Utilisation de l'outil TC pour la configuration de classes de service DiffServ

Baptiste Jacquemin

Utilisation de l'outil TC pour la configuration de classes de service DiffServ

Introduction

Ce papier présente un exemple de configuration des classes de service au niveau d'un routeur DiffServ à partir de l'outil TC (*Traffic Control*) [1] qui permet la configuration de gestionnaires de files d'attente basés sur des classes (*classful queuing discipline* ou *classful qdisc*). On considère alors pour cet exemple les trois classes DiffServ EF, AF et BE, la classe AF étant elle-même divisée en trois sous-classes : AF_1, AF_2 et AF_3 (voir Figure 1). L'ordonnanceur utilisé est de type HTB (*Hierarchical Tocken Bucket*) et on veut que les files soient initialement configurées avec une bande passante maximale de 990 kbit/s en BE et de 100 bit/s en EF, AF_1, AF_2 et AF_3 ; les différentes commandes à exécuter pour obtenir cette configuration sont alors présentées en détail dans la partie I. Ensuite, le démarrage d'un flux vidéo IPv6 enclenche un processus de réservation de QoS (non détaillé ici) qui se traduit par une augmentation de la taille de la file EF et d'une diminution de la file BE en fonction du débit considéré. La partie II nous présente alors les différentes commandes nécessaires pour ces modifications.

