CSE 4840: [Internetworking Protocols Lab]

Lab # 02

Objectives

- Configure static routing in a network topology following given specifications
- Configure Routing Information Protocol (RIP) in a network topology following given specifications
- Configure OSPF in a network topology following given specifications

Static Routing:

By now, you all know that routers take the help of routing table to forward packets to the intended destinations. When a packet reaches the router, it looks up the routing table, finds the corresponding output interfaces for the destination network address and sends the packet through that interface. The question is how this routing table is formed in the first place. The answer is either manually configuring the routing entries or using routing protocols to configure the routes dynamically. The first approach is called static routing.

In static routing, the network administrator manually adds the routing entries to the routing table. The routing entries will not be changed automatically. All changes have to be done manually. If the network condition changes (for example, *some link goes down*), then the necessary changes in the routing table must be done manually whereas these info are updated automatically in dynamic routing.

In practical large networks, static routing is mostly used as a backup to dynamic routing. Unlike dynamic routing, static routing requires very less computational resource and bandwidth as no extra packet is required for routing table update process. But as the network administrator needs to know the whole network topology and network addresses to effectively configure the routing table, static routing is not used as the only routing mechanism in large scale networks.

There are some concepts related to static routing that you need to be familiar with before you get your hands dirty. You know that packets travel from one hop to the next to reach its final destination. In the routing table of a router, the next hop address is associated with a certain destination address. Its not realistic to assume that there would be next-hop entry for every possible destination network. That’s why a routing entry known as the **default route** is present in the routing table. It defines a default exit interface for the packets that don't have any corresponding route in the routing table. When working with CISCO devices and specifically for this lab, you'll encounter two types of static default routes. One is **directly connected** static default route and another is **next-hop** static default route.

The general format of the command to specify static routes is:

```
ip route destination_network_prefix destination_prefix_mask
(nexthop_address | interface) [distance_metric]
ipv6 route ipv6_destination_network_prefix(with CIDR) (ipv6_next-
hop_address
| interface) [distance_metric]
```

In case of directly connected static default routes, you'll specify the ***interface***. In case of next-hop static default routes, you'll specify the ***next_hop address***. One special use case of the above command is to configure a **primary static default route** where both the *destination_network_prefix* and *destination_prefix_mask* are *0.0.0.0*. The IPv4 and IPv6 command format for specifying primary static default route is given below:

```
ip route 0.0.0.0 0.0.0.0 (next-hop_address | interface)
[distance_metric]
ipv6 route ::/0 (ipv6_next-hop_address | interface) [distance_metric]
```

The above commands basically mean that "*packets from any IP address with any subnet mask get sent to the specified next-hop address or interface*".

Another concept when configuring static routing is the **floating static route**. A floating route is nothing but the route that’s used to forward a packet to a certain destination when main route is unavailable. The way floating routes are defined is by providing a higher ***distance_metric*** to a certain route. The default ***distance_metric*** when its not manually specified is **1**. The floating static routes are given higher numbers than 1. Routers always take the route with *lower* ***distance_metric*** when multiple routes to the same destination are available. That's why this floating static route will only be used when main route is down or unavailable.

Configure static routing:

I. Configure R1 Interfaces

```
R1(config)#int g0/0
R1(config-if)#ip add 192.168.1.1 255.255.255.0
R1(config-if)#desc connection-to-PC0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#int s0/1/0
R1(config-if)#ip add 192.168.0.2 255.255.255.252
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#int s0/1/1
R1(config-if)#ip add 192.168.0.6 255.255.255.252
R1(config-if)#no shutdown
R1(config-if)#exit
```

II. Configure R2 Interfaces

```
R2(config)#int s0/1/1
R2(config-if)#ip add 192.168.0.5 255.255.255.252
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#int g0/0
R2(config-if)#ip add 192.168.2.1 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#exit
```

