

Sorbonne Université, Master 2^e informatique, spécialité réseaux

TME 6 : DiffServ Core Configurations

(cf. Cours 09-10)

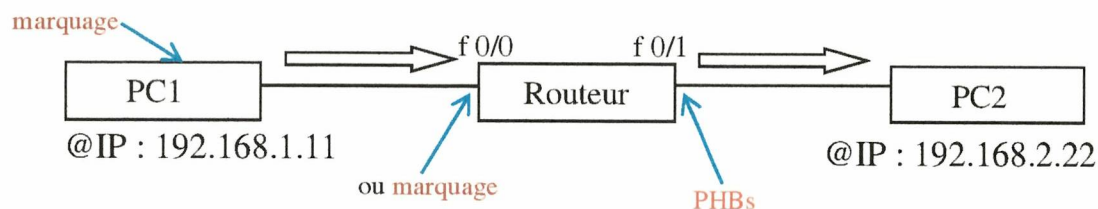
 Compte rendu électronique (pdf)

Introduction

Dans ce TP, vous testerez des éléments DiffServ qui sont destinés à être déployés dans les routeurs de cœur. Il s'agit de tester la mise en place des PHBs (Per-Hop Behavior) correspondant aux différents DSCPs qui ont été définis dans le réseau.

Pour ce faire, vous allez générer directement un trafic marqué à partir de la machine émettrice PC1. Il est possible de marquer les paquets dans le système Linux, au niveau utilisateur, avec l'API des sockets. Cette dernière approche est utilisée par les outils de génération de trafic comme `iperf` avec l'option `-S`. La commande `ping` permet aussi de marquer les paquets avec l'option `-Q`. Néanmoins, il faut spécifier le champ TOS et non pas le champ DSCP à ces deux dernières commandes. (DSCP x 4 → TOS)

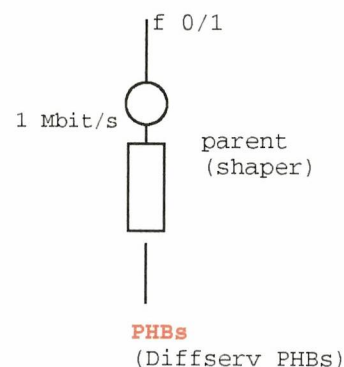
Ainsi, pour tous les prochains tests, vous devez utiliser directement l'option `-S` ou `-Q` afin de marquer les paquets générés par `iperf` ou `ping`. L'unique routeur du TP, et plus précisément l'interface `f 0/1`, jouera alors le rôle d'un routeur de cœur qui reçoit des paquets déjà marqués. En fonction du marquage, vous allez créer les classes de services et les PHBs. Afin de marquer les paquets d'une application autre que `ping` ou `iperf`, nous pouvons utiliser l'interface d'entrée `f0/0`.



Préparation

Créez un goulot d'étranglement (bottleneck) au niveau de l'interface qui connecte le routeur au PC2, avec les commandes ci-dessous. Cette opération n'est pas nécessaire dans une configuration ordinaire *sauf* en cas de contrôle de débit ou si la politique ne peut pas être rattachée directement à l'interface. Dans notre cas, cela nous permet d'émuler un lien à 1Mbit/s auquel on va rattacher les PHBs.

```
Router(config)# policy-map PHBs
Router(config-pmap)# exit
Router(config)# policy-map parent
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 1000000
Router(config-pmap-c)# service-policy PHBs
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# interface FastEthernet 0/1
Router(config-if)# service-policy output parent
```



La politique `policy-map PHBs` implantera les différents PHBs.

Classes de service et PHB

Test 1

→ Les configs :

En utilisant la commande `class-map`, créez les deux classes EF et AF1 correspondant respectivement au codes DSCP suivants : ef et af1i (i = 1 ou 2 ou 3).

Ensuite, à l'intérieur de la politique `policy-map` PHBs :

- Donnez à la classe EF une priorité stricte sur le lien avec une limitation de son débit moyen à 300kbit/s. (Ce débit constitue donc la nouvelle garantie de débit pour le trafic EF. La garantie en termes de **délai** est liée à la **bande passante totale** du lien grâce à la priorité.)

