Dynamic Multipoint VPN (DMVPN)

By stretch | Wednesday, July 23, 2008 at 3:44 a.m. UTC

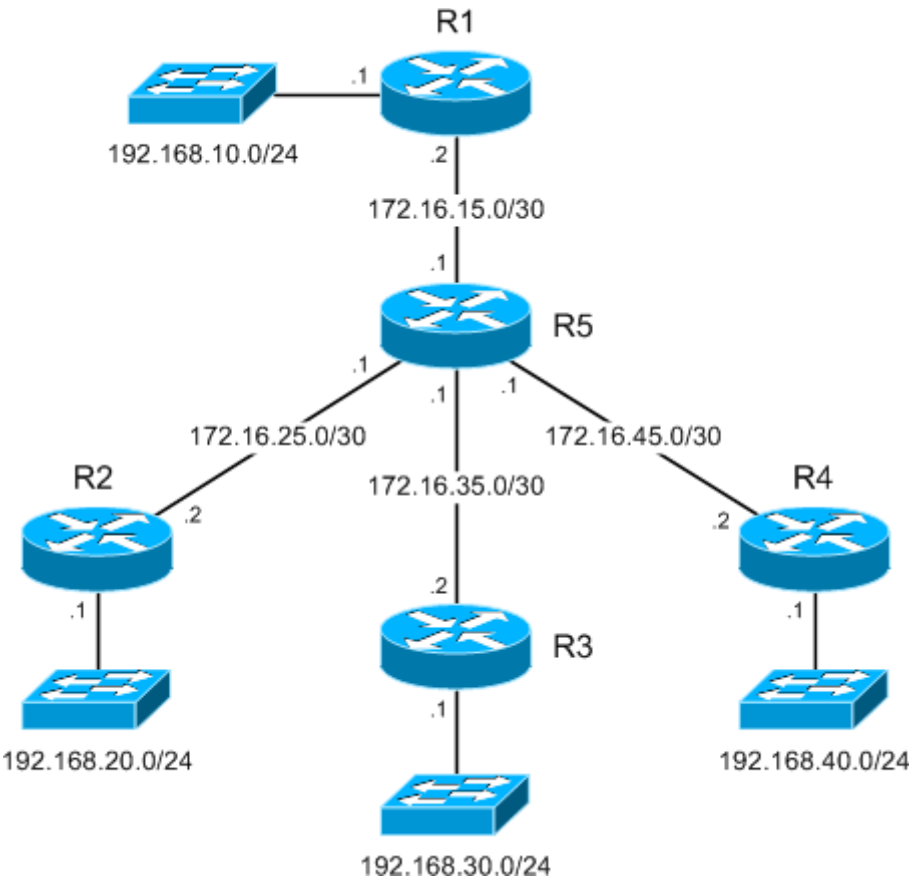
Paul Lavelle wrote in recently to share his experience building a DMVPN lab. He suggested it would make a good blog topic and I agreed. If you're not quite comfortable with GRE tunneling yet, have a look over Visualizing tunnels before continuing.

DMVPN Operation

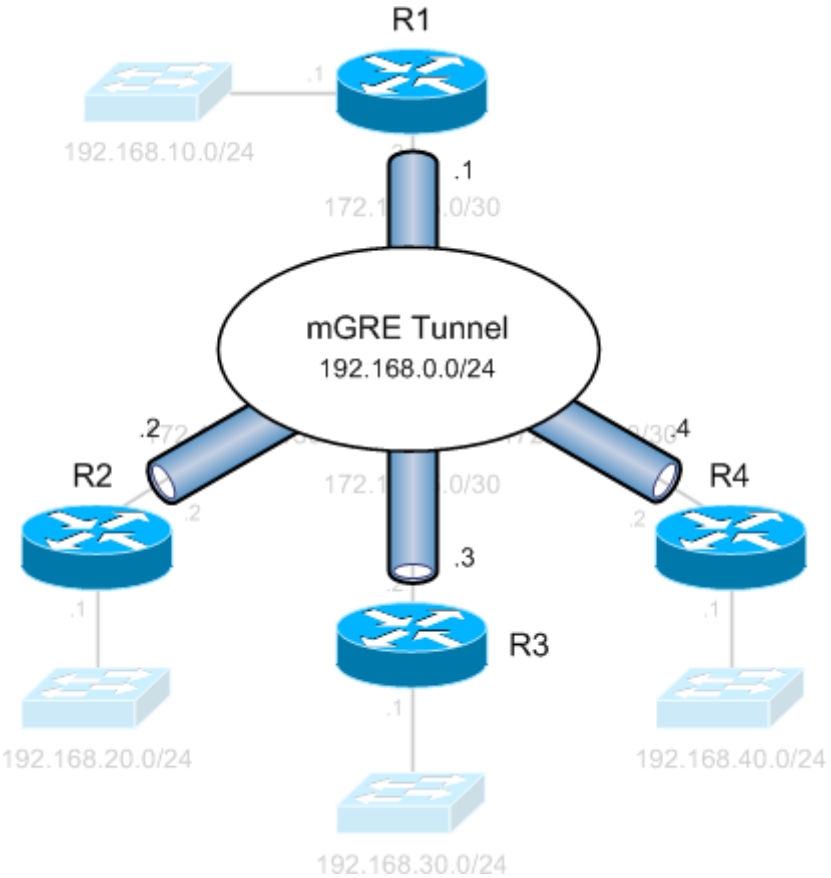
A Dynamic Multipoint VPN is an evolved iteration of hub and spoke tunneling (note that DMVPN itself is not a protocol, but merely a design concept). A generic hub and spoke topology implements static tunnels (using GRE or IPsec, typically) between a centrally located hub router and its spokes, which generally attach branch offices. Each new spoke requires additional configuration on the hub router, and traffic between spokes must be detoured through the hub to exit one tunnel and enter another. While this may be an acceptable solution on a small scale, it easily grows unwieldy as spokes multiply in number.

