UNIVERSIDADE DO MINHO | 2020/2021

ENGENHARIA DE TRÁFEGO COM MPLS

ENCAMINHAMENTO DE TRÁFEGO EM REDES IP

GRUPO 1:

André Gonçalo Ribeiro da Silva Lopes……A75363

Jorge Manuel de Almeida e Sousa……A74230

Leandro Henrique Dantas Alves……A82157

1 Topologia

Neste trabalho prático foram realizadas mais algumas modificações na topologia da rede do sistema autónomo AS65100, mais propriamente, na área 0, para tornar esta mais complexa para a criação de vários túneis e caminhos distintos para a engenharia de tráfego MPLS. A topologia e o cenário experimental estão representados na

Figura 1 e Figura 2 , respetivamente.

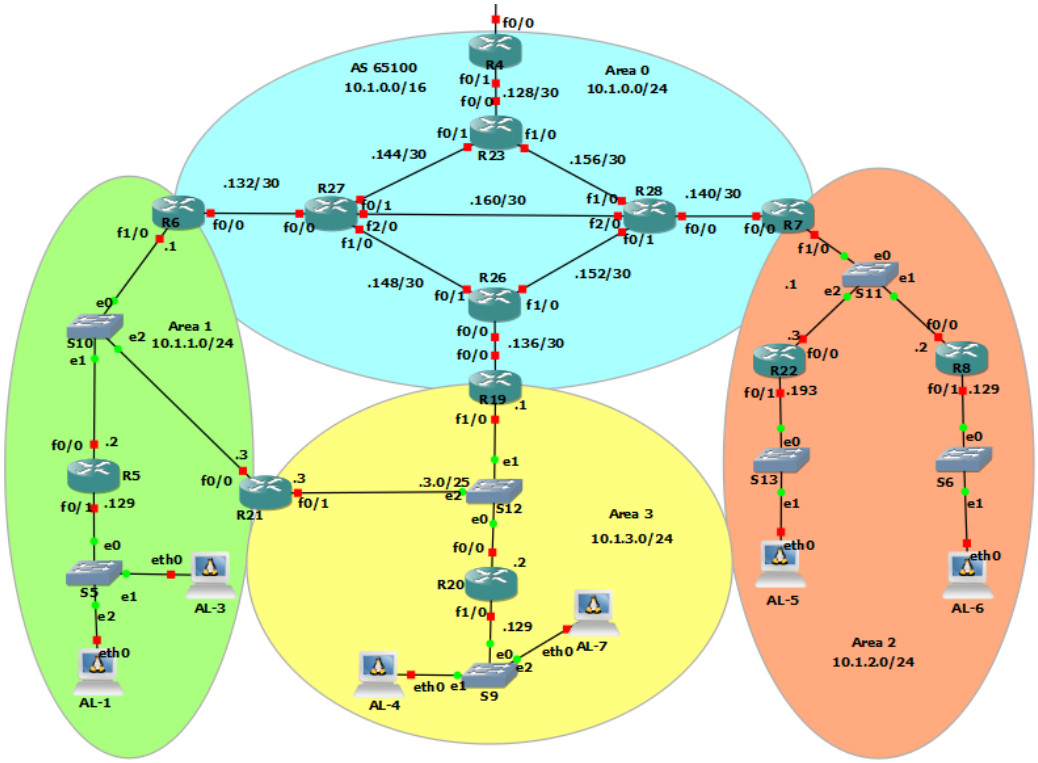

Figura 1 - Topologia física

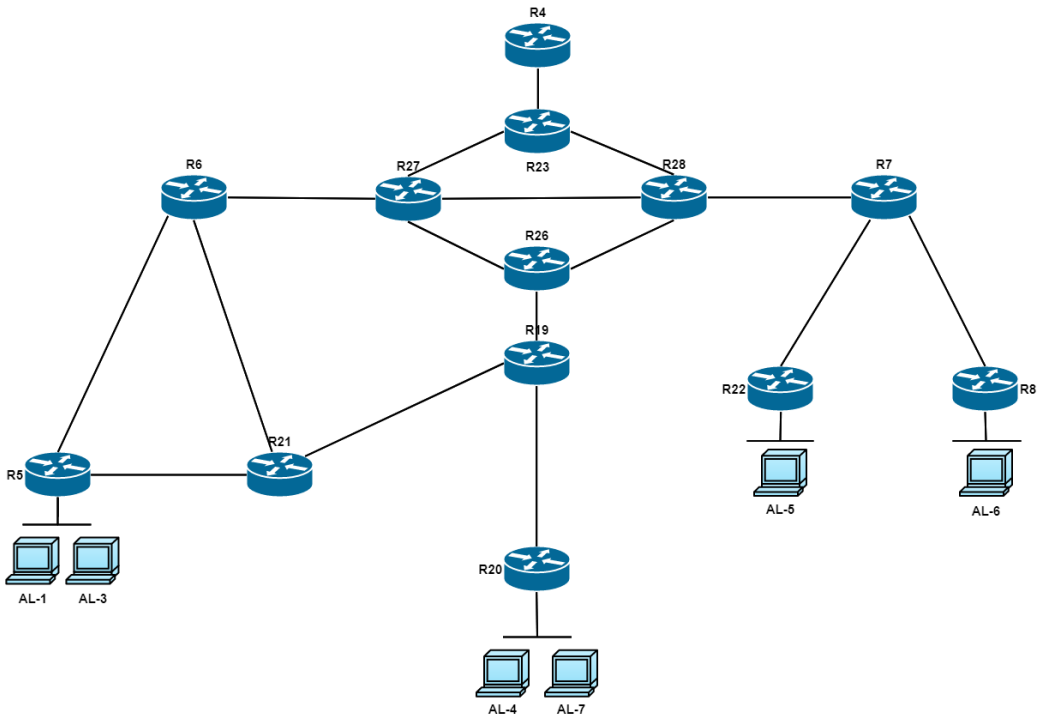

Figura 2 - Cenário experimental AS65100

2 Balanceamento de carga de forma desigual entre dois túneis MPLS

Nesta fase, configurou-se a área 0 (área *backbone*) do Sistema Autónomo AS65100 para suportar a engenharia de tráfego MPLS. Para o correto funcionamento da rede MPLS foram escolhidos, como *routers* LER, todos os *routers* ABR da área 0 e, como *routers* LSR, todos os *routers* internos da área. Na Figura 3 está uma representação da topologia com a identificação dos *routers* LER e LSR.

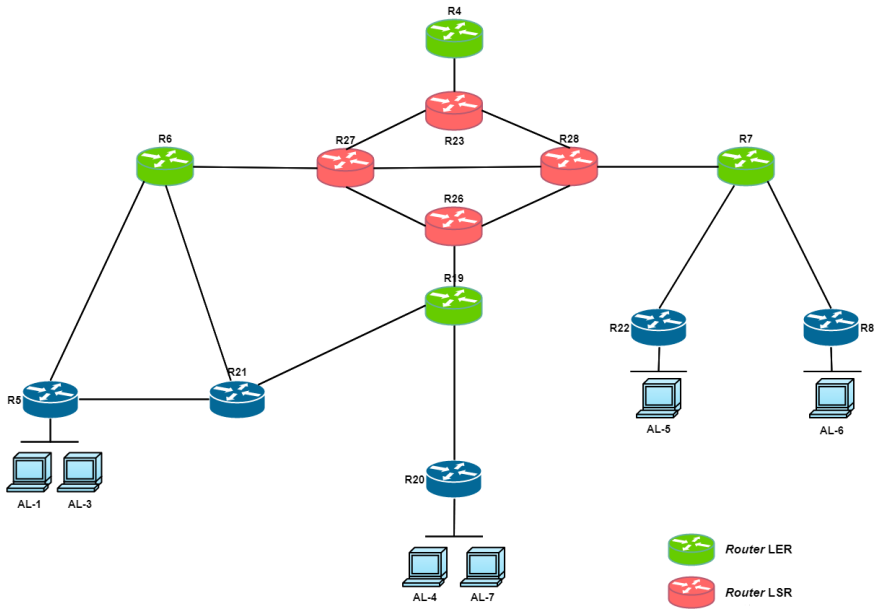

Figura 3 – Topologia com a identificação dos *routers* LER e LSR

Depois de identificados os *routers* internos ao domínios MPLS e os *routers* fronteira, procedeu-se à configuração dos mesmos. A configuração foi dividida em 3 etapas: criação de interfaces de *Loopback* em todos os *routers*, configuração da engenharia de tráfego MPLS e a criação de túneis.

Na primeira etapa, foram configuradas interfaces de *Loopback* com endereços IP **10.10.10.X/32**, onde X corresponde a um valor entre 1 e 8, dependendo do *router*.

Na segunda etapa, para a correta configuração da engenharia de tráfego foram aplicados os seguintes comandos, globalmente na configuração do *router*:

- **`mpls traffic-eng tunnels`**: ativar a engenharia de tráfego MPLS
- **`mpls label protocol ldp`**: definir o protocolo LDP para gerar e distribuir etiquetas

Foi necessário também configurar cada interface de cada *router* que estava dentro da rede MPLS, com os seguintes comandos:

- **`mpls traffic-eng tunnels`**
- **`mpls ip`**
- **`ip rsvp bandwidth`** [*total_reservável*] [*máximo_fluxo*]

Após as configurações globais e em cada interface, procedeu-se à reconfiguração do protocolo OSPF para que passasse a anunciar também a informação da engenharia de tráfego MPLS e as redes IP das interfaces *Loopback*, através dos seguintes comandos:

- **`mpls traffic-eng router-id Loopback0`**: definir como identificador do *router* da engenharia de tráfego o endereço IP da interface de *Loopback0*
- **`mpls traffic-eng area 0`**: indicar a área em que a engenharia de tráfego está ativa

Na Figura 4 pode-se ver a configuração de um *router* com os comandos acima descritos.

