Trabalho Prático 2 - Grupo 4

João Abreu, *pg55895,* Luís Vilas, *pg57888,* e Ricardo Pereira, *pg56001*

Index Terms

Eve-NG, MPLS, PE, CE, Core, MPLS Traffic-eng, RSVP-TE, OSPF, LDP, ACL, PBR, LDP, Ping, Traceroute, iPerf, QoS.

I. Introdução

Este relatório foi desenvolvido no âmbito da unidade curricular de Redes Fixas e Móveis do Mestrado em Engenharia Informática da Universidade do Minho. O objetivo deste projeto é conceber e implementar uma topologia de rede simulada na plataforma EVE-NG, capaz de reproduzir uma rede baseada no protocolo **MPLS**, interligando vários clientes através de equipamentos CE conectados a PEs (LERs). No interior da topologia, diversos routers P (LSRs) asseguram a comutação de etiquetas ao longo de múltiplos túneis estabelecidos entre os PEs.

Adicionalmente, foram configurados túneis específicos para o encaminhamento de tráfego com base em técnicas de Traffic engineering, assim como critérios definidos por Listas de Controlo de Acessos (ACLs) e Policy-Based Routing (PBR). Esta abordagem permite o encaminhamento diferenciado de tráfego, nomeadamente tráfego HTTP e tráfego UDP, através de caminhos distintos dentro da rede.

II. Desenvolvimento

Para o desenvolvimento desta topologia, foram seguidas uma série de etapas. Inicialmente, e após a instalação e configuração do EVE-NG, o grupo realizou alguns testes com a topologia fornecida pelo docente no guia de setup, a fim de se ambientar com a tecnologia. Após percebidos alguns conceitos base, o grupo desenvolveu uma topologia em forma de duplo peixe, que fosse capaz de preencher os requisitos para a utilização de túneis MPLS.

A. Desenvolvimento da Topologia

Na nossa topologia existem 3 Clientes e uma Rede MPLS. Cada cliente tem os seus Hosts (Linux Servers, Tinycore, Alpines, VPCs ou Imagem Linux Personalizada) interligados a um Router CE que serve de ponte para a Rede MPLS. Dentro da Rede MPLS, nas bordas, os Routers PEs (LERs) servem de ponte com uma interface fora da rede MPLS conectada a um CE, e com as interfaces internas ligadas a um ou mais Ps (LSRs). Todos os Routers são VIOS 15.6 com 1 core e 1024 MB de memória RAM, com exceção para os Routers P4 e P5 que têm uma maior capacidade face aos restantes, com 2 cores e 2048 MB de RAM. Isto porque estes routers tem mais interfaces ligadas e encontram-se numa posição mais central da Rede, e como tal, exigem uma maior capacidade de processamento. Já os Routers CEs, como apenas servem de interligação entre os clientes e a rede MPLS, têm apenas 1 core e 512 MB de RAM.

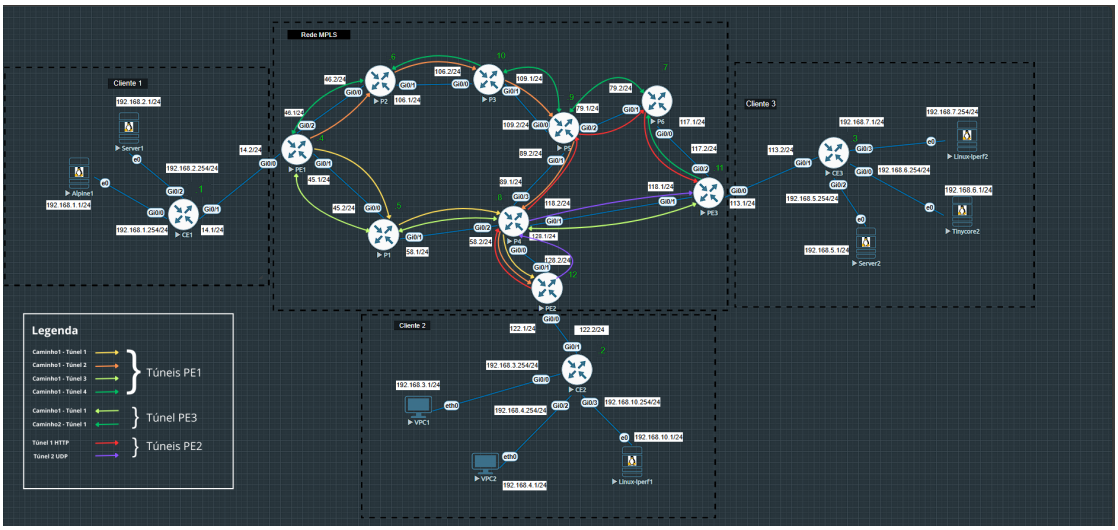

Fig. 1. Visão geral da topologia desenvolvida no EVE-NG, representando a conexão entre Hosts, Customer Edge (CE), Provider Edge (PE) e Provider (P) routers. A configuração ilustra o fluxo de comunicação dentro da infraestrutura, destacando os pontos de entrada e saída do tráfego de rede. As cores repetidas indicam caminhos reversos. Para simplificar e evitar o excesso de linhas, estes foram representados como linhas bidirecionais, embora os túneis sejam, na realidade, unidireccionais.

Na topologia podemos observar os cinco Túneis de tráfego normal (2 túneis de PE1 para PE2, 2 túneis de PE1 para PE3 e 1 túnel com dois caminhos de PE3 para PE1), entre os PEs colocados, assim como os dois Túneis de tráfego HTTP e UDP (2 túneis de PE2 para PE3). De notar que cada túnel é unidirecional, com um destino definido.

B. Descrição dos IPs de encaminhamento

Na Tabela I temos todas as ligações existentes entre os CEs de cada cliente e a Rede MPLS. Também temos todas as ligações entre os PEs, CEs e Ps, com as interfaces de ligação e os IPs correspondentes.

