groups of addresses at a given level or routes learned from other routing protocols
 - Routing for Networks – network for which OSPF is currently injecting routes
- › For RIP:
 - Router Administrative State – RIP protocol state. Enable means that it is allowed to send and receive updates. Disable means that it may be configured but it is not allowed to run yet.
 - System Version – RIP versions allowed for sending and receiving RIP updates. The system version is currently set to RIP1, which sends RIP version 1 but will receive version 1 or 2. If the version is set to RIP2, the system will send and receive version 2 only. The default is configured for RIP1.
 - Update interval – current setting of the update timer (in seconds)
 - Invalid after – current setting of the invalid timer (in seconds)
 - hold down time – current setting of the hold down timer (in seconds)
 - flushed interval – current setting of the flush timer (in seconds)
 - Filter applied to outgoing route update – access list applied to outgoing RIP route updates
 - Filter applied to incoming route update – access list applied to incoming RIP route updates
 - Global route map – route map that specifies all RIP interfaces on the system
 - Distance – value added to RIP routes added to the IP routing table. The default is 120.
 - Interface – interface type on which RIP protocol is running
 - Redistributing – protocol to which RIP is redistributing routes
 - Routing for Networks – network for which RIP is currently injecting routes
- Example

```
host1#show ip protocols
Routing Protocol is "bgp 100"
  Redistributing: ospf
  Default local preference is 100
  IGP synchronization is enabled
  Always compare MED is disabled
  Router flap damping is disabled
  Administrative Distance: external 20 internal 200 local 200
```

```
Neighbor(s):  
  Address 1.1.1.1  
  Outgoing update distribute list is 2  
  Outgoing update prefix list is efg  
  Incoming update prefix tree is abc  
  Incoming update filter list is 1  
Routing for Networks:  
  192.168.1.0/24
```

```
Routing Protocol is "isis isisOne"  
  System Id: 0000.0000.0011.00  IS-Type: level-1-2  
  Distance: 115  
  Address Summarization:  
    None  
  Routing for Networks:  
    fastEthernet0/0
```

```
Routing Protocol is "ospf 1" with Router ID 192.168.1.151  
  Distance is 110  
  Redistributing: isis  
  Address Summarization:  
    None  
  Routing for Networks:  
    192.168.1.0/255.255.255.0 area 0.0.0.0
```

```
Routing Protocol is "rip"  
  Router Administrative State: enable  
  System version RIP1: send = 1, receive = 1 or 2  
  Update interval: 30 seconds  
  Invalid after: 180 seconds  
  hold down time: 120 seconds  
  flushed interval: 300 seconds  
  Filter applied to outgoing route update is not set  
  Filter applied to incoming route update is not set  
  No global route map  
  Distance is 120  
  Interface      Tx    Rx    Auth  
  fastEthernet0/0  1    1,2  none  
  Redistributing: ospf  
  Routing for Networks:  
    192.168.1.0/255.255.255.0
```

show ip redistribute

- Use to display configured route redistribution policy.
- Field descriptions
 - › To – protocol that routes are distributed into
 - › From – protocol that routes are distributed from
 - › status – redistribution status
 - › route map number – number of the route map
- Example


```
host1#show ip red
To ospf, From static is enabled with route map 4
To ospf, From connected is enabled with route map 3
```

show ip route

- Use to display the current state of the routing table, including routes not used for forwarding.
- You can display all routes, a specific route, best route to a resolved domain name, all routes beginning with a specified address, routes for a particular protocol (BGP, IS-IS, OSPF, or RIP), locally connected routes, internal control routes, static routes, or summary counters for the routing table.
- Field descriptions
 - › Type – protocol type
 - › Prefix – IP address prefix
 - › Length – prefix length
 - › Next Hop – IP address of the next hop to the route, whether it is a local interface or another router
 - › Dist – administrative distance for the route; see Table 2-4
 - › Met – number of hops
 - › Intf – interface type and interface specifier
- Example 1

```
host1#show ip route
```

```
Protocol/Route type codes:
```

```
I1- ISIS level 1, I2- ISIS level2,
I- route type intra, IA- route type inter, E- route type external,
i- metric type internal, e- metric type external,
O- OSPF, E1- external type 1, E2- external type2,
N1- NSSA external type1, N2- NSSA external type2
```

