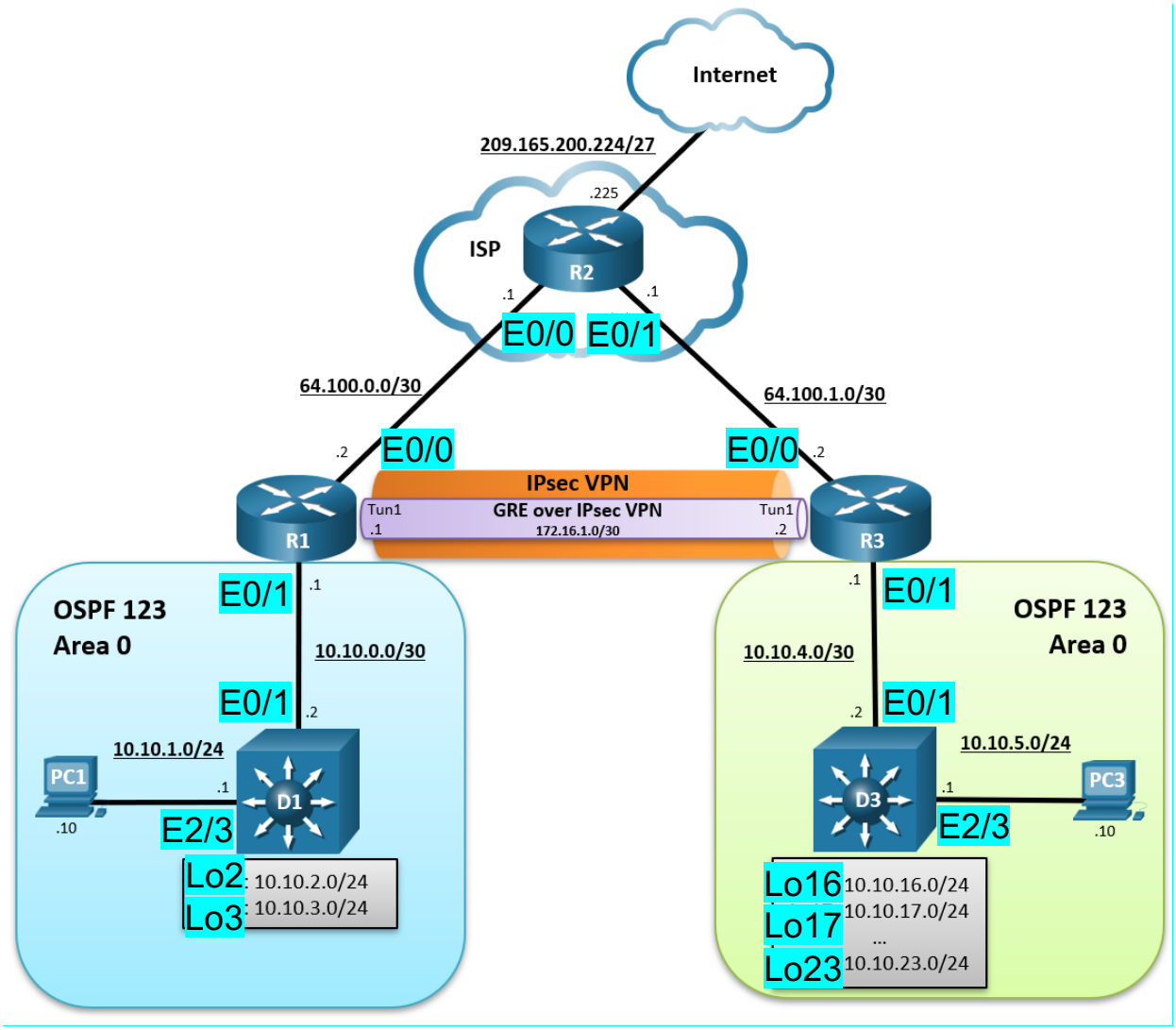
Lab - Implement GRE over IPsec Site-to-Site VPNs

Topology



Addressing Table

Device	Interface	IPv4 Address	Default Gateway
R1	E0/0	64.100.0.2/30	N/A
	E0/1	10.10.0.1/29	
	Tunnel 1	172.16.1.1/30	
R2	E0/0	64.100.0.1/30	N/A

Device	Interface	IPv4 Address	Default Gateway
	E0/1	64.100.1.1/30	
	Lo0	209.165.200.225/27	
R3	E0/0	64.100.1.2/30	N/A
	E0/1	10.10.4.1/30	
	Tunnel 1	172.16.1.2/30	
D1	E0/1	10.10.0.2/29	N/A
	E2/3	10.10.1.1/24	
	Lo2	10.10.2.1/24	
	Lo3	10.10.3.1/24	
D3	E0/1	10.10.0.3/29	N/A
	E2/3	10.10.5.1/24	
	Lo16	10.10.16.1/24	
	Lo17	10.10.17.1/24	
	Lo18	10.10.18.1/24	
	Lo19	10.10.19.1/24	
	Lo20	10.10.20.1/24	
	Lo21	10.10.21.1/24	
	Lo22	10.10.22.1/24	
	Lo23	10.10.23.1/24	
PC1	NIC	10.10.1.10/24	10.10.1.1
PC3	NIC	10.10.5.10/24	10.10.5.1

