

# Configuring IP Tunnels

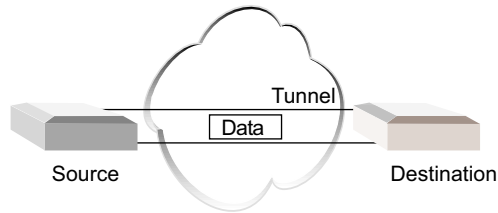
IP tunnels provide a way of transporting datagrams between routers separated by networks that do not support all the protocols that those routers support. This chapter describes how to configure IP tunnels on the ERX system.

Topic	Page
Overview	4-1
Line Module Requirements	4-2
Managing TSMs	4-3
Specifications	4-3
References	4-3
Configuration Tasks	4-3
Monitoring IP Tunnels	4-8

## Overview

---

The ERX system supports static IP tunnels. An IP tunnel is a virtual point-to-point connection between two routers. See Figure 4-1. To establish an IP tunnel, you specify a tunnel type and name, and then configure an interface on each router to act as an endpoint for the tunnel.



**Figure 4-1 IP tunneling**

The ERX system supports the following types of IP tunnels:

- Generic Routing Protocol (GRE) tunnels
- Distance Vector Multicast Routing Protocol (DVMRP) tunnels, also known as IP in IP tunnels

### *GRE Tunnels*

GRE encapsulates IP packets to enable data transmission through an IP tunnel. The resulting encapsulated packet contains a GRE header and a delivery header. Consequently, the packet requires more processing than an IP packet, and GRE can be slower than native routing protocols.

### *DVMRP Tunnels*

DVMRP tunnels allow the exchange of IP multicast traffic between routers separated by networks that do not support multicast routing. For information about DVMRP, see *Chapter 3, Configuring IP Multicasting*.

## Line Module Requirements

---

To create IP tunnels, you must install a Tunnel Service line module (TSM) in the ERX system. For information about installing TSMs in the ERX system, see the *ERX Installation and User Guide*.

Unlike other line modules, the TSM does not pair with a corresponding I/O module that contains ingress and egress ports. The TSM receives data from, and transmits data to, other line modules with ingress and egress ports. However, you must assign interfaces on other line modules or loopback interfaces to act as source endpoints for the tunnel.

All line modules forward traffic to tunnels. For information about which line modules accept traffic for tunnels, see the *Release Notes*.

## Managing TSMs

---

For information about TSM redundancy and tunnel distribution, see *ERX Physical and Link Layers Configuration Guide, Chapter 9, Managing Tunnel Service and IPSec Service Interfaces*.

## Specifications

---

Each Tunnel Service module supports a total of 8,000 tunnels, with a maximum of 4000 IP tunnels. For example, a TSM can support 2000 GRE tunnels, 2000 DVMRP tunnels, and 4000 L2TP tunnels.

## References

---

For more information about IP tunnels, see the following documents:

- RFC 791 – Internet Protocol DARPA Internet Program Protocol Specification (September 1981)
- RFC 1700 – Assigned Numbers (October 1994)
- RFC 1701 – Generic Routing Encapsulation (October 1994)
- RFC 1702 – Generic Routing Encapsulation over IPv4 Networks (October 1994)
- RFC 2003 – IP Encapsulation within IP (October 1996)
- RFC 2784 – Generic Routing Encapsulation (GRE) (March 2000)

## Configuration Tasks

---

To configure an IP tunnel:

- 1 Create or select a physical or loopback interface.  
This interface acts as an anchor for the source of the tunnel.
- 2 Assign an IP address to the physical or loopback interface.
- 3 Create a tunnel interface.
- 4 Set the source address for the tunnel.
- 5 Set the destination address for the tunnel.
- 6 (Optional) Enable error checking across a GRE tunnel.
- 7 Set the maximum transmission unit (MTU) size for the tunnel.



**Note:** On TSM interfaces, issue only the commands listed below. Do not configure protocols such as Multilink PPP or Multilink Frame Relay on TSM interfaces.

### ***interface tunnel***

- Use to create an IP tunnel interface.
- Specify the type and name of the tunnel you want to create.
- You can establish the tunnel on a virtual router other than the current virtual router.
- Example

```
host1(config)#interface tunnel dvmrp:boston-tunnel-1  
                  transport-virtual-router boston
```

- Use the **no** version to remove the tunnel.

### ***tunnel checksum***

- Use to enable checksum computation across a GRE tunnel.
- Checksum computation is not supported for DVMRP tunnels.
- Selecting this feature causes the ERX system to drop corrupted packets it receives on the tunnel interface.
- Example

```
host1(config)#interface tunnel gre:tunnel2  
host1(config-if)#tunnel checksum
```

- Use the **no** version to disable the checksum option.