Na terceira etapa, primeiro, foram criados dois caminhos explícitos entre os dois *routers* LER, através dos seguintes comandos:

- **`ip explicit-path name`** [*name*]
  - **`next-address`** [next_hop_*address*]

Com os caminhos criados, para a criação/configuração do túnel foram utilizados os seguintes comandos:

- **`ip unnumbered Loopback0`**: indicar o endereço IP da interface do túnel
- **`tunnel destination`** [*destination_address*]
- **`tunnel mode mpls traffic-eng`**
- **`tunnel mpls traffic-eng autoroute announce`**: indicar que o protocolo OSPF deve usar o túnel
- **`tunnel mpls traffic-eng bandwidth`** [*bandwidth_value*]
- **`tunnel mpls traffic-eng path-option 10 explicit name`** [*path_name*]

Ambos os túneis criados, foram definidos com a mesma largura de banda para que a proporção de tráfego a enviar por cada um fosse igual, ou seja com uma percentagem de 50%. Cada túnel só tem um caminho possível, o que torna esta divisão de tráfego mais fácil.

Na Figura 5 pode-se ver a configuração de um *router* com a configuração de um túnel.

Depois de configurados todos os *routers* da rede MPLS, verificou-se que ambos os túneis definidos foram criados com sucesso e que o protocolo de distribuição de etiquetas estava a funcionar corretamente, como se pode verificar nas Figura 6, Figura 7, Figura 8, Figura 9, Figura 10 e Figura 11. Na Figura 12 pode-se ver um sumário da representação da rede MPLS na topologia, desenhada através da leitura das tabelas de encaminhamento das etiquetas.

```
hostname R6
!
ip cef
!
!
!
!
no ip domain lookup
ip multicast-routing
!
mpls label protocol ldp
mpls traffic-eng tunnels
multilink bundle-name authenticated
!
interface Loopback0
 ip address 10.10.10.2 255.255.255.255
!
interface FastEthernet0/0
 ip address 10.1.0.133 255.255.255.252
 ip pim sparse-mode
 ip ospf message-digest-key 1 md5 grupo1
 duplex auto
 speed auto
 mpls ip
 mpls traffic-eng tunnels
 ip rsvp bandwidth 512 512
!
router ospf 1
 mpls traffic-eng router-id Loopback0
 mpls traffic-eng area 0
 log-adjacency-changes
 area 0 authentication message-digest
 area 1 authentication message-digest
 area 1 range 10.1.1.0 255.255.255.0
 network 10.1.0.132 0.0.0.3 area 0
 network 10.1.1.0 0.0.0.127 area 1
 network 10.10.10.2 0.0.0.0 area 0
!
```

Figura 4 – Configuração da engenharia de tráfego MPLS do *router* R6

```
interface Tunnel0
 ip unnumbered Loopback0
 tunnel destination 10.10.10.4
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth  256
 tunnel mpls traffic-eng path-option 10 explicit name PATH_TO_A2
 no routing dynamic
!
ip explicit-path name PATH_TO_A2 enable
 next-address 10.1.0.134
 next-address 10.1.0.145
 next-address 10.1.0.157
 next-address 10.1.0.141
!
```

Figura 5 – Configuração do túnel do *router* R6

```
R6#sh mpls traffic-eng tunnels name R6_t0

Name: R6_t0                               (Tunnel0) Destination: 10.10.10.4
  Status:
    Admin: up         Oper: up     Path: valid       Signalling: connected

    path option 10, type explicit PATH_TO_A2 (Basis for Setup, path weight 40)

  Config Parameters:
    Bandwidth: 256      kbps (Global)  Priority: 7  7   Affinity: 0x0/0xFFFF
    Metric Type: TE (default)
    AutoRoute:  enabled   LockDown: disabled  Loadshare: 256      bw-based
    auto-bw: disabled

  InLabel  :  -
  OutLabel : FastEthernet0/0, 16
  RSVP Signalling Info:
       Src 10.10.10.2, Dst 10.10.10.4, Tun_Id 0, Tun_Instance 16
    RSVP Path Info:
      My Address: 10.1.0.133
      Explicit Route: 10.1.0.134 10.1.0.146 10.1.0.145 10.1.0.158
                      10.1.0.157 10.1.0.142 10.1.0.141 10.10.10.4
      Record Route:  NONE
      Tspec: ave rate=256 kbits, burst=1000 bytes, peak rate=256 kbits
    RSVP Resv Info:
      Record Route:  NONE
      Fspec: ave rate=256 kbits, burst=1000 bytes, peak rate=256 kbits
  History:
    Tunnel:
      Time since created: 15 minutes, 35 seconds
      Time since path change: 14 minutes, 39 seconds
    Current LSP:
      Uptime: 14 minutes, 39 seconds
```

Figura 6 – Detalhes do Túnel 0 criado no *router* R6

```
R6#sh mpls traffic-eng tunnels name R6_t1

Name: R6_t1                               (Tunnel1) Destination: 10.10.10.4
  Status:
    Admin: up         Oper: up     Path: valid       Signalling: connected

    path option 10, type explicit PATH2_TO_A2 (Basis for Setup, path weight 40)

  Config Parameters:
    Bandwidth: 256      kbps (Global)  Priority: 7  7   Affinity: 0x0/0xFFFF
    Metric Type: TE (default)
    AutoRoute:  enabled   LockDown: disabled  Loadshare: 256      bw-based
    auto-bw: disabled

  InLabel  :  -
  OutLabel : FastEthernet0/0, 28
  RSVP Signalling Info:
       Src 10.10.10.2, Dst 10.10.10.4, Tun_Id 1, Tun_Instance 16
    RSVP Path Info:
      My Address: 10.1.0.133
      Explicit Route: 10.1.0.134 10.1.0.149 10.1.0.150 10.1.0.153
                      10.1.0.154 10.1.0.142 10.1.0.141 10.10.10.4
      Record Route:  NONE
      Tspec: ave rate=256 kbits, burst=1000 bytes, peak rate=256 kbits
    RSVP Resv Info:
      Record Route:  NONE
      Fspec: ave rate=256 kbits, burst=1000 bytes, peak rate=256 kbits
  History:
    Tunnel:
      Time since created: 15 minutes, 40 seconds
      Time since path change: 14 minutes, 43 seconds
    Current LSP:
      Uptime: 14 minutes, 43 seconds
```

Figura 7 – Detalhes do Túnel 1 criado no *router* R6

```
R27#sh mpls forwarding-table lsp-tunnel
Local  Outgoing    Prefix            Bytes tag  Outgoing    Next Hop
tag    tag or VC   or Tunnel Id      switched   interface
16     16          10.10.10.2 0 [16] 0          Fa0/1       10.1.0.145
28     18          10.10.10.2 1 [16] 0          Fa1/0       10.1.0.150
```

Figura 8 – Tabela de encaminhamento de etiquetas do *router* R27

```
R23#sh mpls forwarding-table lsp-tunnel
Local  Outgoing    Prefix            Bytes tag  Outgoing    Next Hop
tag    tag or VC   or Tunnel Id      switched   interface
16     18          10.10.10.2 0 [16] 0          Fa1/0       10.1.0.157
```

Figura 9 – Tabela de encaminhamento de etiquetas do *router* R23

```
R26#sh mpls forwarding-table lsp-tunnel
Local  Outgoing    Prefix            Bytes tag  Outgoing    Next Hop
tag    tag or VC   or Tunnel Id      switched   interface
18     19          10.10.10.2 1 [16] 0          Fa1/0       10.1.0.154
```

Figura 10 – Tabela de encaminhamento de etiquetas do *router* R26

```
R28#sh mpls forwarding-table lsp-tunnel
Local  Outgoing    Prefix            Bytes tag  Outgoing    Next Hop
tag    tag or VC   or Tunnel Id      switched   interface
18     Pop tag     10.10.10.2 0 [16] 0          Fa0/0       10.1.0.141
19     Pop tag     10.10.10.2 1 [16] 0          Fa0/0       10.1.0.141
```