Objectives

Part 1: Build the Network, Configure Basic Device Settings and Static Routing

Part 2: Configure GRE over IPsec using Crypto Maps on R1

Part 3: Configure GRE over IPsec using a Tunnel IPsec Profile on R3

Part 4: Verify the GRE over IPsec Tunnel on R1 and R3

Background / Scenario

IPsec site-to-site VPNs can only send unicast IP traffic. Therefore, they do not support routing protocols that require multicast or broadcast communication.

GRE is a VPN tunneling technology that permits unicast, multicast, and broadcast traffic. However, GRE does not encrypt traffic.

Combined together, GRE can tunnel multicast and broadcast traffic, and IPsec can secure GRE traffic. GRE over IPsec provides security and support for routing protocols.

In this lab, you will build and configure:

- **GRE over IPsec VPN using a Crypto Map** – This is a common method of enabling multicast traffic over the VPN.
- **GRE over IPsec VPN using an IPsec Tunnel Profile** – This is a newer method of implementing GRE over IPsec using IPsec profiles.

Note: This lab is an exercise in developing, deploying, and verifying how VNPs operates and does not reflect networking best practices.

Note: The routers used with this CCNP hands-on lab are Cisco 4221 routers and the two Layer 3 switches are Catalyst 3650 switches. Other routers and Layer 3 switches and Cisco IOS versions can be used. Depending on the model and Cisco IOS version, the commands available and the output produced might vary from what is shown in the labs.

Note: Ensure that the routers and switches have been erased and have no startup configurations. If you are unsure contact your instructor.

Required Resources

- 3 Routers (Cisco 4221 with Cisco IOS XE Release 16.9.4 universal image or comparable)
- 2 Switches (Cisco 3650 with Cisco IOS XE Release 16.9.4 universal image or comparable)
- 2 PCs (Choice of operating system with a terminal emulation program installed)
- Console cables to configure the Cisco IOS devices via the console ports
- Ethernet cables as shown in the topology

Instructions

Part 1: Build the Network, Configure Basic Device Settings and Static Routing

In Part 1, you will set up the network topology, configure basic settings, interface addressing, and single-area OSPFv2 on the routers.

Step 1: Cable the network as shown in the topology.

Attach the devices as shown in the topology diagram, and cable as necessary.

Step 2: Configure basic settings for the routers.

a. Console into each router and switch, enter global configuration mode, and apply the basic settings, and interface addressing. A command list for each device is provided for your reference.

Routing is enabled as follows:

- R2 has a static route to the networks connected to R1 (i.e., 10.10.0.0/22) and two static routes to the networks connected to R3 (i.e., 10.10.4.0/22, 10.10.16.0/21).
- R1 and R3 each have a default static route to R2.
- OSPFv2 routing is enabled between R1 and D1, and R1 is propagating the default route to D1.
- OSPFv2 routing is enabled between R3 and D3, and R3 is propagating the default route to D3.
- A command list for each device is listed below to perform initial configurations.

Router R1

```
hostname R1
no ip domain lookup
```

```
line con 0
 logging sync
 exec-time 0 0
 exit
banner motd # This is R1, Implement GRE over IPsec Site-to-Site VPNs #
interface Ethernet0/0
 description Connection to R2
 ip add 64.100.0.2 255.255.255.252
 no shut
 exit
interface Ethernet0/1
 description Connection to D1
 ip address 10.10.0.1 255.255.255.252
 no shut
 exit
router ospf 123
 router-id 1.1.1.1
 auto-cost reference-bandwidth 1000
 network 10.10.0.0 0.0.0.3 area 0
 default-information originate
exit
ip route 0.0.0.0 0.0.0.0 64.100.0.1
```

Router R2

```
hostname R2
no ip domain lookup
line con 0
 logging sync
 exec-time 0 0
 exit
banner motd # This is R2, Implement GRE over IPsec Site-to-Site VPNs #
interface Ethernet0/0
 description Connection to R1
 ip add 64.100.0.1 255.255.255.252
 no shut
 exit
interface Ethernet0/1
 description Connection to R3
 ip address 64.100.1.1 255.255.255.252
 no shut
 exit
int lo0
 description Internet simulated address
 ip add 209.165.200.225 255.255.255.224
 exit
ip route 0.0.0.0 0.0.0.0 Loopback0
```

```
ip route 10.10.0.0 255.255.252.0 64.100.0.2
ip route 10.10.4.0 255.255.252.0 64.100.1.2
ip route 10.10.16.0 255.255.248.0 64.100.1.2
```

