ERICSSON

Configuring GRE Tunnels

SYSTEM ADMINISTRATOR GUIDE

67/1543-CRA 119 1170/1-V1 Uen D

Copyright

© Ericsson AB 2009–2011. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

Trademark List

SmartEdge	is a registered trademark of Telefonaktiebolaget LM Ericsson.
NetOp	is a trademark of Telefonaktiebolaget LM Ericsson.

Contents

1 Overview

This document describes how to configure, monitor, and administer Generic Routing Encapsulation (GRE) tunnels and tunnel circuits over IP Version 4 (IPv4), IP Version 6 (IPv6), and GRE Virtual Private Networks (VPNs).

This document applies to both the Ericsson SmartEdge® and SM family routers. However, the software that applies to the SM family of systems is a subset of the SmartEdge OS; some of the functionality described in this document may not apply to SM family routers.

For information specific to the SM family chassis, including line cards, refer to the SM family chassis documentation.

For specific information about the differences between the SmartEdge and SM family routers, refer to the Technical Product Description *SM Family of Systems* (part number 5/221 02-CRA 119 1170/1) in the **`Product Overview`** folder of this Customer Product Information library.

GRE is a simple, stateless protocol that allows for the tunneling of IP in IP. The SmartEdge router implementation of GRE over IPv4 is based on these IETF documents:

- RFC 1702, `Generic Routing Encapsulation over IPv4 Networks`
- RFC 2784, `Generic Routing Encapsulation`
- RFC 2893, `Transition Mechanisms for IPv6 Hosts and Routers`

Note: Unless otherwise noted, the SmartEdge 100 router supports all commands described in this document.

1.1 Using GRE Tunnels and Tunnel Circuits with IPv6 Packets

GRE allows you to connect remote sites using IPv6 addresses over a public network that uses publicly routable IPv4 addresses. IPv6 packets traveling through the tunnel are encapsulated with a GRE header and then with an IPv4 header using addresses from the public IPv4 address as shown in Figure 1.

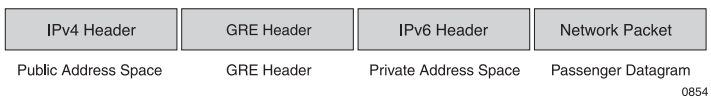

Figure 1 GRE Tunnel Packet Encapsulation for IPv6 Packets (854)

GRE tunnel circuits allow you to multiplex traffic from different users through the same tunnel. Each tunnel uses an IPv4 routing infrastructure to transfer IP

packets through the tunnel. Each tunnel circuit is assigned a unique key and bound to an interface. Each tunnel circuit then acts as a point-to-point circuit connection for traffic associated with that interface.

1.2 Using GRE Tunnels and Tunnel Circuits with IPv4 Packets

GRE allows you to connect remote sites using private IP addresses over a public network that uses publicly routable IP addresses. IP packets traveling through the tunnel are encapsulated with an IP header from the public address space as shown in Figure 2.

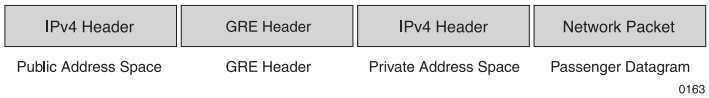

Figure 2 GRE Tunnel Packet Encapsulation for IPv4 Packets (163)

GRE tunnel circuits allow you to multiplex traffic from different users through the same tunnel. Each tunnel circuit is assigned a unique key and bound to an interface. Each tunnel circuit then acts as a point-to-point circuit connection for traffic associated with that interface.

1.3 Using GRE Tunnels and Tunnel Circuits for VPNs

One of the more common applications of GRE tunneling is the creation of VPNs to connect to remote sites. Multiple SmartEdge router contexts and GRE tunnel circuits, one for each VPN, demultiplex traffic for each VPN into its own IP address space. Thus each context acts as a dedicated virtual router for a VPN, where the IP address space (for example, private addresses as described in RFC 1918, `Address Allocation for Private Internets`) and routing databases are maintained separately from other contexts.

In this model, a single tunnel is created between the local site and each remote site. Each GRE tunnel is defined in a context, usually local, and connected to the public network. A single public IP address is assigned to each end of each tunnel and is shared by all tunnel circuits using that tunnel. For each VPN, a context and an interface are created; then a GRE tunnel circuit with a unique key identifier is created for the VPN in the tunnel and bound to the VPN's interface in the VPN's context.

Figure 3 shows the GRE tunnel architecture with multiple contexts. In the figure, each key identifies a tunnel circuit that is bound to an interface in a different context.

