Setting up GRE with IPSEC for a Typical VPN

I made this from a transcription I made of Doug Suida's video at:
https://www.youtube.com/watch?v=2PtK8HgkRvM

A followup that uses the same technique to set up DMVPN (Dynamic Multipoint VPN) is here:
https://www.youtube.com/watch?v=WEzo1UvMpg0

On network mapping it looks like these aren't directly connected but "sh cdp nei only shows f/0/0 - doesn't show as a directly connected neighbor. Neighborship won't happen until you add to OSPF network 10.10.1.0 0.0.0.3 area 0

Outbound connection:

R1 ==> 162.27.193.128/30 ==>========INTERNET========<== 45.12.153.200/30 <== R2

Tunnel inside

10.10.1.1 ==>Tunnel0 << ------> ====== 10.10.1.0/30 ====== < ------- >> Tunnel0 <== 10.10.1.2

Setting up the LANs, uplinks, and tunnel

Local: 192.168.1.0/24

```
R1> int Loopback0
R1> ip address 192.168.1.1
R1> int f0/1
R1> ip address 192.168.1.2
R1> router ospf 123
R1> network 192.168.1.0 0.0.0.255 area 0
```

Uplink

```
R1> int f0/0
R1> ip address 162.27.193.130
R1> exit
```

Tunnel

```
R1> int tunnel0
R1> tunnel source f0/0
R1> tunnel destination 45.12.153.202
R1> ip address 10.10.1.1 255.255.255.252
```

Local: 10.1.1.0/24

```
R2> int Loopback0
R2> ip address 10.1.1.1
R2> int f0/1
R2> ip address 10.1.1.2
R2> router ospf 123
R2> network 10.1.1.0 0.0.0.255 area 0
```

Uplink

```
R2> int f0/0
R2> ip address 45.12.153.202
```

Tunnel

```
R2> int tunnel0
R2> tunnel source f0/0
R2> tunnel destination 162.27.193.130
R2> ip address 10.10.1.2 255.255.255.252
R2>
```

These are also added on both routers but not needed

> tunnel path-mtu-discovery
> ip ospf mtu-ignore

The first one gets the tunnel to do it's own MTU discovery since it is over the internet
The second one is to do the same with OSPF to make sure it stays neat
You can put in a keepalive on the tunnel but don't need it since it's a logical interface
If you are skiddish about OSPF go ahead.
On OSPF: no tunnel = no multicast and no neighborship

So now, after testing and making sure it's all working, we can add standard VPN mechanisms with IPSEC. You might otherwise have many policies and transform sets but here just one.

1. Define traffic to be encrypted

```
R1> ip access-list extended MY-IPSEC-TRAFFIC
R1> remark VPN Traffic
R1> permit gre host 162.27.193.130 host 45.12.153.202
```

You'll do the same on R2 with the addreess reversed

2. Set up Phase I, with a preshared key rather than a certificate

```
R1> crypto isakmp policy 1
R1> authentication pre-share
####### that is where you'd put cert-based rsa-encr, rsa-sig
R1> encryption aes 128
####### other options 3des des aes and 128, 192, 256
R1> hash sha
####### you can also choose to put md5
R1> group 2
####### choose DH group 1, 2 or 5
R1> lifetime 86400
####### 60-86400 seconds (86400 sec = 1 day, is the default)
R1> crypto isakmp key 0 mypassword address 45.12.153.202
####### If the password you are giving here is encrypted or not:  0 is plaintext, 6 is encrypt
```

3. Set up Phase II - make the IPSEC transform set

```
R1> crypto ipsec transform-set TRANSFM-TUN esp-aes 128 esp-sha-hmac
####### There are tons of transforms to use.  The above are the defaults.
R1> mode tunnel
####### Or transport
```

4. Create crypto-map

```
R1>  crypto map CRYPTOMAPPY 1 ipsec-isakmp
R1(config-cryptomap)> description DESC to R2
R1(config-cryptomap)> match address MY-IPSEC-TRAFFIC
####### Name of the ACL made in step 1
```

5. Apply to interfaces

```
R1> set peer 45.12.153.202
R1> set transform-set TRANSFM-TUN

R1> int f0/0
R1> crypto map CRYPTOMAPPY
R1> int tunnel 0
R1> crypto map CRYPTOMAPPY
```

Test

```
R1> show crypto ipsec sa
```

Show stats of # crypted/ decrypted - tells us if it really being encrypted
If you do a sh run config you won't see a lot of the stuff that were used here which were default settings

Valid Encryption Methods
esp-des
esp-3des (default)
esp-aes (128-bit encryption)
esp-aes-192
esp-aes-256
esp-null

Valid Authentication Methods

esp-md5-hmac
esp-sha-hmac (default)

DMVPN - Dynamic Multipoint VPN with mGRE (multipoint GRE), IPSec and NHRP

5 sites (routers), full mesh
R1 has individual tunnels to R2, R3, R4, and R5, and each of those routers have similar tunnels to eachother.
10 tunnels? DMVPN can automate the process. Sort of a hub and spoke configuration mashed in with a frame relay-style network with one hub site (not full mesh FR). With DMVPN, it turns out you don't have to configure all the tunnels after doing those coming out of R1. Those tunnels among the "spokes" routers are built dynamically as-needed by the DMVPN configuration.