Figura 11 – Tabela de encaminhamento de etiquetas do *router* R28

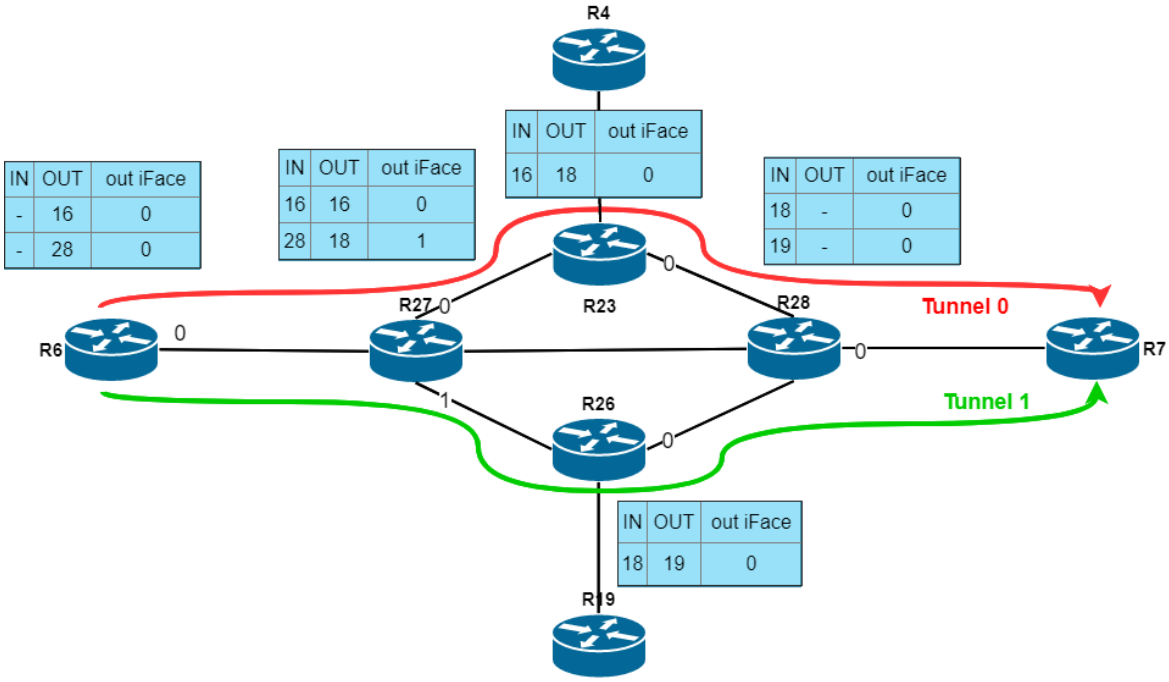

Figura 12 – Rede MPLS com tabela de etiquetas e túneis

Para testar a solução implementada, foram escolhidos como sistemas finais de origem, os sistemas terminais AL-1 e AL-3, e como sistema final de destino, o AL-6. Foram escolhidos dois sistemas finais de origem para que fosse possível testar o balanceamento de tráfego. Como se pode ver pelas Figura 13, Figura 14 e Figura 15, o tráfego é dividido pelos dois túneis criados. O tráfego gerado pelo sistema terminal AL-1 é encaminhado pelo túnel 1, e o tráfego do sistema terminal AL-3 pelo túnel 0.

```
R27#sh mpls forwarding-table lsp-tunnel
Local  Outgoing    Prefix            Bytes tag  Outgoing    Next Hop
tag    tag or VC   or Tunnel Id      switched   interface
16     16          10.10.10.2 0 [16] 1863882    Fa0/1       10.1.0.145
28     18          10.10.10.2 1 [16] 2265996    Fa1/0       10.1.0.150
```

Figura 13 – Tabela de encaminhamento de etiquetas do *router* R27 após envio de pacotes

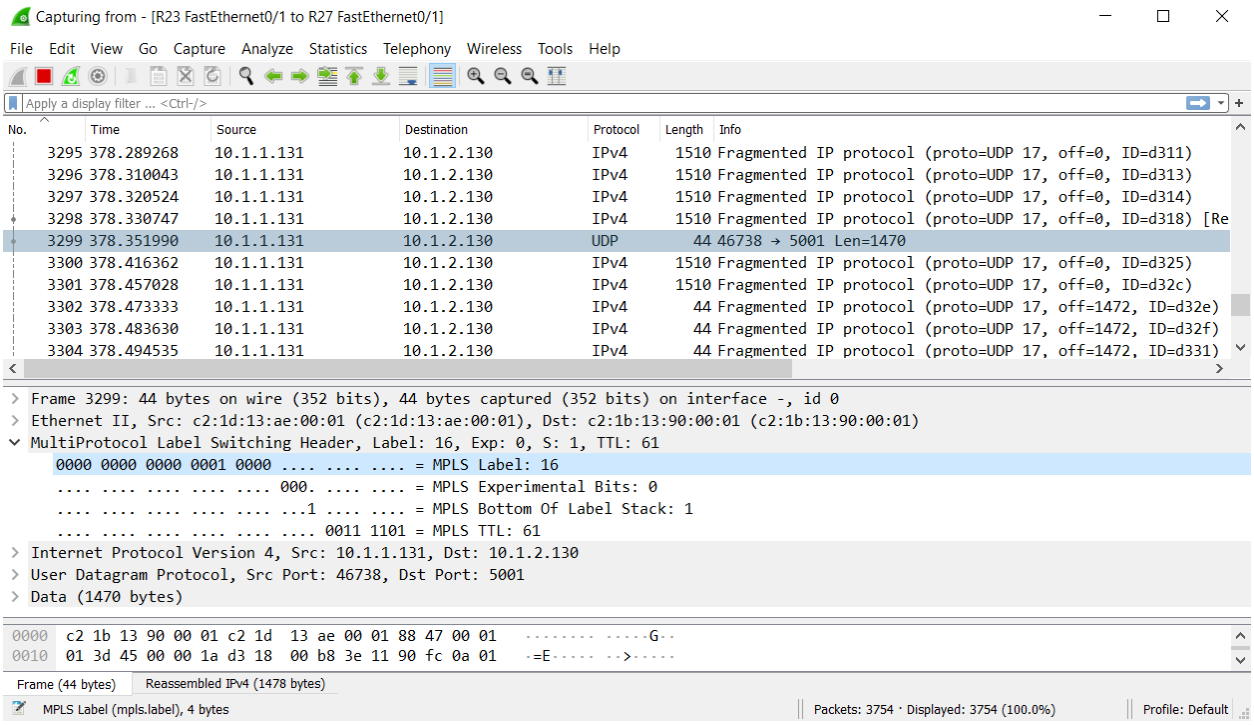

Figura 14 – Captura *Wireshark* no *router* R27 no caminho do Túnel 0

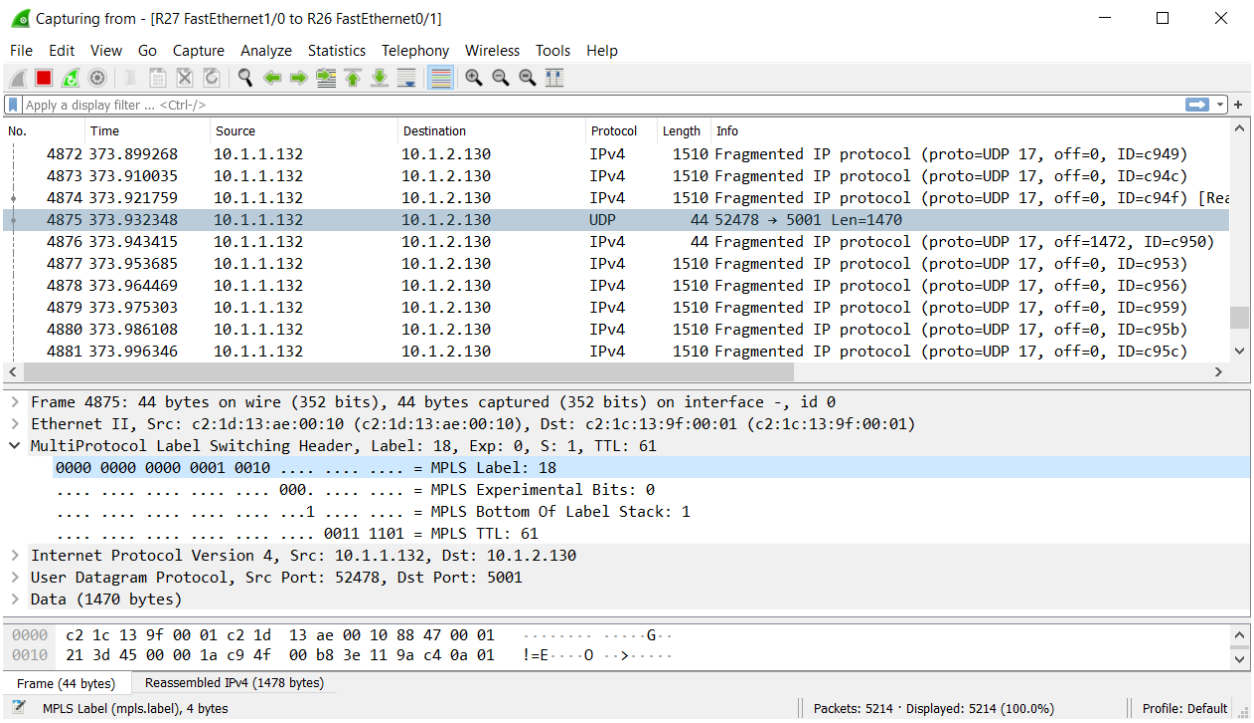

Figura 15 – Captura Wireshark no router R27 no caminho do Túnel 1

3 Engenharia de tráfego por classes de serviço (MPLS DiffServ-TE)

Neste fase, o objetivo era acrescentar um domínio *DiffServ* ao domínio MPLS já criado na fase anterior. No domínio *DiffServ* foram criadas três classes de serviço: EF (*Expedited Forwarding*), AF (*Assured Forwarding*) e BE (*Best Effort*). Para a classe EF, foi definido todo o tráfego de voz que use as portas UDP entre 16384 e 32767. O tráfego HTTP que use as portas TCP 80 ou 8080 foi definido na classe AF. O restante tráfego foi classificado como BE. Para mapear o tráfego a ser utilizado por cada classe, foram utilizadas listas de acesso para definir qual o tipo de tráfego e as respetivas portas, como podemos ver na Figura 16. Estas listas de acesso foram associadas a cada classe através dos seguintes comandos, Figura 17:

- **class-map match-all** [*class_name*]
  - **match access-group** [*access_list_number*]

```
access-list 101 permit udp any any range 16384 32767
access-list 102 permit tcp any any eq www
access-list 102 permit tcp any any eq 8080
access-list 104 permit ip any any
```

Figura 16 – Listas de acesso para cada tipo de tráfego

```
class-map match-all EF
 match access-group 101
class-map match-all AF
 match access-group 102
class-map match-all BE
 match access-group 104
```

Figura 17 – Associação das listas de acesso ao tipo de classe correspondente.

