Configure The Interface facing the ISP to get IP add Dynamically from the ISP DHCP POOL and Configure a static Ip add to the interface the Ubuntu client is connected to, this should be done on both Customer routers

c1(config-if)#int g0/1

c1(config-if)#ip add dhcp

c1(config-if)#no sh

c1(config-if)#int g0/0

c1(config-if)#ip add 10.1.1.1 255.255.255.0

c1(config-if)#no sh

**STEP2**: Confirm the interface has received IP Dynamically as well as the routing table on both Customers device

c1(config-if)#do sh ip int brief

Interface IP-Address OK? Method Status Protocol

GigabitEthernet0/0 10.1.1.1 YES manual up up

GigabitEthernet0/1 8.8.10.4 YES DHCP up up

GigabitEthernet0/2 unassigned YES unset administratively down down

GigabitEthernet0/3 unassigned YES unset administratively down down

c1(config-if)#do sh ip route

Gateway of last resort is 8.8.10.1 to network 0.0.0.0

S\* 0.0.0.0/0 [254/0] via 8.8.10.1

8.0.0.0/8 is variably subnetted, 2 subnets, 2 masks

C 8.8.10.0/24 is directly connected, GigabitEthernet0/1

L 8.8.10.4/32 is directly connected, GigabitEthernet0/1

10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks

C 10.1.1.0/24 is directly connected, GigabitEthernet0/0

L 10.1.1.1/32 is directly connected, GigabitEthernet0/0

**STEP 3**: Configure DHCP POOL on both customer router to dish out Ip add to the Ubuntu client and confirm it has started binding address   
N.B: This should be done on both routers with the respective address

c1#conf t

Enter configuration commands, one per line. End with CNTL/Z.

c1(config)#ip dhcp pool CS1\_pool

c1(dhcp-config)#network 10.1.1.0 255.255.255.0

c1(dhcp-config)#default-router 10.1.1.1

c1(dhcp-config)#

c1(dhcp-config)#do sh ip dhcp bi

c1(dhcp-config)#do sh ip dhcp binding

Bindings from all pools not associated with VRF:

IP address Client-ID/ Lease expiration Type

Hardware address/

User name

10.1.1.2 0102.425f.5804.00 Jun 15 2025 04:33 PM Automatic

STEP 3: Configure GRE Tunnel between C1 and C2 so that Ubuntu 1 and Ubuntu 2 can communicate with each other

1(config)#int tunnel ?

<0-2147483647> Tunnel interface number

c1(config)#int tunnel 0

c1(config-if)#ip add 10.1.3.1 255.255.255.0

c1(config-if)#tunnel source?

source

c1(config-if)#tunnel source g0/1

c1(config-if)#tunnel destination 8.8.11.4

c2(config)#int tunnel 0

c2(config-if)#ip add 10.1.3.2 255.255.255.0

c2(config-if)#tunnel source g0/1

c2(config-if)#tunnel destin

c2(config-if)#tunnel destination 8.8.10.4

c2(config-if)#  
  
STEP 4: Test Pings between the two tunnels

c1(config)#do ping 10.1.3.2

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 10.1.3.2, timeout is 2 seconds:

!!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 12/15/18 ms  
  
STEP5: You need to advertise routes between C1 10.1.1.0 and C2 10.1.2.0 using any of the routing protocol, in my case I use OSPF

c1(config)#router ospf 100

c1(config-router)#network 10.1.0.0 0.0.255.255 area 0

c1(config-router)#

\*Jun 14 17:43:58.465: %OSPF-5-ADJCHG: Process 100, Nbr 10.1.3.2 on Tunnel0 from LOADING to FULL, Loading Done

c1(config-router)#

c1(config-router)#do sh ip ospf neighbor

Neighbor ID Pri State Dead Time Address Interface

10.1.3.2 0 FULL/ - 00:00:36 10.1.3.2 Tunnel0

c1(config-router)#do sh ip route

c2(config)#router ospf 100

c2(config-router)#network 10.1.0.0 0.0.255.255 area 0

Jun 14 17:44:16.837: %OSPF-5-ADJCHG: Process 100, Nbr 10.1.3.1 on Tunnel0 from LOADING to FULL, Loading Done

c2(config-router)#do sh ip ospf neighbor

Neighbor ID Pri State Dead Time Address Interface

10.1.3.1 0 FULL/ - 00:00:30 10.1.3.1 Tunnel0

Finally we can now test pings between the Ubuntu clients

root@Ubuntu-1:~# ping 10.1.2.2

PING 10.1.2.2 (10.1.2.2) 56(84) bytes of data.

64 bytes from 10.1.2.2: icmp\_seq=1 ttl=62 time=21.3 ms

64 bytes from 10.1.2.2: icmp\_seq=2 ttl=62 time=11.4 ms

64 bytes from 10.1.2.2: icmp\_seq=3 ttl=62 time=10.9 ms

64 bytes from 10.1.2.2: icmp\_seq=4 ttl=62 time=11.4 ms

64 bytes from 10.1.2.2: icmp\_seq=5 ttl=62 time=13.1 ms

64 bytes from 10.1.2.2: icmp\_seq=6 ttl=62 time=10.6 ms

Conclution: GRE Tunnel is a technology that basically allows our traffic to flow site to site without beighn visible to the ISP, in my LAB I could confirm the tunnel ip is not visible on all of my ISP routing table this is as a result of GRE9Generic Routing Encapsulation) Tunnel which is a type of virtual tunnel that allows two endpoints to communicate over an ip network as if they were directly connected even though they are seperated by a network