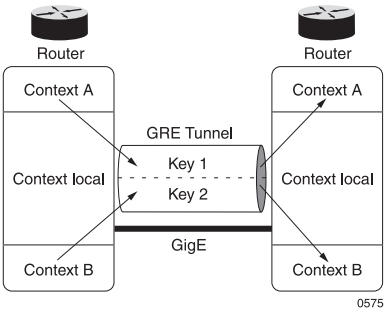

Figure 3 GRE Tunnel Architecture (0575)

Traffic from users in **Context A** travels over the tunnel circuit identified with **Key 1** and is kept separated from traffic from users in **Context B**, which travels over the tunnel circuit identified with **Key 2**, although both circuits share the same GRE tunnel and physical link, the Gigabit Ethernet port (shown as the heavy line labeled "**GigE**").

Using GRE, an arbitrary network topology can be overlaid on the physical topology; that is, each VPN can have a topology independent of the topology to which the physical SmartEdge router is connected. Multiple topologies are supported: full mesh, partial mesh, and hub-and-spoke. To facilitate IP connectivity between VPNs on different SmartEdge router over GRE tunnels, several options exist:

- For simple topologies, static routes in each VPN context can be used.
- For hub-and-spoke and dual hub-and-spoke topologies, a combination of static routing and Routing Information Protocol (RIP) can be used.

  Each spoke VPN is configured with a static default route to the GRE tunnel attached to the hub site, and is configured using RIP to disseminate downstream prefixes to the hub. Each hub VPN is configured to run RIP in passive mode to listen for prefixes from spoke routers.
- For more complex topologies, each VPN can run its own instance of a routing protocol, such as RIP or Open Shortest Path First (OSPF).

1.4 Terminology

Note: When IP Version 6 (IPv6) addresses are not referenced or explicitly specified, the term IP address can refer generally to IP Version 4 (IPv4) addresses, IPv6 addresses, or IP addressing. In instances where IPv6 addresses are referenced or explicitly specified, the term IP address refers only to IPv4 addresses.

2 Configuration and Operations Tasks

Please read the following guidelines before attempting any configurations:

2.1 Configuration Guidelines for GRE Tunnels and Tunnel Circuits

Consider the following guidelines when configuring a GRE tunnel:

- If the traffic receiving end of a GRE tunnel is a SmartEdge router, you must configure the router with a GRE tunnel whose local endpoint is the traffic receiving endpoint of the tunnel. This configuration is required in order to respond to GRE keepalives sent by the traffic sending end of the tunnel.
- Usually, a GRE tunnel is created in the local context to make it available to all tunnel circuits, including those configured for VPNs.
- You can configure multiple tunnels, but usually only one tunnel between sites.
- A GRE tunnel requires at least one tunnel circuit.
- If you create a GRE tunnel in the local context, you can create its tunnel circuits and bind them to interfaces created in any context. If you create a GRE tunnel in any other context (not the local context), you must bind all its tunnel circuits to interfaces that have been created in the same context as the GRE tunnel.
- If you are creating more than one tunnel, they can use the same local interface as long as the remote interfaces are all different. To use an interface and its public IP address for more than one tunnel, use the **`interface`** command with the **`loopback`** keyword in context configuration mode. The **`loopback`** keyword allows you to reuse the public IP address for more than one tunnel.
- The local IP address that you specify must match the IP address of an interface that you have configured in the same context that you are entering this command.
- If the local IP address that you assign to the local tunnel interface is not adjacent to the IP address of the tunnel interface at the remote site and the remote site cannot be reached with a routing protocol, you must also enter the **`ip route`** command in context configuration mode.
- The remote IP address at one end of the GRE tunnel is the same as the local IP address at the other end of the GRE tunnel and conversely. If the remote IP address is not adjacent to the local IP address, and the remote

site cannot be reached with a routing protocol, you must also enter the **ip route** command in context configuration mode.

2.2 Configure a GRE Tunnel

To configure a GRE tunnel, perform the tasks described in Table 1.