Na fase anterior, foi definido em cada interface os valores de largura de banda reserváveis na *global-pool* através do comando **ip rsvp bandwidth** [*total_reservável*] [*máximo_fluxo*]. No entanto, nesta fase, foi necessário alterar os valores de largura de banda reserváveis, tanto na *global-pool* como na *sub-pool*, através do uso do mesmo comando. Para a reserva da largura de banda na *global-pool* foram estabelecidos dois valores: 5000 *kbps* e 2000 *kbps*. O primeiro foi utilizado nas interfaces com os *routers* LER, pois eram as potenciais interfaces em que mais do que um túnel MPLS podia ser estabelecido. Já o segundo valor foi definido para as restantes interfaces. Para a *sub-pool* foi estabelecido um único valor, 256 *kbps*.

De seguida, foram configuradas as filas *DiffServ*, apenas no *router* LER entre a área 0 e 1. Na interface de entrada, interface f1/0, foram definidas a marcação e as políticas de descarte de pacotes e na interface de saída, interface f0/0, foi aplicado o escalonamento do tráfego.

Os pacotes foram marcados através do campo DSCP (*Differentiated Services Code Point*), após serem policiados, com os valores presentes na Tabela 1 e de acordo com a classe de tráfego a que pertencem. Para este efeito, foram criadas três classes de tráfego: *gold*, *silver* e *default*. A classe *gold* representa o tráfego de voz que é sensível a perdas e atrasos. A classe *silver* representa o tráfego HTTP. O restante tráfego foi representado pela classe *default*.

No policiamento de tráfego, foram definidas algumas restrições na largura de banda utilizada por cada um. Para o tráfego de Voz, foi imposto um limite máximo de largura de banda de 256 *kbps* e, caso esteja dentro dos limites é marcado com o valor DSCP 46. Caso contrário, é descartado. Para o tráfego HTTP, foi imposto um limite máximo de largura de banda de 2 *Mbps*. Se o limite não for ultrapassado, é-lhe atribuído o valor DSCP 26. Se a largura de banda exceder o limite máximo até 62.5 *kbytes,* o tráfego é marcado com valor DSCP 0. Caso viole o intervalo anterior, o tráfego é descartado. Para o restante tráfego apenas foi definido o valor DSCP 0. Para o policiamento de tráfego foram utilizados os seguintes comandos:

- **policy-map** [*policy_name*]
  - **class** [*class_name*]
    - **police** [*bandwidth*] **conform-action set-dscp-transmit** [*DSCP_value*] **exceed-action set-dscp-transmit** [*DSCP_value*] **violate-action drop**

Na Figura 18 podemos ver toda a configuração realizada para estabelecer as políticas de descarte das diferentes classes de serviço.

```
policy-map SETDSCP_POLICE
 class EF
   police 256000 conform-action set-dscp-transmit 46 exceed-action drop
 class AF
   police 2000000 conform-action set-dscp-transmit 26 exceed-action set-dscp-transmit 0 violate-action drop
 class BE
  set dscp default
```

Figura 18 – Configuração das políticas de descarte e marcação do tráfego.

Na Figura 19 podemos ver, detalhadamente, o policiamento do tráfego aplicado na entrada da rede MPLS. Como mostra a figura e, visto que, ainda não se gerou tráfego de voz ou HTTP, as classes EF e AF não tem nenhum pacote marcado com os respetivos valores. Na classe BE, apesar de não se gerar tráfego para esta classe, esta já tem pacotes marcados. Isto acontece porque protocolos como o OSPF estão constantemente a trocar mensagens para a sua manutenção. Apesar de estarem a ser marcados, este tráfego não utiliza os túneis MPLS, pois a sua marcação não tem qualquer efeito na mudança do valor DSCP já pré-definido, possivelmente por se tratar de tráfego fundamental para o correto funcionamento do respetivo protocolo.

```
R6#sh policy-map interface f1/0
 FastEthernet1/0

  Service-policy input: SETDSCP_POLICE

    Class-map: EF (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 101
      police:
          cir 256000 bps, bc 8000 bytes
        conformed 0 packets, 0 bytes; actions:
          set-dscp-transmit ef
        exceeded 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps

    Class-map: AF (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 102
      police:
          cir 2000000 bps, bc 62500 bytes, be 1500 bytes
        conformed 0 packets, 0 bytes; actions:
          set-dscp-transmit af31
        exceeded 0 packets, 0 bytes; actions:
          set-dscp-transmit default
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps, violate 0 bps

    Class-map: BE (match-all)
      1245 packets, 144364 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 104
      QoS Set
        dscp default
          Packets marked 1245
```

Figura 19 – Mapa de políticas na entrada da rede MPLS em detalhe

Na interface de saída, foi aplicada uma política para a atribuição de prioridades de fila, da largura de banda assegurada e, quando necessário, o descarte preventivo WRED (*Weighted Random Early Detection*). Para a classe de tráfego *gold* foi definida uma prioridade de 256 e para a classe de tráfego *silver*, uma largura de banda de 2 *Mbps* e descarte preventivo WRED. Para a classe *default* não foi atribuída nenhuma política, visto que não se trata de tráfego essencial nem prioritário. Nas Figura 20 e Figura 21 podemos ver toda a configuração relativa a este policiamento.

```
class-map match-all gold
 match mpls experimental topmost 5
class-map match-all silver
 match mpls experimental topmost 3
```

Figura 20 – Configuração das classes de tráfego.

```
policy-map DIST_CLASSES
 class gold
  priority 256
 class silver
  bandwidth 2000
  random-detect
!
```

Figura 21 – Configuração das políticas de escalonamento.

Como estamos perante um domínio MPLS, foi necessário comparar o campo EXP da camada MPLS para determinar a que classe pertencia o tráfego. Estes valores são estabelecidos de acordo com os três bits mais significativos do campo DSCP, atribuído no policiamento do tráfego como já foi dito anteriormente. Um exemplo desta atribuição está representado na Figura 22

	6th bit	5th bit	4th bit	3rd bit	2nd bit	1st bit	
DSCP	**1**	**0**	**1**	**1**	**1**	**0**	**46**
EXP	**1**	**0**	**1**				**5**

Figura 22 – Exemplo da atribuição do valor EXP através do campo DSCP.

Classe de tráfego	Tipo de tráfego	Valor DSCP	Valor EXP
gold	Voz	46 (EF)	5
silver	HTTP	26 (AF31)	3
default	Restante tráfego IP	0 (CS0)	0

Tabela 1 – Tabela com os valores DSCP e EXP para cada classe de tráfego.

Na Figura 23 podemos ver a política de escalonamento em detalhe. À semelhança do que aconteceu na entrada, aqui também vemos que nenhum pacote de qualquer classe de tráfego foi policiado. Isto acontece, porque na entrada não foram marcados quaisquer pacotes nas diferentes classes criadas.

```
R6#sh policy-map interface f0/0
 FastEthernet0/0

  Service-policy output: DIST_CLASSES

    Class-map: gold (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: mpls experimental topmost 5
      Queueing
        Strict Priority
        Output Queue: Conversation 264
        Bandwidth 256 (kbps) Burst 6400 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: silver (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: mpls experimental topmost 3
      Queueing
        Output Queue: Conversation 265
        Bandwidth 2000 (kbps)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
         exponential weight: 9
         mean queue depth: 0

  class    Transmitted      Random drop      Tail drop    Minimum Maximum  Mark
           pkts/bytes       pkts/bytes       pkts/bytes    thresh  thresh  prob
      0       0/0              0/0              0/0           20      40  1/10
      1       0/0              0/0              0/0           22      40  1/10
      2       0/0              0/0              0/0           24      40  1/10
      3       0/0              0/0              0/0           26      40  1/10
      4       0/0              0/0              0/0           28      40  1/10
      5       0/0              0/0              0/0           30      40  1/10
      6       0/0              0/0              0/0           32      40  1/10
      7       0/0              0/0              0/0           34      40  1/10
   rsvp       0/0              0/0              0/0           36      40  1/10


    Class-map: default (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps
      Match: mpls experimental topmost 0
```

Figura 23 – Política de escalonamento em detalhe.

Após a configuração *DiffServ*, com a marcação e o policiamento necessário, foram criados três túneis, um para cada classe de tráfego. Os túneis foram configurados através dos mesmos comandos que na fase anterior. Como cada classe de tráfego tem o seu limite de largura de banda, foi necessário também estabelecer a largura de landa dos túneis de acordo com esse limite. Nos túneis das classes de tráfego *silver* e *default* foi definida uma largura de banda de 2 *Mbps* na *global-pool* e na classe *gold* foi definido uma largura de banda de 256 *kbps* na *sub-pool*. Na Figura 24 está representada a configuração dos túneis criados no *router* LER.

```
interface Tunnel0
 ip unnumbered Loopback0
 tunnel destination 10.10.10.4
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth  2000
 tunnel mpls traffic-eng path-option 10 explicit name PATH_TO_A2
 no routing dynamic
!
interface Tunnel1
 ip unnumbered Loopback0
 tunnel destination 10.10.10.4
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth  2000
 tunnel mpls traffic-eng path-option 10 explicit name PATH2_TO_A2
 no routing dynamic
!
interface Tunnel2
 ip unnumbered Loopback0
 tunnel destination 10.10.10.4
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng priority 7 7
 tunnel mpls traffic-eng bandwidth sub-pool 256
 tunnel mpls traffic-eng path-option 10 explicit name PATH3_TO_A2
 no routing dynamic
!
```

Figura 24 – Configuração dos túneis criados

Apesar de os túneis estarem criados e estabelecidos, a rede não sabia como encaminhar o tráfego para o túnel respetivo e, por isso, as rotas foram configuradas, manualmente, através do encaminhamento baseado em políticas (*Policy-based Routing*). Para esta configuração foi necessário a criação de três listas de acesso, uma para cada classe de tráfego. Cada lista de acesso filtra o tráfego de acordo com o valor DSCP, como podemos ver na Figura 25.

```
access-list 111 permit ip any any dscp ef
access-list 112 permit ip any any dscp af31
access-list 114 permit ip any any dscp default
```

Figura 25 – Listas de acesso definidas para o encaminhamento de tráfego.