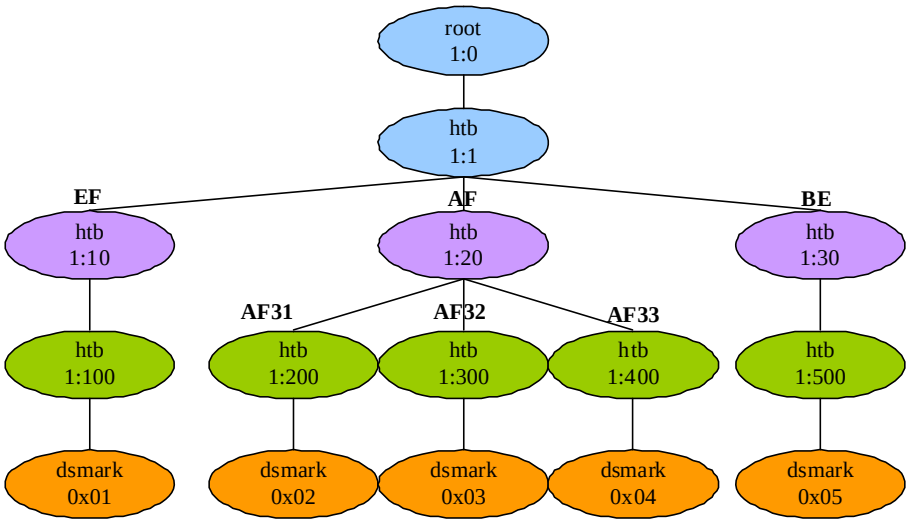

Figure 1 – Configuration des classes de service DiffServ par TC

I. Initialisation des classes de service DiffServ

Au démarrage du routeur DiffServ, les différentes classes de trafic souhaitées sont créées et configurées en exécutant la série de commandes TC suivante que nous détaillons par la suite :

```
Initialization of TC tree:
sudo /sbin/tc qdisc del dev tun0 root
```

```
sudo /sbin/tc qdisc add dev tun0 root handle 1:0 htb default 500
sudo /sbin/tc class add dev tun0 parent 1:0 classid 1:1 htb rate 990400bit
ceil 990400bit

Processing service classes:
sudo /sbin/tc class add dev tun0 parent 1:1 classid 1:10 htb rate 100bit
ceil 990400bit prio 1
sudo /sbin/tc class add dev tun0 parent 1:1 classid 1:20 htb rate 300bit
ceil 990400bit prio 2
sudo /sbin/tc class add dev tun0 parent 1:1 classid 1:30 htb rate 990000bit
ceil 990400bit prio 3
sudo /sbin/tc class add dev tun0 parent 1:10 classid 1:100 htb rate 100bit
ceil 100bit
sudo /sbin/tc qdisc add dev tun0 parent 1:100 handle 100:0 dsmark indices 1
set_tc_index
sudo /sbin/tc class change dev tun0 classid 100:0 dsmark mask 0x00 value
0x01
sudo /sbin/tc class add dev tun0 parent 1:20 classid 1:200 htb rate 100bit
ceil 100bit
sudo /sbin/tc qdisc add dev tun0 parent 1:200 handle 200:0 dsmark indices 1
set_tc_index
sudo /sbin/tc class change dev tun0 classid 200:0 dsmark mask 0x00 value
0x02
sudo /sbin/tc class add dev tun0 parent 1:20 classid 1:300 htb rate 100bit
ceil 100bit
sudo /sbin/tc qdisc add dev tun0 parent 1:300 handle 300:0 dsmark indices 1
set_tc_index
sudo /sbin/tc class change dev tun0 classid 300:0 dsmark mask 0x00 value
0x03
sudo /sbin/tc class add dev tun0 parent 1:20 classid 1:400 htb rate 100bit
ceil 100bit
sudo /sbin/tc qdisc add dev tun0 parent 1:400 handle 400:0 dsmark indices 1
set_tc_index
sudo /sbin/tc class change dev tun0 classid 400:0 dsmark mask 0x00 value
0x04
sudo /sbin/tc class add dev tun0 parent 1:30 classid 1:500 htb rate
990000bit ceil 990000bit
sudo /sbin/tc qdisc add dev tun0 parent 1:500 handle 500:0 dsmark indices 1
set_tc_index
sudo /sbin/tc class change dev tun0 classid 500:0 dsmark mask 0x00 value
0x05
```

Tout d'abord, l'arbre TC doit être initialisé à l'aide des 3 commandes qui suivent « Initialization of TC tree » :

- La 1ère commande, « `tc qdisc del dev tun0 root` », permet de supprimer une *queuing discipline* (`qdisc del`) de la sortie (`root`) de l'interface considéré (`dev tun0`), ici « tun0 ». Ceci permet de s'assurer qu'aucune *queuing discipline* n'est déjà mise en place pour l'interface considérée.
- La 2ème commande, « `tc qdisc add dev tun0 root handle 1:0 htb default 500` », permet d'ajouter une *queuing discipline* (`qdisc add`) à la sortie de l'interface tun0 (`dev tun0 root`) en lui attribuant un identifiant unique de la forme « major : minor » (le nombre « major » est complètement libre mais traditionnellement, il commence à 1 pour un objet attaché directement au qdisc root ; le nombre « minor » est forcément à 0 lorsqu'il s'agit d'un qdisc, tout autre valeur identifiant l'objet comme une classe), ici 1:0, et en précisant quel type de *queing discipline* doit être

attachée, ici htb. Enfin, la commande `default 500` indique que tout flux non classifié est dirigé vers la classe d'identifiant 500.

- La 3ème commande, «`tc class add dev tun0 parent 1:0 classid 1:1 htb rate 990400bit ceil 990400bit` », permet d'ajouter une classe (`class add`) qui doit être attachée à l'interface « tun0 » (`dev tun0`). Cette classe est attachée à un parent d'identifiant 1:0 (`parent 1:0`) et a elle-même pour identifiant 1:1 (`classid 1:1`). Ensuite, étant donné qu'une *queuing discipline* contient toujours des classes du même type qu'elle-même, la classe est de type HTB (`htb`). Enfin, cette classe est configurée pour avoir une bande passante garantie de 990400bps (`rate 990400bit`) et une bande passante maximale en « burst » de 990400bps (`ceil 990400bit`). Dans le cas de cette classe dont hériterons toutes les autres sous-classes, les deux valeurs *rate* et *ceil* sont les mêmes et définissent en fait la bande passante maximale au niveau du nœud concerné.

