Exercise: OSPF

Summary of OSPF commands

router ospf 1

Enter OSPF configuration mode. The **1** is simply the OSPF process ID - a router can run more than one OSPF process if required.

network *n.n.n.n w.w.w.w* area 0

The network statement has two functions and its use varies depending on the function required.

1. The first use is to enable OSPF on all interfaces which match the given IP address + "wildmask". A wildmask is used in access control lists to select a range of addresses. A "0" bit in the wildmask means the corresponding address bit must match, and a "1" means the corresponding address bit is "don't care". Examples:

   ```
   network 196.200.220.17 0.0.0.0 area 0
   network 196.200.220.98 0.0.0.0 area 0
      -- talk OSPF only on those interfaces with these addresses

   network 196.200.220.192 0.0.0.3 area 0
      -- talk OSPF on the serial interface with this address (this is an
         alternative to the previous examples and has the same functionality)

   network 196.200.220.0 0.0.0.255 area 0
      -- talk OSPF on every interface whose IP address is 196.200.220.X

   network 0.0.0.0 255.255.255.255 area 0
      -- talk OSPF on every interface we have
   ```

   By talking OSPF on an interface the router will automatically inject the network block used on that interface into OSPF.

   Note that it is important that we only talk OSPF to our own networks, never to customers or other ISPs (they could break our network by injecting bad information), so usually it is best just to list the interfaces we want to talk on.

2. The second use of the network statement is to inject prefixes into OSPF. For this function the network statement must match both the network and the network's mask on that interface. The mask is called an "inverse mask", and is the one's complement of the network mask for the network block. This technique is used to inject prefixes from non-OSPF speaking interfaces into OSPF.

   ```
   network 196.200.220.192 0.0.0.3 area 0
      -- announce the network 196.200.220.192/30 to OSPF
   ```

```
network 196.200.220.0 0.0.0.255 area 0
   -- announce the network 196.200.220.0/24 to OSPF
```

redistribute connected subnets

Advertise all networks to which we are connected, including those which are not being used to talk to other OSPF routers. The use of this command is not recommended within ISP backbones, but is included here for completeness.

("redistributed connected" by itself only distributes classful routes, i.e. whole class A/B/C networks, so it's important to add "subnets")

redistribute static subnets

Advertise all static routes we have (except defaultroute). The use of this command isn't ordinarily required or recommended. However, there will be circumstances where redistribution of statics into OSPF are required. An example will be shown in the lab.

default-information originate [always] metric *n*

Advertise a default route into OSPF, with a cost of "n". Typically this would go on your border router(s). Without 'always', the announcement will only be made if the router already has a valid defaultroute from somewhere else (e.g. a static route to a link which is up)

area 0 authentication message-digest

Use MD5 authentication on all OSPF packets

Per-interface configuration

```
ip ospf cost <n>                              Set outbound interface cost
ip ospf message-digest-key 1 md5 <string>     Set MD5 authentication key
```

Part 1: Loopback interfaces

A loopback interface is a single (/32) IP address which belongs to a device, independent of its physical interface addresses. It's very convenient when managing routers, because you can use the loopback address as a fixed address to telnet to, or monitor using SNMP, which will continue to work even if one or more of the interfaces has failed.

1. Check that there is no OSPF process running from any previous exercises. If it is still there, remove it.

   ```
   router-a#conf t
   router-a(config)#no router ospf 1
   ```

```
router-a(config-if)#  [Hit ctrl-Z]
router-a#write mem
```

2. Check that you have no static routes. If you do have some, remove them.

```
router-a#show ip route
The only routes you should see are (C)onnected routes for your own
interfaces. If you have any (S)tatic routes, delete them like this:
router-a#conf t
router-a(config)#no ip route 196.200.220.128 255.255.255.240 196.200.220.12
router-a(config-if)#  [Hit ctrl-Z]
router-a#write mem
```

3. Now reboot your router - this will clear any remaining OSPF data structures and the router will be ready for this exercise.

```
router-a#reload
```

4. When the router has rebooted, log in and check your router can still ping the PC on your desk, and other routers on the backbone.