De seguida, foi criado um mapa de rotas para a aplicação das listas de acesso criadas. Se o valor DSCP do tráfego corresponder ao valor 46 (EF), este é encaminhado para o túnel 2, túnel da classe *gold*. Se o valor DSCP for 26 (AF31), então o tráfego é encaminhado par ao túnel 1, túnel da classe *silver.* Por último, se o valor DSCP corresponder ao valor 0 (*default*), o tráfego é encaminhado para o túnel 0, tunél da classe *default*. Podemos ver este encaminhamento na Figura 26. Este mapa foi definido à entrada da rede MPLS, no *router* LER.

Na Figura 27 podemos ver, em detalhe, o mapa das rotas. Reparamos que, a cada correspondência fica associada a etiqueta respetiva de cada túnel. A etiqueta 22 foi associada ao túnel da classe *gold*, a etiqueta 21 ao túnel da classe *silver* e a etiqueta 23 ao túnel da classe *default*.

```
route-map tunnels permit 10
 match ip address 111
 set interface Tunnel2
!
route-map tunnels permit 20
 match ip address 112
 set interface Tunnel1
!
route-map tunnels permit 30
 match ip address 114
 set interface Tunnel0
!
```

Figura 26 – Configuração do mapa de rotas para o encaminhamento.

```
R6#sh route-map
route-map tunnels, permit, sequence 10
  Match clauses:
    ip address (access-lists): 111
  Set clauses:
    interface Tunnel2
      Tu2 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{22}, via Fa0/0
        C21D13AE0000C206276A00008847 00016000
  Policy routing matches: 0 packets, 0 bytes
route-map tunnels, permit, sequence 20
  Match clauses:
    ip address (access-lists): 112
  Set clauses:
    interface Tunnel1
      Tu1 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{21}, via Fa0/0
        C21D13AE0000C206276A00008847 00015000
  Policy routing matches: 0 packets, 0 bytes
route-map tunnels, permit, sequence 30
  Match clauses:
    ip address (access-lists): 114
  Set clauses:
    interface Tunnel0
      Tu0 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{23}, via Fa0/0
        C21D13AE0000C206276A00008847 00017000
  Policy routing matches: 0 packets, 0 bytes
```

Figura 27 – Mapa das rotas criadas detalhado.

Para testar o correto funcionamento das classes de tráfego, foram escolhidos dois sistemas terminais, o AL-1 como gerador de tráfego e o AL-6 como recetor. Foi utilizada a ferramenta *iperf* para gerar o tráfego.

Para um primeiro teste, foi gerado tráfego UDP para a porta destino 20000, porta que está no intervalo definido para a respetiva classe e com uma largura de banda de 250 *kbps*. Notar que esta largura de banda não é constante em todo o tráfego gerado, podendo haver oscilações.

Na entrada da rede MPLS, Figura 28, o tráfego gerado foi classificado na classe *gold*, uma vez quer pertencia à classe EF definida. Reparamos que, dois pacotes foram descartados, pois excederam o limite imposto nas políticas. Esta perda confirma-se através dos *logs* da ferramenta *iperf*, Figura 29.

```
R6#sh policy-map interface f1/0
 FastEthernet1/0

  Service-policy input: SETDSCP_POLICE

    Class-map: EF (match-all)
      439 packets, 341564 bytes
      5 minute offered rate 10000 bps, drop rate 0 bps
      Match: access-group 101
      police:
          cir 256000 bps, bc 8000 bytes
        conformed 437 packets, 338552 bytes; actions:
          set-dscp-transmit ef
        exceeded 2 packets, 3012 bytes; actions:
          drop
        conformed 28000 bps, exceed 0 bps

    Class-map: AF (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 102
      police:
          cir 2000000 bps, bc 62500 bytes, be 1500 bytes
        conformed 0 packets, 0 bytes; actions:
          set-dscp-transmit af31
        exceeded 0 packets, 0 bytes; actions:
          set-dscp-transmit default
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps, violate 0 bps

    Class-map: BE (match-all)
      2387 packets, 277236 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 104
      QoS Set
        dscp default
          Packets marked 2387
```

Figura 28 – Policiamento à entrada da rede MPLS

```
/ # iperf -c 10.1.2.130 -u -p 20000 -b 250k -i 1
------------------------------------------------------------
Client connecting to 10.1.2.130, UDP port 20000
Sending 1470 byte datagrams, IPG target: 47040.00 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 10.1.1.130 port 54860 connected with 10.1.2.130 port 20000
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  33.0 KBytes   270 Kbits/sec
[  3]  1.0- 2.0 sec  30.1 KBytes   247 Kbits/sec
[  3]  2.0- 3.0 sec  30.1 KBytes   247 Kbits/sec
[  3]  3.0- 4.0 sec  31.6 KBytes   259 Kbits/sec
[  3]  4.0- 5.0 sec  30.1 KBytes   247 Kbits/sec
[  3]  5.0- 6.0 sec  30.1 KBytes   247 Kbits/sec
[  3]  6.0- 7.0 sec  30.1 KBytes   247 Kbits/sec
[  3]  7.0- 8.0 sec  31.6 KBytes   259 Kbits/sec
[  3]  8.0- 9.0 sec  30.1 KBytes   247 Kbits/sec
[  3]  0.0-10.0 sec   306 KBytes   250 Kbits/sec
[  3] Sent 213 datagrams
[  3] Server Report:
[  3]  0.0-10.0 sec   303 KBytes   248 Kbits/sec   8.649 ms    2/  213 (0.94%)
```

Figura 29 – *Logs* da ferramenta *iperf* para o tráfego UDP gerado.

Este tráfego, como foi marcado na classe *gold,* foi encaminhado para o túnel respetivo, o túnel 2, e foi aplicada a etiqueta com o número 22, o valor DSCP 46 e o valor 5 no campo EXP. Como podemos ver na Figura 30 e na captura *wireshark* nas Figura 32 e Figura 33.

Na saída do *router* LER, o tráfego foi, novamente, policiado e colocado numa fila prioritária, Figura 31.

```
R6#sh route-map
route-map tunnels, permit, sequence 10
  Match clauses:
    ip address (access-lists): 111
  Set clauses:
    interface Tunnel2
      Tu2 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{22}, via Fa0/0
        C21D13AE0000C206276A00008847 00016000
  Policy routing matches: 437 packets, 338552 bytes
route-map tunnels, permit, sequence 20
  Match clauses:
    ip address (access-lists): 112
  Set clauses:
    interface Tunnel1
      Tu1 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{21}, via Fa0/0
        C21D13AE0000C206276A00008847 00015000
  Policy routing matches: 0 packets, 0 bytes
route-map tunnels, permit, sequence 30
  Match clauses:
    ip address (access-lists): 114
  Set clauses:
    interface Tunnel0
      Tu0 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{23}, via Fa0/0
        C21D13AE0000C206276A00008847 00017000
  Policy routing matches: 0 packets, 0 bytes
```

Figura 30 – Mapa das rotas do encaminhamento de tráfego para os túneis.

```
R6#sh policy-map interface f0/0
 FastEthernet0/0

  Service-policy output: DIST_CLASSES

    Class-map: gold (match-all)
      434 packets, 335752 bytes
      5 minute offered rate 8000 bps, drop rate 0 bps
      Match: mpls experimental topmost 5
      Queueing
        Strict Priority
        Output Queue: Conversation 264
        Bandwidth 256 (kbps) Burst 6400 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: silver (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: mpls experimental topmost 3
      Queueing
        Output Queue: Conversation 265
        Bandwidth 2000 (kbps)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
         exponential weight: 9
         mean queue depth: 0

  class    Transmitted      Random drop      Tail drop    Minimum Maximum  Mark
           pkts/bytes       pkts/bytes       pkts/bytes    thresh  thresh  prob
      0       0/0              0/0              0/0           20      40  1/10
      1       0/0              0/0              0/0           22      40  1/10
      2       0/0              0/0              0/0           24      40  1/10
      3       0/0              0/0              0/0           26      40  1/10
      4       0/0              0/0              0/0           28      40  1/10
      5       0/0              0/0              0/0           30      40  1/10
      6       0/0              0/0              0/0           32      40  1/10
      7       0/0              0/0              0/0           34      40  1/10
   rsvp       0/0              0/0              0/0           36      40  1/10


    Class-map: default (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps
      Match: mpls experimental topmost 0
```

Figura 31 – Policiamento na interface de saída do *router* LER.