Prefix/Length	Type	Next Hop	Dist/Met	Intf
-----	----	-----	-----	-----
172.16.2.0/24	Bgp	192.168.1.102	20/1	fastEthernet0/0
10.10.0.112/32	Static	192.168.1.1	1/1	fastEthernet0/0
10.1.1.0/24	Connect	10.1.1.1	0/1	atm3/0.100

- Example 2

```

host1#show ip route static
Protocol/Route type codes:
  I1- ISIS level 1, I2- ISIS level2,
  I- route type intra, IA- route type inter, E- route type external,
  i- metric type internal, e- metric type external,
  O- OSPF, E1- external type 1, E2- external type2,
  N1- NSSA external type1, N2- NSSA external type2

```

```

Prefix/Length  Type   Next Hop   Dist/Met  Intf
-----
10.10.0.112/32  Static 192.168.1.1  1/1      fastEthernet0/0

```

- Example 3

```

host1#show ip route summary
5 total routes, 720 bytes in route entries
0 isis routes
0 rip routes
1 static routes
1 connected routes
0 bgp routes
0 ospf routes
3 other internal routes
0 access routes
0 internally created access host routes

Last route added/deleted: 0.0.0.0/0 by StaticLow
At THU MAR 09 2000 05:22:49 UTC

```

- Example 4

```

host1#show ip route all
Protocol/Route type codes:
  I1- ISIS level 1, I2- ISIS level2,
  I- route type intra, IA- route type inter, E- route type external,
  i- metric type internal, e- metric type external,
  O- OSPF, E1- external type 1, E2- external type2,
  N1- NSSA external type1, N2- NSSA external type2

```

```

Prefix/Length  Type   Next Hop   Dist/Met  Intf
-----
0.0.0.0/0      Static 192.168.1.1  1/1      fastEthernet0/0
1.1.1.1/32     I2-E-i 192.168.1.105 115/10   fastEthernet0/0
6.6.6.0/24     Static 192.168.1.1  1/1      fastEthernet0/0
6.33.5.0/24    Static 0.0.0.0      1/1      loopback2
8.8.8.0/24     I2-E-i 192.168.1.105 115/10   fastEthernet0/0
9.9.9.9/32     I2-E-i 192.168.1.105 115/10   fastEthernet0/0
10.0.0.0/8     I2-E-i 192.168.1.105 115/10   fastEthernet0/0
10.10.0.156/32 Static 192.168.1.1  1/1      fastEthernet0/0
11.1.1.1/32    I2-E-i 192.168.1.105 115/10   fastEthernet0/0
11.11.11.12/32 I2-I-i 192.168.1.105 115/10   fastEthernet0/0
22.2.0.0/16    I2-I-i 92.168.1.105 115/10   fastEthernet0/0

```

```

34.0.0.0/8      I2-E-i  192.168.1.105  115/10  fastEthernet0/0
172.20.32.0/24 Static  192.168.1.1    1/1     fastEthernet0/0
174.20.32.0/24 I2-I-i  192.168.1.105  115/20  fastEthernet0/0
176.20.32.0/24 Connect 176.20.32.1    0/1     loopback1
192.168.1.0/24 Connect 192.168.1.214  0/1     fastEthernet0/0
201.1.1.0/24   I2-E-i  192.168.1.105  115/10  fastEthernet0/0
201.2.1.0/24   I2-E-i  192.168.1.105  115/10  fastEthernet0/0
201.3.1.0/24   I2-E-i  192.168.1.105  115/10  fastEthernet0/0
202.1.1.1/32   I2-E-i  192.168.1.105  115/10  fastEthernet0/0
207.1.1.0/24   I2-E-i  192.168.1.105  115/10  fastEthernet0/0

```

show ip route slot

- Use to display the interface and next hop for an IP address in the routing table of a line module.
 - › slotNumber – number of slot containing the line module for which the information is displayed
 - › ipAddress – IP address to look up in the routing table
- A next hop is displayed only for protocols where ARP is used to resolve the addresses, such as for FastEthernet, GigabitEthernet, bridged Ethernet over ATM, and so on.
- Field descriptions
 - › IP address – address reachable via the interface
 - › Interface – interface type and specifier associated with the IP address; displays “Local Interface” if a special interface index is present in the routing table for special IP addresses, such as broadcast addresses
 - › Next Hop – IP address of the next hop router to reach the IP address; displays “---” if no next hop is associated with the IP address
- Example 1

```

host1#show ip route slot 6 10.10.0.231
IP address      Interface      Next Hop
-----
10.10.0.231    fastEthernet 6/0  10.10.0.231