Router R3

```
hostname R3
no ip domain lookup
line con 0
 logging sync
 exec-time 0 0
 exit
banner motd # This is R3, Implement GRE over IPsec Site-to-Site VPNs #
interface Ethernet0/0
 description Connection to R2
 ip add 64.100.1.2 255.255.255.252
 no shut
 exit
interface Ethernet0/1
 description Connection to D3
 ip address 10.10.4.1 255.255.255.252
 no shut
 exit
ip route 0.0.0.0 0.0.0.0 64.100.1.1
router ospf 123
 router-id 3.3.3.1
 auto-cost reference-bandwidth 1000
 network 10.10.4.0 0.0.0.3 area 0
 default-information originate
exit
```

Switch D1

```
hostname D1
no ip domain lookup
line con 0
 exec-timeout 0 0
 logging synchronous
 exit
banner motd # This is D1, Implement GRE over IPsec Site-to-Site VPNs #
interface Ethernet0/1
 description Connection to R1
 no switchport
 ip address 10.10.0.2 255.255.255.252
 no shut
 exit
interface Ethernet2/3
 description Connection to PC1
```

```
 no switchport
 ip address 10.10.1.1 255.255.255.0
 no shut
 exit
int Lo2
 description Loopback to simulate an OSPF network
 ip add 10.10.2.1 255.255.255.0
 ip ospf network point-to-point
exit
int Lo3
 description Loopback to simulate an OSPF network
 ip add 10.10.3.1 255.255.255.0
 ip ospf network point-to-point
exit
ip routing
router ospf 123
 router-id 1.1.1.2
 auto-cost reference-bandwidth 1000
 network 10.10.0.0 0.0.3.255 area 0
exit
```

Switch D3

```
hostname D3
no ip domain lookup
line con 0
 logging sync
 exec-time 0 0
 exit
banner motd # This is D3, Implement GRE over IPsec Site-to-Site VPNs #
interface Ethernet0/1
 description Connection to R3
 no switchport
 ip address 10.10.4.2 255.255.255.252
 no shut
 exit
interface Ethernet2/3
 description Connection to PC3
 no switchport
 ip address 10.10.5.1 255.255.255.0
 no shut
 exit
int Lo16
 description Loopback to simulate an OSPF network
 ip add 10.10.16.1 255.255.255.0
 ip ospf network point-to-point
exit
```

```
int Lo17
 description Loopback to simulate an OSPF network
 ip add 10.10.17.1 255.255.255.0
 ip ospf network point-to-point
 exit
int Lo18
 description Loopback to simulate an OSPF network
 ip add 10.10.18.1 255.255.255.0
 ip ospf network point-to-point
 exit
int Lo19
 description Loopback to simulate an OSPF network
 ip add 10.10.19.1 255.255.255.0
 ip ospf network point-to-point
 exit
int Lo20
 description Loopback to simulate an OSPF network
 ip add 10.10.20.1 255.255.255.0
 ip ospf network point-to-point
 exit
int Lo21
 description Loopback to simulate an OSPF network
 ip add 10.10.21.1 255.255.255.0
 ip ospf network point-to-point
 exit
int Lo22
 description Loopback to simulate an OSPF network
 ip add 10.10.22.1 255.255.255.0
 ip ospf network point-to-point
 exit
int Lo23
 description Loopback to simulate an OSPF network
 ip add 10.10.23.1 255.255.255.0
 ip ospf network point-to-point
 exit
ip routing
router ospf 123
 router-id 3.3.3.2
 auto-cost reference-bandwidth 1000
 network 10.10.4.0 0.0.1.255 area 0
 network 10.10.16.0 0.0.7.255 area 0
 exit
```

b. Save the running configuration to startup-config.

Step 3: Configure PC1 and PC3 with IP addressing.

Configure the two PCs with the IP addresses listed in the Address Table. Also configure their respective default gateways.

Step 4: On PC1, verify end-to-end connectivity.

a. From PC1, **ping** PC3 (10.10.5.10).

```
PC1> ping 10.10.5.10
```

The pings should be successful. If the pings are unsuccessful, troubleshoot the basic device configurations before continuing.

b. From PC1, **ping** the first loopback on D3 (10.10.16.1).

```
PC1> ping 10.10.16.1
```

c. Finally, from PC1, **ping** the default gateway loopback on R2 (209.165.200.225).

```
PC1> ping 209.165.200.225
```

Step 5: Verify the routing table of R1 and R3.