If some host off of R4 needs something off of the network hosted by R2, it can dynamically create the tunnel so it can be used, wait for it to time out, and then tear it down again. Say R4 queries the hub router (R1), which provides the next hop information it needs to build the tunnel; it does so, and when they finish tear the tunnel down. In this scenario you don't need anything but interior routing protocols
On routing, lets say R3 adds a network and sends out an advertising update. The update only goes to the hub router- not the others, and it's the hub router that sends the update to the others. Sortof like a NBMA setup, and have to turn off split horizon, add Next Hop Resolution Protocol (NHRP), etc. Also- participant routers are never neighbors with any other router than the "hub" router, even when a tunnel is created. R1 is depended on for information needed to build the tunnels and propagate the routing updates, and no other participating router can communicate with another without this "hub" router as it's resource.

Here we set up R1, then just R2 and R3 - it will become apparent after one spoke is done, that it's a replication. After setting up the "hub router" and the first spoke router, the rest of the spoke routers can use the same stuff with only the tunnel and gateway IPs needing to be set, and EIGRP tuned on.

Overview: EIGRP 10 with:
R1 - f0/0 - 54.45.12.1 to default GW .2/30 - internal network (loopback): 192.168.17.0/24 DMVPN IP: 192.168.1.1
R2 - f0/0 - 54.45.12.5 to 6/30 - internal network (loopback): 192.168.200.0/24 DMVPN IP: 192.168.1.12
R3 - f0/0 - 54.45.12.9 to 10/30 - internal network (loopback): 10.1.1.0/24 DMVPN IP: 192.168.1.3
Set up all of this before starting the exercise. Ping-test all the links and you are ready:

Differences from previous: we create a transform, no cryptomap with ACL. Create ipsec profile instead
1. Set ISAKMP policy and define pre-shared key
R1> crypto isakmp policy 10
R1> authentication pre-share
R1> encryption aes 192
R1> hash md5
R1> group 2
R1> crypto isakmp key 0 ISAKEY address 0.0.0.0 0.0.0.0 <---said isakey was name - no password?
####### Here we don't have a specific IP address like the other example. Put ANY with matching key (with 0's)

2. Phase II - make the IPSEC transform set
R1> crypto ipsec transform-set DMVPN-TRANSFM esp-aes 192 esp-md5-hmac

3. Make a profile (instead of the cryptomap for single tunnel)
R1> ipsec profile DMVPN-PROFILE
R1(ipsec profile)> set security-association lifetime seconds 120
####### 120 sec is the minimum. Is lifetime before tunnel gets torn down!
R1(ipsec profile)> set transform-set DMVPN-TRANSFM
####### DONE WITH IPSEC

Tunnel
R1> int tunnel0
R1> ip address 192.168.1.1 255.255.255.0
R1> no ip redirects
####### MTU for encryption on top of packets
R1> ip mtu 1440

####### Next Hop Resolution Protocol (NHRP) stuff
R1> ip nhrp authentication ISAKEY
R1> ip nhrp network-id 1
####### Allow multicast protocols to use dynamic tunnels, shut off split horizon and next-hop-self
R1> ip nhrp map multicast dynamic
R1> no ip split-horizon eigrp 10
R1> no next-hop-self
R1> tunnel source fastEthernet 0/0
R1> tunnel mode gre multipoint
R1> tunnel key 0
R1> tunnel protect profile DMVPN-PROFILE
R!> router eigrp 10
R1> network 192.168.1.0

So it should be understood at this point:
R2 and R3 both use tunnel0 to get to R1; tunnel0 goes from R2 to R3. It is the SAME tunnel for ALL connected routers and that is what multipoint GRE provides to us. Very much like FR- here one tunnel for the whole mess instead of 5, or 10 tunnels.

R2 - needs the same IPSEC config we just did. On R1, do a sh running config and copy ipsec material. Just copy and paste it into the R2 command line after config t is entered. Done.
Like the IPSEC config, the tunnel config is almost the same... don't copy and paste this quite yet- this is the configuration that is going to be copied, pasted into all the other spoke routers.

R2> int tunnel0
R2> ip address 192.168.1.2 255.255.255.0
R2> no ip redirects
R2> ip mtu 1440
R2> ip nhrp authentication ISAKEY
R2> ip nhrp map multicast dynamic
######## Set the hub for next hop info and map with it's gateway address:
R2> ip nhrp nhs 192.168.1.1
R2> ip nhrp map 192.168.1.1 54.45.12.1
R2> ip nhrp map multicast 54.45.12.1
R2> ip nhrp network-id 1
R2> tunnel source fastEthernet 0/0
R2> tunnel mode gre multipoint
R2> tunnel key 0
R2> tunnel protection profile DMVPN-PROFILE
R2> router eigrp 10
R2> network 192.168.1.0

Other spokes will be the same as R2 except for the router's ip address:
####### config t and copy IPSEC info from sh running confg on R2, paste.
R3> int tunnel0
R3> ip address 192.168.1.3 255.255.255.0
####### Copy and paste the rest of the tunnel from first running config here, then make EIGRP declaration:
R3> router eigrp 10
R3> network 192.168.1.0
That's all. Do this for as many spoks as you want and it "just works"

sh crypto isakmp sa -- this is really just to show phase 1 info - does show tunnel is up or not
sh crypto ipsec sa -- this is for actual tunnel stats - shows traffic data and ecryption %'s

So, what happens when the "hub" goes down? Is there a ways to have failover/redundancy? It turns out, when naming NHS's you're not limited to 1 IP addy- you just need map commands for all added. This addresses the SPOF problem.