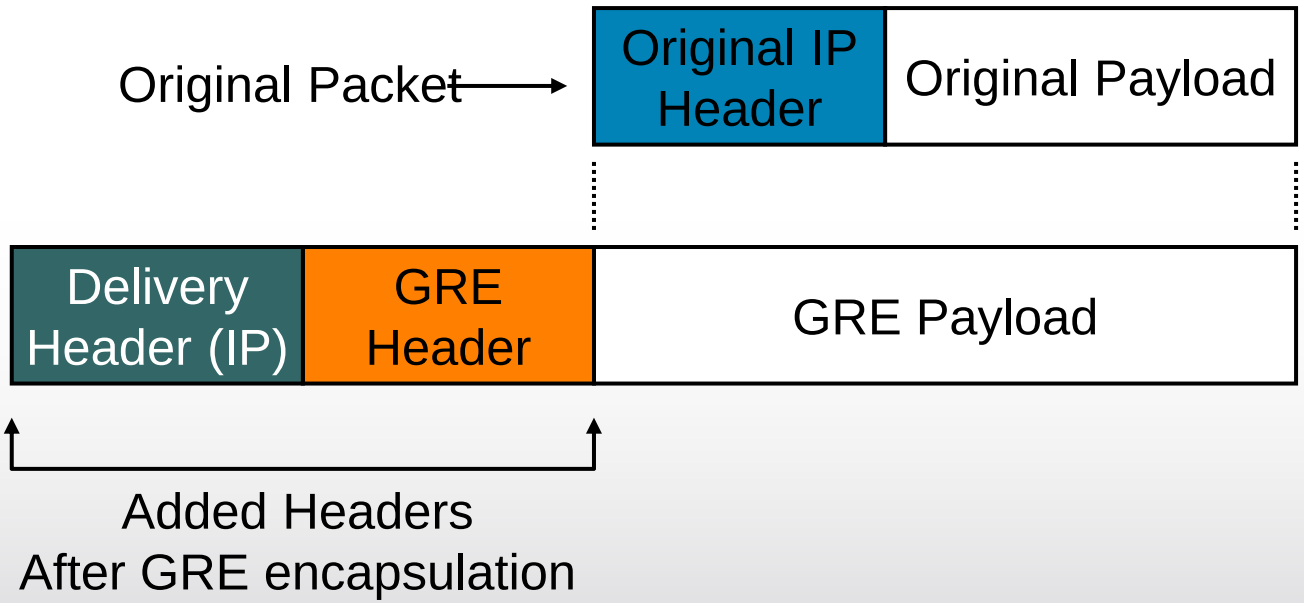
GRE Overview and Troubleshooting

Kyuhwan Kim

GRE Overview

What is GRE?