a. Verify the OSPF routing table of R1.

```
R1# show ip route ospf | begin Gateway
Gateway of last resort is 64.100.0.1 to network 0.0.0.0

      10.0.0.0/8 is variably subnetted, 5 subnets, 3 masks
O        10.10.1.0/24 [110/11] via 10.10.0.2, 00:02:45, Ethernet0/1
O        10.10.2.0/24 [110/2] via 10.10.0.2, 00:02:45, Ethernet0/1
O        10.10.3.0/24 [110/2] via 10.10.0.2, 00:02:45, Ethernet0/1
```

The routing table of R1 only contains the local OSPF routes. The routing table confirms that R1 has knowledge of the networks connected to D1. Notice however, that R1 has no knowledge of the routes connected to the R3 OSPF domain. The reason why PC1 can still reach PC3 is because R1 has a default static route to R2. R1 forwarded the traffic to R2 because it did not know where the 10.10.5.0 network was. R2 has a static route to this network and therefore forwarded it to R3.

b. Verify the routing table of R3.

```
R3# show ip route ospf | begin Gateway
Gateway of last resort is 64.100.1.1 to network 0.0.0.0

      10.0.0.0/8 is variably subnetted, 11 subnets, 3 masks
O        10.10.5.0/24 [110/11] via 10.10.4.2, 00:00:41, Ethernet0/1
O        10.10.16.0/24 [110/2] via 10.10.4.2, 00:00:41, Ethernet0/1
O        10.10.17.0/24 [110/2] via 10.10.4.2, 00:00:41, Ethernet0/1
O        10.10.18.0/24 [110/2] via 10.10.4.2, 00:00:41, Ethernet0/1
O        10.10.19.0/24 [110/2] via 10.10.4.2, 00:00:41, Ethernet0/1
O        10.10.20.0/24 [110/2] via 10.10.4.2, 00:00:41, Ethernet0/1
O        10.10.21.0/24 [110/2] via 10.10.4.2, 00:00:41, Ethernet0/1
O        10.10.22.0/24 [110/2] via 10.10.4.2, 00:00:41, Ethernet0/1
O        10.10.23.0/24 [110/2] via 10.10.4.2, 00:00:41, Ethernet0/1
```

Like R1, the routing table of R3 only contains its local routes.

Part 2: Configure GRE over IPsec using a Crypto Map on R1

A limitation of IPsec VPNs is that it only forwards unicast traffic across the VPN tunnel. Therefore, routing protocol traffic is not propagated across the VPN tunnel.

What if the network policy was to enable the OSPF routing protocol to operate over the IPsec VPN tunnel? This could be accomplished by configuring GRE over IPsec VPN.

There are two methods to configure GRE over IPsec:

- **Crypto maps** - This is an older, classic method of configuring GRE over IPsec.
- **Tunnel IPsec profiles** - This is a newer and simpler method of configuring GRE over IPsec.

Note: Crypto maps and IPsec profile configuration options are compatible with each other.

Although Tunnel IPsec profiles is now the preferred method to configure GRE over IPsec, crypto maps are still widely deployed and should be understood.

In this part, we will configure GRE over IPsec using crypto maps on R1.

Step 1: On R1, configure the ISAKMP policy and pre-shared key.

Like site-to-site VPNs using crypto maps, GRE over IPsec also requires an ISAKMP policy configuration and pre-shared key configured.

In this lab, we will use the following parameters for the ISAKMP policy 10 on R1:

- Encryption: **aes 256**
- Hash: **sha256**
- Authentication method: **pre-share key**
- Diffie-Hellman group: **14**
- Lifetime: **3600** seconds (60 minutes / 1 hour)

a. Configure ISAKMP policy 10 on R1:

```
R1(config)# crypto isakmp policy 10
R1(config-isakmp)# encryption aes 256
R1(config-isakmp)# hash sha256
R1(config-isakmp)# authentication pre-share
R1(config-isakmp)# group 14
R1(config-isakmp)# lifetime 3600
R1(config-isakmp)# exit
```

b. Configure the pre-shared key of **cisco123** on R1. This command points to the remote peer R3 G0/0/0 IP address.

Note: Production networks should use longer and more complex keys.

```
R1(config)# crypto isakmp key cisco123 address 64.100.1.2
```

Step 2: On R1, configure the transform set and VPN ACL.

a. Create a transform set called GRE-VPN using AES 256 cipher with ESP and the SHA 256 hash function.

```
R1(config)# crypto ipsec transform-set GRE-VPN esp-aes 256 esp-sha256-hmac
```

b. Unlike a site-to-site IPsec VPN, the transform must use **transport** mode. The **mode** command is used to identify the type of tunnel that will be established. The default is mode tunnel mode. However, GRE over IPsec should be configured using the **mode transport** command.

```
R1(cfg-crypto-trans)# mode transport
R1(cfg-crypto-trans)# exit
```

c. Next, create a named extended ACL called GRE-VPN-ACL that makes the tunnel interface traffic interesting.

```
R1(config)# ip access-list extended GRE-VPN-ACL
R1(config-ext-nacl)# permit gre host 64.100.0.2 host 64.100.1.2
R1(config-ext-nacl)# exit
```

