Introduction

This guide describes Generic Routing Encapsulation (GRE) and its configuration. GRE is a mechanism for encapsulating any network layer protocol over any other network layer protocol.

Contents

Introduction ............................................................................................................................................................................. 1
Products and software version that apply to this guide........................................................................................................ 1
What is GRE? ....................................................................................................................................................................... 2
Configuration Example ............................................................................................................................................................ 4

Products and software version that apply to this guide

This Guide applies to AlliedWare Plus products that supports GRE, running version 5.4.5 or later.

However, implementation varies between products. To see whether a product supports a feature or command, see the following documents:

- The product’s Datasheet
- The AlliedWare Plus Datasheet
- The product’s Command Reference
What is GRE?

GRE is a mechanism for encapsulating any network layer protocol over any other network layer protocol. The general specification was originally described in RFC 1701, and the encapsulation of IP packets over IP is defined in RFC 1702 as a specific implementation of GRE. The GRE specification has been formalized in RFC 2784 and is commonly used for encapsulating IPv4 and IPv6 packets inside IPv4 packets. RFC 2890 extends RFC 2784 with the edition of key and sequence number.

The IPv4 protocol 47 is used when GRE packets are encapsulated in IPv4. GRE is widely used in VPNs as the mechanism for transporting IP packets between private IP networks across public networks with globally routed IP addresses. The advantage of GRE over other tunneling protocols is that it can encapsulate broadcast, multicast traffic (multicast streaming or routing protocols) or other non-IP protocols. GRE packets can be protected by using Internet Protocol Security (IPSec) ensuring confidentiality and integrity of the tunneled traffic.

GRE is stateless and has no knowledge of the configuration or even existence of the remote tunnel endpoint. Once GRE is configured, packets are encapsulated and forwarded whether the decapsulating device is present or not.

GRE allows hosts in one private IP network to communicate with hosts in another private IP network by providing a tunnel between two routers across the Internet.

The GRE connection endpoints are terminated via a Virtual Tunnel Interface (VTI) configured in each device.

**Figure 1: A GRE encapsulated packet form**

<table>
<thead>
<tr>
<th>Delivery Header (IPv4)</th>
<th>GRE Header</th>
<th>Payload Header (IPv4/IPv6)</th>
<th>Payload</th>
</tr>
</thead>
</table>

**Figure 2: A GRE packet header structure**

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| C | Reserved0 | Ver | Protocol Type |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Checksum (optional) | Reserved1 (Optional) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Key (optional) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Sequence Number (Optional) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

**Virtual Tunnel Interface (VTI)**

A Virtual Tunnel Interface has similar characteristics to any other interface on the device. It is virtual because it does not directly map to any of the physical interfaces on the device, but
instead is actually the endpoint of a tunnel from another device. VTls are commonly layer 3 interfaces, can have IP configuration applied directly to them and are compatible with layer 3 routing protocols. The actual tunneling mechanism depends on the protocol used (GRE, RFC2473, L2TP and so on), but commonly uses IP as its transport.

**Delivery header**
This is the outer or encapsulating header. The IPv4 delivery header uses protocol 47 to indicate the next header is a GRE header.

**GRE Header**
The 4-16 byte header is placed between the delivery and payload headers. At least, the GRE header stores the GRE version and payload protocol type. Optionally the GRE header can store a packet checksum, a tunnel key and packet sequence number.

**Payload header**
This is the inner or encapsulated header. GRE is commonly and only used to transport IPv4 and IPv6 packets.

**Figure 3: GRE IP network tunneling protocols**

IP packets from the private IP network destined for a host in the private IP network are encapsulated by Router A and forwarded to Router B. Intermediate routers route the packets using addresses in the delivery protocol header. Router B extracts the original payload packet and routes it to the appropriate destination within network.

The device supports the following features.
- GRE as specified in RFC2784
- Virtual Tunnel Interfaces for terminating GRE encapsulated traffic
- IPv4 as the delivery protocol, used to transport the private data across the public network
- IPv4 as the payload
- IPv6 as the payload
- Up to 256 GRE connections can be configured, with a single GRE tunnel per VTI
- Configurable tunnel source using IPv4 address
- Configurable tunnel source using interface
- Configurable tunnel destination IPv4 address
- Configurable tunnel destination using hostname
- Configurable checksum insertion and checking (disabled by default)
- Configurable TTL value for insertion into the outer header
- Configurable DSCP value for insertion into the outer header (copied from the inner header by default)
- Display of tunnel parameters via show interface tunnel (GRE) command output
- Tunnels are compatible with dynamic IPv4 and IPv6 routing protocols (RIPv1, RIPv2, RIPNG, OSPF, OSPFv3, BGP, BGP4+)
- Existing interface MTU command can be used to set the MTU of a tunnel interface
- Setting DF value in the outer header by copying from the inner header