III. Configure R3 Interfaces

```
R3(config)#int s0/1/0
R3(config-if)#ip add 192.168.0.1 255.255.255.252
R3(config-if)#no shutdown
R3(config-if)#exit
R3(config)#int g0/0
R3(config-if)#ip add 192.168.2.2 255.255.255.0
R3(config-if)#no shutdown
R3(config-if)#exit
```

IV. Configure PC0

```
IP: 192.168.1.10
Mask: 255.255.255.0
Gateway: 192.168.1.1
```

V. Configure PC1

```
IP: 192.168.2.10
Mask: 255.255.255.0
Gateway: 192.168.2.1
```

VI. Configure static routing to Remote LAN in R1

```
R1(config)#ip route 192.168.2.0 255.255.255.0 s0/1/1
```

It’s a **directly connected** static default route.

```
R1(config)#ip route 192.168.2.0 255.255.255.0 192.168.0.1 5
```

It’s a **next-hop floating** static default route.

VII. Configure static routing to Local LAN in R2

```
R2(config)#ip route 192.168.1.0 255.255.255.0 s0/1/1
```

It’s a **directly connected** static default route.

```
R2(config)#ip route 192.168.1.0 255.255.255.0 192.168.0.6 5
```

It’s a **next-hop floating** static default route

VIII. Configure static routing to Local LAN in R3

```
R2(config)#ip route 192.168.1.0 255.255.255.0 s0/1/0
```

It’s a **directly connected** static default route.

```
R2(config)#ip route 192.168.1.0 255.255.255.0 192.168.0.2 5
```

It’s a **next-hop floating** static default route

IX. Verify

```
Ping PC1 from PC0
```

Dynamic Routing:

Static routing has some serious disadvantages when it comes to configure large-scale networks. Network administrator needs to know about the whole topology and the network changes have to be reflected manually in the configuration. In this lab, you'll learn about dynamic routing which overcomes all these issues but at the cost of bandwidth overhead. Still, its better to use extra bandwidth than to configure everything manually.

Basically, dynamic routing lets routers to select routes based on the real-time network condition. Always there will be packets travelling around the networks to keep the routers up-to-date about the present network condition. Then the routers will select an optimal route to a given destination based on some set of metrics. The dynamic routing protocols perform this basic function. As you know by now, there are different types of dynamic routing protocols following different algorithms. Two most common ones are **Routing Information Protocol** (RIP) following distance-vector algorithm and **Open Shortest Path First** (OSPF) protocol following link state routing algorithm.

Routing Information Protocol (RIP)

Before starting, know that RIP is somewhat an obsolete protocol due to its limitations and the advent of more modern and sophisticated protocols like OSPF, EIGRP etc. Still, as this was one of the pioneering dynamic routing protocols and dominated the networking world for quite some time, you need to understand this. This will help you in turn to understand the improvements made in newer protocols.

There are two versions of RIP namely RIPv1 and RIPv2. **RIPv2** is the dominant one and is used almost in all cases where RIP is used. So, we'll focus on RIPv2 only for our lab. Major problem of RIPv1 was that it used to broadcast routing table updates every 30 seconds. You can imagine the flood of packets every 30 seconds that would take place where millions of networks are now interconnected, even if they are configured to send updates at random times. RIPv2 was designed to overcome this issue along with other improvements. You can learn more on these two versions here.

RIP uses hop count as the metric for routes. Paths with lower metric are selected always for a given destination. The max hop count is 15 for both versions of RIP to prevent loops. Any route further than 15 hops away are considered unreachable. To configure RIP, three steps are followed:

I. Enable RIP by using the *router rip* global configuration command.
II. Tell router to use the RIPv2 by the *version 2* command.
III. Tell RIP which networks to advertise by using one or more *network* commands.

In the 3rd step, you don't need to specify any subnet mask when using the network command. The network command takes a classful address as the parameter. As any address beginning with `10` belongs to class A, RIP will be enabled on *all the interfaces* that has IP address that begins with `10`.

