

---

# INGI 2142 - Group 8

## Assignment 2 : TE with MPLS (OSPF)

---

Laurent Lamouline - 3597-05-00  
Thibault Leruitte - 2972-05-00  
Vincent Nuttin - 5772-05-00

March 17, 2010

### Contents

<b>1 Quick reminder about MPLS TE</b>	<b>1</b>
<b>2 Guidelines and requirements</b>	<b>1</b>
<b>3 LSPs set-up and CSPF</b>	<b>2</b>
3.1 Bandwidth reservation . . . . .	2
3.2 Explicit and loose hops . . . . .	2
3.3 Attribute tags . . . . .	2
<b>4 Forward traffic on our tunnel</b>	<b>3</b>
<b>A Guidelines and requirements</b>	<b>4</b>
<b>B LSPs set-up and CSPF : listings</b>	<b>4</b>
B.1 Explicit and loose hops . . . . .	5
<b>C Forward traffic on our tunnel : listings</b>	<b>6</b>

The aim of this second work was to configure an entire network using our knowledge in traffic engineering by using MPLS. Actually, we had eight routers and some dedicated high/low priority and high/low bandwidth tunnels to configure. All this stuff must work together to provide a reliable network to our customers.

## 1 Quick reminder about MPLS TE

It's not suitable to perform TE (traffic engineering) in a network where LDP (labelled distribution protocol) mechanism is based on OSPF. Indeed, if the LDP distribution mechanism is based on an underlying IGP such as OSPF, it's not necessary to perform TE because the labels are already distributed in an "optimal" way based on the criteria of the underlying protocol. The main problem of this solution is that it's not possible to dispatch the traffic to avoid congestion because OSPF doesn't care about this feature.

## 2 Guidelines and requirements

Here is a scenario in which we should not activate MPLS on all interfaces. In the case of a VPN, it's not necessary to activate MPLS on all the routers interfaces because we are just interested in one way to go from one router to another. Nevertheless, it's allowed to activate MPLS on each interfaces to create multiple VPNs transparently inside the same network.

To check our configuration, we use these commands to list the configuration summary on the corresponding router:

`show running-config` Thanks to this command, we saw the summary of the configuration previously set up on the corresponding router. For example, we can see the resource reservation that we make on each interface. As previously, we can also see the corresponding IP addresses matching these interfaces and the loopback address. Furthermore, we also see the OSPF process running with the id 8 that we choose.

`show ip ospf mpls traffic-eng link` We used this command to see what will be advertised by OSPF at a given router. Indeed, when we configured the router, we used the command "mpls traffic-eng area 0" to "extend" OSPF such that it floods traffic engineering for the indicated OSPF area (in our case there is only the area 0). So, when using this command, we get some informations about the maximum bandwidth (here we didn't configure anything so that we get the default value which is 12500000), the maximum reservable bandwidth, the interfaces IDs, ... (listing 1)

`show ip ospf database opaque-area` This command allows us to see all the informations known about the other routers by the current router. In our case, we can see that router 8 knows all the other routers thanks to OSPF that runs at the background. It shows the database that is used by the MPLS TE process to calculate the best route (for TE) for dynamic tunnels.

## 3 LSPs set-up and CSPF

### 3.1 Bandwidth reservation

CSPF (constraints shortest path first) begins by pruning away all the paths that don't respect the constraints of the tunnel (bandwidth, explicit path, excluded hops, ...) then it simply selects the shortest path that remains in the list. We can then easily check on each routers between R5 and R7 if the selected path goes through them by using the following command: `show mpls forwarding-table`. The label processing, the stack usage, the outgoing interface can be known by using `show mpls forwarding-table detail`. An example of result is listed in listing 2.

We can see that if R5 receives a packet with label 17, it forwards it with the label 18 on the outgoing interface f1/1 which corresponds to the next hop 10.0.11.2.

After having set up the second tunnel, we can see that the first one that we configured is redirected via other hops because CSPF. Actually, the first tunnel wants 85Mbps of bandwidth and the second one wants 25Mbps. The sum equals 105Mbps which is a problem because all the links of our topology has a maximum of 100Mbps. It seems to be clear that if we activate both tunnels, there will be a problem with shared links on the path because the maximum bandwidth is only 100Mbps.

