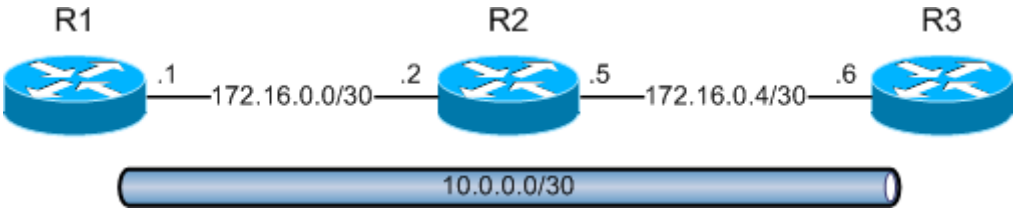
Visualizing tunnels

By stretch | Friday, July 11, 2008 at 11:08 a.m. UTC

When I was first introduced to tunnel interfaces, it took me a while to work out a solid visualization in my mind. Referencing documentation on the topic at the time probably would have sped things along considerably, but I digress. For the novice, I suggest thinking of a tunnel interface as simply adding or removing protocol headers, rather than transmitting packets. Consider the following illustration.



Routers 1, 2, and 3 are physically connected in series using point-to-point links in the 172.16.0.0/24 range. A tunnel between routers 1 and 3 is configured and addressed as 10.0.0.0/30. R1 has been configured to source the tunnel from its physical interface FastEthernet0/0, with the destination address of R3's 172.16.0.6 interface.

```
hostname R1
!
interface Tunnel0
ip address 10.0.0.1 255.255.255.252
tunnel source 172.16.0.1
tunnel destination 172.16.0.6
!
interface FastEthernet0/0
ip address 172.16.0.1 255.255.255.252
```

R3's configuration is a mirror image of R1:

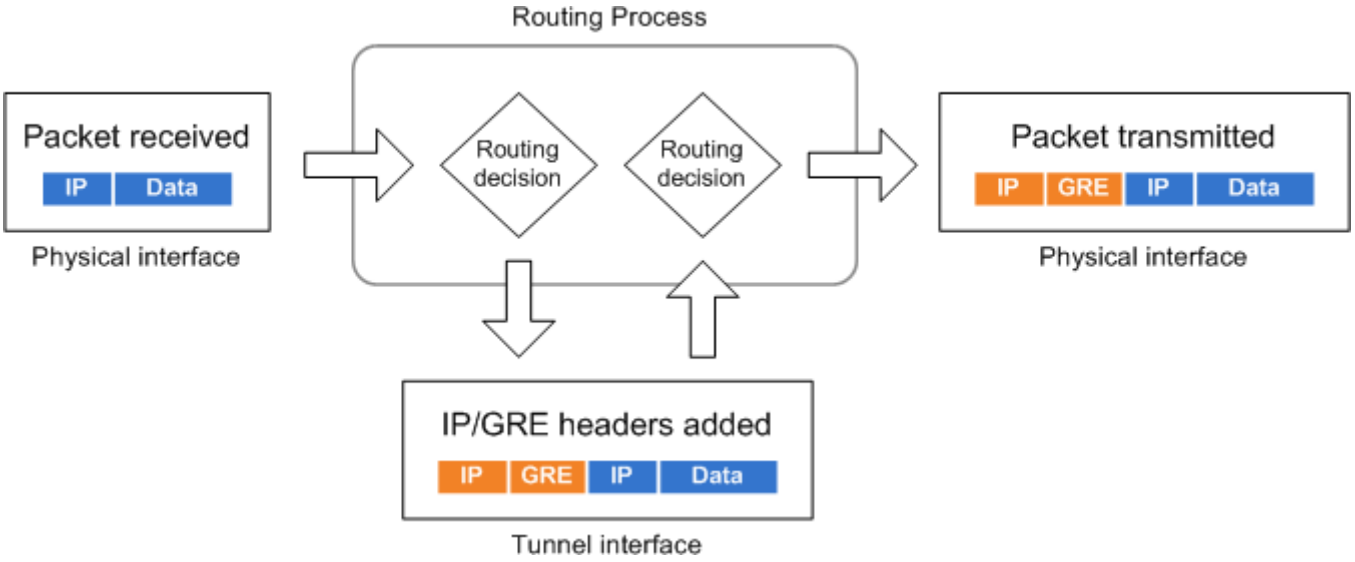
```
hostname R3
!
interface Tunnel0
ip address 10.0.0.2 255.255.255.252
tunnel source 172.16.0.6
tunnel destination 172.16.0.1
!
interface FastEthernet0/0
ip address 172.16.0.6 255.255.255.252
```

R1 and R3 both view the 10.0.0.0/30 network as directly connected. Because we didn't specify a tunnel type in our setup, GRE encapsulation (the default mode) is used.

```
R1# show interface tun0
Tunnel0 is up, line protocol is up
 Hardware is Tunnel
 Internet address is 10.0.0.1/30
 MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
reliability 255/255, txload 1/255, rxload 1/255
 Encapsulation TUNNEL, loopback not set
 Keepalive not set
 Tunnel source 172.16.0.1, destination 172.16.0.6
 Tunnel protocol/transport GRE/IP
Key disabled, sequencing disabled
```

```
  Checksumming of packets disabled
   Tunnel TTL 255
   Fast tunneling enabled
   ...
```

Although the tunnel terminates at a (virtual) interface, it isn't responsible for obtaining layer two adjacency information or for transmitting packets. Instead, packets routed into or out of a tunnel interface have a protocol header added or removed, respectively. Consider what happens when R1 receives a packet to be routed through the GRE tunnel.



1. A packet is received on or sourced from R1.
2. A routing decision is made, and the packet is forwarded "out" the tunnel interface.
3. The tunnel interface encapsulates the packet with a new IP header and a GRE header. Its destination IP address is that of the tunnel destination (172.16.0.6).
4. A second routing decision is made to determine the new packet's outbound interface based on the outermost IP header.
5. The packet is transmitted out the appropriate physical interface.

Having illustrated the outbound tunnel process, it's simple to reverse the flow and examine the inbound process.

1. A packet is received on R3's physical interface.
2. A routing decision determines that the destination address belongs to R3.
3. The router recognizes the destination IP address and GRE header as belonging to the tunnel interface. The tunnel interface removes the outer IP and GRE headers, and the original IP packet is sent back "in" to the router.
4. A second routing decision is performed based on the original destination IP address.
5. The IP TTL is decremented by one and the packet is transmitted out the appropriate interface.

Note that R2 never sees packets destined to or from the 10.0.0.0/30 subnet, only for the tunnel endpoints 172.16.0.1 and 172.16.0.6. There are other, more interesting modes of tunnel encapsulation, such as IPv6-in-IP and IPsec, but the general flow remains the same.

Posted in Routing