DMVPN offers an elegant solution to this problem: multipoint GRE tunneling. Recall that a GRE tunnel encapsulates IP packets with a GRE header and a new IP header for transport across an untrusted network. Point-to-point GRE tunnels have exactly two endpoints, and each tunnel on a router requires a separate virtual interface with its own independent configuration. Conversely, a multipoint GRE tunnel allows for more than two endpoints, and is treated as a non-broadcast multiaccess (NBMA) network.

Our lab:



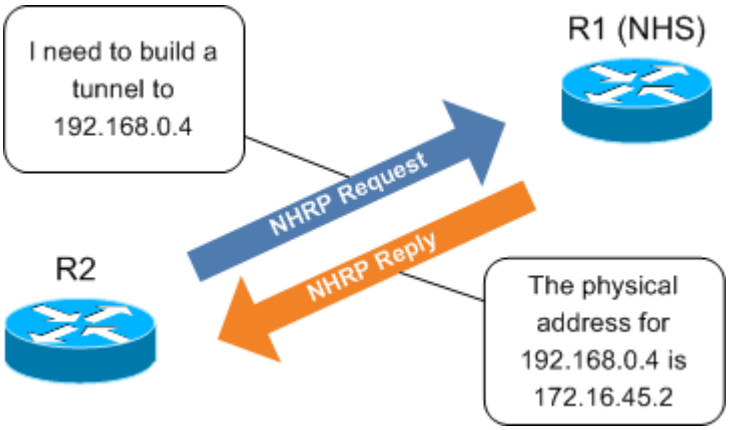
Formation of multipoint GRE tunnels:



While a legacy hub and spoke setup would require three separate tunnels spanning from R1 to each of the spoke routers, we see that multipoint GRE allows all four routers to have a single tunnel interface in the same IP subnet (192.168.0.0/24). This NBMA configuration is enabled by Next Hop Resolution Protocol, which allows multipoint tunnels to be built dynamically.

Next Hop Resolution Protocol (NHRP)

NHRP (defined in RFC 2332) is the catalyst which facilitates dynamic tunnel establishment, providing tunnel-to-physical interface address resolution. NHRP clients (spoke routers) issue requests to the next hop server (hub router) to obtain the physical address of another spoke router.



It is interesting to note that, in our scenario, designation as the NHS is the only attribute which distinguishes R1 as the hub router.

DMVPN Configuration

Let's start by examining the configuration of R1:

```
interface FastEthernet0/0
ip address 172.16.15.2 255.255.255.252
```

```
!
interface Tunnel0
ip address 192.168.0.1 255.255.255.0
ip nhrp map multicast dynamic
ip nhrp network-id 1
tunnel source 172.16.15.2
tunnel mode gre multipoint
```

The first thing you're likely to notice is that the tunnel does not have an explicit destination specified. This is because multipoint tunnels are built dynamically from the DMVPN spokes to the hub router; the hub router doesn't need to be preconfigured with spoke addresses. Also note that the tunnel mode has been designated as multipoint GRE. `ip nhrp network-id 1` uniquely identifies the DMVPN network; tunnels will not form between routers with differing network IDs. `ip nhrp multicast dynamic` enables forwarding of multicast traffic across the tunnel to dynamic spokes (required by most routing protocols).