### 3.2 Explicit and loose hops

To achieve a tunnel from R6 to R8 through an explicit path, we have to do it in two steps. First, we must create a tunnel (Tunnel 3) on R6 (listing 3).

Then, for the second step, we need to explicitly build the path with the lines in listing 4.

Thanks to this, we are sure that the packets sent via the Tunnel3 interface of R6 will follow the path R6-R1-R2-R4-R8 via the IP addresses written before. A good way to check the path is to execute the command in listing 5.

Concerning the fourth tunnel that we had to create, the first step is the same. We need a Tunnel4 interface on R5 with no bandwidth reservation but with an explicit path "ane" (listing 6).

The second step is a little bit different. We need to enter some excluded addresses. Actually, we want to avoid R2 at all cost. So, we will enter all IP addresses of R2 as excluded addresses (listing 7).

Once again, we can perform this command in order to verify our configuration (listing 8).

### 3.3 Attribute tags

To set the attribute flags for each interface, we use the `mpls traffic-eng attribute-flags interface` configuration command. Thanks to this command, the interface is flooded globally so that it can be used as a tunnel path selection criterion. We choose to represent the 3 different colors with the following codes:

- green : 0x0001
- yellow : 0x0010

- red : 0x0100

To configure the affinity for a color to our MPLS traffic engineering tunnel, we use the `tunnel mpls traffic-eng affinity` command. Thanks to this command, we can configure criteria for the path traversal. To configure the tunnel such that it traverses only yellow or green links, it's necessary to make a mapping with the colors that are allowed. To do so, we use a property which is set to 0x0000 and a mask which is 0x0100. By this way, we force that the attribute flag is a 0 at the red place, i.e. the flag can not be red.

In conclusion, we can see that the path from R8 to R5 through the tunnel doesn't cross any red link (listing 9)

## 4 Forward traffic on our tunnel

In order to configure the route-map on R6, we need three steps.

The first step is to configure the access list (listing 10). The aim of the access list is to identify the packets we want to route. Here, the access list stands for the packets which are destined to 10.10.10.10.

The second step is to set the route-map itself (listing 11). The route map matches the packets with respect to the access list we've just configured. These packets are sent via the tunnel 2. (This is the tunnel between R6 and R7.)

The last step is to configure the interface on which the packets should arrive. In our model, it would be an external link of our network. Since we haven't configured such a link, we will use R3 as the sender. First, we have configured the IP policy of the f1/1 interface of R6 (listing 12). Then, we have set up a static route on R3. The packets destined to 10.10.10.10 are sent to R6 via R6's f1/1.

We can see that the packets sent from R3 to 10.10.10.10 go first to R6, and then to R7 via our tunnel 2 (listing 13). The path followed by the tunnel is dependent of the MPLS negotiation. The path followed by a traceroute to 7.7.7.7 is not the same (listing 14).

For the next scenario, we can imagine that R5 receives a lot of torrent traffic that are destined to R7. We will thus decide to sent this flow to the tunnel 1. We will modelize the sender of this flow as R1. The configuration is similar than before, and can be found in listing 15. The traceroute commands we've made are shown in listings 16 and 17.

If we have had more time, we could have think at a more elaborated scenario. For instance, we would be able to redirect only the traffic sent from R1 to any destination.