### ***tunnel destination***

- Use to configure the remote end of the tunnel.
- Specify either the IP address of an interface on the remote system or the hostname of the remote system.
  - › The IP address is the address for the destination interface.
  - › The hostname is the name of the destination interface.
- Example 1

```
host1(config)#interface tunnel dvmrp:tunnel2  
host1(config-if)#tunnel destination 192.13.7.1
```

- Example 2

```
host1(config)#interface tunnel dvmrp:tunnel2  
host1(config-if)#tunnel destination remoteHost
```

- Use the **no** version to remove the destination of a tunnel.

### **tunnel mtu**

- Use to set the MTU for the tunnel.
- Specify a value in the range 1024 to 10240 bytes.
- Example

```
host1(config-if)#tunnel mtu 7500
```
- Use the **no** version to restore the default, 10240 bytes.

### **tunnel source**

- Use to configure the source of the tunnel.
- Specify either the primary IP address or the type and specifier of an interface.
- Do not specify an unnumbered interface.
- Example 1

```
host1(config)#interface tunnel dvmrp:boston-tunnel-1  
host1(config-if)#tunnel source 192.10.2.1
```
- Example 2

```
host1(config)#interface tunnel dvmrp:boston-tunnel-1  
host1(config-if)#tunnel source atm 5/0.12
```
- Use the **no** version to remove the source of a tunnel.

### *Configuration Example*

In this example, two GRE tunnel interfaces are configured on different virtual routers of an ERX system. The source of the first tunnel interface matches the destination of the second tunnel interface and vice versa.

- 1 Configure a virtual router called boston that will support one end of the tunnel.

```
host1#virtual-router boston
```

- 2 Configure a physical or loopback interface for the end of the tunnel on the virtual router boston.

The IP address of this interface appears in the header of tunneled frames and is used for forwarding traffic.

```
host1:boston#interface atm 12/0.5  
host1:boston(config-if)#ip address 5.5.5.5 255.255.255.0
```

- 3 Configure the tunnel interface on virtual router boston.

- a Create the tunnel interface.

```
host1:boston(config)#interface tunnel gre:ChicagoTunnel
```

- b Configure the source and destination points of the tunnel interface.

```
host1:boston(config-if)#tunnel source 5.5.5.5  
host1:boston(config-if)#tunnel destination 6.6.6.6
```

- c Set the MTU for the tunnel.

```
host1:boston(config-if)#tunnel mtu 8000
```

- d Configure the IP address of the tunnel interface.

```
host1:boston(config-if)#ip address 7.7.7.7 255.255.255.0
```

- 4 Configure a virtual router called `chicago` that will support the other end of the tunnel.

```
host1(config)#virtual-router chicago
```

- 5 Configure a physical or loopback interface for the end of the tunnel on the virtual router `chicago`.

```
host1:chicago(config)#interface atm 12/1.5  
host1:chicago(config-if)#ip address 6.6.6.6 255.255.255.0
```

- 6 Configure the tunnel interface on virtual router `chicago`.

- a Create the tunnel interface.

The name of the tunnel interface can differ from the tunnel interface configured in step 3.

```
host1:chicago(config-if)#interface tunnel gre:BostonTunnel
```

- b Configure the source and destination points of the tunnel interface.

The destination of this tunnel interface matches the source of the tunnel interface configured in step 3 and vice versa.

```
host1:chicago(config-if)#tunnel source 6.6.6.6  
host1:chicago(config-if)#tunnel destination 5.5.5.5
```

- c Set the MTU for the tunnel.

The MTU must match the MTU configured in step 3.

```
host1:chicago(config-if)#tunnel mtu 8000
```

- d Configure the IP address of the tunnel interface.

```
host1:chicago(config-if)#ip address 9.9.9.9 255.255.255.0
```

### *Configuring IP Tunnels to Forward IP Frames*

When a line module receives IP frames destined for a tunnel, the module forwards the frames to the TSM. The TSM encapsulates the frames and forwards them to the tunnel through an interface determined by a route lookup of an IP frame. The source and destination addresses in the IP frame are the source and destination addresses of the tunnel.

Similarly, when a line module receives traffic from a tunnel, the module forwards the traffic to the TSM for deencapsulation. After deencapsulation, the TSM forwards the resultant IP frames to an interface determined by a route lookup.

When you have configured a tunnel interface, treat it in the same way as any IP interface on the router. For example, you can configure static IP routes or enable routing protocols on the tunnel interface. The IP configurations you apply to the tunnels control how traffic travels through the network.