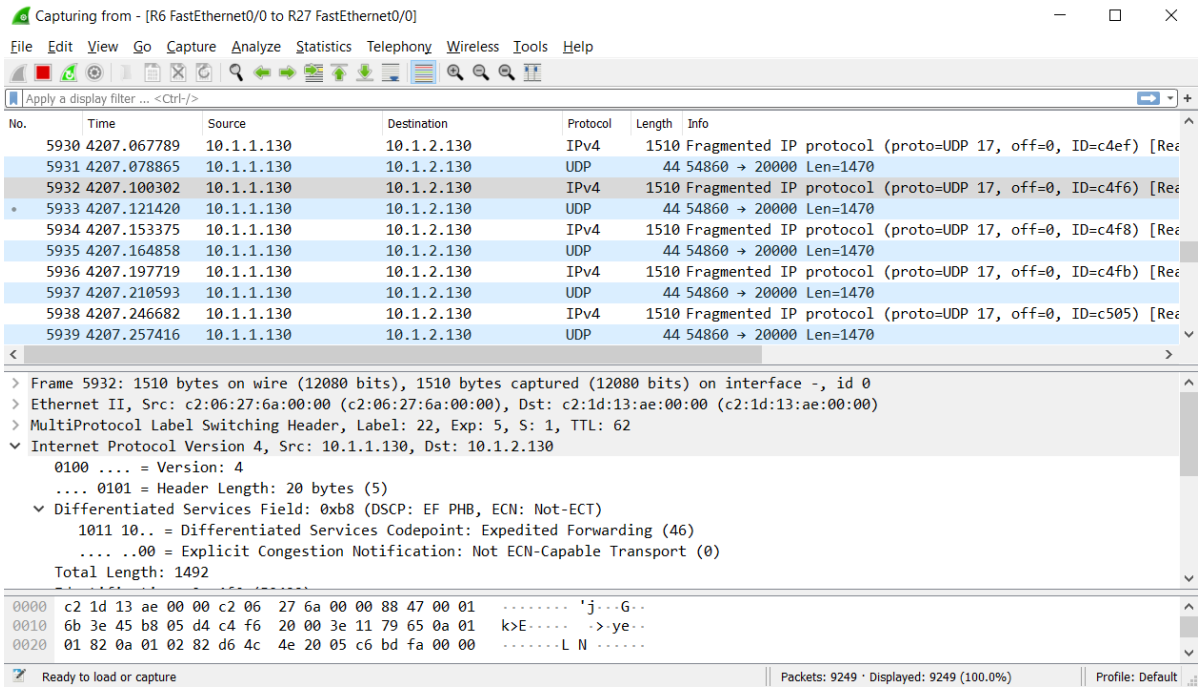

Figura 32 – Captura *wireshark* de um pacote UDP fragmentado (pacote 1/2)

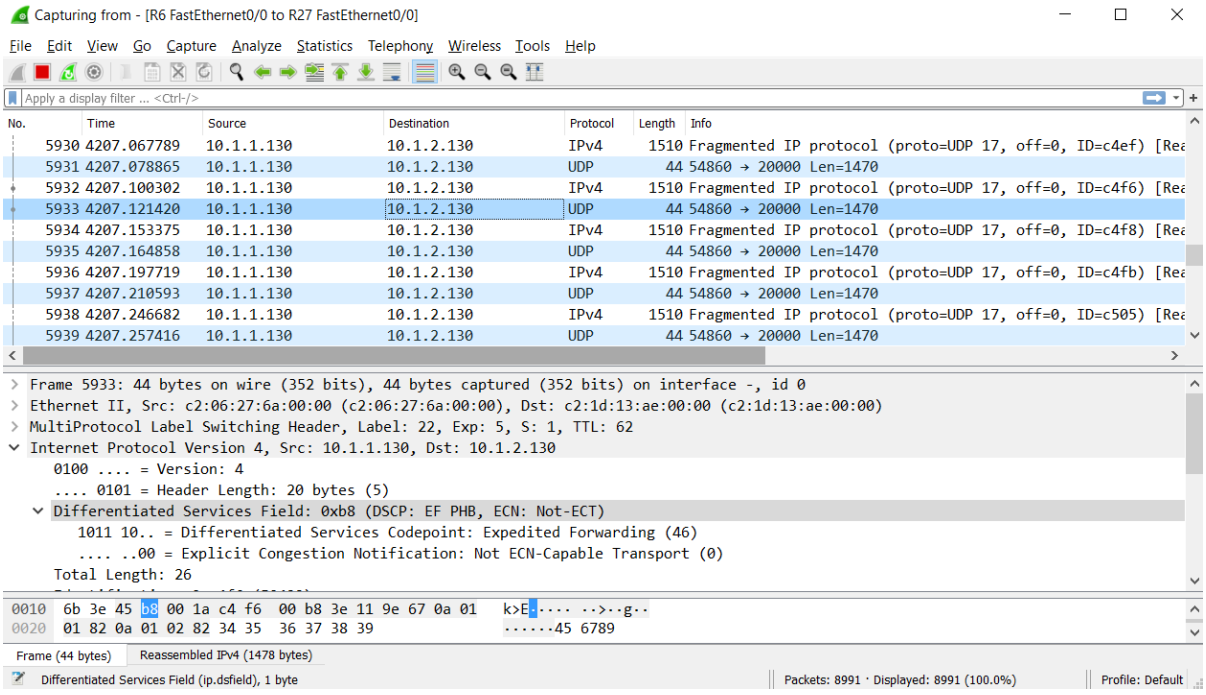

Figura 33 – Captura *wireshark* de um pacote UDP (pacote 2/2)

Num segundo teste foi gerado tráfego HTTP na porta TCP 8080. Neste teste não foi possível indicar qual a largura de banda, pois o protocolo TCP a ajusta automaticamente.

À entrada da rede MPLS, todo o tráfego gerado na porta TCP 8080 foi classificado na classe *silver*, sendo marcado com o valor DSCP 26 (AF31), Figura 34. Podemos também reparar que não houve qualquer remarcação/perda de pacotes, devido ao ajuste automático da largura de banda no protocolo TCP.

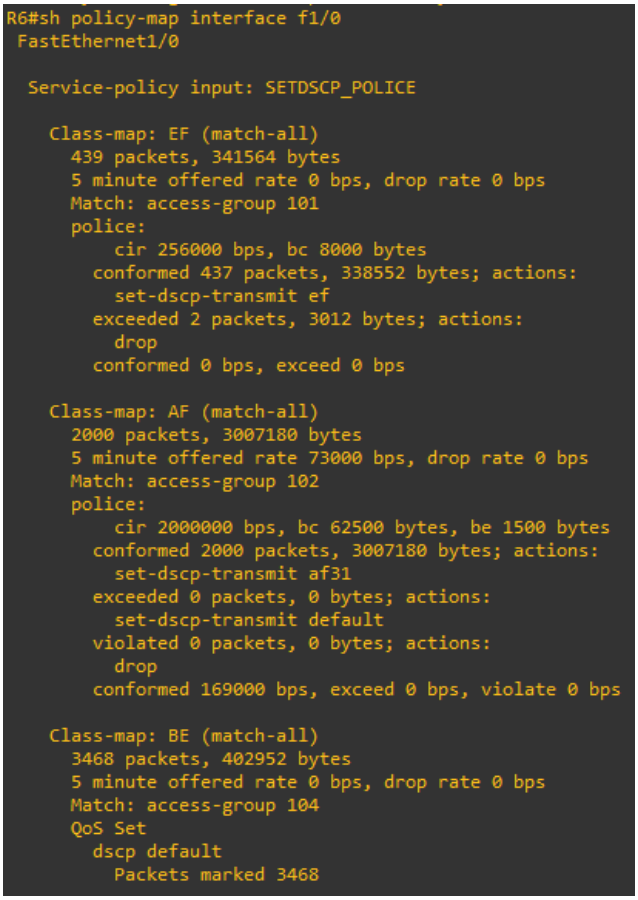

Figura 34 – Policiamento à entrada da rede MPLS.

Como o tráfego foi marcado na classe *silver*, este foi encaminhado para o túnel 1, túnel responsável pela classe de tráfego *silver*. A este tráfego foi lhe atribuído a etiqueta 21 e o valor EXP correspondente ao valor DSCP. Como podemos ver na Figura 35 e na captura *wireshark*, Figura 37.