## A Guidelines and requirements

Listing 1: R6's TE links

```
1 R6#show ip ospf mpls traffic-eng link
2
3         OSPF Router with ID (6.6.6.6) (Process ID 8)
4
5     Area 0 has 3 MPLS TE links. Area instance is 6.
6
7     Links in hash bucket 49.
8     Link is associated with fragment 2. Link instance is 6
9     Link connected to Broadcast network
10    Link ID : 10.0.4.6
11    Interface Address : 10.0.4.6
12    Admin Metric te: 1 igp: 1
13    Maximum bandwidth : 12500000
14    Maximum reservable bandwidth : 12500000
15    Number of Priority : 8
16    Priority 0 : 12500000    Priority 1 : 12500000
17    Priority 2 : 12500000    Priority 3 : 12500000
18    Priority 4 : 12500000    Priority 5 : 12500000
19    Priority 6 : 12500000    Priority 7 : 1875000
20    Affinity Bit : 0x10
21    Link is associated with fragment 1. Link instance is 6
22    Link connected to Broadcast network
23    Link ID : 10.0.2.6
24    Interface Address : 10.0.2.6
25    Admin Metric te: 1 igp: 1
26    Maximum bandwidth : 12500000
27    Maximum reservable bandwidth : 12500000
28    Number of Priority : 8
29    Priority 0 : 12500000    Priority 1 : 12500000
30    Priority 2 : 12500000    Priority 3 : 12500000
31    Priority 4 : 12500000    Priority 5 : 12500000
32    Priority 6 : 12500000    Priority 7 : 12500000
33    Affinity Bit : 0x10
34    Link is associated with fragment 0. Link instance is 6
35    Link connected to Broadcast network
36    Link ID : 10.0.1.6
37    Interface Address : 10.0.1.6
38    Admin Metric te: 1 igp: 1
39    Maximum bandwidth : 12500000
40    Maximum reservable bandwidth : 12500000
41    Number of Priority : 8
42    Priority 0 : 9375000    Priority 1 : 9375000
43    Priority 2 : 9375000    Priority 3 : 9375000
44    Priority 4 : 9375000    Priority 5 : 9375000
45    Priority 6 : 9375000    Priority 7 : 9375000
46    Affinity Bit : 0x1
```

## B LSPs set-up and CSPF : listings

Listing 2: MPLS forwarding-table

```
1 R8#show mpls forwarding-table detail
2 Local  Outgoing  Prefix          Bytes tag  Outgoing  Next Hop
3 tag    tag or VC      or Tunnel Id    switched   interface
4 16     Pop tag        6.6.6.6 2 [10]    0          Fa0/0     10.0.13.7
5     MAC/Encaps=14/14, MRU=1504, Tag Stack{}
6     CA031428001CCA07142800008847
7     No output feature configured
8 17     18            5.5.5.5 1 [12]    0          Fa1/1     10.0.11.2
9     MAC/Encaps=14/18, MRU=1500, Tag Stack{18}
10     CA0514280038CA071428001D8847 00012000
```

```
11 No output feature configured
```

## B.1 Explicit and loose hops

Listing 3: Tunnel 3 configuration

```
1 interface Tunnel3
2 ip unnumbered Loopback0
3 tunnel destination 8.8.8.8
4 tunnel mode mpls traffic-eng
5 tunnel mpls traffic-eng path-option 1 explicit name poney
```

Listing 4: Tunnel 3 explicit path configuration

```
1 ip explicit-path name poney
2 next-address 10.0.2.1
3 next-address 10.0.5.2
4 next-address 10.0.9.4
5 next-address 10.0.12.8
```

Listing 5: Tunnel 3 explicit path

```
1 R6#show ip explicit-paths
2 PATH poney (strict source route, path complete, generation 10)
3   1: next-address 10.0.2.1
4   2: next-address 10.0.5.2
5   3: next-address 10.0.9.4
6   4: next-address 10.0.12.8
```

Listing 6: Tunnel 4 configuration

```
1 tunnel mpls traffic-eng path-option 1 explicit name ane
```

Listing 7: Tunnel 4 explicit path configuration

```
1 ip explicit-path name ane
2 exclude-address 10.0.5.2
3 exclude-address 10.0.10.2
4 exclude-address 10.0.11.2
5 exclude-address 10.0.9.2
6 exclude-address 2.2.2.2
```

Listing 8: Tunnel 4 explicit path

```
1 R5#show ip explicit-paths
2 PATH ane (loose source route, path complete, generation 7)
3   1: exclude-address 10.0.5.2
4   2: exclude-address 10.0.10.2
5   3: exclude-address 10.0.11.2
6   4: exclude-address 10.0.9.2
7   5: exclude-address 2.2.2.2
```