### *Preventing Recursive Tunnels*

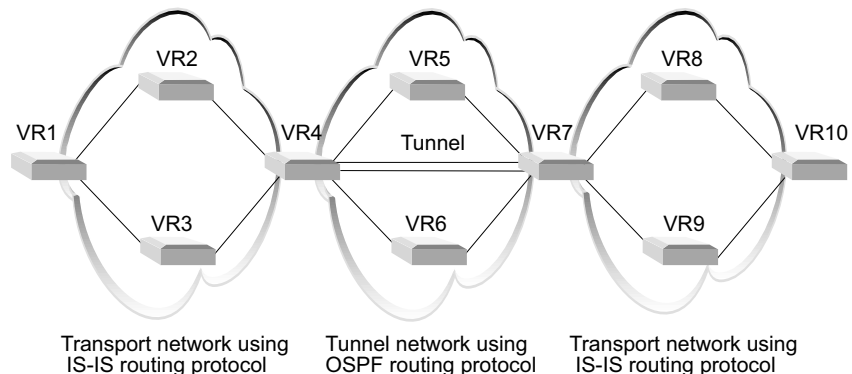
If routing information about the tunnel network combines with routing information about the transport networks (the networks that the tunnel services), a *recursive* tunnel can occur. In this case, the routing table defines the tunnel itself as the best path to a tunnel destination. To prevent recursive tunnels, differentiate routing information for the tunnel network and the transport networks with one or both of the following techniques:

- Use different routing protocols for the tunnel network and the transport networks.
- Define a static route to the tunnel destination.



**Note:** If you define a static route to a tunnel destination, be careful not to create routing loops.

Figure 4-2 illustrates how to prevent recursive tunnels by using different routing protocols for the tunnel network and the transport networks.



**Figure 4-2** Transport and tunnel networks using different routing protocols

## Monitoring IP Tunnels

You can monitor DVMRP and GRE tunnels by using the following commands.

### *show dvmrp tunnel*

- Use to display information about DVMRP tunnels.
- Optionally, specify the **detail** keyword to view detailed information about tunnels.
- Optionally, specify the **state** keyword and the state of the tunnel (disabled, down, enabled, lower-down, not-present, up) to view the number of tunnels in a specific state.
- Optionally, specify a tunnel name to view the state of a specific tunnel.
- Optionally, specify an IP address to view the number of tunnels associated with that IP address.
- Optionally, specify an IP address with the **virtual-router** keyword and the name of the virtual router to view the number of tunnels associated with that IP address on the virtual router.
- Field descriptions
  - › Tunnel status
    - up – tunnel is operational
    - down – tunnel is not operational
    - not-present – tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible
  - › Tunnel name – name of the tunnel
  - › Tunnel mtu – maximum transmission unit for the tunnel
  - › Tunnel source address – IP address of the source of the tunnel
  - › Tunnel destination address – IP address of the destination of the tunnel
  - › Tunnel transport virtual router – virtual router associated with the tunnel

- › Tunnel up/down trap is enabled – indicates whether or not the ERX system sends traps to SNMP when the operational state of the tunnels changes
  - › Tunnel server location – location of the TSM in slot/port format. The port is internal to the TSM.
  - › Tunnel administrative state – configured state of the tunnel: up or down
  - › Statistics – details of packets received or transmitted by the tunnel
  - › Packets – number of packets received or transmitted by the tunnel
  - › Octets – number of octets received or transmitted by the tunnel
  - › Discards – number of packets not accepted by the tunnel
  - › Errors – number of packets with errors received or transmitted by the tunnel
  - › Data rx – received data
  - › Data tx – transmitted data
- Example 1

```
host1#show dvmrp tunnel
DVMRP tunnel boston1 is up
1 DVMRP tunnel found
```

- Example 2

```
host1#show dvmrp tunnel detail
DVMRP tunnel boston1 is up
Tunnel operational configuration
  Tunnel name is 'boston1'
  Tunnel mtu is '10240'
  Tunnel source address is '0.0.0.0'
  Tunnel destination address is '0.0.0.0'
  Tunnel transport virtual router is vr1
  Tunnel up/down trap is enabled
  Tunnel server location is 4/0
  Tunnel administrative state is up
Statistics   packets      octets      discards     errors
Data rx     0             0           0            0
Data tx     0             0           0            0
1 DVMRP tunnel found
```

- Example 3

```
host1#show dvmrp tunnel state enabled
DVMRP tunnel boston1 is up
1 DVMRP tunnel found
```

- Example 4

```
host1#show dvmrp tunnel virtual-router vr1 ip 0.0.0.0
DVMRP tunnel boston1 is up
1 DVMRP tunnel found
```