No policiamento à saída do *router* LER, como se trata de tráfego HTTP foi atribuída uma largura de banda de 2 *Mbps* com descarte preventivo, Figura 36, como foi estipulado nas configurações do *DiffServ*.

```
R6#sh route-map
route-map tunnels, permit, sequence 10
  Match clauses:
    ip address (access-lists): 111
  Set clauses:
    interface Tunnel2
      Tu2 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{22}, via Fa0/0
        C21D13AE0000C206276A00008847 00016000
  Policy routing matches: 437 packets, 338552 bytes
route-map tunnels, permit, sequence 20
  Match clauses:
    ip address (access-lists): 112
  Set clauses:
    interface Tunnel1
      Tu1 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{21}, via Fa0/0
        C21D13AE0000C206276A00008847 00015000
  Policy routing matches: 2000 packets, 3007180 bytes
route-map tunnels, permit, sequence 30
  Match clauses:
    ip address (access-lists): 114
  Set clauses:
    interface Tunnel0
      Tu0 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{23}, via Fa0/0
        C21D13AE0000C206276A00008847 00017000
  Policy routing matches: 0 packets, 0 bytes
```

Figura 35 – Mapa das rotas do encaminhamento de tráfego para os túneis.

```
R6#sh policy-map interface f0/0
 FastEthernet0/0

  Service-policy output: DIST_CLASSES

    Class-map: gold (match-all)
      434 packets, 335752 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: mpls experimental topmost 5
      Queueing
        Strict Priority
        Output Queue: Conversation 264
        Bandwidth 256 (kbps) Burst 6400 (Bytes)
        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: silver (match-all)
      1990 packets, 3000000 bytes
      5 minute offered rate 68000 bps, drop rate 0 bps
      Match: mpls experimental topmost 3
      Queueing
        Output Queue: Conversation 265
        Bandwidth 2000 (kbps)
        (pkts matched/bytes matched) 1741/2635874
        (depth/total drops/no-buffer drops) 0/0/0
         exponential weight: 9
         mean queue depth: 0

  class    Transmitted      Random drop      Tail drop    Minimum Maximum  Mark
           pkts/bytes       pkts/bytes       pkts/bytes    thresh  thresh  prob
      0       0/0              0/0              0/0           20      40  1/10
      1       0/0              0/0              0/0           22      40  1/10
      2       0/0              0/0              0/0           24      40  1/10
      3       0/0              0/0              0/0           26      40  1/10
      4       0/0              0/0              0/0           28      40  1/10
      5       0/0              0/0              0/0           30      40  1/10
      6       0/0              0/0              0/0           32      40  1/10
      7       0/0              0/0              0/0           34      40  1/10
   rsvp       0/0              0/0              0/0           36      40  1/10


    Class-map: default (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps
      Match: mpls experimental topmost 0
```

Figura 36 – Policiamento na interface de saída do router LER.

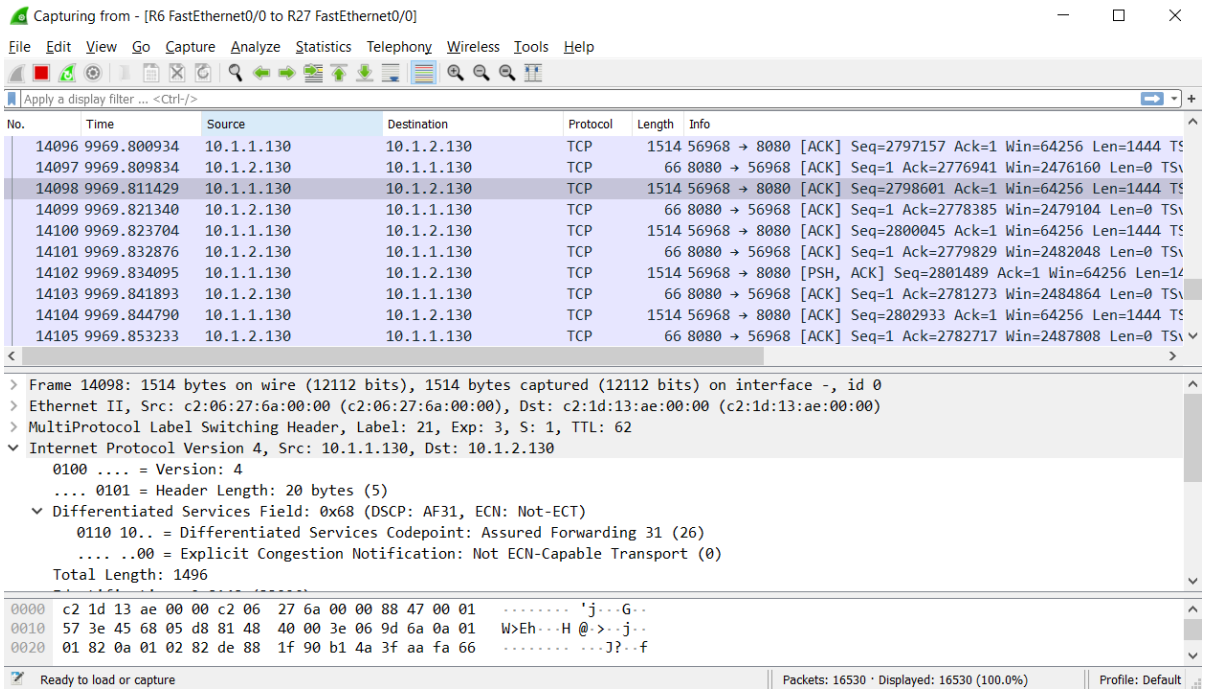

Figura 37 – Captura *wireshark* de um pacote TCP.

Num último teste, foi gerado tráfego na porta UDP 5000, com uma largura de banda de 256 *Kbps*. Como era de esperar, todo este tráfego foi classificado na classe *default*, visto que não se trata de tráfego de Voz nem HTTP, Figura 38. Porém, por razões desconhecidas, apenas os pacotes fragmentados do protocolo UDP foram classificados na classe *default*. Os pacotes UDP foram classificados na classe *gold* apesar do tráfego se destinar a uma porta que não se encontra dentro do intervalo definido, entre 16384 e 32767. Esta informação encontra-se nas capturas *wireshark* Figura 41 e Figura 42.

Foi também gerado tráfego na mesma porta UDP, mas com largura de banda de 1 *Mbps*. Porém foram perdidos a maior parte dos pacotes, devido ao problema dito anteriormente. Como podemos ver através dos *logs* da ferramenta *iperf*, Figura 43 e Figura 44.

```
R6#sh policy-map interface f1/0
 FastEthernet1/0

  Service-policy input: SETDSCP_POLICE

    Class-map: EF (match-all)
      1758 packets, 394324 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 101
      police:
          cir 256000 bps, bc 8000 bytes
        conformed 1756 packets, 391312 bytes; actions:
          set-dscp-transmit ef
        exceeded 2 packets, 3012 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps

    Class-map: AF (match-all)
      2000 packets, 3007180 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 102
      police:
          cir 2000000 bps, bc 62500 bytes, be 1500 bytes
        conformed 2000 packets, 3007180 bytes; actions:
          set-dscp-transmit af31
        exceeded 0 packets, 0 bytes; actions:
          set-dscp-transmit default
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 0 bps, exceed 0 bps, violate 0 bps

    Class-map: BE (match-all)
      7062 packets, 4210692 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: access-group 104
      QoS Set
        dscp default
          Packets marked 7062
```

Figura 38 – Policiamento à entrada da rede MPLS.

O tráfego que foi classificado na classe *default* foi encaminhado para o túnel 0, túnel definido para todo o tráfego da classe *default*. Como podemos ver na Figura 39. A este tráfego foi atribuído a etiqueta 23 e no campo EXP foi atribuído o valor 0, uma vez que pertence à classe de serviço BE.

Como se trata de tráfego da classe *default*, não foram estabelecidas quaisquer prioridades nem largura de banda, Figura 40.

```
R6#sh route-map
route-map tunnels, permit, sequence 10
  Match clauses:
    ip address (access-lists): 111
  Set clauses:
    interface Tunnel2
      Tu2 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{22}, via Fa0/0
        C21D13AE0000C206276A00008847 00016000
  Policy routing matches: 1756 packets, 391312 bytes
route-map tunnels, permit, sequence 20
  Match clauses:
    ip address (access-lists): 112
  Set clauses:
    interface Tunnel1
      Tu1 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{21}, via Fa0/0
        C21D13AE0000C206276A00008847 00015000
  Policy routing matches: 2000 packets, 3007180 bytes
route-map tunnels, permit, sequence 30
  Match clauses:
    ip address (access-lists): 114
  Set clauses:
    interface Tunnel0
      Tu0 forwarding info:
        MAC/Encaps=14/18, MTU=1500, Label Stack{23}, via Fa0/0
        C21D13AE0000C206276A00008847 00017000
  Policy routing matches: 2451 packets, 3674730 bytes
```

Figura 39 – Mapa das rotas do encaminhamento de tráfego para os túneis.

```
R6#sh policy-map interface f0/0
 FastEthernet0/0

  Service-policy output: DIST_CLASSES

    Class-map: gold (match-all)
      1753 packets, 393788 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: mpls experimental topmost 5
      Queueing
        Strict Priority
        Output Queue: Conversation 264
        Bandwidth 256 (kbps) Burst 6400 (Bytes)
        (pkts matched/bytes matched) 788/34672
        (total drops/bytes drops) 0/0

    Class-map: silver (match-all)
      1990 packets, 3000000 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: mpls experimental topmost 3
      Queueing
        Output Queue: Conversation 265
        Bandwidth 2000 (kbps)
        (pkts matched/bytes matched) 1741/2635874
        (depth/total drops/no-buffer drops) 0/0/0
         exponential weight: 9
         mean queue depth: 0

  class    Transmitted      Random drop      Tail drop    Minimum Maximum  Mark
           pkts/bytes       pkts/bytes       pkts/bytes    thresh  thresh  prob
      0       0/0              0/0              0/0           20      40  1/10
      1       0/0              0/0              0/0           22      40  1/10
      2       0/0              0/0              0/0           24      40  1/10
      3       0/0              0/0              0/0           26      40  1/10
      4       0/0              0/0              0/0           28      40  1/10
      5       0/0              0/0              0/0           30      40  1/10
      6       0/0              0/0              0/0           32      40  1/10
      7       0/0              0/0              0/0           34      40  1/10
   rsvp       0/0              0/0              0/0           36      40  1/10


    Class-map: default (match-all)
      2449 packets, 3681502 bytes
      5 minute offered rate 0 bps
      Match: mpls experimental topmost 0
```

Figura 40 – Policiamento na interface de saída do router LER.