   (Your PC won't be able to ping any other PCs though, because your router doesn't have the routes any more)

5. As described in the presentation the loopback interface address is that used to form the router-id used by OSPF. If no loopback is configured, the router uses the highest IP address configured at the time the OSPF process was started.

   You will need to subdivide your assigned network space. You will need one subnet for your desk network, and a /32 for the loopback. (In real life you would have a number of routers, and say a /29 block would be enough for 8 loopbacks). Configure your loopback address into your router.

   The example below is for the router belonging to Team A. They have decided to subdivide their address block 196.200.220.80 such that 196.200.220.95/32 is the address of the loopback interface of their router.

```
router-a#conf t
router-a(config)#int loopback0
router-a(config-if)#ip address 196.200.220.30 255.255.255.255
router-a(config-if)#  [Hit ctrl-Z]
```

6. Now save your configuration.

```
router-a#write mem
```

Part 2: Learning routes via OSPF

The initial network topology is the same as for the static routing exercise

1. Enable OSPF on the backbone interface (only).

   We will use a new feature in IOS 12.0 which explicitly disables OSPF on all

interfaces except those you nominate.

```
router-a#conf t
router-a(config)#router ospf 1
router-a(config-router)#log-adjacency-changes
router-a(config-router)#passive-interface default
router-a(config-router)#no passive-interface e0/0
router-a(config-router)#network 196.200.220.0 0.0.0.15 area 0
router-a(config-router)#network 196.200.220.30 0.0.0.0 area 0
router-a(config-router)#network 196.200.220.16 0.0.0.7 area 0
router-a(config-router)#  [Hit ctrl-Z]
```

If you are still using a router with an IOS image prior to 12.0, you cannot use the above **passive-interface default** but will have to explicitly passive every interface which is not talking OSPF. The above example would then become:

```
router-a(config)#router ospf 1
router-a(config-router)#ospf log-adjacency-changes
router-a(config-router)#passive-interface Loopback0
router-a(config-router)#passive-interface e0/1
router-a(config-router)#network 196.200.220.1 0.0.0.15 area 0
router-a(config-router)#network 196.200.220.30 0.0.0.0 area 0
router-a(config-router)#network 196.200.220.16 0.0.0.7 area 0
...
```

This configuration can get somewhat unwieldy on a router with a large number of interfaces, hence the new schema introduced in 12.0 is the preferred option to use. (The "workaround" in releases prior to 12.0 was to use **redistribute connected subnets** and then only list interfaces which spoke OSPF with explicit **network** statements.)

2. We will use MD5 authentication, to ensure that we only listen to OSPF packets from machines which know the secret key; we will also set the link cost to 100.

```
router-a(config)#router ospf 1
router-a(config-router)#area 0 authentication message-digest
router-a(config-router)#int e0/0  (or int e0/1)
router-a(config-if)#ip ospf message-digest-key 1 md5 t2@afnog
router-a(config-if)#ip ospf cost 100
router-a(config-if)#  [Hit ctrl-Z]
```

In real life you should use an MD5 key which is different from your login, enable and SNMP strings

3. Look at OSPF status

```
router-a#show ip ospf int
router-a#show ip ospf neighbor
```

To interpret the neighbor information:

```
2WAY    =  we are neighbors (we have established 2-way exchange of hellos),
           but neither of us is a designated router
FULL    =  we are neighbors and we exchange routes (one of us is DR or BDR)
```

```
DR      =  we are the Designated Router for this network
BDR     =  we are the Backup Designated Router for this network
DROTHER =  we are neither DR nor BDR
```

If you see other states, they are intermediate steps on the way to establishing the final relationship, and should change after a few seconds.

4. Question: who is the Designated Router (DR) and Backup Designated Router (BDR) on the backbone network?

   ```
   DR =                          BDR =
   ```

5. Once you have established a neighbor relationship with another router, you should automatically have learned some new routes:

   ```
   router-a#show ip route
   ```

   Routes learned through OSPF are tagged with **O**. Check that the *next hop IP address* for each route is correct

   Also, the *far* router should also have picked up *your* route. You can go over to the other desk and ask to see "show ip route"

6. Check that your PC can ping the PCs on other desks

   ```
   $ ping 196.200.220.129
   ...
   ```

7. Once all desks are running, the instructors will add a class router into the OSPF cloud, and get it to announce a default route.