Listing 9: Traceroute from R8 to R5 through tunnel 5

```
1 R8#traceroute 5.5.5.5 numeric
2
3 Type escape sequence to abort.
4 Tracing the route to 5.5.5.5
5
6  1 10.0.12.4 [MPLS: Label 17 Exp 0] 8 msec 28 msec 32 msec
7  2 10.0.6.1 [MPLS: Label 18 Exp 0] 28 msec 8 msec 64 msec
8  3 10.0.2.6 [MPLS: Label 16 Exp 0] 8 msec 64 msec 80 msec
9  4 10.0.1.5 80 msec * 48 msec
```

## C Forward traffic on our tunnel : listings

Listing 10: R6's access list

```
1 R6#show ip access-lists
2 Extended IP access list 101
3     10 permit ip any host 10.10.10.10 (18 matches)
```

Listing 11: R6's route-map

```
1 R6#show route-map
2 route-map reroute10, permit, sequence 10
3   Match clauses:
4     ip address (access-lists): 101
5   Set clauses:
6     interface Tunnel2
7     Tu2 forwarding info:
8     MAC/Encaps=14/18, MTU=1500, Label Stack{17}, via Fa1/1
9     CA061428001DCA021428001D8847 00011000
10    Policy routing matches: 15 packets, 900 bytes
```

Listing 12: R6's ip policy

```
1 R6#show ip policy
2 Interface      Route map
3 Fa1/1         reroute10
```

Listing 13: Traceroute to 10.10.10.10 through tunnel 2.

```
1 R3#traceroute 10.10.10.10 numeric
2
3 Type escape sequence to abort.
4 Tracing the route to 10.10.10.10
5
6   1 10.0.4.6 8 msec 28 msec 4 msec
7   2 10.0.4.3 [MPLS: Label 17 Exp 0] 60 msec 84 msec 136 msec
8   3 10.0.7.4 [MPLS: Label 17 Exp 0] 96 msec 152 msec 56 msec
9   4 10.0.9.2 [MPLS: Label 16 Exp 0] 108 msec 152 msec 84 msec
10  5 10.0.10.7 84 msec 88 msec 28 msec
11  6 10.0.10.7 !H * !H
```

Listing 14: Traceroute to 10.10.10.10 through OSPF.

```
1 R6#traceroute 7.7.7.7 numeric
2
3 Type escape sequence to abort.
4 Tracing the route to 7.7.7.7
5
6   1 10.0.2.1 8 msec 4 msec 20 msec
7   2 10.0.5.2 28 msec 52 msec 60 msec
8   3 10.0.10.7 32 msec * 12 msec
```

Listing 15: R5 route-map configuration

```
1 R5#show ip access-lists
2 Extended IP access list 101
3     10 permit ip any host 10.10.10.10
4
5 R5#show route-map
6 route-map reroute10, permit, sequence 10
7   Match clauses:
8     ip address (access-lists): 101
9   Set clauses:
10    interface Tunnel1
11    Tu1 forwarding info:
12    MAC/Encaps=14/18, MTU=1500, Label Stack{21}, via Fa0/0
```

```
13 CA0414280038CA01142800008847 00015000
14 Policy routing matches: 0 packets, 0 bytes
15
16 R5#show ip policy
17 Interface      Route map
18 Fa0/0          reroute10
```

Listing 16: Traceroute from R5 to 10.10.10.10 through tunnel 1.

```
1 R1#traceroute 10.10.10.10 numeric
2
3 Type escape sequence to abort.
4 Tracing the route to 10.10.10.10
5
6  1 10.0.3.5 8 msec 192 msec 112 msec
7  2 10.0.3.1 [MPLS: Label 21 Exp 0] 132 msec 260 msec 160 msec
8  3 10.0.6.4 [MPLS: Label 18 Exp 0] 92 msec 64 msec 100 msec
9  4 10.0.12.8 [MPLS: Label 16 Exp 0] 80 msec 136 msec 52 msec
10 5 10.0.13.7 128 msec 104 msec 68 msec
11 6 10.0.13.7 !H * !H
```

Listing 17: Traceroute from R5 to 10.10.10.10 through OSPF.

```
1 R5#traceroute 7.7.7.7 numeric
2
3 Type escape sequence to abort.
4 Tracing the route to 7.7.7.7
5
6  1 10.0.3.1 60 msec 28 msec 48 msec
7  2 10.0.5.2 64 msec 40 msec 64 msec
8  3 10.0.10.7 60 msec * 52 msec
```