Table 1 Configure a GRE Tunnel

Step	Task	Root Command	Notes
1.	Create or select the context for the tunnel interface and access context configuration mode.	*context*	Enter this command in global configuration mode.
2.	Create or select the local tunnel endpoint and access interface configuration mode.	*Interface (context)*	Enter this command in the context mode created or selected in the previous step.
3.	Assign a public IP address to the interface.	*ip address (interface)*	Enter this command in interface mode. This is an IPv4 address.
4.	Create a Generic Routing Encapsulation (GRE) tunnel.	*tunnel*	Enter this command with the **gre** keyword in global configuration mode. This command access the tunnel configuration mode.
5.	Specify tunnel attributes in tunnel configuration mode:		
	Configure the local and remote endpoints of the tunnel circuit, and specify the context of the tunnel.	*peer-end-point*	Creates GRE tunnel circuit with default key “0.” Set the local endpoint IP address to the local IP address of the tunnel specified in step 3.
	Optional. Specify that the DF flag be cleared in inbound packets.	*clear-df*	
	Associate a description with the GRE tunnel.	*description (tunnel)*	
	Specify the name of a tunnel to which the output of the current tunnel is forwarded.	*forward output (tunnel)*	
	Associate the IP address of the remote host.	*source-address*	

Table 1 Configure a GRE Tunnel

Step	Task	Root Command	Notes
	Enable the sending of keepalive packets.	*keepalive (tunnel)*	Consider the following guidelines when configuring any GRE tunnel circuit: • To configure keepalive packets for a tunnel circuit, it must be configured in the same context as the tunnel. • To allow multiple tunnel circuits through a tunnel, you must assign a unique key to each tunnel circuit associated with the tunnel. **Note:** Do not configure your SmartEdge router so that keepalive packets might be forwarded and received on a tunnel endpoint that is bound to different context than the local interface in step 3.
	Enable the logging of state changes.	*log-state-changes*	
	Set the MTU for the tunnel.	*mtu (tunnel)*	
	Bind the tunnel to its interface you created in step 3.	*bind interface*	The binding created by this command in the specified context is used to route inner IP packets.
	Enable the tunnel (begin operations on it).	*shutdown (Tunnel)*	Use the **no** form to enable the tunnel. You can leave a table in the disabled state until you are ready to begin operations on it.

Table 1 Configure a GRE Tunnel

Step	Task	Root Command	Notes
	Specify one or more key IDs in the GRE tunnel. Each key ID creates a circuit (sometimes called a tunnel channel) with the GRE tunnel.	*gre*	Enter the command **gre key** with the ***key-id*** argument. This command enters the GRE key configuration mode, where the tunnel circuit (specified by the ***key-id***) can be bound to an interface and the other attributes of the tunnel circuit can be specified. For data to flow through a GRE tunnel, you must configure at least one tunnel circuit.
6.	Bind each tunnel circuit to an interface.	*bind interface*	Enter this command in GRE tunnel key configuration mode. Consider the following guidelines when configuring a GRE tunnel circuit as a VPN: • To keep traffic separate from different users, you must create a context for each tunnel circuit (VPN) that use the tunnel. For this reason, keepalive packets are not supported for tunnel circuits used as VPNs. • You must assign a private IP address to the interface you create for the tunnel circuit (or VPN). You can reuse this IP address for each tunnel circuit (or VPN) that you create, because you have defined the interface for each tunnel circuit in a different context. If the circuit is bound to the local interface, the routing table of the local interface determines traffic forwarding.

2.3 GRE Tunnel and Tunnel Circuit Operations

To monitor and troubleshoot GRE tunnels and tunnel circuits, perform the appropriate task listed in Table 2. Enter the **`clear`** and **`debug`** commands in exec mode; enter the **`show`** commands in any mode.

Table 2 GRE Tunnel and Single-Circuit Tunnel Circuit Operations

Task	**Root Command**
Clear the circuit counters for one or more GRE tunnel circuits in the system.	*clear circuit counters*
Enable the generation of debug messages for GRE events.	*debug gre*
Display debug messages for circuit events.	*debug circuit*
Enable the generation of debug messages for dynamic tunnel client events.	*debug tunnel client*
Display circuit information for one or more tunnel circuits in the system.	*show circuit*
Display general counters and counters specific to the GRE tunnel circuit type for one or more GRE tunnel circuits in the system.	*show circuit counters*
Display configuration commands for a GRE tunnel circuit.	*show configuration (circuits)*
Display GRE tunnel or tunnel circuit information.	*show gre*
Display GRE tunnel counters.	*show gre counters*
Display GRE and IP-IP tunnel, or tunnel circuit information, or the IP Version 6 (IPv6) tunnel.	*show tunnel*
Display information about dynamic tunnel clients that are registered with the tunnel manager.	*show tunnel client*

Note: To display the bindings for GRE tunnels and the interfaces to which they are bound, enter the **`show ip interface`** command (in any mode).