   ```
   Don't type this - it goes on the class border router
   t2-border-1(config)#router ospf 1
   t2-border-1(config-router)#default-information originate metric 100
   ```

8. Check that you can see the new router as a neighbor, and that you have picked up a default route (0.0.0.0)

   This should be sufficient to establish connectivity to the outside Internet! Use ping, traceroute etc. to test this

9. Save your config

   ```
   router-a#write mem
   ```

10. If you want DNS, you'll have to create `/etc/resolv.conf` on your PC

    ```
    domain ws.afnog.org
    nameserver 196.200.220.1
    ```

    You should then be able to ssh/telnet to the outside world.

Part 3: Configuration storage via TFTP

As discussed in the preceding TFTP exercise, save your configuration to your TFTP

server

Part 4: Dynamic changes in topology

Above you showed how OSPF can learn routes from the rest of your network, without having to manually insert static routes. Now you can show how OSPF can adapt to topology changes and choose better (lower cost) routes when they are available

1. Work in pairs with an adjacent desk
2. Connect a DTE/DCE cable pair between Serial 0 (or Serial 0/0) on one router and the other.
3. One of you will need to allocate a /30 subnet out of your address space for the link, and assign an IP address to each end. Then both of you need to configure your end of the link.

   ```
   router-a#conf t
   router-a(config)#int s0/0  (or int s0/1)
   router-a(config-if)#description Serial link to desk B
   router-a(config-if)#ip address 196.200.220.25 255.255.255.252
   router-a(config-if)#no shutdown
   ```

   Note: If the remote router does not support HDLC, then you will need to use PPP as the encapsulation.

   ```
   router-a#conf t
   router-a(config)#int s0/0
   router-a(config-if)#encap ppp
   ```

   Once this is done on both routers, "show int s0" should show that the Interface is up (layer 1), but Line protocol is down (layer 2).
4. On the router which has the DCE cable, set it to generate clock. *If you can't tell which end is the DCE, just try it on both routers; the DTE end will refuse the command*

   ```
   router-a(config-if)#clock rate 64000
   ```

   This is only because this is a back-to-back cable; normally you would use synchronous modems which generate clock
5. Line protocol should be up. Check you can ping the remote IP address. Because we have set the link to be only 64K, you should see a longer round-trip time reported by ping.

   ```
   router-a#ping 196.200.220.26
   ```
6. Enable OSPF on the serial line. We will use a larger cost of 500 to reflect the fact that this is a slower-speed link

   ```
   router-a#conf t
   router-a(config)#router ospf 1
   router-a(config-router)#network 196.200.220.24 0.0.0.3 area 0
   ```

```
router-a(config-router)#no passive-interface s0/0   (or s0/1)
router-a(config-router)#int s0/0  (or s0/1)
router-a(config-if)#ip ospf message-digest-key 1 md5 t2@afnog
router-a(config-if)#ip ospf cost 500
router-a(config-if)#  [Hit ctrl-Z]
```

7. Both desks: look at your forwarding table

   ```
   router-a#show ip route
   ```

   Look carefully at the route to your neighbor's desk network, and your neighbor's router loopback interface, and make a note of it.

8. On one desk, unplug the ethernet connection into the backbone, wait a few seconds, and look at the forwarding table again.

   Does the desk which had its ethernet unplugged still have connectivity to the Internet? When you traceroute, what route do the packets take?

9. Plug the ethernet back in. Set the cost of the serial link to 50.

   Look at the routes again. What has happened to the route to your neighbor's desk network, and to their loopback interface?

10. Try setting the cost of the serial link to 100

    Check the forwarding table ("show ip route"). Now what do you notice about the route to your neighboring desk's network?

Before finishing this exercise, remove the serial links and reboot the router to get back to the saved configuration.

Part 5: Extras

1. There are cross ethernet cables available if you wish to try some more complex topologies linking to other desks.
2. When finished, revert to the saved configuration from the end of Part 3. **HINT**: Use the file saved to the tftp server.

Last updated 2007-04-21