```

- Example 2

```

host1#show ip route slot 9 90.248.1.2
IP address      Interface      Next Hop
-----
90.248.1.2     serial9/23:2   ---

```

- Example 3

```

host1#show ip route slot 9 90.249.255.255
IP address      Interface      Next Hop
-----
90.249.255.255 Local Interface ---

```

show ip static

- Use to display the status of static routes in the routing table.
- You can specify an IP mask that filters specific routes.

- Field descriptions
 - › Prefix – IP address prefix
 - › Length – prefix length
 - › Next Hop – IP address of the next hop
 - › Met – number of hops
 - › Dist – administrative distance of the route; see Table 2-4
 - › Tag – tag value of the route
 - › Intf – interface type and interface specifier
- Example

```

host1#show ip route static
Prefix/Length:  Next Hop:      Met:  Dist:  Tag:  Intf:
10.2.0.0/24     192.168.1.1  1     1     0     ethernet6/0
10.2.1.0/24     192.168.1.1  1     1     1     ethernet6/0
172.31.1.48/32  172.18.2.2   1     1     0     atm5/1.1
  
```

show ip tcp statistics

- Use to display TCP statistics.
- Field descriptions
 - › TCP Global Statistics Connections:
 - attempted – number of outgoing TCP connections attempted
 - accepted – number of incoming TCP connections accepted
 - established – number of TCP connections established
 - › TCP Global Statistics Rcvd:
 - total pkts – total number of packets received
 - in-sequence pkts – number of packets received in sequence
 - bytes – number of bytes received
 - chksum err pkts – number of checksum error packets received
 - authentication err pkts – number of authentication error packets received
 - bad offset pkts – number of bad offset packets received
 - short pkts – number of short packets received
 - duplicate pkts – number of duplicate packets received
 - out of order pkts – number of packets received out of order
 - › TCP Global Statistics Sent:
 - total pkts – total number of packets sent
 - data pkts – number of data packets sent
 - bytes – number of bytes sent
 - retransmitted pkts – number of packets retransmitted
 - retransmitted bytes – number of bytes retransmitted
 - › TCP Session Statistics
 - Local addr – local address of the TCP connection
 - Local port – local port number of the TCP connection
 - Remote addr – remote address of the TCP connection

- Remote port – remote port number of the TCP connection
- State – current state of the TCP connection
- Authentication – authentication status of the TCP connection
- › TCP Session Statistics Rcvd:
 - total pkts – total number of packets received on the TCP connection
 - in-sequence pkts – number of packets received in sequence on the TCP connection
 - bytes – number of bytes received on the TCP connection
 - chksum err pkts – number of checksum error packets received on the TCP connection
 - bad offset pkts – number of bad offset packets received on the TCP connection
 - short pkts – number of short packets received on the TCP connection
 - duplicate pkts – number of duplicate packets received on the TCP connection
 - out of order pkts – number of packets received out of order on the TCP connection
- › TCP Session Statistics Sent:
 - total pkts – total number of packets sent on the TCP connection
 - data pkts – number of data packets sent on the TCP connection
 - bytes – number of bytes sent on the TCP connection
 - retransmitted pkts – number of packets retransmitted on the TCP connection
 - retransmitted bytes – number of bytes retransmitted on the TCP connection
- Example

```
host1#show ip tcp statistics
```

```
TCP Global Statistics:
```

```
Connections: 7358 attempted, 4 accepted, 7362 established  
             0 dropped, 14718 closed
```

```
Rcvd: 75923 total pkts, 53608 in-sequence pkts, 3120303  
      bytes
```

```
         0 chksum err pkts, 0 authentication err pkts, 0 bad  
offset pkts
```

```
         0 short pkts, 0 duplicate pkts, 0 out of order pkts
```

```
Sent: 82352 total pkts, 44404 data pkts, 657095 bytes  
      34 retransmitted pkts, 487 retransmitted bytes
```

```
TCP Session Statistics:
```

```
Local addr: 0.0.0.0, Local port: 23
```

```
Remote addr: 0.0.0.0, Remote port: 0
```

```
State: LISTEN Authentication: None
```

```
Rcvd: 4 total pkts, 0 in-sequence pkts, 0 bytes
```

```
         0 chksum err pkts, 0 bad offset pkts, 0 short pkts
```

```
         0 duplicate pkts, 0 out of order pkts
```

```
Sent: 0 total pkts, 0 data pkts, 0 bytes
      0 retransmitted pkts, 0 retransmitted bytes

Local addr: 192.168.1.250, Local port: 23
Remote addr: 10.10.0.77, Remote port: 2170
State: ESTABLISHED Authentication: None
Rcvd: 61 total pkts, 34 in-sequence pkts, 41 bytes
      0 chksum err pkts, 0 bad offset pkts, 0 short pkts
      0 duplicate pkts, 0 out of order pkts
Sent: 64 total pkts, 45 data

Local addr: 192.168.1.250, Local port: 23
Remote addr: 10.10.0.77, Remote port: 2170
State: ESTABLISHED Authentication: None
Rcvd: 61 total pkts, 34 in-sequence pkts, 41 bytes
      0 chksum err pkts, 0 bad offset pkts, 0 short pkts
      0 duplicate pkts, 0 out of order pkts
Sent: 64 total pkts, 45 data pkts, 2304 bytes
      0 retransmitted pkts, 0 retransmitted bytes

Local addr: 192.168.1.250, Local port: 23
Remote addr: 192.168.1.139, Remote port: 1038
State: ESTABLISHED Authentication: None
Rcvd: 295 total pkts, 159 in-sequence pkts, 299 bytes
      0 chksum err pkts, 0 bad offset pkts, 0 short pkts
      0 duplicate pkts, 0 out of order pkts
Sent: 281 total pkts, 210 data pkts, 3089 bytes
      0 retransmitted pkts, 0 retransmitted bytes
```