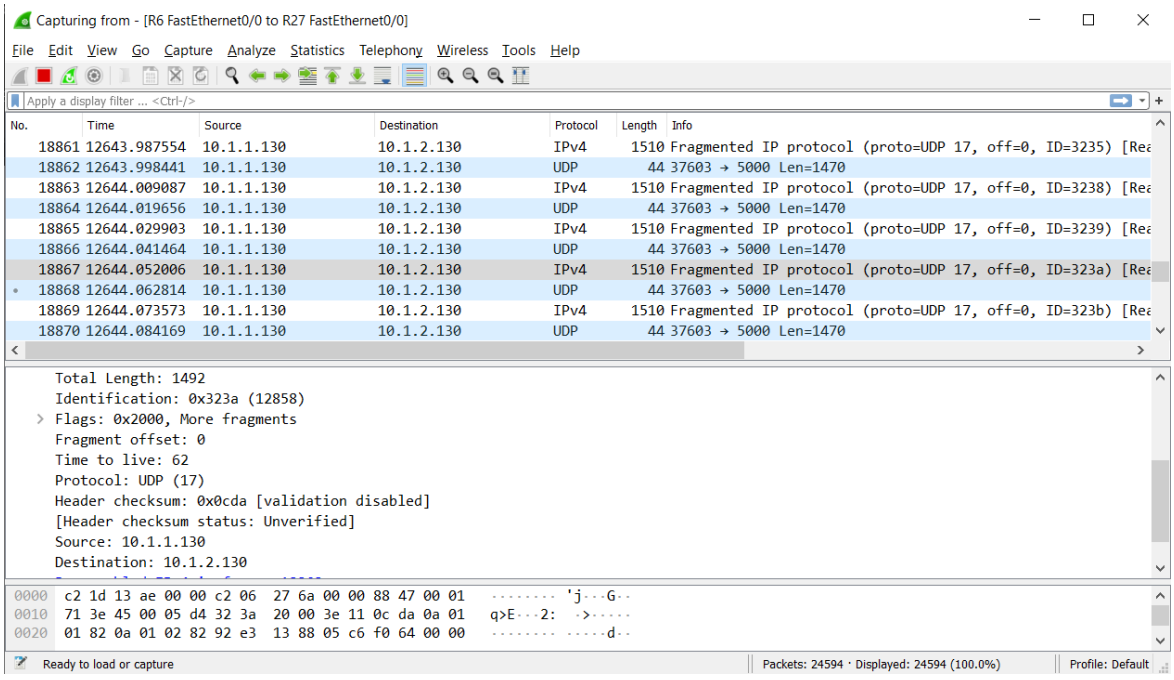

Figura 41 – Captura wireshark de um pacote IP fragmentado (pacote 1/2)

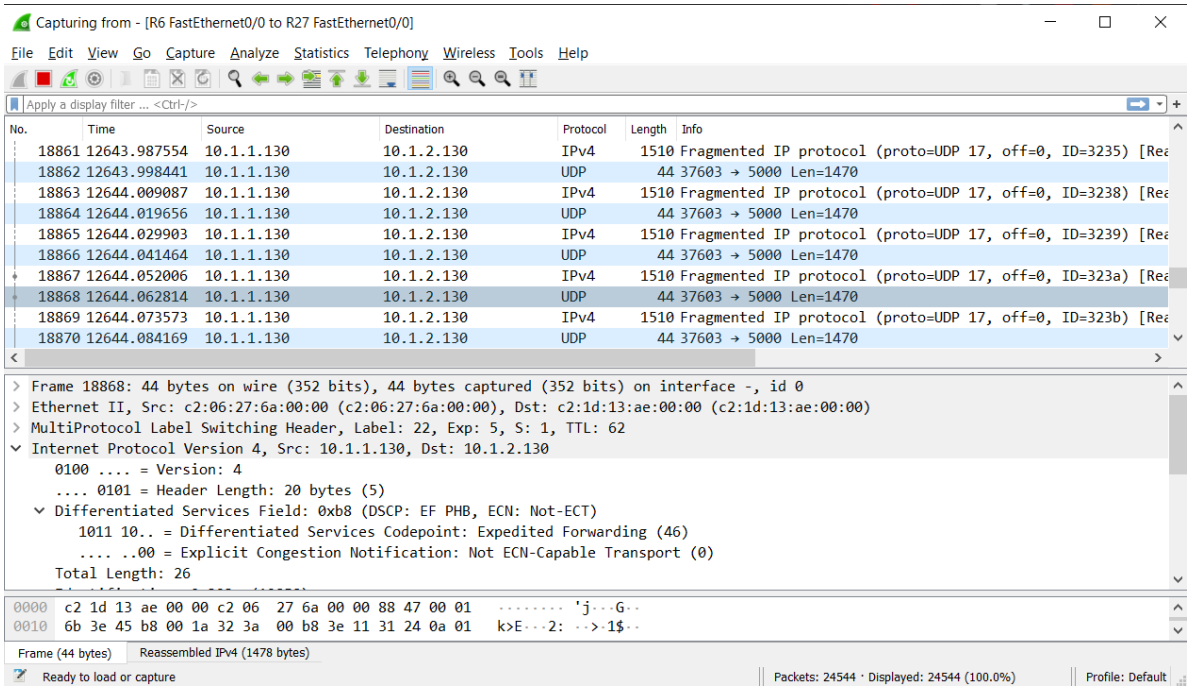

Figura 42 – Captura wireshark de um pacote UDP (pacote 2/2)

```
/ # iperf -c 10.1.2.130 -u -p 5000 -b 1m -i 1
------------------------------------------------------------
Client connecting to 10.1.2.130, UDP port 5000
Sending 1470 byte datagrams, IPG target: 11760.00 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 10.1.1.130 port 51745 connected with 10.1.2.130 port 5000
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   125 KBytes  1.02 Mbits/sec
[  3]  1.0- 2.0 sec   122 KBytes  1000 Kbits/sec
[  3]  2.0- 3.0 sec   122 KBytes  1000 Kbits/sec
[  3]  3.0- 4.0 sec   122 KBytes  1000 Kbits/sec
[  3]  4.0- 5.0 sec   122 KBytes  1000 Kbits/sec
[  3]  5.0- 6.0 sec   122 KBytes  1000 Kbits/sec
[  3]  6.0- 7.0 sec   122 KBytes  1000 Kbits/sec
[  3]  7.0- 8.0 sec   122 KBytes  1000 Kbits/sec
[  3]  8.0- 9.0 sec   122 KBytes  1000 Kbits/sec
[  3]  0.0-10.0 sec  1.19 MBytes  1000 Kbits/sec
[  3] Sent 851 datagrams
[  3] Server Report:
[  3]  0.0-12.2 sec   449 KBytes   303 Kbits/sec  17.039 ms  539/  852 (63%)
```

Figura 43 – *Logs* da ferramente *iperf* no sistema terminal gerador.

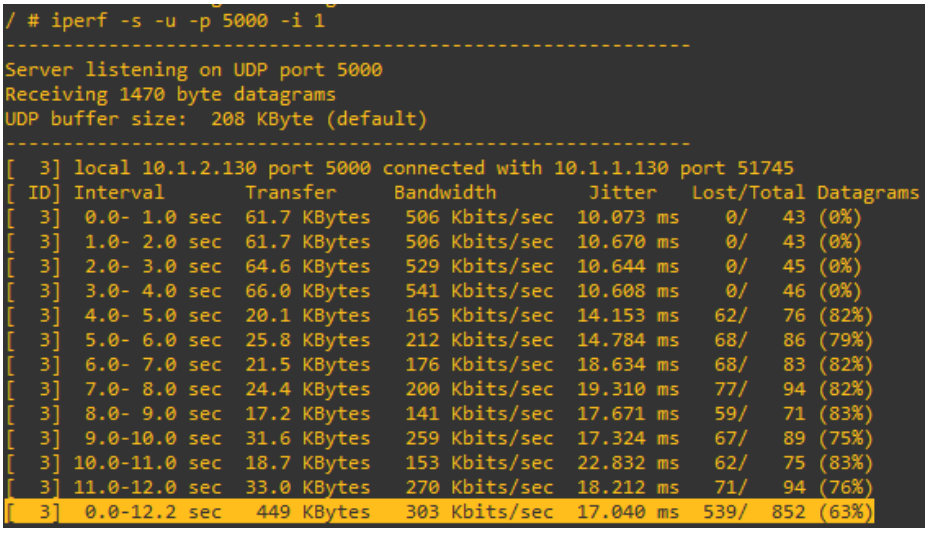

Figura 44 – *Logs* da ferramenta *iperf* no sistema recetor.

Após a realização dos testes e análise dos resultados obtidos, observámos que existe um tratamento diferenciado nas duas classes de serviço implementadas, EF e AF. Esta diferenciação deve-se maioritariamente ao *DiffServ*, pois foi o responsável pela análise do tráfego, marcando-o na respetiva classe de serviço. Com o *DiffServ* também se garantiu uma rede fluida, sem atrasos nem perdas. Porém, por vezes, foi necessário descartar de pacotes, para manter a estabilidade da rede intacta.

O MPLS também teve um papel importante, pois foi ele que distribuiu o tráfego para os respetivos túneis estabelecidos, de acordo com a classe de tráfego.

4 Conclusão

Neste trabalho prático houve alguns contratempos, nomeadamente no uso de alguns comandos da Cisco, uma vez que estamos a trabalhar com uma versão mais antiga. Com o uso de alternativas, este obstáculo foi ultrapassado. Achamos que concluímos este trabalho prático com sucesso, mas ficaram algumas dúvidas sobre o porquê de certas coisas acontecerem, particularmente no último teste realizado.