3 Configuration Examples

This section includes GRE configuration examples:

3.1 GRE Tunnel with a Single Circuit

The following example shows how to configure the GRE tunnel named `tunnel01` with a single circuit (without a key identifier), all in the `local` context:

```
[local]Redback(config)#context local
[local]Redback(config-ctx)#interface upstream

!Assign a public IP address to the local tunnel interface
[local]Redback(config-if)#ip address 172.16.1.1/30

!Assign a private IP address to the tunnel circuit interface
[local]Redback(config-if)#exit
[local]Redback(config-ctx)#interface link
[local]Redback(config-if)#ip address 10.1.1.1/24
[local]Redback(config-if)#exit
[local]Redback(config-ctx)#exit
[local]Redback(config)#tunnel gre tunnel01
[local]Redback(config-tunnel)#peer-end-point local 172.16.1.1 remote 172.16.1.2
[local]Redback(config-tunnel)#description tunnel-with-a-single-circuit
[local]Redback(config-tunnel)#bind interface link local
[local]Redback(config-tunnel)#end
```

3.2 GRE Tunnel with Multiple Circuits Used as VPNs

Figure 4 shows a basic mesh configuration with tunnels between three sites and two tunnel circuits (VPNs) sharing each tunnel. The labels, **A VPN** and **B VPN**, represent contexts, **vpnA** and **vpnB**, in the example commands; not shown in each context are the interfaces, **toHartford**, in each context in the example commands. Private IP addresses are also reused in each VPN context.

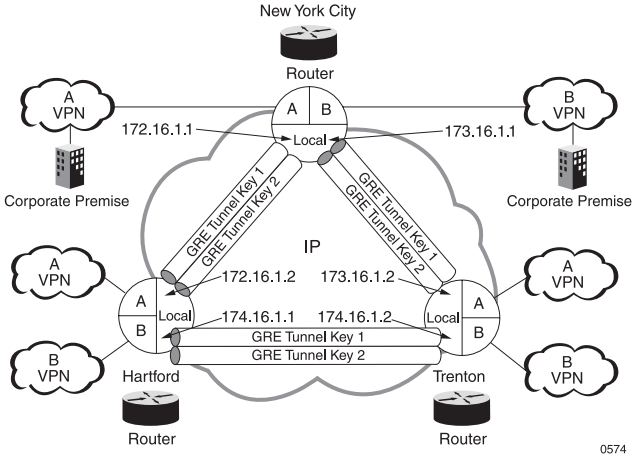

Figure 4 GRE Tunneling Example (0574)

The following examples configure the tunnel to **Hartford** on the router in **New York:**

```
!Create the local interface for the tunnel and
!assign a public IP address to the local tunnel interface
[local]Redback(config)#context local
[local]Redback(config-ctx)#interface toHartford
[local]Redback(config-if)#ip address 172.16.2.1/30
[local]Redback(config-if)#exit
[local]Redback(config-ctx)#exit

!Create the local interface for a tunnel circuit for VPN A, in its own context and
!assign a private IP address to the tunnel circuit interface
[local]Redback(config)#context vpnA
[local]Redback(config-ctx)#interface toHartford
[local]Redback(config-if)#ip address 10.1.2.1/24
[local]Redback(config-if)#exit
[local]Redback(config-ctx)#exit

!Create the local interface for a tunnel circuit for VPN B, in its own context and
!assign a private IP address to the tunnel circuit interface
[local]Redback(config)#context vpnB
[local]Redback(config-ctx)#interface toHartford
[local]Redback(config-if)#ip address 10.1.2.1/24
[local]Redback(config-if)#exit
[local]Redback(config-ctx)#exit

!Configure the tunnel with the local IP address of its interface
[local]Redback(config)#tunnel gre HartfordTnl
[local]Redback(config-tunnel)#peer-end-point local 172.16.2.1 remote 172.16.2.2

!Create the tunnel circuit for VPN A (key 1) and
!Bind the tunnel circuit to its interface, which is in the vpnA context
[local]Redback(config-tunnel)#gre key 1
[local]Redback(config-tunnel-key)#description VPN-A-to-Hartford
[local]Redback(config-tunnel-key)#bind interface toHartford vpnA
[local]Redback(config-tunnel-key)#exit

!Create the tunnel circuit for VPN B (key 2) and
!Bind the tunnel circuit to its interface, which is in the vpnB context
[local]Redback(config-tunnel)#gre key 2
[local]Redback(config-tunnel-key)#description VPN-B-to-Hartford
[local]Redback(config-tunnel-key)#bind interface toHartford vpnB
[local]Redback(config-tunnel-key)#end
```