TABLE I: Ligações entre os Routers CEs, PE e Ps

Dispositivo A	**Interface A**	**IP A**	**Dispositivo B**	**Interface B**	**IP B**
CE1	Gi0/1	192.168.14.1/24	PE1	Gi0/0	192.168.14.2/24
PE1	Gi0/1	192.168.45.1/24	P1	Gi0/0	192.168.45.2/24
PE1	Gi0/2	192.168.46.1/24	P2	Gi0/0	192.168.46.2/24
P1	Gi0/1	192.168.58.1/24	P4	Gi0/2	192.168.58.2/24
P4	Gi0/3	192.168.89.1/24	P5	Gi0/1	192.168.89.2/24
P5	Gi0/2	192.168.79.1/24	P6	Gi0/1	192.168.79.2/24
P2	Gi0/1	192.168.106.1/24	P3	Gi0/0	192.168.106.2/24
P3	Gi0/1	192.168.109.1/24	P5	Gi0/0	192.168.109.2/24
P6	Gi0/0	192.168.117.1/24	PE3	Gi0/2	192.168.117.2/24
PE3	Gi0/0	192.168.113.1/24	CE3	Gi0/1	192.168.113.2/24
PE3	Gi0/1	192.168.118.1/24	P4	Gi0/1	192.168.118.2/24
PE2	Gi0/0	192.168.122.1/24	CE2	Gi0/1	192.168.122.2/24
PE2	Gi0/1	192.168.128.1/24	P4	Gi0/0	192.168.128.2/24

TABLE II: Tabela de Routers e os Loopback correspondentes

Router	Loopback
CE1	1.1.1.1/32
CE2	2.2.2.2/32
CE3	3.3.3.3/32
PE1	4.4.4.4/32
P1	5.5.5.5/32
P2	6.6.6.6/32
P6	7.7.7.7/32
P4	8.8.8.8/32
P5	9.9.9.9/32
P3	10.10.10.10/32
PE3	11.11.11.11/32
PE2	12.12.12.12/32

Na Tabela II, o Loopback corresponde ao identificador único de cada Router. O Loopback vai ter um grande importância para a implementação do protocolo MPLS, uma vez que será utilizado para distinguir de forma exclusiva cada dispositivo na rede.

TABLE III: Tabela de Hosts e suas Interfaces

Dispositivo	Interface	IP / Máscara	Conectado a	Gateway
Alpine1	eth0	192.168.1.1/24	CE1	192.168.1.254
Server1	eth0	192.168.2.1/24	CE1	192.168.2.254
VPC1	eth0	192.168.4.1/24	CE2	192.168.4.254
VPC2	eth0	192.168.5.1/24	CE2	192.168.5.254
Linux-Iperf1	eth0	192.168.10.1/24	CE2	192.168.10.254
Linux-Iperf2	eth0	192.168.7.1/24	CE3	192.168.7.254
Server2	eth0	192.168.5.1/24	CE3	192.168.5.254
TinyCore2	eth0	192.168.6.1/24	CE3	192.168.6.254

Na Tabela III, temos a descrição da conexão dos vários Hosts ao CE respetivo, que fará o encaminhamento à rede.

C. Configuração dos Hosts Linux (Server, Alpine, Tinycore, VPC e Personalizado)

Para os Hosts utilizamos quatro imagens Linux (`linux-ubuntu-22.04-server`, `linux-alpine-3.18.4`, o `linux-tinycore-6.4` e uma imagem Linux Ubuntu com **Iperf instalado**) e o VPC padrão do EVE-NG. Como cada imagem é diferente, precisamos de diferentes tipos de configurações. Em cada Host definimos o seu IP, assim como o seu gateway correspondente.

```
ip addr add 192.168.x.1/24 dev <interface>
ip link set <interface> up
ip route add default via 192.168.x.254
```

O código acima mostra o processo para definir um endereço IP no Linux Server, na imagem personalizada e no Alpine. O comando configura o endereço IP na `interface` especificada, ativa-a e define

o gateway padrão. As letras `x` nos IPs devem ser substituídas pela rede correspondente de cada Host. As interfaces do Server e da imagem personalizada são `ens3`, enquanto que a do Alpine é `eth0`.

```
sudo ifconfig eth0 192.168.x.1 netmask 255.255.255.0 up
sudo route add default gw 192.168.x.254
```

Já o código acima exemplifica a configuração manual de rede num sistema TinyCore Linux. O comando `ifconfig` atribui um endereço IP estático à interface `eth0`, ativando-a com a máscara de sub-rede especificada. De seguida, o comando `route` define o gateway padrão para a rede. Tal como antes, a letra `x` deve ser substituída pelo número correspondente à rede utilizada.

```
ip 192.168.x.1 255.255.255.0
ip 192.168.x.1 192.168.x.254
```

Este código especifica a configuração manual de um IP num VPC, com a máscara de sub-rede 255.255.255.0, assim como a definição do gateway do mesmo.

D. Configuração dos Routers CEs

Na nossa topologia, representada na Figura 1, temos três Routers CEs que fazem a ligação dos Clientes à Rede MPLS. Nesta demonstração, vamos utilizar o router CE1 como exemplo, já que os restantes são praticamente iguais, mudando apenas os IPs nas interfaces assim como o loopback. Todos os routers CEs são Routers VIOS (`vios-adventerprisek9-m.SPA.159-3.M6`) com 1 Core e 512 MB de RAM.

```
en
configure terminal
hostname CE1
interface GigabitEthernet0/0
ip address 192.168.1.254 255.255.255.0
no shutdown
exit
interface loopback 0
ip address 1.1.1.1 255.255.255.255
no shut
exit
interface GigabitEthernet0/1
ip address 192.168.14.1 255.255.255.0
no shut
exit
interface GigabitEthernet0/2
ip address 192.168.2.254 255.255.255.0
no shut
exit
router ospf 1
network 1.1.1.1 0.0.0.0 area 0
network 192.168.14.0 0.0.0.255 area 0
network 192.168.1.0 0.0.0.255 area 0
network 192.168.2.0 0.0.0.255 area 0
exit
```

Começamos por configurar o nome do dispositivo com `hostname CE1`, e de seguida definimos as interfaces de rede com os respetivos endereços IP. Cada interface é ativada com o comando `no shutdown`, garantindo que a interface fica operacional (*up*).