show ip traffic

- Use to display statistics about IP traffic.
- You can use the ipTraffic log to show consumable IP traffic to the SRP module; the traffic is filterable per router and IP interface. You can show ICMP, TCP, and UDP traffic with the icmpTraffic, udpTraffic, and tcpTraffic logs.
- Field descriptions
 - › IP Statistics Rcvd:
 - router Id – router ID number
 - total – number of frames received
 - local destination – frames with this router as their destination
 - hdr errors – number of packets containing header errors
 - addr errors – number of packets containing addressing errors
 - unkn proto – number of packets received containing unknown protocols
 - discards – number of discarded packets

- › IP Statistics Frags:
 - reassembled – number of reassembled packets
 - reasm timed out – number of reassembled packets that timed out
 - reasm req – number of requests for reassembly
 - reasm fails – number of reassembly failures
 - frag ok – number of fragmented packets reassembled successfully
 - frag fail – number of fragmented packets reassembled unsuccessfully
 - frag creates – number of packets created by fragmentation
- › IP Statistics Sent:
 - forwarded – number of packets forwarded
 - generated – number of packets generated
 - out disc – number of outbound packets discarded
 - no routes – number of packets that could not be routed
 - routing discards – number of packets that could not be routed and were discarded
- › IP Statistics Route:
 - routes in table – number of routes in the routing table
- › ICMP Statistics Rcvd:
 - total – total number of ICMP packets received
 - errors – number of error packets received
 - dst unreach – number of packets received with destination unreachable
 - time exceed – number of packets received with time-to-live exceeded
 - param probs – number of packets received with parameter errors
 - src quench – number of source quench packets received
 - redirects – number of receive packet redirects
 - echo req – number of echo request (PING) packets
 - echo rpy – number of echo replies received
 - timestamp req – number of requests for a timestamp
 - timestamp rpy – number of replies of timestamp requests
 - addr mask req – number of mask requests sent
 - addr mask rpy – number of mask replies sent
- › ICMP Statistics Sent:
 - total – total number of ICMP packets sent
 - errors – number of error packets sent
 - dest unreach – number of packets sent with destination unreachable
 - time excd – number of packets sent with time-to-live exceeded
 - param prob – number of packets sent with parameter errors
 - src quench – number of source quench packets sent
 - redirects – number of send packet redirects
 - echo req – number of echo request (ping) packets
 - echo rpy – number of echo replies received