Step 3: On R1, configure the crypto map and apply it to the interface.

a. Create a crypto map called **GRE-CMAP** that associates the new GRE-VPN-ACL, transform set, and peer.

```
R1(config)# crypto map GRE-CMAP 10 ipsec-isakmp
% NOTE: This new crypto map will remain disabled until a peer
        and a valid access list have been configured.
R1(config-crypto-map)# match address GRE-VPN-ACL
R1(config-crypto-map)# set transform-set GRE-VPN
R1(config-crypto-map)# set peer 64.100.1.2
R1(config-crypto-map)# exit
```

b. Finally, assign a crypto map called **GRE-MAP** on G0/0/0.

```
R1(config)# interface e0/0
R1(config-if)# crypto map GRE-CMAP
```

Step 4: On R1, configure the GRE tunnel interface.

Configure a GRE tunnel interface as shown. To enable GRE on the tunnel interface, the **tunnel mode gre ipv4** command is required. However, this command is enabled by default and will therefore not be configured in our example.

```
R1(config)# interface Tunnel1
R1(config-if)# bandwidth 4000
R1(config-if)# ip address 172.16.1.1 255.255.255.252
R1(config-if)# ip mtu 1400
R1(config-if)# tunnel source 64.100.0.2
R1(config-if)# tunnel destination 64.100.1.2
R1(config-if)# end
*Feb 19 17:54:21.381: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel1, changed
state to down
*Feb 19 17:54:23.689: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel1, changed
state to up
```

Note: To support IPv6 traffic, configure the **tunnel mode gre ipv6** command.

Part 3: Configure GRE over IPsec using a Tunnel IPsec Profile on R3

In this part, you will configure R3 to support GRE over IPsec. Although we could configure R3 to also use crypto maps, we will instead configure GRE over IPsec using IPsec profiles to demonstrate how both methods are compatible with each other. This means that a router configured with GRE over IPsec using crypto maps can communicate with GRE over IPsec using tunnel IPsec profiles.

A tunnel IPsec profiles requires fewer commands than the crypto map method. However, it still requires an ISAKMP policy configuration, pre-shared key, and transform set to be configured. It will then still require a tunnel interface to be created.

In this part, we will configure GRE over IPsec using tunnel IPsec profiles on R3.

Step 1: On R3, configure the ISAKMP policy, pre-shared key, and transform set.

In this step, we will configure the same parameters for the ISAKMP policy 10 that we configured on R1.

a. Configure ISAKMP policy 10 on R3:

```
R3(config)# crypto isakmp policy 10
R3(config-isakmp)# encryption aes 256
R3(config-isakmp)# hash sha256
R3(config-isakmp)# authentication pre-share
R3(config-isakmp)# group 14
R3(config-isakmp)# lifetime 3600
R3(config-isakmp)# exit
```

b. Configure the pre-shared key of **cisco123** on R1. This command points to the remote peer R3 G0/0/0 IP address.

Note: Production networks should have longer and more complex keys.

```
COMMENT: Changed address in following command
R3(config)# crypto isakmp key cisco123 address 64.100.0.2
```

a. Create a new transform set called GRE-VPN using the same security parameters and transport mode that we configured on R1. Also configure the **mode transport** command.

```
R3(config)# crypto ipsec transform-set GRE-VPN esp-aes 256 esp-sha256-hmac
R3(cfg-crypto-trans)# mode transport
R3(cfg-crypto-trans)# exit
```

Step 2: On R3, configure the IPsec profile.

a. Instead of a crypto map, we will configure an IPsec profile called GRE-PROFILE using the **crypto ipsec profile** *ipsec-profile-name* global configuration command.

```
R3(config)# crypto ipsec profile GRE-PROFILE
```

b. In IPsec profile configuration mode, specify the transform set to be negotiated using the **set transform-set** *transform-set-name* command. Multiple transform sets can be specified in order of priority. The fist transform-set-name specified is the highest priority.

```
R3(ipsec-profile)# set transform-set GRE-VPN
R3(ipsec-profile)# exit
```

Step 3: On R3, configure the tunnel interface.

a. On R3, configure a GRE tunnel interface.

```
R3(config)# interface Tunnel1
R3(config-if)# bandwidth 4000
R3(config-if)# ip address 172.16.1.2 255.255.255.252
R3(config-if)# ip mtu 1400
R3(config-if)# tunnel source 64.100.1.2
R3(config-if)# tunnel destination 64.100.0.2
R3(config-if)#
*Feb 19 17:57:10.415: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Tunnel1, changed state to down
```

```
R3(config-if)#
*Feb 19 17:57:12.660: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Tunnel1, changed state to up
```

b. Apply the IPsec profile GRE-PROFILE to the Tunnel 1 interface using the **tunnel protection ipsec profile** *profile-name* command.

```
R3(config-if)# tunnel protection ipsec profile GRE-PROFILE
*Feb 19 17:58:09.299: %CRYPTO-6-ISAKMP_ON_OFF: ISAKMP is ON
R3(config-if)# end
```

Step 4: On R1 and R3, enable OSPF routing on the tunnel interface.