As interfaces estão organizadas da seguinte forma:

- **Gi0/0** — ligada a um dos **Hosts** do cliente;
- **Gi0/1** — ligada ao Router **PE (Provider Edge)**, que dá acesso à rede **MPLS**;
- **Gi0/2** — ligada ao segundo **Host** do cliente.

Adicionalmente, é criada a interface **Loopback0** com o IP `1.1.1.1`, que serve como identificador único e estável do router. Esta interface mantém-se ativa independentemente do estado das interfaces físicas e é essencial para a operação fiável de protocolos de roteamento.

Por fim, é configurado o protocolo **OSPF (Open Shortest Path First)**, um protocolo de encaminhamento dinâmico do tipo *IGP (Interior Gateway Protocol)*.

O OSPF permite que o **CE1** troque informações de roteamento com outros routers na rede, garantindo o conhecimento dos melhores caminhos para os diferentes destinos.

Todas as interfaces configuradas, incluindo a *loopback*, são incluídas na área `0`, que é a **área backbone** do OSPF, essencial para a interligação entre redes.

E. Configuração dos Routers Ps (LSRs)

Na topologia implementada, existem 6 routers **P** com configurações gerais idênticas, variando apenas no número de interfaces ativas em cada um.

Todos os routers são Routers VIOS (`vios-adventerprisek9-m.SPA.159-3.M6`). Os routers **P4** e **P5** apresentam um maior número de interfaces ligadas, o que justifica uma atribuição de recursos superior, portanto foram configurados com **2 cores** de CPU e **2048 MB de memória RAM**, enquanto os restantes routers da rede core do **MPLS** operam com apenas **1 core** e **1024 MB de RAM**, dado que possuem menor carga de processamento.

O objetivo com esta distinção na alocação de recursos é assegurar o desempenho da rede nos nós que desempenham um papel mais central na topologia e por onde se espera que passe mais tráfego.

```
en
configure terminal
hostname P4
mpls traffic-eng tunnels
interface gi 0/0
ip address 192.168.128.1 255.255.255.0
mpls ip
mpls traffic-eng tunnels
ip rsvp bandwidth 800000
no shut
exit
interface gi 0/1
ip address 192.168.118.2 255.255.255.0
mpls ip
mpls traffic-eng tunnels
ip rsvp bandwidth 800000
no shut
exit
interface gi 0/2
ip address 192.168.58.2 255.255.255.0
mpls ip
```

```
mpls traffic-eng tunnels
ip rsvp bandwidth 800000
no shut
exit
interface gi 0/3
ip address 192.168.89.1 255.255.255.0
mpls ip
mpls traffic-eng tunnels
ip rsvp bandwidth 800000
no shut
exit
interface loopback 0
ip address 8.8.8.8 255.255.255.255
no shut
exit
router ospf 1
network 8.8.8.8 0.0.0.0 area 0
network 192.168.128.0 0.0.0.255 area 0
network 192.168.118.0 0.0.0.255 area 0
network 192.168.58.0 0.0.0.255 area 0
network 192.168.89.0 0.0.0.255 area 0
mpls traffic-eng router-id Loopback0
mpls traffic-eng area 0
exit
mpls ldp router-id loopback 0 force
```

Neste exemplo, temos a configuração do router **P4**, um dos routers do núcleo da rede **MPLS (Multiprotocol Label Switching)**. Este foi o exemplo escolhido porque é o router P mais complexo, já que é o que tem mais interfaces conectadas.

Nesta configuração temos 4 interfaces físicas, uma interface loopback, e a ativação de funcionalidades essenciais como **MPLS**, **OSPF** e **RSVP**.

Começamos por configurar o nome do dispositivo com `hostname P4`, seguido da ativação do suporte a `mpls traffic-eng tunnels`, que permite o uso de engenharia de tráfego (TE) com túneis MPLS. **Traffic Engineering (TE)** é uma funcionalidade avançada do MPLS que visa otimizar o encaminhamento do tráfego na rede. Ao contrário do IP tradicional, que envia os pacotes sempre pelo caminho de menor custo, o TE permite definir manualmente ou dinamicamente os caminhos que o tráfego deve seguir, com base em critérios como a *bandwidth*, *latency*, etc.

Em seguida, são configuradas quatro interfaces físicas:

- **Gi0/0** com IP `192.168.128.1`;
- **Gi0/1** com IP `192.168.118.2`;
- **Gi0/2** com IP `192.168.58.2`;
- **Gi0/3** com IP `192.168.89.1`.

Todas as interfaces são ativadas com `no shutdown`, têm o `mpls ip` habilitado, e suportam **RSVP (Resource Reservation Protocol)** com `ip rsvp bandwidth 800000`, o que permite reservar uma largura de banda de **800 Mbps** para túneis de engenharia de tráfego. É importante destacar que a capacidade total do router VIOS é de 1 Gbps, o que significa que a reserva de 800 Mbps deixa cerca de 200 Mbps disponíveis para outros tipos de tráfego ou serviços na mesma interface. O RSVP assegura que, uma vez estabelecido o túnel de TE, os 800 Mbps estarão disponíveis para o tráfego específico do túnel.

É também criada a interface **Loopback0** com o IP `8.8.8.8`, que funciona como **router-id** estável, assim como o **OSPF**, que é utilizado para disseminar as rotas dentro da rede. Esta última é praticamente

igual à configuração dos CEs, mas agora definimos o **Loopback0** como o **router-id** para o traffic engineering (`mpls traffic-eng router-id Loopback0`) e ativamos a engenharia de tráfego na **área 0** com `mpls traffic-eng area 0`.

Por fim, é ativado o **LDP (Label Distribution Protocol)** com o comando `mpls ldp router-id loopback 0 force`, que configura a interface **Loopback0** como o **router-id** para a distribuição de rótulos MPLS. O **LDP** é responsável por distribuir rótulos entre os routers para permitir o encaminhamento de pacotes MPLS de forma eficiente. Usar a interface `Loopback0` como **router-id** assegura uma identificação estável e o comando `force` garante que o **router-id** seja sempre o `Loopback0`.