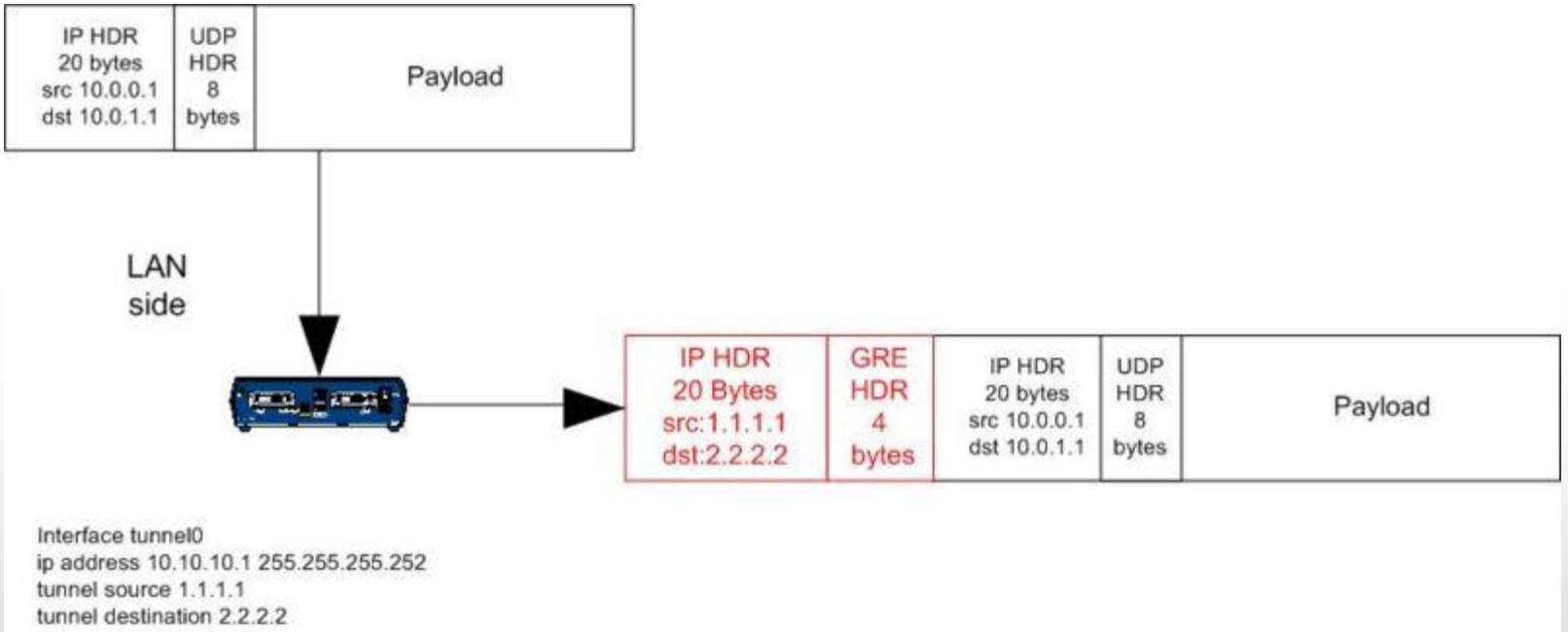
Generic Routing Encapsulation is defined by the IETF in RFC 2784 as a protocol for the encapsulation of a network layer protocol inside another network layer protocol



After GRE encapsulation, two new headers are added to what was the original packet an IP header.

GRE Overview
What is GRE?

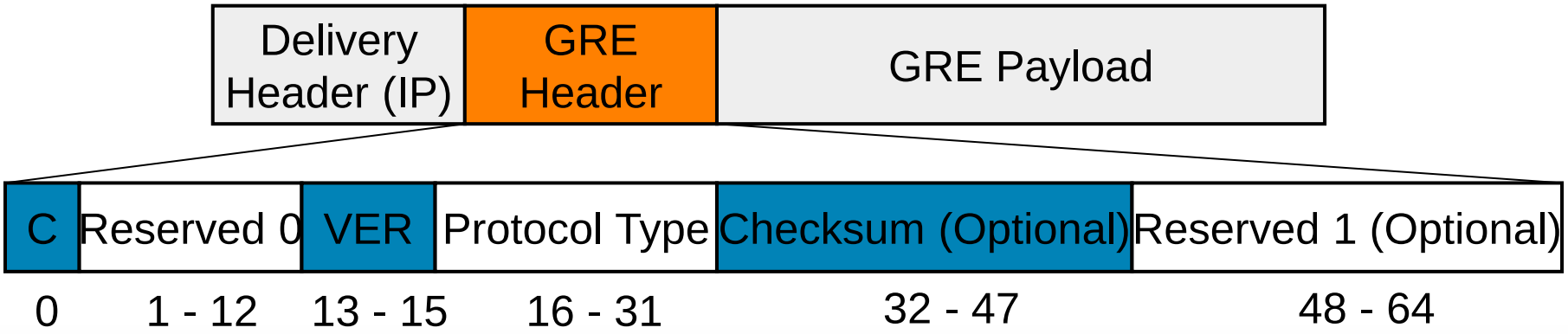
When a packet enters a GRE tunnel, it has an encapsulation process applied where the new IP header and GRE header are appended to the packet - after encapsulation, the addresses in the attached IP header are used for forwarding…



When the packet reaches the other end of the tunnel, it goes through a de-encapsulation process where the additional headers are removed. From this point on, addresses in the original IP header are used for forwarding