Configure RIP:

I. **Configure R1 Interfaces**

```
R1(config)# int g0/0
R1(config-if)# ip address 10.0.1.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# exit
R1(config)# int g0/1
R1(config-if)# ip address 172.16.0.1 255.255.0.0
R1(config-if)# no shutdown
R1(config-if)# exit
R1# copy running-config startup-config
```

II. **Configure R2 Interfaces**

```
R2(config)# int g0/1
R2(config-if)# ip address 192.168.0.1 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# exit
R2(config)# int g0/0
R2(config-if)# ip address 172.16.0.2 255.255.0.0
R2(config-if)# no shutdown
R2(config-if)# exit
R2# copy running-config startup-config
```

III. Configure PC0

```
IP: 10.0.1.10
Mask: 255.255.255.0
Gateway: 10.0.1.1
```

IV. Configure PC1

```
IP: 192.168.0.10
Mask: 255.255.255.0
Gateway: 192.168.0.1
```

V. Configure RIP in R1

```
R1(config)#router rip
R1(config-router)#version 2
R1(config-router)#network 10.0.0.0
R1(config-router)#network 172.16.0.0
```

VI. Configure RIP in R2

```
R2(config)#router rip
R2(config-router)#version 2
R2(config-router)#network 192.168.0.0
R2(config-router)#network 172.16.0.0
```

VII. Verify

```
R1# show ip route rip
Ping PC1 from PC0
```

Open Shortest Path First (OSPF) Routing Protocol:

As you know already, OSPF uses Link State Routing (LSR) algorithm. In OSPF, each router sends its link information to the directly connected routers, known as neighbors, which in turn sends the info to other neighbors. After that each router runs the Shortest Path Algorithm on the received information to determine the optimal route to different networks. The running of the algorithm is one of the reason why OSPF is a CPU-intensive protocol as whenever there's a change the algorithm is run. Still OSPF is better than RIP due to its fast convergence and better load-balancing. Remember that, RIP sends the whole routing table to its neighbors whereas OSPF only sends the link information with various optimizations in place. You can learn more about OSPF and RIP here.

There's a plethora of terminologies and concepts related to OSPF that one needs to know for full understanding of the protocol. Some most common ones are listed here. But for our purpose, we'll just cover the ones necessary for our lab.

- **Area**: OSPF networks are divided into **areas** which are a logical collection of routers, links having the *same area identification*. A router within an area must maintain a topological database *for the area to which it belongs*. The router does not have detailed information about network topology outside of its area, which thereby reduces the size of its database. There's a special area called the **backbone area** (*area 0*) to which all other areas must be connected. Different areas can communicate with each other through area 0.
- **Area Border Router (ABR)**: A router with interfaces in two areas is called an ABR. This router
is the boundary between two areas.
- **Autonomous System (AS):** OSPF operates within a single Autonomous System which is a collection of areas. AS is basically a group of networks running under a single administrative control. It controls how far the routing information should be propagated and facilitates filtering of information for sharing with other AS.
- **Designated Router (DR):** A router is elected as the Designated Router (DR) and another as Backup Designated Router (BDR) on a multi-access network (like LAN) in OSPF. DR and BDR serve as the *central point* for exchanging OSPF routing information. Each non-DR or non-BDR router will exchange routing information *only* with the DR and BDR, instead of exchanging updates with every router on the network segment. DR will then distribute topology information to every other router inside the same area, which greatly reduces OSPF traffic. For more, you can read on here.
- **Router ID**: Each router running the OSPF protocol is assigned a router ID to **uniquely identify that router within an AS**. This is a 32 bit number and can be set *manually* by using the `router-id` command. If router ID is not set manually then the highest IP address of the router's *loopback address* will be the router ID. If there's no loopback address then the highest *active IP address* on any of the router's interface will be the router ID. Remember to restart the router to reflect the new router-id assignment. The `reload` command will restart the router. Also, the `clear ip ospf process` command will work for new router-id assignment.
- **Cost**: The cost in OSPF is calculated as **Reference Bandwidth / Interface Bandwidth**. The

default value of reference bandwidth is *100 Mbps* but it can be set manually. The command to manually set the reference bandwidth is `auto-cost reference-bandwidth <value>`. Note that, `value` here is in units of *Mbps*. So, `value`=100 would mean 100 Mbps. You can change the interface bandwidth with the `bandwidth <value>` command. But, here the `value` is in units of *kbps*.

