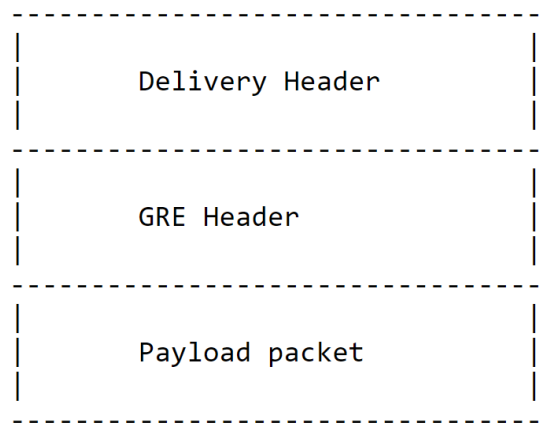
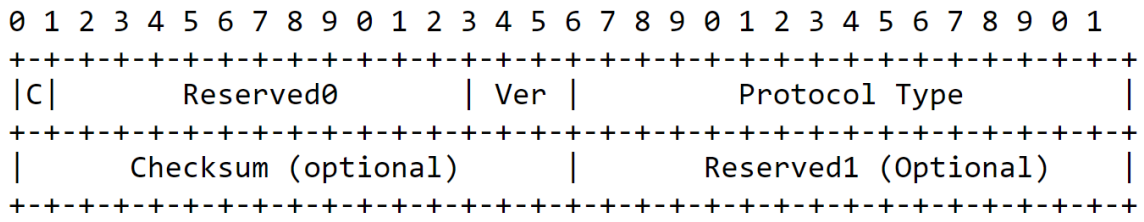
GRE Tunnels
Generic Routing Encapsulation Tunneling

- Point to Point Tunnel
- Allows you to transport multiple higher layer protocols (e.g. IPv4. IPv6, IPX, etc.)
- Does not supply Authentication and Encryption
- Emulates a Point-to-Point tunnel / Point-to-Point serial link
- Supports Multicast Routing Protocols
- GRE tunnels used to be put inside an IPSec tunnel (provides the encryption and authentication)
- GRE encapsulates other traffic inside a 20 byte IP header + a 4 byte GRE header
- RFC 2784 – describes the details of GRE

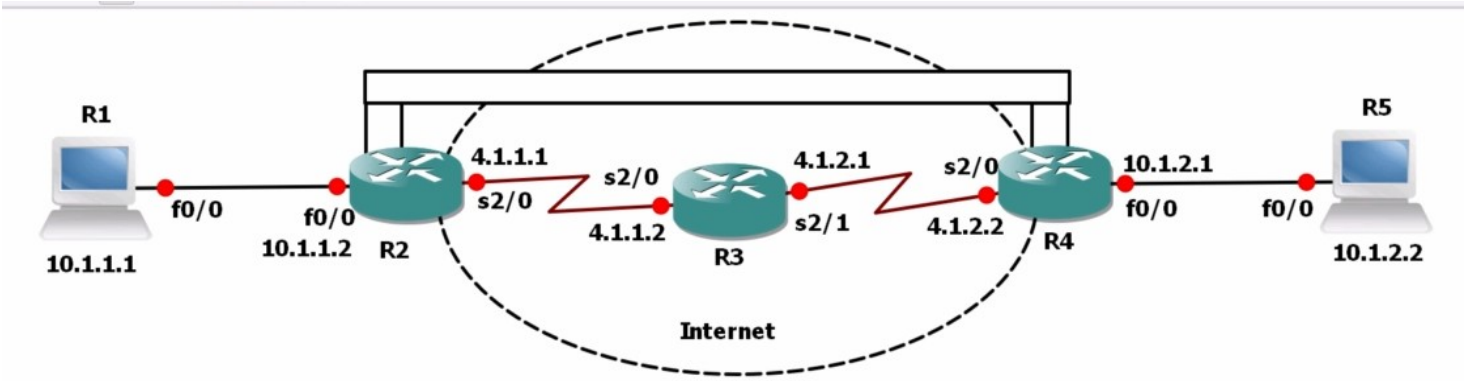
Structure of a GRE Encapsulated Packet



The GRE packet header has the form:



Example of GRE Tunnel



R1 (acting as PC) sends data to R5 (acting as a PC) over a GRE Tunnel between R2 and R4. The GRE Tunnel looks like a Point to Point serial link between R2 and R4.