GRE Overview
GRE Packet Detail

The Generic Rote Encapsulation header consists of the following fields

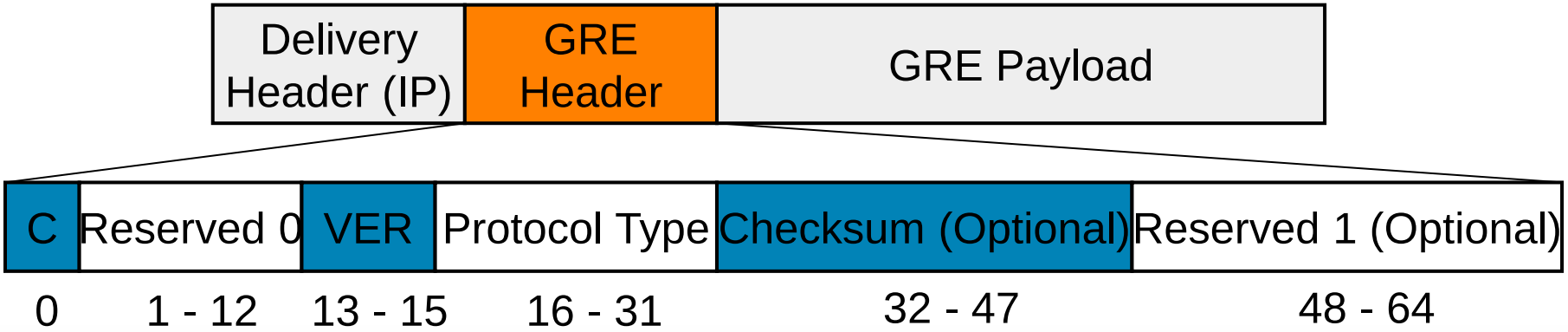


Bits	Field Description
0	The checkpoint present field is a one bit value that, if set to one, indicates the presence of the Checksum and Reserved1 fields and that the Checksum field contains a valid value.
1-12	The receiver must discard any packet where bits 1 through 5 are non-zero unless the receiver implements RFC 1701. Bits 6 through 12 are reserved for future use; however, they must be set to zero and must be ignored on receipt.
13-15	The Version field must contain the value zero.

Continued on next page…

GRE Overview
What is GRE con't?

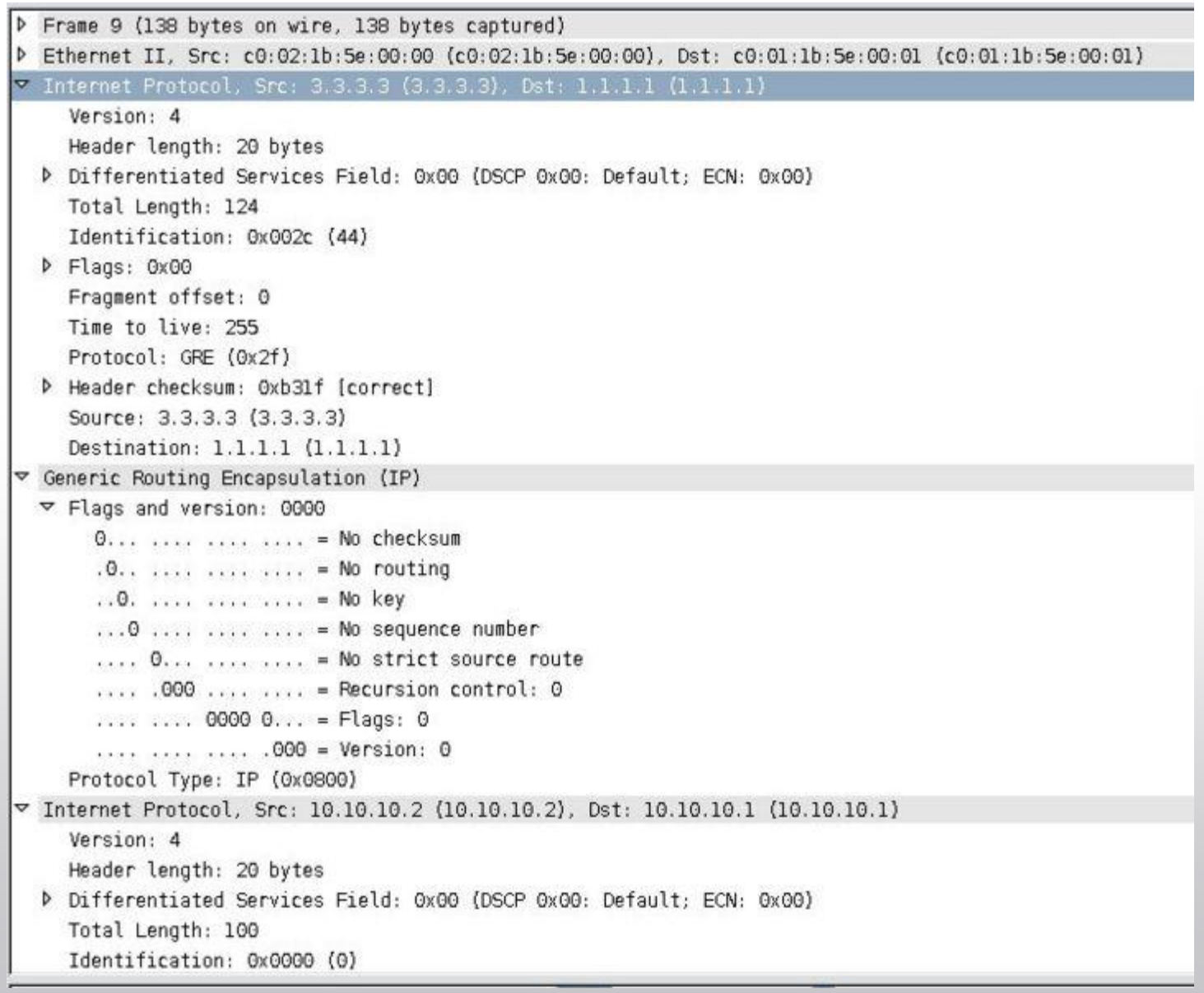
The Generic Rote Encapsulation header consists of the following fields