- Attribuez 80% de la bande passante (bandwidth) restante à la classe AF1

Vous implanterez ces différents PHBs avec les commandes `priority` et `bandwidth`. (**Voir l'annexe en fin de ce document**). N'oubliez pas aussi de consulter le document de référence : « **Cisco IOS Quality of Service Solutions Command Reference** »

Pour résoudre les problèmes et vérifiez vos configurations utilisez les commandes suivantes :

```
show policy-map
show policy-map interface f 0/1 output
show policy-map interface f 0/1 output | begin EF
show policy-map interface f 0/1 output | begin AF1
```

→ Le test :

Établissez une connexion UDP marquée par le code **EF** qui envoie à **200kbit/s**, et une autre connexion UDP **AF11** qui envoie à **2Mbit/s**. Pour cela, utilisez les options `-u`, `-c`, `-s`, `-b` de `iperf` sur PC1, et `-u` et `-s` sur PC2 (Commandes complètes mentionnées ci-dessous). Puis observez les mesures fournies par `iperf` sur PC2.

```
PC2:iperf -u -s -f k -i 1
```

```
PC1:iperf -u -c pc2 -f k -t 1800 -i 10 -S 184 -b 200k
```

```
PC1:iperf -u -c pc2 -f k -t 1800 -i 10 -S 40 -b 2m
```

	DSCP (déc)	TOS (déc)
ef	46	184
af11	10	40
af12	12	48
af13	14	56
af21	18	72
af22	20	80
af23	22	88
df	0	0

Question 1 : Quel est le taux de perte mesuré par le trafic EF ?

Question 2 : La connexion EF obtient-elle le débit de 200kbit/s généré par la source ?

Question 3 : Quel est le délai d'aller-retour moyen observé par le trafic EF ? Comparez-le avec ceux des trafics AF1 et Best-Effort : Mesurez tous ces délais avec `ping` et l'option `-Q` **SANS** arrêter les trafics EF et AF. (Remarque : La VoIP est souvent associée au PHB EF).

Question 4 : Arrêtez le trafic AF1. Avec la commande `ping` et l'option `-Q`, mesurez à nouveau le délai d'aller-retour moyen observé par le trafic EF. Normalement, vous devriez trouver presque la même moyenne que la question précédente. Que cela signifie-t-il ?

Test 2

Copiez après chaque test vos configurations dans un fichier à sauvegarder dans votre compte

→ Les configs :

Gardez les configurations précédentes. En utilisant la commande `class-map`, rajoutez la classe AF2 qui correspond au codes DSCP suivants : af2i (i = 1 ou 2 ou 3).

Ensuite, toujours avec la politique `policy-map` PHBs :

- Attribuez 20% de la bande passante restante à la classe AF2.

Pour résoudre les problèmes et vérifiez vos configurations, vous utilisez toujours les commandes suivantes :

```
Show policy-map
Show policy-map interface f 0/1 output
Show policy-map interface f 0/1 output | begin ClassName
```

En plus, la commande :

```
clear counters f 0/1
```

vous permet d'initialiser les compteurs afin de mieux suivre les résultats des tests.

→ Le test :

Établissez une connexion UDP marquée par le code **EF** qui envoie à **500kbit/s**, et une autre connexion UDP **AF11** qui envoie à **500kbit/s**, et enfin une connexion UDP **AF22** qui envoie aussi à **500kbit/s**.

Question 1 : Comment la bande passante est-elle allouée dans ce cas (`iperf`) ? Ces valeurs correspondent-elles aux valeurs devant être obtenues par la configuration ? Afin de répondre à cette dernière question, vous devez calculer les valeurs théoriques.

Remarque : Pour obtenir des résultats `iperf` plus précis, on devrait augmenter la durée de mesure du serveur (option `-i`). N'oubliez pas surtout que les entêtes IP et UDP ne sont pas comptabilisés par `iperf`.

Question 2 : Calculez théoriquement le taux de perte de chaque connexion et comparez les valeurs obtenues avec celles affichées par `iperf` en fin de chaque ligne des statistiques (%).

Question 3 : Mesurez à nouveau le délai des paquets EF et comparez-le à celui du test précédent (test1, question 3). Expliquez.