Verify that the GRE over IPsec VPN is operational.

a. On R1, perform an extended ping to the R3 10.10.16.1 interface.

```
R1# ping 10.10.16.1 source 10.10.0.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.10.16.1, timeout is 2 seconds:
Packet sent with a source address of 10.10.0.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 2/2/3 ms
```

b. The pings are successful, and it appears that the VPN is operational. On R1, verify the IPsec SA encrypted and decrypted statistics.

```
R1# show crypto ipsec sa | include encrypt|decrypt
      #pkts encaps: 0, #pkts encrypt: 0, #pkts digest: 0
      #pkts decaps: 0, #pkts decrypt: 0, #pkts verify: 0
```

No packets were encrypted.

c. From D1, trace the path taken to the R3 10.10.16.1 interface.

```
D1# trace 10.10.16.1
Type escape sequence to abort.
Tracing the route to 10.10.16.1
VRF info: (vrf in name/id, vrf out name/id)
  1 10.10.0.1 0 msec 9 msec 0 msec
  2 64.100.0.1 0 msec 8 msec 0 msec
  3 64.100.1.2 0 msec 0 msec 9 msec
  4 10.10.4.2 0 msec *  0 msec
```

The packet sent by D1 takes the route R1, R2, R3, and then D3 path. It does not use the GRE over IPsec VPN.

The reason for this is because the VPN ACL has been altered to make GRE traffic interesting. Therefore, we must generate traffic that will use the tunnel interface. To do so, we will configure the OSPF routing protocol to enable OSPF on the tunnel interface.

d. On R1, configure OSPF to advertise the tunnel interfaces.

```
R1(config)# router ospf 123
R1(config-router)# network 172.16.1.0 0.0.0.3 area 0
```

e. On R3, configure OSPF to advertise the tunnel interfaces.

```
R3(config)# router ospf 123
R3(config-router)# network 172.16.1.0 0.0.0.3 area 0
R3(config-router)#
*Feb 19 18:01:18.613: %OSPF-5-ADJCHG: Process 123, Nbr 1.1.1.1 on Tunnel1
from LOADING to FULL, Loading Done
```

Notice the OSPF adjacency message that appears when the **network** command is entered.

Part 4: Verify the GRE over IPsec Tunnel on R1 and R3

Now that the GRE over IPsec has been configured, we must verify that the tunnel interfaces are correctly enabled, that the crypto session is active, and then generate traffic to confirm it is traversing securely over the IPsec tunnel.

Step 1: On R1 and R3, verify the tunnel interfaces.

a. Use the **show interfaces tunnel 1** command to verify the interface settings.

```
R1# show interfaces tunnel 1
Tunnel1 is up, line protocol is up
  Hardware is Tunnel
  Internet address is 172.16.1.1/30
  MTU 9976 bytes, BW 4000 Kbit/sec, DLY 50000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel linestate evaluation up
  Tunnel source 64.100.0.2, destination 64.100.1.2
  Tunnel protocol/transport GRE/IP
    Key disabled, sequencing disabled
    Checksumming of packets disabled
  Tunnel TTL 255, Fast tunneling enabled
  Tunnel transport MTU 1476 bytes
  Tunnel transmit bandwidth 8000 (kbps)
  Tunnel receive bandwidth 8000 (kbps)
  Last input 00:00:01, output 00:00:04, output hang never
  Last clearing of "show interface" counters 00:02:01
  Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/0 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     21 packets input, 2344 bytes, 0 no buffer
     Received 0 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     26 packets output, 3360 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 unknown protocol drops
     0 output buffer failures, 0 output buffers swapped out
```

The highlighted output identifies various aspects of the R1 tunnel 1 interface. Notice that the tunnel protocol is the default GRE/IP.

b. On R3, use the **show interfaces tunnel 1** command to verify the interface settings.

```
R3# show inter tunnel 1 | include is up|Internet address|Enc|Tunnel protocol
Tunnel1 is up, line protocol is up
  Internet address is 172.16.1.2/30
  Encapsulation TUNNEL, loopback not set
  Tunnel protocol/transport GRE/IP
```

Again, the highlighted output identifies various aspects of the R3 Tunnel 1 interface.