- **Wildcard Mask:** The command to configure which networks to advertise is `network <ip_address> <wildcard_mask> area <area_id>`. Unlike RIP, the `network` command in OSPF supports classless routing and that support is achieved by the `wildcard_mask`. This mask is like an inverted subnet_mask but with different interpretation. The **0** bits in the mask indicate the corresponding bit positions that *must match* the same bit positions in the IP address. The **1** bits indicate the corresponding bit positions *don't need to match* the same bit positions in the IP address. Its best understood by an example. Suppose, there's a `10.0.1.0` directly connected subnet to our router that we want to advertise in the OSPF routing process. The command to include that subnet in the advertisement would be:
```
network 10.0.1.0 0.0.0.255 area 0
```
According to the wildcard_mask used in the above command, first 24 bits of the addresses must match and rest 8 bits don't need to match. So, any interface having IP address in the format **`10.0.1.X`** will match in this case. A more detailed explanation of wildcard_mask with more examples can be found here.
- **Process ID**: When enabling OSPF in a router, you need to mention a process ID. All OSPF functions will then be performed under that process. OSPF configuration mode is entered using the command `router ospf <process_id>`. You can have more than one OSPF processes in a single router. The process IDs will be different for different processes. Each OSPF process has its separate database, topology table etc. More on process ID can be read from here.

Now lets mention some key concepts in OSPF. OSPF enabled routers send *hello* packets to each other in certain intervals known as the *hello interval*. This is required to establish neighbor relationship and to let other router know the availability of the router. For example, if the *hello interval* is 5 seconds, then each router will send *hello* packets to the neighboring router in every 5 seconds. There's another interval known as the *dead interval* which is usually `4 times` of the hello interval. If a router doesn't receive *hello* packets from its neighbors then after this *dead interval* amount of time that neighbor will be declared *non-operational* and the routing table will be updated accordingly.

In OSPF, some interfaces are configured as passive-interfaces. Passive interfaces don't send hello packets. This is usually done for the local-LAN facing interfaces. Note that, the network connected with the passive interface will still be advertised as OSPF has been enabled in that interface. Check out this to learn more about passive interfaces.

Configure OSPF:

I. **Configure R1 Interfaces**

```
R1(config)# int g0/0
R1(config-if)# ip address 10.0.1.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# exit
R1(config)# int g0/1
R1(config-if)# ip address 172.16.0.1 255.255.0.0
R1(config-if)# no shutdown
R1(config-if)# exit
R1# copy running-config startup-config
```

II. **Configure R2 Interfaces**

```
R2(config)# int g0/1
R2(config-if)# ip address 192.168.0.1 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# exit
R2(config)# int g0/0
R2(config-if)# ip address 172.16.0.2 255.255.0.0
R2(config-if)# no shutdown
R2(config-if)# exit
R2# copy running-config startup-config
```

III. Configure PC0

```
IP: 10.0.1.10
Mask: 255.255.255.0
Gateway: 10.0.1.1
```

IV. Configure PC1

```
IP: 192.168.0.10
Mask: 255.255.255.0
Gateway: 192.168.0.1
```

V. Configure OSPF in R1

```
R1(config)# router ospf 1
R1(config-router)# network 10.0.1.0 0.0.0.255 area 0
R1(config-router)# network 172.16.0.0 0.0.255.255 area 0
```

VI. Configure OSPF in R2

```
R2(config)# router ospf 1
R2(config-router)# network 192.168.0.0 0.0.0.255 area 0
R2(config-router)# network 172.16.0.0 0.0.255.255 area 0
```

VII. Verify

```
R1# show ip ospf neighbor
R2# show ip ospf neighbor
Ping PC1 from PC0
```