Test 3

→ Les configs :

Gardez toutes les configurations précédentes. Ensuite, toujours à l'intérieur de la politique `policy-map PHBs` :

- Au sein de la classe AF1, créez trois niveaux de précedence (rejet) tels que le trafic correspondant au DSCP af11 est nettement mieux protégé en cas de congestion que celui correspondant à af12, qui à son tour est nettement mieux protégé que af13. Utilisez des probabilités de rejet élevées et déclenchez le rejet de paquets dès que les mémoires tampons se remplissent de quelques paquets. Choisissez donc convenablement vos seuils de rejet ainsi que vos probabilités de rejet.

Vous implanterez ces différents PHBs avec la commande `random-detect`. **(Voir l'annexe en fin de ce document)**. N'oubliez pas aussi de consulter à nouveau si nécessaire le « **Cisco IOS Quality of Service Solutions Command Reference** »

Pour résoudre les problèmes et vérifiez vos nouvelles configurations `random-detect` utilisez les commandes suivantes :

```
Show policy-map
Show policy-map interface NomInterface output
Show policy-map interface NomInterface output | begin NomClass
```

En particulier, les deux dernières commandes vous permettent de voir les paramètres WRED et certains compteurs utiles par DSCP:

dscp	Transmitted pkts/bytes	Random drop pkts/bytes	Tail drop pkts/bytes	Minimum thresh	Maximum thresh	Mark prob
------	---------------------------	---------------------------	-------------------------	-------------------	-------------------	--------------

→ Le test :

En utilisant trois serveurs et trois clients différents, générez trois trafics TCP marqués respectivement par les DSCP AF11, AF12 et AF13 comme suit :

```
PC2:iperf -s -f k -i 1
PC2:iperf -s -f k -i 1 -p 12345
PC2:iperf -s -f k -i 1 -p 2979

PC1:iperf -c pc2 -M 536 -f k -t 1200 -i 10 -S 40 (trafic AF11)
PC1:iperf -c pc2 -p 12345 -M 536 -f k -t 1200 -i 10 -S 48 (AF12)
PC1:iperf -c pc2 -p 2979 -M 536 -f k -t 1200 -i 10 -S 56 (AF13)
```

Question 1 : Est-ce normal que le débit le plus faible (ou nul) est celui du trafic AF13 ?

Question 2 : Reportez les débits des trois trafics AF. Déterminez aussi les taux de pertes à partir des compteurs par DSCP affichés par le routeur et interprétez ce résultat. (Vous devriez voir une différence assez nette entre les performances des trois trafics)

Question 3 : Arrêtez les trafics AF11 et AF12 et reportez l'évolution du débit du trafic AF13 ? Est-ce normal que le trafic AF13 obtient un débit nettement plus élevé ? La valeur de ce débit est-elle cohérente avec la configuration du routeur ?

Test 4

→ Les configs :

Mêmes configurations. En plus rajoutez une classe nommée `http` correspondant au trafic du protocole HTTP, et une policy-map nommée `pmarker` afin de marquer les paquets de la classe `http` avec le DSCP `af11`. Attachez cette politique en entrée de l'interface `f 0/0`.

→ Le test :

D'abord, lancez un trafic AF11 avec un débit proche de 1 mbit/s afin de créer une situation de surcharge de réseau :

```
PC2:iperf -u -s -f k -i 1
PC1:iperf -u -c pc2 -f k -t 1200 -i 10 -S 40 -b 950k
```

Avec un autre terminal, lancez sur PC1, le serveur web apache avec la commande suivante :

```
sudo httpd -c "ServerName pc1"
```

Ensuite, en dehors de PC1, créez le répertoire `public_html` sous `/home/itqos` et copiez dedans le fichier `image.jpg` comme suit :

```
cd /home/itqos
mkdir public_html
chmod 755 .
chmod 755 public_html
```

ensuite, activez le réseau :  →  PCI Ethernet  Connecté , puis avec firefox téléchargez <https://www.npa.lip6.fr/~malouch/M2/ITQoS/tme6/timage.jpg> dans le répertoire `public_html`, enfin

```
chmod +r public_html/timage.jpg
```

Sur PC2 lancez firefox et accédez à l'url <http://pc1/~itqos/timage.jpg>. Vous constateriez que le téléchargement est très long.

Arrêtez maintenant le client iperf précédent et relancez-le cette fois-ci avec le DSCP `af13` :

```
PC1:iperf -u -c pc2 -f k -t 1200 -i 10 -S 56 -b 950k
```

Téléchargez à nouveau l'image.