F. Configuração dos Routers PEs (LERs)

Na nossa topologia, contamos com três routers Provider Edge (PEs), que atuam como interfaces entre os routers internos da rede MPLS e os routers de cliente (CEs), que por sua vez estabelecem ligação com os hosts. Todos os equipamentos utilizados são Routers VIOS (`vios-adventerprisek9-m.SPA.159-3.M6`), com configurações padrão de 1 core e 1024 MB de RAM.

```
en
configure terminal
hostname PE1
mpls traffic-eng tunnels
int gi0/0
ip address 192.168.14.2 255.255.255.0
no shut
exit
int gi0/1
ip address 192.168.45.1 255.255.255.0
mpls ip
mpls traffic-eng tunnels
ip rsvp bandwidth 800000
no shut
exit
int gi0/2
ip address 192.168.46.1 255.255.255.0
mpls ip
mpls traffic-eng tunnels
ip rsvp bandwidth 800000
no shut
exit
interface loopback 0
ip address 4.4.4.4 255.255.255.255
no shut
exit
router ospf 1
network 4.4.4.4 0.0.0.0 area 0
network 192.168.14.0 0.0.0.255 area 0
network 192.168.45.0 0.0.0.255 area 0
network 192.168.46.0 0.0.0.255 area 0
mpls traffic-eng router-id Loopback0
mpls traffic-eng area 0
exit
mpls ldp router-id loopback 0 force
```

Antes da configuração dos túneis MPLS-TE, os routers PE apresentam uma configuração bastante semelhante à dos routers P, variando essencialmente nas interfaces ativas, que dependem do papel de cada PE na topologia. No exemplo acima, utilizámos o PE1 como referência, mas a configuração base

pode ser adaptada de forma análoga aos restantes PEs, com pequenas variações consoante os endereços IP e a conectividade com os CEs.

Nas subsecções seguintes, será realizada a configuração de túneis MPLS Traffic Engineering (TE) neste router PE1, com o objetivo de direcionar o tráfego de forma mais eficiente e controlada pela rede. Estes túneis irão permitir a definição de caminhos explícitos, otimizando a utilização da largura de banda e assegurando um maior controlo sobre o encaminhamento do tráfego através da rede MPLS.

G. Testes de Conectividade da Rede

Nesta fase, fizemos diversos testes de ligação na topologia, para garantir que a configuração básica da rede foi concretizada com sucesso. Os testes também serviram como base e controlo perante os testes posteriores à aplicação de Túneis.

Inicialmente, para garantir que toda a topologia estivesse devidamente estabelecida e que a conectividade entre os dispositivos fosse funcional, foram enviados `pings` ICMP a partir de hosts distribuídos pelos três clientes.

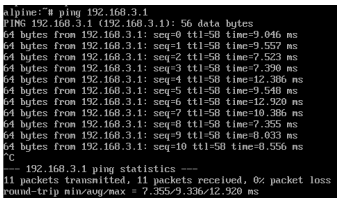

Fig. 2. Ping do Cliente 1 para o 2

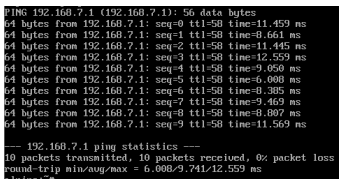

Fig. 3. Ping do Cliente 1 para o 3

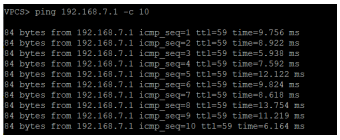

Fig. 4. Ping do Cliente 2 para o 3

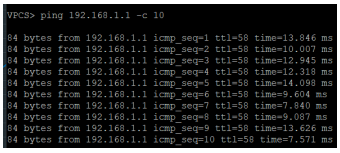

Fig. 5. Ping do Cliente 2 para o 1

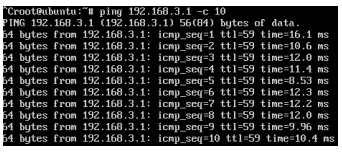

Fig. 6. Ping do Cliente 3 para o 1

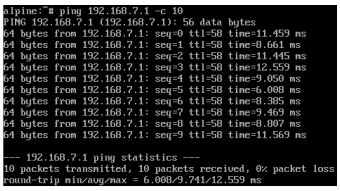

Fig. 7. Ping do Cliente 3 para o 2

Para além dos pings, foi utilizado o comando `show ip ospf neighbor` para garantir que os loopbacks de todos os routers vizinhos são reconhecidos. Esta verificação é fundamental para a atribuição correta das labels no processo de desenvolvimento da rede MPLS.

Fig. 8. Vizinhos OSPF do CE1

Fig. 9. Vizinhos OSPF do CE2

Fig. 10. Vizinhos OSPF do CE3

Fig. 11. Vizinhos OSPF do P1

Fig. 12. Vizinhos OSPF do P2

Fig. 13. Vizinhos OSPF do P3

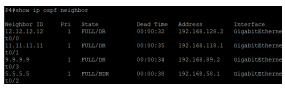

Fig. 14. Vizinhos OSPF do P4

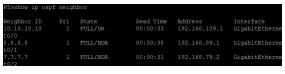

Fig. 15. Vizinhos OSPF do P5

Fig. 16. Vizinhos OSPF do P6

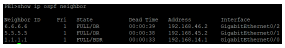

Fig. 17. Vizinhos OSPF do PE1

Fig. 18. Vizinhos OSPF do PE2

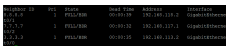

Fig. 19. Vizinhos OSPF do PE3

H. Configuração dos Túneis MPLS

Os túneis MPLS (Multiprotocol Label Switching) são utilizados para criar uma ligação eficiente entre dois *Provider Edge* (PE), também chamados de *Label Edge Routers* (LERs). O objetivo principal de um túnel MPLS é estabelecer uma comunicação entre esses dispositivos, proporcionando uma forma de encaminhamento de pacotes através da rede utilizando *labels*.

A criação de um túnel MPLS começa com a atribuição da *label*. Esta *label* é associada a um caminho predefinido na rede, conhecido como LSP (Label Switched Path). O LSP é, na prática, uma sequência de *labels* que orientam os pacotes através de diferentes routers da rede, permitindo que o encaminhamento seja feito com base na *label* ao invés de endereços IP. Este processo é eficiente, porque em vez de avaliar todo o cabeçalho IP, os Routers apenas analisam as *labels*, o que permite um aumento de performance no envio dos pacotes pela rede.