Step 2: On R1 and R3, verify the crypto settings.

a. On R1, use the **show crypto session** command to verify the operation of the VPN tunnel.

```
R1# show crypto session
Crypto session current status

Interface: Ethernet0/0
Session status: UP-ACTIVE
Peer: 64.100.1.2 port 500
  Session ID: 0
  IKEv1 SA: local 64.100.0.2/500 remote 64.100.1.2/500 Active
  IPSEC FLOW: permit 47 host 64.100.0.2 host 64.100.1.2
        Active SAs: 4, origin: crypto map
```

Notice how the output identifies G0/0/0 as the crypto interface. This is because R1 was configured using the classic crypto map method of configuring GRE over IPsec.

b. On R3, use the **show crypto session** command to verify the operation of the VPN tunnel.

```
R3# show crypto session
Crypto session current status

Interface: Tunnel1
Session status: UP-ACTIVE
Peer: 64.100.0.2 port 500
  Session ID: 0
  IKEv1 SA: local 64.100.1.2/500 remote 64.100.0.2/500 Active
  IPSEC FLOW: permit 47 host 64.100.1.2 host 64.100.0.2
        Active SAs: 4, origin: crypto map
```

Notice how the output identifies Tunnel 1 as the crypto interface on R3. This is because R3 is configured with GRE over IPsec using a tunnel IPsec profile.

Step 3: On R1 and R3, verify OSPF routing.

a. On R1 and R3, verify which interfaces are configured for OSPF using the **show ip ospf interface brief** command.

```
R1# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Tu1          123   0               172.16.1.1/30      250   P2P   1/1
Ethernet0/1  123   0               10.10.0.1/30       1     BDR   1/1
```

```
R3# show ip ospf interface brief
Interface    PID   Area            IP Address/Mask    Cost  State Nbrs F/C
Tu1          123   0               172.16.1.2/30      250   P2P   1/1
Ethernet0/1  123   0               10.10.4.1/30       1     BDR   1/1
```

b. On R1 and R3, verify the OSPF neighbors using the **show ip ospf interface brief** command.

```
R1# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
3.3.3.1           0   FULL/  -        00:00:32    172.16.1.2      Tunnel1
1.1.1.2           1   FULL/DR         00:00:35    10.10.0.2       Ethernet0/1

R3# show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.1.1.1           0   FULL/  -        00:00:36    172.16.1.1      Tunnel1
3.3.3.2           1   FULL/DR         00:00:36    10.10.4.2       Ethernet0/1
```

c. Verify the R1 routing table for OSPF routes.

```
R1# show ip route ospf | begin Gateway
Gateway of last resort is 64.100.0.1 to network 0.0.0.0

      10.0.0.0/8 is variably subnetted, 15 subnets, 3 masks
O        10.10.1.0/24 [110/11] via 10.10.0.2, 00:25:20, Ethernet0/1
O        10.10.2.0/24 [110/2] via 10.10.0.2, 00:25:20, Ethernet0/1
O        10.10.3.0/24 [110/2] via 10.10.0.2, 00:25:20, Ethernet0/1
O        10.10.4.0/30 [110/251] via 172.16.1.2, 00:05:22, Tunnel1
O        10.10.5.0/24 [110/261] via 172.16.1.2, 00:05:22, Tunnel1
O        10.10.16.0/24 [110/252] via 172.16.1.2, 00:05:22, Tunnel1
O        10.10.17.0/24 [110/252] via 172.16.1.2, 00:05:22, Tunnel1
O        10.10.18.0/24 [110/252] via 172.16.1.2, 00:05:22, Tunnel1
O        10.10.19.0/24 [110/252] via 172.16.1.2, 00:05:22, Tunnel1
O        10.10.20.0/24 [110/252] via 172.16.1.2, 00:05:22, Tunnel1
O        10.10.21.0/24 [110/252] via 172.16.1.2, 00:05:22, Tunnel1
O        10.10.22.0/24 [110/252] via 172.16.1.2, 00:05:22, Tunnel1
O        10.10.23.0/24 [110/252] via 172.16.1.2, 00:05:22, Tunnel1
```

Notice how R1 has now learned about the R3 OSPF networks via the tunnel interface.