Bits	Field Description
16-31	This contains the value of the payload packet protocol type and valid values are found in RFC 1700 as ETHER TYPES. The receiver should discard Ethertypes not found in RFC 1700. If the payload being carried is an IPv4 payload, then the Protocol Type should be set to 0x800
32-47	this field contains the checksum of the GRE header and Payload packet. This field is present only if the C bit is set to one.
48-64	This field is reserved for future use and if transmitted must be set to all zeroes.

GRE Overview

- GRE Packet in wireshark



GRE Configuration

- Configuring a GRE tunnel is pretty straight forward.
- The minimum configuration looks like this:

```
interface Tunnel0
    ip address 10.10.10.2 255.255.255.252
    tunnel source 2.2.2.2
    tunnel destination 1.1.1.1
```

- The minimum required is:
  - an ip address used to route through the tunnel.
  - a interface that anchors the tunnel source [should be up].
  - a tunnel destination which *could* be reached via the routing table.

- If all of the 3 above conditions are present, then the GRE tunnel interface on the router will remain up
- If not the tunnel interface will go down → troubleshooting step

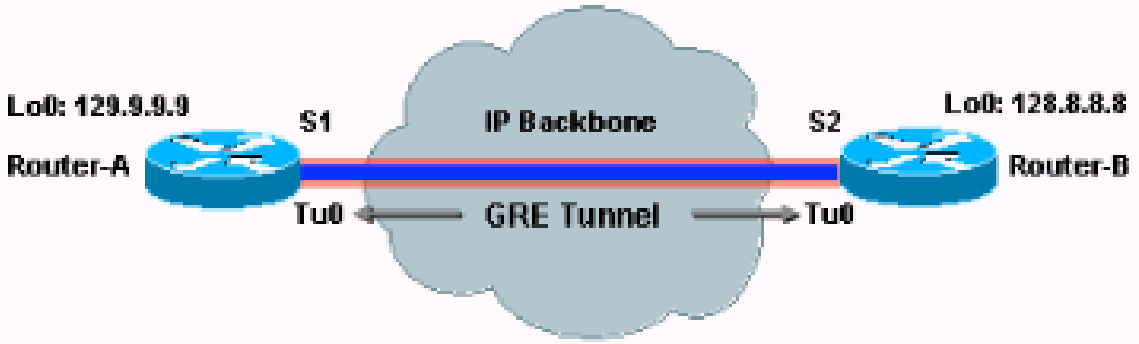
GRE Keepalive

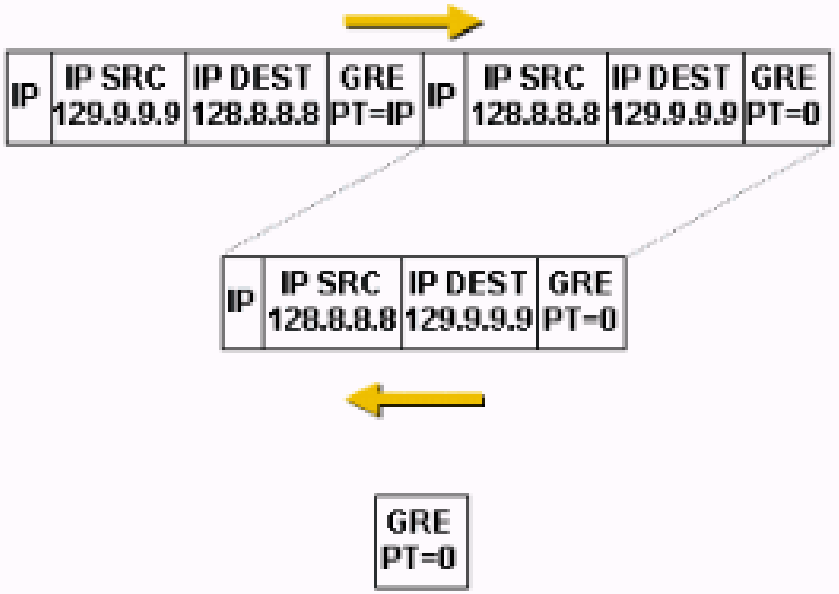
- GRE tunnels are designed to be completely stateless.
  - Each tunnel end-point does not care if other end is up or down
  - Tunnel come up if you just configure tunnel int and source int is up

- How GRE Keepalive works
  - Sender pre-builds keepalive response packet inside of original GRE keepalive
  - Sender sends out keepalive on GRE tunnel
  - Receiver only does GRE decapsulation of the outer GRE IP header and then forward the inner IP GRE packet back to sender by non-GRE interface
  - So, GRE keepalive works fine even if receiver router doesn't support it.

- Keepalive Configuration
  - "Keepalive 5 4" – 5 is interval in seconds and 4 is retry limit
  - Sender increases keepalive counters when it sends out keepalive to receiver
  - Sender reset keepalive counter to zero when it receives response pkt from receiver
  - Sender sends out keepalive and data traffic until keepalive counter reaches retry limit then GRE tunnel become up/down and stop sending data traffic but keepalive still go out

How GRE Keepalive works





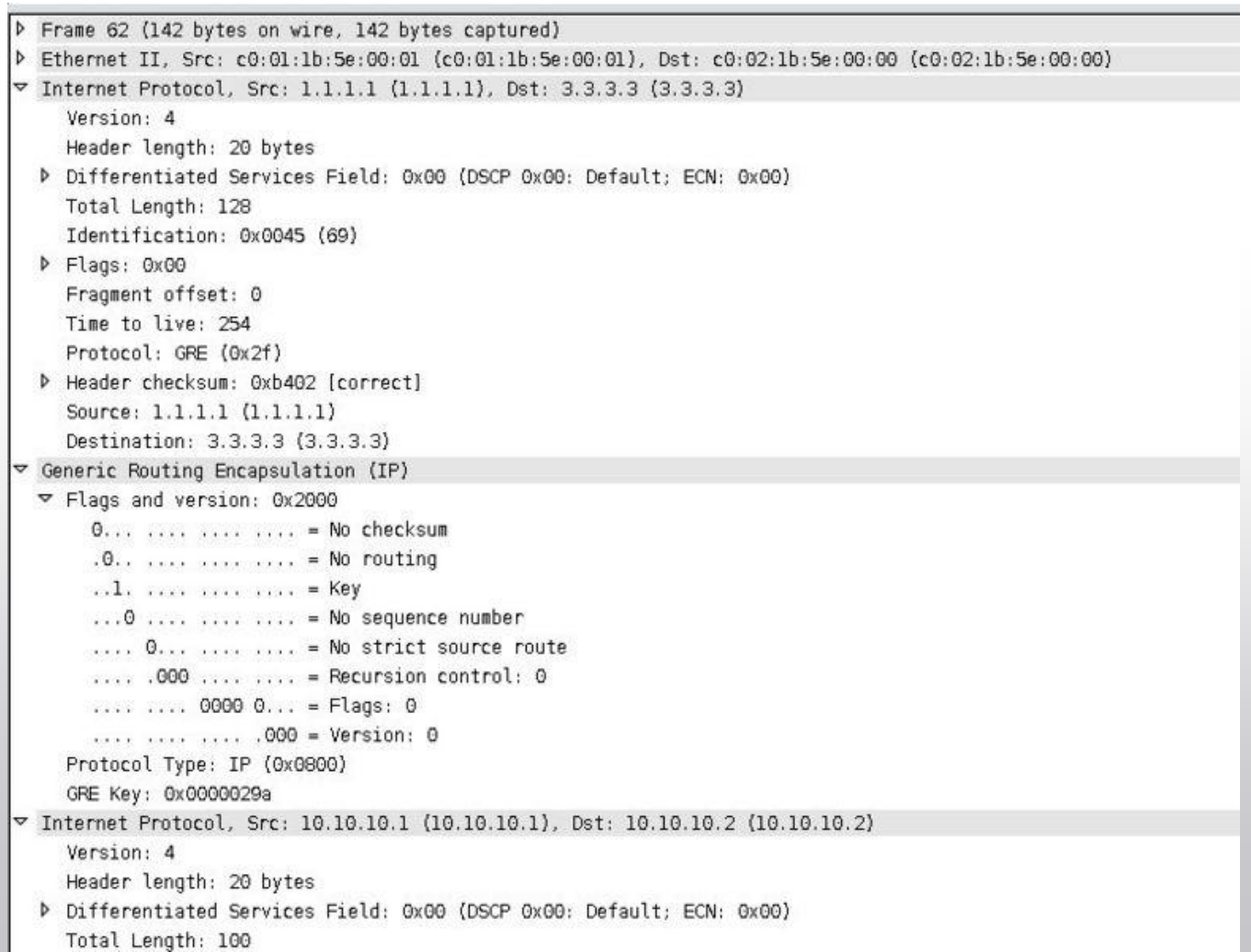
1. Router-A sends a GRE Keepalive to Router-B, PT=IP