- timestamp req – number of request for a timestamp
- timestamp rpy – number of replies to timestamp requests
- addr mask req – number of address mask requests
- addr mask rpy – number of replies to address mask requests
- › UDP Statistics Rcvd:
 - total – total number of UDP packets received
 - checksum – number of checksum error packets received
 - no port – number of packets received for which no ERX application listener was listening on the destination port
- › UDP Statistics Sent:
 - total – total number of UDP packets sent
 - errors – number of error packets sent
- › TCP Global Statistics Connections:
 - attempted – number of outgoing TCP connections attempted
 - accepted – number of incoming TCP connections accepted
 - established – number of TCP connections established
 - dropped – number of TCP connections dropped
 - closed – number of TCP connections closed
 - currently established – number of TCP connections currently established
- › TCP Global Statistics Rcvd:
 - total pkts – total number of TCP packets received
 - in-sequence pkts – number of packets received in sequence
 - bytes – number of bytes received
 - chksum err pkts – number of checksum error packets received
 - authentication err pkts – number of authentication error packets received
 - bad offset pkts – number of packets received with bad offsets
 - short pkts – number of short packets received
 - duplicate pkts – number of duplicate packets received
 - out of order pkts – number of packets received out of order
- › TCP Global Statistics Sent:
 - total pkts – total number of TCP packets sent
 - data pkts – number of data packets sent
 - bytes – number of bytes sent
 - retransmitted pkts – number of packets retransmitted
 - retransmitted bytes – number of retransmitted bytes
- › OSPF Statistics – provides statistics on OSPF
- › IGMP Statistics – provides statistics about queries, reports sent or received
- › ARP Statistics – not supported for this version of the system

- Example

```
host1#show ip traffic
IP statistics: Router Id: 172.31.192.217
  Rcvd: 97833 total, 171059 local destination
    0 hdr errors, 0 addr errors
      167 unkn proto, 0 discards
  Frags: 4 reassembled, 30 reasm timed out, 8 reasm req
    0 reasm fails, 145 frag ok, 0 frag fail
    290 frag creates
  Sent: 15 forwarded, 25144 generated, 0 out disc
    0 no routes, 0 routing discards
  Route: 57680 routes in table
    0 timestamp req, 0 timestamp rpy
    0 addr mask req, 0 addr mask rpy
ICMP statistics:
  Rcvd: 561 total, 0 errors, 15 dst unreach
    0 time exceed, 0 param probs, 0 src quench
    0 redirects, 0 echo req, 0 echo rpy
    0 timestamp req, 0 timestamp rpy
    0 addr mask req, 0 addr mask rpy
  Sent: 463866 total, 0 errors, 163676 dest unreach
    0 time excd, 0 param prob, 0 src quench
    20 redirects, 463846 echo req, 0 echo rpy
    0 timestamp req, 0 timestamp rpy
    0 addr mask req, 0 addr mask rpy
UDP Statistics:
  Rcvd: 93326 total, 0 checksum errors, 90610 no port
  Sent: 0 total, 0 errors
TCP Global Statistics:
  Connections: 7358 attempted, 4 accepted, 7362 established
    0 dropped, 14718 closed
  Rcvd: 75889 total pkts, 53591 in-sequence pkts, 3120283
    bytes
    0 chksum err pkts, 0 authentication err pkts, 0 bad
    offset
    0 short pkts, 0 duplicate pkts, 0 out of order pkts
  Sent: 82318 total pkts, 44381 data pkts, 656321 bytes
    34 retransmitted pkts, 487 retransmitted bytes
OSPF Statistics:
IGMP Statistics:
ARP Statistics:
```

show ip udp statistics

- Use to display UDP statistics.
- Example

```
host1#show ip udp statistics
UDP Statistics:
  Rcvd: 39196 total, 0 checksum errors, 29996 no port
  Sent: 210 total, 0 errors
```

show profile brief

- Use to list all profile names.
- Field descriptions
 - › profile – profile name
- Example

```
host1#show profile brief
Profile :
foo
trill
profile4
```

show route-map

- Use to display the configured route maps.
- The displayed information includes the instances of each access list such as **match** and **set** commands.
- Example

```
host1(config)#route-map westford permit 10
host1(config-route-map)#match community 44
host1(config-route-map)#set local-pref 400
host1(config-route-map)#exit
host1(config)#exit
host1#show route-map westford
route-map 1, permit, sequence 10
  Match clauses:
    match community 44
  Set clauses:
    set local-pref 400
```