The configuration of spoke routers is very similar to that of the hub. The configuration presented here is taken from R2.

```
interface FastEthernet0/0
ip address 172.16.25.2 255.255.255.252
!
interface Tunnel0
ip address 192.168.0.2 255.255.255.0
ip nhrp map 192.168.0.1 172.16.15.2
ip nhrp map multicast 172.16.15.2
ip nhrp network-id 1
ip nhrp nhs 192.168.0.1
tunnel source 172.16.25.2
tunnel mode gre multipoint
```

You'll notice two new commands in addition to those found on the hub. `ip nhrp nhs 192.168.0.1` designates R1 as the NHS (the only functionality unique to the hub router), and `ip nhrp map 192.168.0.1 172.16.15.2` statically maps the NHS address to R1's physical address. The `ip nhrp multicast` command also differs slightly from its application on the hub in that multicast traffic is only being allowed from spokes to the hub, not from spoke to spoke.

After completing the tunnel configuration on each router, we can verify that DMVPN sessions have been established between the hub and each spoke:

```
R1# show dmvpn
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
   N - NATed, L - Local, X - No Socket
   # Ent --> Number of NHRP entries with same NBMA peer

Tunnel0, Type:Hub, NHRP Peers:3,
# Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
----- --------------- --------------- ----- -------- -----
1     172.16.25.2     192.168.0.2    UP 00:57:47 D
1     172.16.35.2     192.168.0.3    UP 00:45:56 D
1     172.16.45.2     192.168.0.4    UP 00:45:46 D
```

Dynamic Tunneling

While DMVPN certainly provides a tidy configuration, its brilliance lies in its ability to dynamically establish spoke-to-spoke tunnels. In a legacy hub and spoke design, a packet destined from R2 to R4 would need to be routed through R1, to exit the R2 tunnel and be reencapsulated to enter the R4 tunnel. Clearly a better path lies directly via R5, and DMVPN allows us to take advantage of this.

Check out this packet capture of traffic from R2 to R4. Traffic initially follows the path through R1 as described above, while a dynamic tunnel is built from R2 to R4 using NHRP. After the new tunnel has been established, traffic flows across it, bypassing R1 completely. We can see a new tunnel has been established after traffic destined for R4 has been detected:

```
R2# show dmvpn
...

Tunnel0, Type:Spoke, NHRP Peers:1,
# Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
----- --------------- --------------- ----- -------- -----
1     172.16.15.2     192.168.0.1    UP 01:08:02 S

R2# ping 192.168.0.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.0.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/37/56 ms

R2# show dmvpn
...

Tunnel0, Type:Spoke, NHRP Peers:2,
# Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
----- --------------- --------------- ----- -------- -----
1     172.16.15.2     192.168.0.1    UP 01:08:27 S
1     172.16.45.2     192.168.0.4    UP 00:00:03 D
```

Notice that the tunnel to R4 has been flagged as dynamic, in contrast to the static tunnel to the hub/NHS.

Adding Crypto

Up to this point the tunnels have been configured as cleartext for the sake of simplicity, but in the real world we probably want to include IPsec encryption to protect tunnels traversing an untrusted path. Fortunately, this is as simple as applying an IPsec protection policy to the tunnel interface on each router. (For a brief review of IPsec configuration, check out IPsec quick and dirty.) A bare IPsec profile using a pre-shared ISAKMP key is included below for demonstration.

```
crypto isakmp policy 10
authentication pre-share
crypto isakmp key P4ssw0rd address 172.16.0.0 255.255.0.0
!
crypto ipsec transform-set MyTransformSet esp-aes esp-sha-hmac
!
crypto ipsec profile MyProfile
set transform-set MyTransformSet
!
interface Tunnel0
tunnel protection ipsec profile MyProfile
```

After bumping the tunnel interfaces, we can see the DMVPN sessions have been rebuilt, this time sporting some slick military-grade encryption.

```
R1# show dmvpn
...
```

```
Tunnel0, Type:Hub, NHRP Peers:3,
# Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
----- --------------- --------------- ----- -------- -----
1     172.16.25.2     192.168.0.2    UP 00:02:28 D
1     172.16.35.2     192.168.0.3    UP 00:02:26 D
1     172.16.45.2     192.168.0.4    UP 00:02:25 D

R1# show crypto isakmp sa
IPv4 Crypto ISAKMP SA
dst             src             state          conn-id slot status
172.16.15.2     172.16.35.2     QM_IDLE           1002    0 ACTIVE
172.16.15.2     172.16.25.2     QM_IDLE           1001    0 ACTIVE
172.16.15.2     172.16.45.2     QM_IDLE           1003    0 ACTIVE
```

Posted in Routing, Security