A configuração dos túneis MPLS envolve a definição de parâmetros como o destino do túnel, o *traffic engineering*, entre outros. Estas configurações dos túneis MPLS permitem criar caminhos eficientes para a transmissão dos dados, aplicando políticas de qualidade de serviço (QoS).

```
en
configure terminal
mpls traffic-eng tunnels
mpls ip
ip cef load-sharing algorithm universal
//mpls ip em todas as interfaces internas da rede MPLS
//mpls traffic-eng tunnels em todas as interfaces internas da rede
MPLS
//ip rsvp bandwidth em todas as interfaces internas da rede MPLS

//PE1->P1->P4->PE2
ip explicit-path name CAMINHO1P12 enable
 next-address 5.5.5.5
 next-address 8.8.8.8
 next-address 12.12.12.12

interface Tunnel1
ip unnumbered Loopback0
tunnel mpls traffic-eng
tunnel destination 12.12.12.12
tunnel mpls traffic-eng autoroute announce
tunnel mpls traffic-eng priority 2 2
tunnel mpls traffic-eng bandwidth 200000
tunnel mpls traffic-eng path-option 1 explicit name CAMINHO1P12
tunnel mpls traffic-eng load-share 1
load-interval 30
ip load-sharing per-packet
exit
```

Este exemplo de configuração do Túnel 1 do PE1, representado na Figura 1, ilustra como criar e configurar um túnel MPLS numa rede.

O comando `mpls traffic-eng tunnels` ativa o suporte para túneis MPLS na rede, permitindo a utilização de *traffic engineering*. Este comando é aplicado em todas as interfaces dentro da Rede MPLS, assim como dentro de cada interface dos Túneis.

O comando `mpls ip` é também utilizado em todas as interfaces internas da rede, garantindo que o roteamento IP seja suportado no contexto MPLS.

Em seguida, é criado um *explicit path* chamado de CAMINHO1P12, que define o caminho exato pelo qual o tráfego deverá ser encaminhado. Com os comandos `next-address`, são especificados os routers de cada hop no caminho, representando tanto os dispositivos intermediários como o destino final do túnel MPLS.

O comando `ip unnumbered Loopback0` garante que a interface do túnel utilize o endereço IP do Loopback0 do dispositivo, permitindo que o túnel seja logicamente associado ao endereço IP do PE1, que é único, sem precisar de um endereço específico para a interface do túnel.

O `tunnel destination 12.12.12.12` especifica o destino final do túnel. O comando `tunnel mpls traffic-eng autoroute announce` permite que o túnel seja anunciado automaticamente na rede MPLS para outros dispositivos PEs.

Além disso, também definimos a prioridade do túnel com o `tunnel mpls traffic-eng priority 2 2`, que indica a prioridade de encaminhamento do tráfego através do túnel. A largura de banda do túnel é definida como 200 Mbps utilizando o comando `tunnel mpls traffic-eng bandwidth 200000`.

A opção de caminho do túnel é configurada com o comando `tunnel mpls traffic-eng path-option 1 explicit name CAMINHO1P12`, ou seja, o caminho explícito definido anteriormente. O comando `tunnel mpls traffic-eng load-share 1` define a distribuição de carga, garantindo que o tráfego seja distribuído uniformemente.

Por fim, para garantir que os pacotes sejam distribuídos igualmente entre todos os túneis disponíveis, é necessário ativar o balanceamento de carga por pacote globalmente com o comando: `ip cef load-sharing algorithm universal`. Já os comandos `load-interval 30` e `ip load-sharing per-packet` serviram para definir o intervalo de medição de carga em 30 segundos para melhor visibilidade e forçar o envio de pacotes alternadamente entre os túneis, respetivamente.

1) Testes práticos dos túneis: Para verificar a configuração e o estado dos túneis MPLS estabelecidos nos PEs, utilizamos o comando `show ip cef`, que exibe a tabela de encaminhamento Cisco Express Forwarding (CEF) para os Ips de destino dos túneis MPLS, e o comando `show ip route`, que exibe a tabela de routing IP que contém todas as rotas configuradas para os endereços IP do PE de destino dos túneis MPLS.

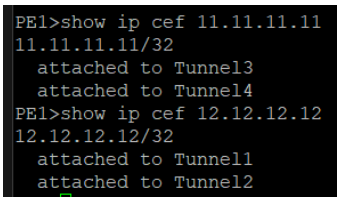

Fig. 20. Túneis estabelecidos sobre o PE1

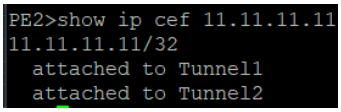

Fig. 21. Túneis estabelecidos sobre o PE2

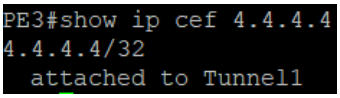

Fig. 22. Túneis estabelecidos sobre o PE3

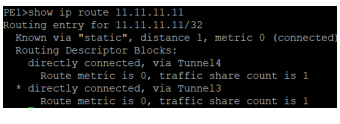

Fig. 23. Show Ip Route PE1

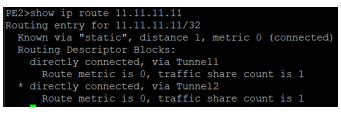

Fig. 24. Show Ip Route PE2

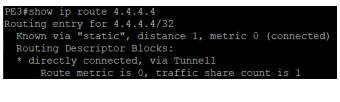

Fig. 25. Show Ip Route PE3

Para analisar os nossos Túneis, através do comando `show mpls traffic-eng tunnels` em cada PE, verificamos algumas características importantes destacadas nas Tabelas IV, V e VI.