2. Router-B decapsulates the outer GRE packet and forwards the inner GRE packet out the physical interface to Router-A

3. As Router-A decapsulates this GRE packet it sees PT=0, which is its keepalive response. It drops the GRE packet and resets the tunnel keepalive counter to 0.

GRE Tunnel Keys

- When you need tunel keys
  - When Multiple GRE tunnels exist between two router → tunnel identification
  - DMVPN on 12.2T or 12.3 require tunnel key → not any more on 12.4 or later
- Tunnel Key must be configured on both end routers
- Tunnel Key adds 4 more bytes to header → 4 Bytes GRE header becomes 8 bytes



GRE troubleshooting

- GRE tunnels is not coming up
  - Each tunnel end-point does not care if other end is up or down
  - Check if
    - Tunnel source interface is up
    - Tunnel destination is routable
    - Tunnel IP address is configured (normal IP or unnumbered IP)
- Debug tunnel

```
R3#debug tunnel ?
  databases  Tunnel databases debugging
  groups     Tunnel groups debugging
  keepalive  Tunnel keepalive debugging
  rbscp      Tunnel RBSCP debugging
  route-via  Tunnel route-via debugging
  source     Debug tunnel source (honors debug condition)

R4#sh run int tu 24
interface Tunnel24
 ip address 3.3.24.4 255.255.255.0
 tunnel source 3.3.34.4
 tunnel destination 3.3.23.2

*Feb  1 00:07:32.265: Tunnel24: GRE/IP encapsulated 3.3.34.4->3.3.23.2 (linktype=7, len=124)
*Feb  1 00:07:32.265: Tunnel24 count tx, adding 0 encap bytes
*Feb  1 00:07:32.269: Tunnel24: GRE/IP to classify 3.3.23.2->3.3.34.4 (tbl=0,"Default" len=124 ttl=253
tos=0x0)
*Feb  1 00:07:32.269: Tunnel24: GRE/IP (PS) to decaps 3.3.23.2->3.3.34.4 (tbl=0,"default" len=124 ttl=253)
*Feb  1 00:07:32.269: Tunnel24: GRE decapsulated IP packet (linktype=7, len=100)
```