d. Verify the R3 routing table for OSPF routes.

```
R3# show ip route ospf | begin Gateway
Gateway of last resort is 64.100.1.1 to network 0.0.0.0

      10.0.0.0/8 is variably subnetted, 15 subnets, 3 masks
O        10.10.0.0/30 [110/251] via 172.16.1.1, 00:40:45, Tunnel1
O        10.10.1.0/24 [110/261] via 172.16.1.1, 00:40:45, Tunnel1
O        10.10.2.0/24 [110/252] via 172.16.1.1, 00:40:45, Tunnel1
O        10.10.3.0/24 [110/252] via 172.16.1.1, 00:40:45, Tunnel1
```

```
O        10.10.5.0/24 [110/11] via 10.10.4.2, 01:00:49, Ethernet0/1
O        10.10.16.0/24 [110/2] via 10.10.4.2, 01:00:49, Ethernet0/1
O        10.10.17.0/24 [110/2] via 10.10.4.2, 01:00:49, Ethernet0/1
O        10.10.18.0/24 [110/2] via 10.10.4.2, 01:00:49, Ethernet0/1
O        10.10.19.0/24 [110/2] via 10.10.4.2, 01:00:49, Ethernet0/1
O        10.10.20.0/24 [110/2] via 10.10.4.2, 01:00:49, Ethernet0/1
O        10.10.21.0/24 [110/2] via 10.10.4.2, 01:00:49, Ethernet0/1
O        10.10.22.0/24 [110/2] via 10.10.4.2, 01:00:49, Ethernet0/1
O        10.10.23.0/24 [110/2] via 10.10.4.2, 01:00:49, Ethernet0/1
```

Notice how R3 has learned about the R1 OSPF networks via the tunnel interface.

e. Verify that there is an operational logical point-to-point link between R1 and R3 using the GRE tunnel interface.

```
R1# show ip route 172.16.0.0
Routing entry for 172.16.0.0/16, 2 known subnets
  Attached (2 connections)
  Variably subnetted with 2 masks
C        172.16.1.0/30 is directly connected, Tunnel1
L        172.16.1.1/32 is directly connected, Tunnel1

R3# show ip route 172.16.0.0
Routing entry for 172.16.0.0/16, 2 known subnets
  Attached (2 connections)
  Variably subnetted with 2 masks
C        172.16.1.0/30 is directly connected, Tunnel1
L        172.16.1.2/32 is directly connected, Tunnel1
```

Step 4: Test the GRE over IPsec VPN tunnel.

a. From D1, trace the path taken to the R3 10.10.16.1 interface.

```
D1# trace 10.10.16.1
Type escape sequence to abort.
Tracing the route to 10.10.16.1
VRF info: (vrf in name/id, vrf out name/id)
  1 10.10.0.1 2 msec 2 msec 2 msec
  2 172.16.1.2 3 msec 2 msec 2 msec
  3 10.10.4.2 4 msec *  3 msec
```

Notice now that the path taken is through the VPN tunnel interface.

b. On R1, verify the IPsec SA encrypted and decrypted statistics.

```
R1# show crypto ipsec sa | include encrypt|decrypt
     #pkts encaps: 296, #pkts encrypt: 296, #pkts digest: 296
     #pkts decaps: 295, #pkts decrypt: 295, #pkts verify: 295
```

The output verifies that the GRE over IPsec VPN tunnel is properly encrypting traffic between both sites. The packets encrypted include the trace packets along with OSPF packets.

Router Interface Summary Table

Router Model	Ethernet Interface #1	Ethernet Interface #2	Serial Interface #1	Serial Interface #2
1800	Fast Ethernet 0/0 (F0/0)	Fast Ethernet 0/1 (F0/1)	Serial 0/0/0 (S0/0/0)	Serial 0/0/1 (S0/0/1)
1900	Gigabit Ethernet 0/0 (G0/0)	Gigabit Ethernet 0/1 (G0/1)	Serial 0/0/0 (S0/0/0)	Serial 0/0/1 (S0/0/1)
2801	Fast Ethernet 0/0 (F0/0)	Fast Ethernet 0/1 (F0/1)	Serial 0/1/0 (S0/1/0)	Serial 0/1/1 (S0/1/1)
2811	Fast Ethernet 0/0 (F0/0)	Fast Ethernet 0/1 (F0/1)	Serial 0/0/0 (S0/0/0)	Serial 0/0/1 (S0/0/1)
2900	Gigabit Ethernet 0/0 (G0/0)	Gigabit Ethernet 0/1 (G0/1)	Serial 0/0/0 (S0/0/0)	Serial 0/0/1 (S0/0/1)
4221	Gigabit Ethernet 0/0/0 (G0/0/0)	Gigabit Ethernet 0/0/1 (G0/0/1)	Serial 0/1/0 (S0/1/0)	Serial 0/1/1 (S0/1/1)
4300	Gigabit Ethernet 0/0/0 (G0/0/0)	Gigabit Ethernet 0/0/1 (G0/0/1)	Serial 0/1/0 (S0/1/0)	Serial 0/1/1 (S0/1/1)

Note: To find out how the router is configured, look at the interfaces to identify the type of router and how many interfaces the router has. There is no way to effectively list all the combinations of configurations for each router class. This table includes identifiers for the possible combinations of Ethernet and Serial interfaces in the device. The table does not include any other type of interface, even though a specific router may contain one. An example of this might be an ISDN BRI interface. The string in parenthesis is the legal abbreviation that can be used in Cisco IOS commands to represent the interface.