No PE1 existem 4 túneis, sendo dois para o mesmo destino mas seguindo caminhos diferentes. O objetivo é forçar a repartição da transmissão de pacotes pelos túneis, de forma distribuída. Anteriormente, foi adotada uma política de repartir os túneis em dois caminhos explícitos como na tabela VI, porém essa técnica revelou não permitir a distribuição pelos caminhos pois era escolhido apenas 1 caminho por túnel, aquele que tivesse um path weight menor. Cada ligação na topologia corresponde a um valor no path weight. Isto significa que os túneis de maior valor são considerados mais pesados e, por conseguinte, são tratados como os túneis menos desejáveis para a retransmissão pelos métodos OSPF.

Como todos os PEs têm a interface **Gi0/0** conectada à rede externa com os clientes, nas restantes interfaces são estabelecidos túneis com interfaces de saída distintas. Isto garante que cada túnel siga um caminho separado. A prioridade é uma medida baseada em *Differentiated Services* (DiffServ), que define a importância relativa de um túnel na rede. Através dessa definição, é possível controlar a forma como o tráfego é tratado e encaminhado, permitindo que túneis com maior prioridade recebam maior atenção na alocação de recursos de rede, enquanto túneis com prioridade inferior são prejudicados.

Túnel	Origem	Destino	Caminho	Path Weight	Interface	Label	Banda (kbps)	Prioridade
PE1_t1	4.4.4.4	12.12.12.12	CAMINHO1P12	3	Gi0/1	16	200000	2/2
PE1_t2	4.4.4.4	12.12.12.12	CAMINHO2P12	5	Gi0/2	19	200000	2/2
PE1_t3	4.4.4.4	11.11.11.11	CAMINHO1P13	3	Gi0/1	17	200000	2/2
PE1_t4	4.4.4.4	11.11.11.11	CAMINHO2P13	5	Gi0/2	20	200000	2/2

TABLE IV
CONFIGURAÇÃO DOS TÚNEIS MPLS TE NO PE1

Túnel	Origem	Destino	Caminho	Path Weight	Interface	Label	Banda (kbps)	Prioridade
PE2_t1	12.12.12.12	11.11.11.11	CAMINHO_TCP	4	Gi0/1	27	200000	1/1
PE2_t2	12.12.12.12	11.11.11.11	CAMINHO_UDP	2	Gi0/1	26	200000	1/1

TABLE V
CONFIGURAÇÃO DOS TÚNEIS MPLS TE NO PE2. NESTA CONFIGURAÇÃO TEMOS OS TÚNEIS TCP E UDP QUE SERÃO ABORDADOS NUMA SECÇÃO POSTERIOR

Túnel	Origem	Destino	Caminho	Path Weight	Interface	Label	Banda (kbps)	Prioridade
PE3_t1	11.11.11.11	4.4.4.4	CAMINHO2P31, CAMINHO1P31	5	Gi0/2	40	200000	1/1

TABLE VI
CONFIGURAÇÃO DOS TÚNEIS MPLS TE NO PE3. APESAR DE SER DEFINIDO DOIS CAMINHOS, O CAMINHO 2P31 FOI O ESCOLHIDO, PORQUE APRESENTA UM MENOR PESO

Em seguida, decidimos observar o caminho do pacote pela nossa rede com o comando `traceroute x.x.x.x source Loopback0 numeric probe 1`. Como exibido nas Figuras 26 e 27, o pacote é encapsulado com uma Label MPLS ao entrar num PE, e é enviado até ao PE de destino nesta rede. Quando chega ao destino, antes de atingir o PE, o pacote passa por um processo de `POP Label`, onde a Label MPLS é removida. Importa ainda referir que, geralmente, o envio de pacotes com Label MPLS apresenta uma latência mais baixa do que o envio convencional por IP.

```
CE1#traceroute 11.11.11.11 source Loopback 0 numeric probe 1
Type escape sequence to abort.
Tracing the route to 11.11.11.11
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.14.2 6 msec
  2 192.168.45.2 [MPLS: Label 17 Exp 0] 12 msec
  3 192.168.58.2 [MPLS: Label 47 Exp 0] 24 msec
  4 192.168.118.1 13 msec
CE1#traceroute 12.12.12.12 source Loopback 0 numeric probe 1
Type escape sequence to abort.
Tracing the route to 12.12.12.12
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.14.2 4 msec
  2 192.168.46.2 [MPLS: Label 19 Exp 0] 12 msec
  3 192.168.106.2 [MPLS: Label 32 Exp 0] 10 msec
  4 192.168.109.2 [MPLS: Label 38 Exp 0] 8 msec
  5 192.168.89.1 [MPLS: Label 38 Exp 0] 15 msec
  6 192.168.128.2 13 msec
```

Fig. 26. Percurso do pacote percorrido de dois Túneis de CE1 para o PE2 (12.12.12.12) e CE1 para o PE3 (11.11.11.11) através do comando `traceroute`

```
CE3#traceroute 4.4.4.4 source Loopback 0 numeric probe 1
Type escape sequence to abort.
Tracing the route to 4.4.4.4
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.113.1 5 msec
  2 192.168.118.2 [MPLS: Label 48 Exp 0] 9 msec
  3 192.168.58.1 [MPLS: Label 47 Exp 0] 7 msec
  4 192.168.45.1 12 msec
```

Fig. 27. Percurso do pacote percorrido do Túnel de CE3 para o PE1 (4.4.4.4) através do comando `traceroute`

I. Testes para garantir Load-balance entre os Túneis com o mesmo destino

O nosso objetivo nesta secção é demonstrar que dos 4 Túneis construídos no PE1, os que partilham o mesmo destino, também parrtilham 50% dos pacotes entre eles.

Para testar, utilizamos o comando `traceroute x.x.x.x source Loopback0 numeric probe 1` duas vezes seguidas para o mesmo destino. O esperado é que a digitar duas vezes este comando, o percurso do pacote seja diferente, o que é verificado nas Figuras 28 e 29.