GRE troubleshooting

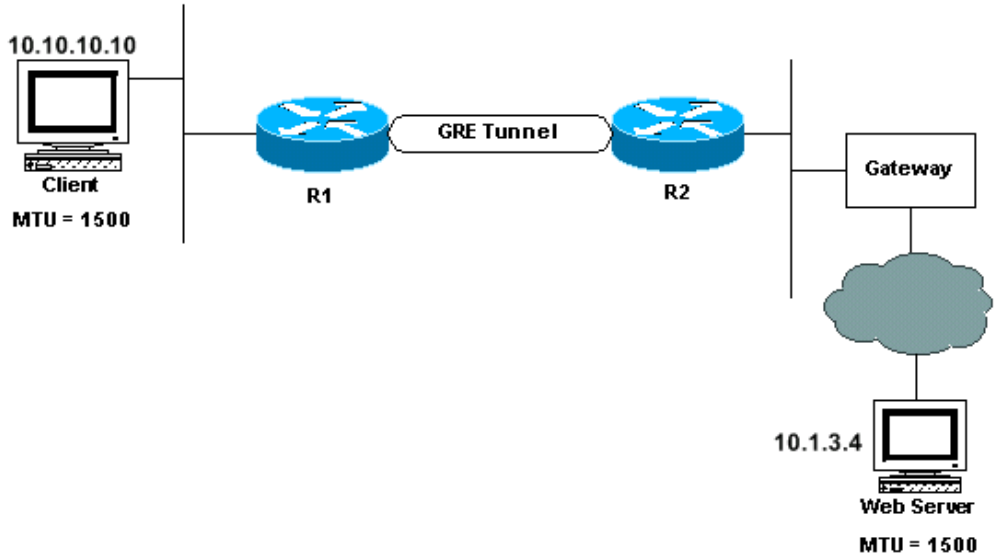
- Recursive routes

```
%TUN-RECURDOWN Interface Tunnel 0 temporarily disabled due to recursive routing
```

  - Peering over GRE tunnel might cause recursive routing loop and tear down GRE
  - GRE tunnel has better metric over physical link as they are directly connected
  - Use proper BW or Cost command on tunnel interface to avoid this problem
- Check CEF and Adjacency

```
R4#sh ip cef 3.3.24.2 det
3.3.24.0/24, epoch 0, flags attached, connected, cover dependents, need deagg
  Interest List:
    - ipv4fib connected receive
  Covered dependent prefixes: 2
    need deagg: 2
  attached to Tunnel24
R4#sh adjacency tunnel 24 det
Protocol Interface              Address
IP      Tunnel24                point2point(6)
                        0 packets, 0 bytes
                        epoch 0
                        sourced in sev-epoch 0
                        Encap length 24
                        4500000000000000FF2F7CC303032204
                        0303170200000800
                        P2P-ADJ
                        Next chain element:
                         IP adj out of Ethernet0/0, addr 3.3.34.3
```

GRE Fragmentation



- How GRE affects MTU
  - Client and Web server negotiate and agree on 1460 MSS
  - MSS = MTU – IP header (20 bytes) – TCP header (20 bytes)
  - WEB server sets “DF BIT” on HTTP return traffic
  - R2 receives 1500 bytes packet from server and tries to send over GRE tunnel
  - GRE MTU is 24 bytes less than Interface (20 outer IP header + 4 b GRE header)
  - R2 can’t fragment packet so it sends out ICMP type 3 code 4 back to web server
  - If DF bit is not set, then R2 fragments 1500 byte packet into 1476 bytes and 44 bytes (24 bytes data + new IP header) → 1500 bytes and 68 bytes GRE pkt
  - Fragmentation uses router CPU power little bit but reassembly uses more CPU and more memory so this will cause more problem on R1
  - “ip tcp adjust-mss 1436” on tunnel interface

http://www.cisco.com/en/US/tech/tk827/tk369/technologies_white_paper09186a00800d6979.shtml

QoS on GRE tunnel

- Cisco QoS on GRE
  - Shaping – GTS and MQC
  - Policing – Hierarchical Policy with shaping on parent/queuing on child
- QoS pre-classify
  - From 11.3, IOS copy Type of Serivce bit to Tunnel or GRE IP header
  - QoS feature can't examine original IP header after it is encrypted or encaptulated
  - "qos pre-classify" make a temp copy of original IP header in memory so QoS can match packets based on original IP header
  - Matching ip address requires "qos pre-classfy" on tunnel interface
  - Matching TOS values, (IP PREC or DSCP) doesn't requre "qos pre-classify"
- Where to apply Service Policy
  - Depends on your goals
  - You can apply either on Tunnel or Physical Interface
  - On tunnel – classify based on pre-tunnel headers
  - On physical int – classify all traffic on interface including all tunnels but only based on post-tunnel hedaers
  - On physical int with qos pre-classfy on tunnel – classify with pre-tunnel header

http://www.cisco.com/en/US/tech/tk543/tk545/technologies_tech_note09186a008017405e.shtml