The following GRE feature components are not supported.
- Non-IPv4 protocols as the delivery protocol
- Non-IPv4/v6 protocols as the payload
- Insertion or processing of Tunnel Key in the GRE header (received packets including a key are dropped)
- Insertion or processing of Sequence Numbers in the GRE header (sequence numbers in received packets are ignored)
- Insertion or processing of Source Route Entries in the GRE header (received packets including a route entry are dropped)
- Path-MTU-discovery in the underlying tunnel interface
- Keep-alives at the GRE protocol level
- Configurable DF value for insertion into the outer header
- Hardware acceleration of GRE encapsulation/decapsulation processes
- Layer 2 features

**Configuration Example**

This example shows the step-by-step instructions to configure a GRE tunnel between Device A and Device B. It assumes that IP has been configured correctly and is operational on both devices.

The following table lists the parameter values in the example. Note public IP addresses are used in this example.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DEVICE A</th>
<th>DEVICE B</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP address of Ethernet interface eth1</td>
<td>1.1.1.1/30</td>
<td>2.2.2.2/30</td>
</tr>
<tr>
<td>tunnel source IP address</td>
<td>1.1.1.1</td>
<td>2.2.2.2</td>
</tr>
</tbody>
</table>
### Figure 4: GRE tunnel

<table>
<thead>
<tr>
<th></th>
<th>DEVICE A</th>
<th>DEVICE B</th>
</tr>
</thead>
<tbody>
<tr>
<td>tunnel destination IP address</td>
<td>2.2.2.2</td>
<td>1.1.1.1</td>
</tr>
<tr>
<td>IP address of tunnel interface</td>
<td>172.168.1.1/24</td>
<td>172.168.1.2/24</td>
</tr>
</tbody>
</table>

**Step 1: Configuring Device A**

```
awplus# configure terminal
Enter the Global Configuration mode.

awplus(config)# interface eth1
Enter the Interface Configuration mode.

awplus(config-if)# ip address 1.1.1.1/30
To assign an IP address for interface eth1.

awplus(config-if)# interface tunnel1
Create tunnel interface tunnel1.

awplus(config-if)# ip address 172.168.1.1/24
Assign an IP address to the tunnel interface.

awplus(config-if)# tunnel mode gre
Set the encapsulation tunneling mode to GRE.

awplus(config-if)# tunnel source 1.1.1.1
Assign an IP address to tunnel source for the tunnel.

awplus(config-if)# tunnel destination 2.2.2.2
Designate the tunnel destination address.

awplus(config-if)# exit
Return to the Global Configuration mode.

awplus(config)# ip route 192.168.2.0 255.255.255.0 172.168.1.2
Configure a static route.
```
### Step 2: Configuring Device B

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>awplus# configure terminal</code></td>
<td>Enter the Global Configuration mode.</td>
</tr>
<tr>
<td><code>awplus(config)# interface eth1</code></td>
<td>Enter the Interface Configuration mode.</td>
</tr>
<tr>
<td><code>awplus(config-if)# ip address 2.2.2.2/30</code></td>
<td>To assign an IP address for interface eth1.</td>
</tr>
<tr>
<td><code>awplus(config-if)# interface tunnel1</code></td>
<td>Create tunnel interface tunnel1.</td>
</tr>
<tr>
<td><code>awplus(config-if)# ip address 172.168.1.2/24</code></td>
<td>Assign an IP address to the tunnel interface.</td>
</tr>
<tr>
<td><code>awplus(config-if)# tunnel mode gre</code></td>
<td>Set the encapsulation tunneling mode to GRE.</td>
</tr>
<tr>
<td><code>awplus(config-if)# tunnel source 2.2.2.2</code></td>
<td>Assign an IP address to tunnel source for the tunnel.</td>
</tr>
<tr>
<td><code>awplus(config-if)# tunnel destination 1.1.1.1</code></td>
<td>Designate the tunnel destination address.</td>
</tr>
<tr>
<td><code>awplus(config-if)# exit</code></td>
<td>Return to the Global Configuration mode.</td>
</tr>
<tr>
<td><code>awplus(config)# ip route 192.168.1.0 255.255.255.0 172.168.1.1</code></td>
<td>Configure a static route.</td>
</tr>
</tbody>
</table>

### Step 3: Verifying connectivity

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>awplus# ping 192.168.2.1</code></td>
<td>You can use the ping command to verify that the tunnel is established.</td>
</tr>
</tbody>
</table>

You should receive ICMP Echo reply message.