Outro teste realizado foi o envio, a partir do CE1 de vários pings para o destino final com o comando: `ping 11.11.11.11 size 1000 repeat 1000` e `ping 12.12.12.12 size 1000 repeat 1000`. Para verificar se a distribuição de pacotes foi feita corretamente para cada Túnel, utilizamos o `show interface Tunnelx | include packets`. Como observado na Figura 30, os pacotes encontram-se distribuídos de forma igual entre os Túneis.

```
CE1#traceroute 11.11.11.11 source Loopback 0 numeric probe 1
Type escape sequence to abort.
Tracing the route to 11.11.11.11
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.14.2 3 msec
  2 192.168.45.2 [MPLS: Label 17 Exp 0] 6 msec
  3 192.168.106.2 [MPLS: Label 48 Exp 0] 8 msec
  4 192.168.118.1 7 msec
CE1#traceroute 11.11.11.11 source Loopback 0 numeric probe 1
Type escape sequence to abort.
Tracing the route to 11.11.11.11
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.14.2 4 msec
  2 192.168.46.2 [MPLS: Label 48 Exp 0] 9 msec
  3 192.168.58.2 [MPLS: Label 24 Exp 0] 9 msec
  4 192.168.109.2 [MPLS: Label 50 Exp 0] 10 msec
  5 192.168.118.1 12 msec
```

Fig. 28. Dois túneis diferentes para o mesmo destino (PE3 - 11.11.11.11)

```
CE1#traceroute 12.12.12.12 source Loopback 0 numeric probe 1
Type escape sequence to abort.
Tracing the route to 12.12.12.12
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.14.2 5 msec
  2 192.168.45.2 [MPLS: Label 49 Exp 0] 7 msec
  3 192.168.106.2 [MPLS: Label 32 Exp 0] 8 msec
  4 192.168.128.2 6 msec
CE1#traceroute 12.12.12.12 source Loopback 0 numeric probe 1
Type escape sequence to abort.
Tracing the route to 12.12.12.12
VRF info: (vrf in name/id, vrf out name/id)
  1 192.168.14.2 3 msec
  2 192.168.46.2 [MPLS: Label 17 Exp 0] 6 msec
  3 192.168.58.2 [MPLS: Label 23 Exp 0] 6 msec
  4 192.168.109.2 [MPLS: Label 32 Exp 0] 9 msec
  5 192.168.128.2 6 msec
```

Fig. 29. Dois túneis diferentes para o mesmo destino (PE2 - 12.12.12.12)

```
PE1#show interface Tunnel1 | include packets
     0 packets input, 0 bytes, 0 no buffer
     500 packets output, 509000 bytes, 0 underruns
PE1#show interface Tunnel2 | include packets
     0 packets input, 0 bytes, 0 no buffer
     500 packets output, 509000 bytes, 0 underruns
PE1#show interface Tunnel3 | include packets
     0 packets input, 0 bytes, 0 no buffer
     500 packets output, 509000 bytes, 0 underruns
PE1#show interface Tunnel4 | include packets
     0 packets input, 0 bytes, 0 no buffer
     500 packets output, 509000 bytes, 0 underruns
```

Fig. 30. Distribuição igual dos pacotes pelos Túneis

J. Configuração dos Túneis HTTP e UDP

Na configuração dos Túneis HTTP e UDP, assim como nos túneis MPLS tradicionais, é necessário definir túneis MPLS. No nosso caso, configurámos dois túneis do PE2 para o PE3, conforme descrito na tabela V.

```
ip access-list extended HTTP_TRAFFIC
 permit tcp any any eq www
 permit tcp any any eq 8080

ip access-list extended UDP_TRAFFIC
```

```
permit udp any any range 18384 32677
permit tcp any any

route-map PBR-TRAFICO permit 10
 match ip address HTTP_TRAFFIC
 set interface Tunnel1

route-map PBR-TRAFICO permit 20
 match ip address UDP_TRAFFIC
 set interface Tunnel2

route-map PBR-TRAFICO permit 30

interface Tunnel1
 no shutdown
 ip unnumbered Loopback0
 tunnel mode mpls traffic-eng
 tunnel destination 11.11.11.11
 tunnel mpls traffic-eng priority 1 1
 tunnel mpls traffic-eng bandwidth 200000
 tunnel mpls traffic-eng path-option 1 explicit name CAMINHO_TCP
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng fast-reroute

interface Tunnel2
 no shutdown
 ip unnumbered Loopback0
 tunnel mode mpls traffic-eng
 tunnel destination 11.11.11.11
 tunnel mpls traffic-eng priority 1 1
 tunnel mpls traffic-eng bandwidth 200000
 tunnel mpls traffic-eng path-option 1 explicit name CAMINHO_UDP
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng fast-reroute

ip route 11.11.11.11 255.255.255.255 Tunnel1
ip route 11.11.11.11 255.255.255.255 Tunnel2
```

A principal diferença nesta configuração está na utilização de **Access Lists (ACLs)** e **Policy-Based Routing (PBR)**. As ACLs são utilizadas para filtrar o tráfego com base na porta, permitindo especificar quais pacotes TCP (portas 80 e 8080) e UDP (intervalo de portas 18384-32677) devem ser encaminhados através dos túneis correspondentes.

No caso do tráfego UDP enviado pelo `iperf3`, embora a informação seja inteiramente transmitida por UDP, foi necessário permitir também o tráfego TCP em qualquer porta. Isto deve-se ao facto de o `iperf3`, mesmo em modo UDP, recorrer inicialmente a uma ligação TCP para negociar os parâmetros da sessão e transmitir estatísticas no final do teste.

Como tal, no PBR do UDP_TRAFFIC teria que permitir qualquer tráfego HTTP através do `permit tcp any any`. O PBR (Policy-Based Routing) é então aplicado na interface **Gi0/0** para redirecionar esse tráfego para os túneis apropriados, garantindo que apenas o tráfego desejado seja encaminho através do túnel especificado. Para verificar que a configuração foi bem implementada utilizamos os comandos `show ip policy` e `show ip acess-lists`, como se pode verificar nas Figuras 31 e 32.

```
PE2#show ip policy
Interface             Route map
Gi0/0                 PBR-TRAFICO
```

Fig. 31. Interface Gi0/0 do PE2 associado ao ip policy PBR-TRAFICO

```
PE2#show ip access-lists
Extended IP access list HTTP_TRAFFIC
    10 permit tcp any any eq www (1733 matches)
    20 permit tcp any any eq 8080
Extended IP access list UDP_TRAFFIC
    10 permit udp any any range 18384 32677 (4211 matches)
    20 permit tcp any any (130 matches)
```

Fig. 32. ACLs com as especificacoes das portas. De notar os matches dos pacotes que satisfazeram as condições.