Question : Quand le trafic exogène (iperf) passe par le niveau de précedence `af13`, le temps de téléchargement de l'image est-il nettement plus court ? Pourquoi ?

Test 7

→ Les configs :

Mêmes configurations. En plus rajoutez une classe nommée `ssh` correspondant au trafic du protocole SSH, et dans la policy-map `pmarker` marquez les paquets de la classe `ssh` avec le DSCP `af11`. (La policy-map est déjà attaché en entrée de l'interface `f 0/0`.)

→ Le test :

D'abord, lancez un trafic AF11 avec un débit supérieur à 1 mbit/s afin de créer une situation de congestion réseau :

```
PC2:iperf -s -f k -i 1
```

```
PC1:iperf -u -c pc2 -f k -t 1200 -i 10 -S 40 -b 2m
```

Avec un autre terminal, lancez sur PC1, le serveur ssh avec la commande suivante :

```
sudo /usr/sbin/sshd
```

Ensuite, sur PC2 connectez-vous à PC1 avec la commande `ssh pc1` et tapez des commandes telles que `ls -l` ou éditez un texte quelconque avec `vi`. Vous constateriez que l'interactivité de votre session ssh est plutôt très faible.

Arrêtez maintenant le client iperf précédent et relancez-le cette fois-ci avec le DSCP `af13` :

```
PC1:iperf -u -c pc2 -f k -t 1200 -i 10 -S 56 -b 2m
```

Testez à nouveau l'interactivité de votre session ssh.

Question : L'interactivité de la connexion ssh est-elle acceptable quand le trafic exogène (iperf) est dirigé vers le niveau de précedence `af13` ? Pourquoi ?

Test 8

→ Les configs :

Gardez toutes les configurations précédentes. Ensuite, afin de compléter la configuration de la classe AF2 et toujours à l'intérieur de la politique `policy-map PHBs` :

- Créez trois niveaux de précedence (rejet) dans AF2 tels que le trafic correspondant au DSCP `af21` est nettement mieux protégé en cas de congestion que celui correspondant à `af22`, qui à son tour est nettement mieux protégé que `af23`. Utilisez des probabilités de rejet élevées et déclenchez le rejet de paquets dès que les mémoires tampons se remplissent de quelques paquets. Choisissez donc convenablement vos seuils ainsi que vos probabilités de rejet.

→ Le test : Il n'y a pas de test à faire ici puisque la classe AF2 et ses niveaux de précedence sont équivalents à ceux de la classe AF1 déjà testée. Vérifiez uniquement votre configuration avec les commandes :

```
sh run | begin class AF2 et  
show policy-map interface f 0/1 output | begin AF2
```

Test 9 – Perfectionnement du service Best-Effort en utilisant RED et ECN

→ Les configs

Gardez les configurations précédentes. Ensuite :

- Modifiez l'allocation de la classe AF2 de 20% à 19% afin de laisser 1% au trafic best-effort.
- Créez par la suite, la classe BE pour le trafic avec le code DSCP 0.
- Dans la policy-map PHBs, attribuez 1% de la bande passante restante à la classe BE.
- Activez RED pour le trafic BE et configurez-le.
- Activez aussi la notification explicite de congestion (ECN) pour le trafic BE.

→ Le test :

Activez la prise en charge de la notification explicite de congestion par TCP dans le noyau du système linux avec la commande :

```
sudo sysctl net.ipv4.tcp_ecn=1
```

Ensuite, établissez une connexion TCP comme suit :

```
PC2:iperf -s -f k -i 1
```

```
PC1:iperf -c pc2 -M 536 -f k -t 1200 -i 10
```