### ***show dvmrp tunnel summary***

- Use to display a summary of information about DVMRP tunnels.
- Field descriptions
  - › Administrative status
    - enabled – tunnel is available for use
    - disabled – tunnel is not available for use
  - › Operational status
    - up – tunnel is operational
    - down – tunnel is not operational
    - not-present – tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible
- Example

```
host1#show dvmrp tunnel summary
Administrative status  enabled  disabled
                      1          0
Operational status    up       down    not-present
                      1          0      0
```

### ***show gre tunnel***

- Use to display information about a GRE tunnel or a list of GRE tunnels.
- Optionally, specify the **detail** keyword to view detailed information about tunnels.
- Optionally, specify the **state** keyword and the state of the tunnel (disabled, down, enabled, lower-down, not-present, up) to view the number of tunnels in a specific state.
- Optionally, specify a tunnel name to view the state of a specific tunnel.
- Optionally, specify an IP address to view the number of tunnels associated with that IP address.
- Optionally, specify an IP address with the **virtual-router** keyword and the name of the virtual router to view the number of tunnels associated with that IP address on the virtual router.
- Field descriptions
  - › Tunnel name – name of the tunnel
  - › Tunnel mtu – maximum transmission unit for the tunnel
  - › Tunnel source address – IP address of the source of the tunnel
  - › Tunnel destination address – IP address of the destination of the tunnel
  - › Tunnel transport virtual router – virtual router associated with the tunnel
  - › Tunnel checksum option – state of the checksum feature: enabled or disabled
  - › Tunnel up/down trap is enabled – indicates whether or not the ERX system sends traps to SNMP when the operational state of the tunnels changes
  - › Tunnel server location – location of the TSM in slot/port format. The port is internal to the TSM.
  - › Tunnel administrative state – configured state of the tunnel: up or down

- › Statistics – details of packets received or transmitted by the tunnel
- › Packets – number of packets received or transmitted by the tunnel
- › Octets – number of octets received or transmitted by the tunnel
- › Discards – number of packets not accepted by the tunnel
- › Errors – number of packets with errors received or transmitted by the tunnel
- › Data rx – received data
- › Data tx – transmitted data

- Example 1

```
host1#show gre tunnel
3 GRE tunnels found
```

- Example 2

```
host1#show gre tunnel detail
```

```
Tunnel operational configuration
```

```
Tunnel name is 'vr1'
Tunnel mtu is '10240'
Tunnel source address is '15.0.0.2'
Tunnel destination address is '15.0.0.1'
Tunnel transport virtual router is vr1
Tunnel checksum option is disabled
Tunnel up/down trap is enabled
Tunnel server location is 4/0
Tunnel administrative state is up
```

Statistics	packets	octets	discards	errors
Data rx	0	0	0	0
Data tx	0	0	0	0

```
Tunnel operational configuration
```

```
Tunnel name is 'default'
Tunnel mtu is '10240'
Tunnel source address is '15.0.0.1'
Tunnel destination address is '15.0.0.2'
Tunnel transport virtual router is default
Tunnel checksum option is disabled
Tunnel up/down trap is enabled
Tunnel server location is 4/0
Tunnel administrative state is up
```

Statistics	packets	octets	discards	errors
Data rx	0	0	0	0
Data tx	0	0	0	0

```
Tunnel operational configuration
```

```
Tunnel name is 'london2'
Tunnel mtu is '10240'
Tunnel source address is '0.0.0.0'
```

```
Tunnel destination address is '0.0.0.0'
Tunnel transport virtual router is vr1
Tunnel checksum option is disabled
Tunnel up/down trap is enabled
Tunnel server location is 4/0
Tunnel administrative state is up
```

Statistics	packets	octets	discards	errors
Data rx	0	0	0	0
Data tx	0	0	0	0

3 GRE tunnels found

- Example 3

```
host1#show gre tunnel state up
GRE tunnel VR1 is up
GRE tunnel default is up
GRE tunnel london2 is down
3 GRE tunnels found
```

Example 4

```
host1#show gre tunnel virtual-router vr1 ip 15.0.0.1
GRE tunnel VR1 is up
1 GRE tunnel found
```

***show gre tunnel summary***

- Use to display a summary of information about GRE tunnels.
- Field descriptions
  - › Administrative status
    - enabled – tunnel is available for use
    - disabled – tunnel is not available for use
  - › Operational status
    - up – tunnel is operational
    - down – tunnel is not operational
    - not-present – tunnel is not operational, because the hardware (such as a line module) supporting the tunnel is inaccessible
- Example

```
host1#show gre tunnel summary
Administrative status   enabled   disabled
                        3         0
Operational status     up        down      not-present
                        3         0         0
```