1) Testes práticos dos túneis HTTP e UDP: Para realizar estes testes práticos tivemos de acrescentar uma nova máquina Linux que tivesse instaladas ferramentas essenciais para a realização dos testes, como o *Iperf3*. Assim, e como ambos os túneis HTTP e UDP ligam unilateralmente o PE2 ao PE3 (como é possível visualizar nas setas vermelhas e roxas da Figura 1), colocamos uma máquina com essa imagem na rede do cliente 2 e outra na rede do cliente 3, atuando a máquina do cliente 2 como emissor e a do cliente 3 como servidor.

O objetivo dos testes HTTP é experimentar se os pacotes seriam direcionados para o Túnel 1 ao chegarem à interface **Gi0/0** do Router PE2.

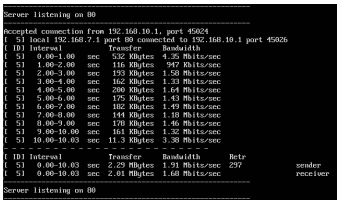

Fig. 33. O servidor Iperf3 a escutar na porta 80.

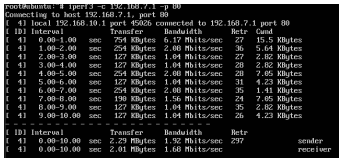

Fig. 34. O emissor Iperf3 a enviar pacotes para a porta 80 do servidor.

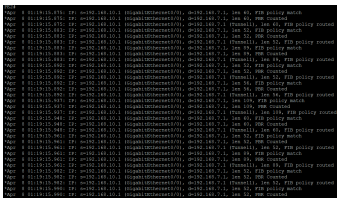

Fig. 35. Logs do comando `debug ip policy`.

Para testar, através do Iperf3, o emissor (Figura 34) envia tráfego TCP para a porta 80 do Servidor (Figura 33). Na Figura 35, podemos verificar que os pacotes TCP estão a ser enviados pelo Túnel 1.

Do mesmo modo, simulámos também o envio de tráfego UDP de um emissor (Figura 38) para um servidor (Figuras 36 e 37), a ouvir na porta 19000. Na Figura 35, é possível verificar que os pacotes UDP estão a ser direcionados e enviados pelo Túnel 2.

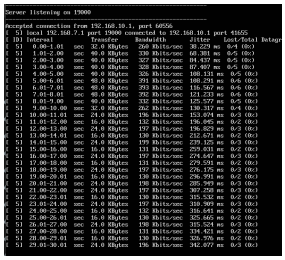

Fig. 36. O servidor Iperf3 a escutar na porta 19000.

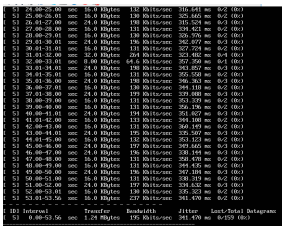

Fig. 37. O servidor Iperf3 a escutar na porta 19000.

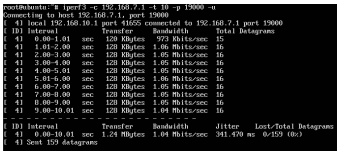

Fig. 38. O emissor Iperf3 a enviar pacotes para a porta 19000 do servidor.

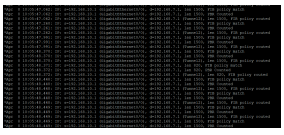

Fig. 39. Logs do comando `debug ip policy`.

K. Implementações Tentadas (sem sucesso)

Ao longo do desenvolvimento deste trabalho fizemos diversas alterações na topologia, nas imagens escolhidas e até nas configurações de cada router. Inicialmente, com a previsão de uma maior necessidade de processamento nos routers P4 e P5, optámos pela utilização dos Routers CSR1000V uma vez que são routers de maior potência comparadamente aos VIOS. No entanto, a diferença de versões entre os VIOS e os CSR1000V tornou a interligação entre eles bastante complicada, o que resultou numa perda significativa de tempo na tentativa de configuração correta. Além disso, devido às elevadas exigências computacionais e de memória para a virtualização dos CSR1000V, algo que dificultava o progresso do trabalho, o grupo decidiu optar pela sua remoção. Na topologia inicial também escolhemos utilizar switches VIOS L2 para conectar os diversos hosts aos routers CE correspondentes. Contudo, o uso deste switch, além de aumentar a carga de recursos da nossa topologia, apresentou problemas de conectividade com os routers CEs, o que levou também à reavaliação dessa escolha.

III. Conclusões

Este trabalho prático revelou-se, na sua essência, bastante desafiante. A configuração do ambiente necessário para a sua execução foi algo demorada, principalmente devido à instalação das diversas imagens e à resolução de erros que surgiram ao longo do processo.

Ainda assim, permitiu-nos explorar o ambiente de simulação e conhecer uma ferramenta amplamente utilizada na configuração de topologias de rede. Para além disso, possibilitou-nos familiarizar com diferentes tipos de equipamentos, em particular diversos modelos de routers e as suas respetivas características.

Relativamente ao trabalho em si, o maior desafio esteve associado à necessidade de trabalhar numa plataforma nova, com todos os obstáculos naturais da sua aprendizagem e adaptação. No entanto, esta experiência permitiu-nos aplicar, de forma prática, os conceitos discutidos nas aulas teóricas, nomeadamente a implementação de uma rede MPLS com técnicas de traffic engineering, filtragem e encaminhamento personalizado através de túneis, consoante a porta de envio.

Ao contrário da plataforma CORE, o EVE-NG apresenta uma abordagem mais low-level, o que nos permitiu compreender de forma mais detalhada o funcionamento de vários protocolos, como o OSPF para a descoberta de vizinhos, o RSVP para a definição da largura de banda máxima nas interfaces, o LDP para a distribuição de labels, entre outros.

De um modo geral, consideramos que foi um excelente trabalho introdutório, tanto ao EVE-NG como aos principais conceitos associados à construção de uma topologia básica com suporte a MPLS para o envio e receção de pacotes.