R1 to R2 – Traffic is sent as normal ethernet frames
At R2 – the traffic is encapsulated in a tunnel for transmission to R4
R3 – acts as an internet router – R3 forwards traffic based on the outer header (aka delivery header)
Because the Tunnel is established from R2 to R4, R3 only sees traffic going from R2's IP address to R4's IP address
R3 routes based on the outer header (Delivery Header) and does not look at the traffic that originated from R1
(Note: GRE does not encrypt, so one could run wire shark between R3 and R4 and capture the internal traffic that was sent from R1 to R5)(The data captured would be contained in the Payload packet portion of a GRE encapsulated Packet)
R4 receives the GRE encapsulated packet and de-encapsulates before forwarding the packets to R5

In setting up this GNS3 lab

R1 and R5 are both acting as PC, with IP routing disabled and a default gateway set to the R2 and R4 respectively
R1(config):default-gateway 10.1.1.2
R1(config):no ip routing

R2 and R4 – create a new interface – Tunnel 0

R2(config):int tunnel 0
R2(config-if):ip add 10.1.3.1 255.255.255.252 (no shut)
R2(config-if):tunnel mode gre ip
R2(config-if):tunnel source 4.1.1.1
R2(config-if):tunnel destination 4.1.2.2

R4(config):int tunnel 0
R4(config-if):ip add 10.1.3.2 255.255.255.252 (no shut)
R4(config-if):tunnel mode gre ip
R4(config-if):tunnel source 4.1.2.2
R4(config-if):tunnel destination 4.1.1.1

Now we need a routing protocol running between R2 and R4., so they are aware of each other's 10.0.0.0/24 routes

R2 is connected to
10.1.1.0/24 via f0/0
10.1.3.0/24 via tun 0

R4 is connected to:
10.1.2.0/24 via g0/0
10.1.3.0/24 via g0/0

Configure EIGRP on R2 and R4
R2(config):router eigrp 100
R2(config-rtr):network 10.0.0.0 (this is the same as 10.0.0.0/8)
R2(config-rtr):no auto-summary

R4(config):router eigrp 100
R4(config-rtr):network 10.0.0.0 (this is the same as 10.0.0.0/8)
R4(config-rtr):no auto-summary

R2 and R4 now have a neighbor relationship over Tunnel 0

```
R4#sh ip eigrp 100 neighbors
EIGRP-IPv4 Neighbors for AS(100)
H   Address                 Interface              Hold Uptime   SRTT   RTO  Q  Seq
                                                   (sec)         (ms)       Cnt Num
0   10.1.3.1                Tu0                      11 00:10:32   26  1470  0  3
```

R4's Routing table

```
Gateway of last resort is 4.1.2.1 to network 0.0.0.0

S*    0.0.0.0/0 [1/0] via 4.1.2.1
      4.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        4.1.2.0/30 is directly connected, Serial1/0
L        4.1.2.2/32 is directly connected, Serial1/0
      10.0.0.0/8 is variably subnetted, 5 subnets, 3 masks
D        10.1.1.0/24 [90/26905600] via 10.1.3.1, 00:11:43, Tunnel0
C        10.1.2.0/24 is directly connected, Ethernet0/0
L        10.1.2.1/32 is directly connected, Ethernet0/0
C        10.1.3.0/30 is directly connected, Tunnel0
L        10.1.3.2/32 is directly connected, Tunnel0
```

There is a static route to R3
There is also a D (EIGRP route to 10.1.1.0 /24 via 10.1.3.1 0– tunnel 0 on R2

R2's Routing table

```
Gateway of last resort is 4.1.1.2 to network 0.0.0.0

S*    0.0.0.0/0 [1/0] via 4.1.1.2
      4.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        4.1.1.0/30 is directly connected, Serial1/0
L        4.1.1.1/32 is directly connected, Serial1/0
      10.0.0.0/8 is variably subnetted, 5 subnets, 3 masks
C        10.1.1.0/24 is directly connected, Ethernet0/0
L        10.1.1.2/32 is directly connected, Ethernet0/0
D        10.1.2.0/24 [90/26905600] via 10.1.3.2, 00:25:20, Tunnel0
C        10.1.3.0/30 is directly connected, Tunnel0
L        10.1.3.1/32 is directly connected, Tunnel0
```

There is a static route to R3
There is also a D (EIGRP route to 10.1.2.0 /24 via 10.1.3.2 0– tunnel 0 int on R4

The traceroute shows the ping leaves R1 to R2 (via R1's default gateway setting)

The 10.1.2./24 network is advertised to R2 via 10.1.3.2 (tunnel 0 int on r4)
R2 -The ip packet is encapsulated with GRE (source 10.1.3.1) (destination 10.1.3.2)
R4 – the packet is de-encapsulated and IP (source 10.1.1.1) (destination 10.1.2.2)
R4 forwards the packet to 10.1.2.2 (locally conncected on g0/0 – 10.1.2.0/24)

R3 Routing Table

```
Gateway of last resort is not set

      4.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C        4.1.1.0/30 is directly connected, Serial1/0
L        4.1.1.2/32 is directly connected, Serial1/0
C        4.1.2.0/30 is directly connected, Serial1/1
L        4.1.2.1/32 is directly connected, Serial1/1
R3#
```

Router 3 does not see 10.1.1.0 or 10.1.2.0 or 10.1.3.0 (tunnel 0)
R3 just forwards the frames between the GRE Source and Destination