Maintenant, avec le routeur et la commande


```
show policy-map interface f 0/1 output | begin BE
```

vérifiez que le routeur ne jette plus les paquets mais les marque. En plus, si votre configuration RED est adéquate, vous n'observerez jamais de pertes 'Tail drop' :

```
Router# show policy-map interface f 0/1 output | begin BE
Class-map: BE (match-all)
  351 packets, 391014 bytes
  5 minute offered rate 1002000 bps, drop rate 0 bps
Match: ip dscp default (0)
Queueing
  Output Queue: Conversation 75
  Bandwidth remaining 1 (%)
  (pkts matched/bytes matched) 351/391014
  (depth/total drops/no-buffer drops) 42/0/0
  exponential weight: 9
  explicit congestion notification
  mean queue depth: 20
```

dscp	Transmitted pkts/bytes	Random drop pkts/bytes	Tail drop pkts/bytes
af11	0/0	0/0	0/0
af12	0/0	0/0	0/0
af13	0/0	0/0	0/0
af21	0/0	0/0	0/0
af22	0/0	0/0	0/0
af23	0/0	0/0	0/0
af31	0/0	0/0	0/0
af32	0/0	0/0	0/0
af33	0/0	0/0	0/0
af41	0/0	0/0	0/0
af42	0/0	0/0	0/0

af43	0/0	0/0	0/0
cs1	0/0	0/0	0/0
cs2	0/0	0/0	0/0
cs3	0/0	0/0	0/0
cs4	0/0	0/0	0/0
cs5	0/0	0/0	0/0
cs6	0/0	0/0	0/0
cs7	0/0	0/0	0/0
ef	0/0	0/0	0/0
rsvp	0/0	0/0	0/0
default	351/391014	0/0	0/0



dscp	ECN Mark pkts/bytes
af11	0/0
af12	0/0
af13	0/0
af21	0/0
af22	0/0
af23	0/0
af31	0/0
af32	0/0
af33	0/0
af41	0/0
af42	0/0
af43	0/0
cs1	0/0
cs2	0/0
cs3	0/0
cs4	0/0
cs5	0/0
cs6	0/0
cs7	0/0
ef	0/0
rsvp	0/0
default	14/15596

Question : Quels sont les paramètres de RED que vous avez configurés ? Donnez aussi le résultat de la commande `show policy-map interface f 0/1 output | begin BE`.

Question finale : Terminez votre rapport avec une copie du résultat de la commande `sh run` afin d'inclure la configuration complète de votre routeur.

Avant de quitter la salle, éteignez la machine virtuelle avec le menu du bureau de la machine virtuelle et NON Virtual Box.

Appendix – A lire !!

Three commands to know and that implement both edge and core router functions:

class-map: create classes of packets

policy-map: create actions to apply on these packets

service-policy: apply these actions to packets going through an interface

Two documents to consult:

1 – « [Cisco IOS Quality of Service Solutions Configuration Guide](#) », contains general rules and some tutorials on DiffServ. Besides, it contains a complete practical example scenario for Diffserv deployment → ‘Sample DiffServ Implementation/Sample DiffServ Configurations’. *It is recommended to study this example as a complementary reading.*

2 - « [Cisco IOS Quality of Service Solutions Command Reference](#) »: Detailed explanations of all commands related to QoS supported by CISCO IOS.

PHB commands:

Bandwidth is the command that enables scheduling using a pre-defined queueing mechanism such that WFQ. (The default strategy before you apply DiffServ is FIFO).

random-detect is the command to enable and configure Weighted Random Early Detection (RED with different thresholds: single average multiple thresholds). Here the RED parameters are different for each DSCP.

Inside one class, you drop differently the packets to create finer differentiation between the packets belonging to the same class. Packets with the lower drop probability will be protected more from losses in periods of congestion than the others.

This is one of the objectives of the AF class (Assured Forwarding, which corresponds somewhat to intermediate types of applications or better than best-effort services)

For instance you give higher drop probability to ftp than that of ssh.

General command syntax: `random-detect dscp xx min max 1/p_max`

The **priority** command implements priority queueing (PQ) which means that packets belonging to the platinum class will not see other traffics in the same router, it is like the router bandwidth is totally allocated to the class: Whenever a packet from that class arrives at the router then the router starts always by this packet even if there are other packets from other classes in the buffers waiting for transmission.

However, to avoid starvation of other traffics we limit the rate of packets of higher classes. That is why there is a “500” after the command `priority` in the above mentioned cisco example. The rate of VoIP packets is limited to 500 kbits if there is not enough bandwidth, but this value is also a guarantee for VoIP. In other words, if the router is empty and other traffics are non present then VoIP can get more than 500, otherwise it gets at least 500. *It is important to notice that delays of allowed EF packets are not related to 500 but to the bandwidth of the output link thanks to the priority.*