Ensuite, prenons le cas de la file EF et analysons les différentes commandes TC nécessaires à sa configuration :

- Tout d'abord, la commande «`tc class add dev tun0 parent 1:1 classid 1:10 htb rate 100bit ceil 990400bit prio 1` » permet la création d'une classe d'identifiant 1:10, attachée à l'interface tun0 et ayant pour parent la classe 1:1. De plus, cette classe est de type HTB et est configurée pour avoir une bande passante garantie initiale de 100bps (tc n'accepte pas la valeur 0 pour les valeurs *rate* et *ceil* donc nous mettrons 100 pour indiquer une valeur proche de 0) et une bande passante maximale en « burst » initiale de 990400bps. Enfin, la priorité de cette classe est 1 (`prio 1`), ce qui correspond à la priorité maximale. On peut constater que la valeur du paramètre *ceil* d'une sous-classe est au maximum égale au paramètre *rate* de la classe parente.
- Ensuite, la commande « `tc class add dev tun0 parent 1:10 classid 1:100 htb rate 100bit ceil 100bit` » permet de créer une classe de d'identifiant 1:100, attachée à l'interface tun0 et ayant pour parent la classe 1:10. Cette classe est de type HTB et est configurée pour avoir une bande passante garantie de 100bps et une bande passante maximale en « burst » de 100bps, c'est-à-dire des valeurs proches de 0.
- Puis, la commande « `tc qdisc add dev tun0 parent 1:100 handle 100:0 dsmark indices 1 set_tc_index` » permet d'ajouter une *queuing discipline* d'identifiant 100:0 (`handle 100:0`), ayant pour parent la classe 1:100. Ensuite, la commande `dsmark` offre les fonctionnalités dont a besoin l'architecture DiffServ. Ainsi, l'option `set_tc_index` indique que le contenu du champ DS original d'un paquet entrant est enregistré tandis que l'option `indices 1` indique le nombre d'indice ou encore la taille de la table des couples [masque, valeur], ici 1.
- Enfin, la commande «`tc class change dev tun0 classid 100:0 dsmark mask 0x00`» permet de modifier la classe 100:0 en indiquant un masque (`mask 0x00`), ici 0 et une valeur (`value 0x01`), ici 1 qui permettront de définir la valeur du champ DS du paquet en sortie de la manière suivante : DS sortie = (DS entrée && mask) || value.

Les autres classes sont ensuite configurées de la même manière que la classe EF sauf que dans le cas de AF, trois sous-classes ont pour parent la classe AF d'identifiant 1:20.

II. Modification des classes de service

Lorsque le flux vidéo IPv6 débute, différentes commandes doivent alors être exécutées pour modifier les classes TC et ajouter une règle de filtrage de paquet. Elles sont décrites ci-dessous et détaillées par la suite :

```
Adding of the filter:
sudo /sbin/ip6tables -t mangle -I POSTROUTING 1 -o tun0 -d
2001:660:6602:103:218:8bff:fea7:579c -m multiport -p UDP --dports 2502 -j
MARK --set-mark 1
sudo /sbin/tc filter add dev tun0 protocol ipv6 parent 1:0 prio 1 handle 1
fw flowid 1:100

Reservation of rate for this flow:
sudo /sbin/tc class change dev tun0 parent 1:0 classid 1:1 htb rate
990400bit ceil 990400bit
sudo /sbin/tc class change dev tun0 parent 1:1 classid 1:10 htb rate
180100bit ceil 990400bit prio 1
sudo /sbin/tc class change dev tun0 parent 1:1 classid 1:20 htb rate 300bit
ceil 990400bit prio 2
sudo /sbin/tc class change dev tun0 parent 1:1 classid 1:30 htb rate
810000bit ceil 990400bit prio 3
sudo /sbin/tc class change dev tun0 parent 1:10 classid 1:100 htb rate
180100bit ceil 180100bit
sudo /sbin/tc class change dev tun0 parent 1:20 classid 1:200 htb rate
100bit ceil 100bit
sudo /sbin/tc class change dev tun0 parent 1:20 classid 1:300 htb rate
100bit ceil 100bit
sudo /sbin/tc class change dev tun0 parent 1:20 classid 1:400 htb rate
100bit ceil 100bit
sudo /sbin/tc class change dev tun0 parent 1:30 classid 1:500 htb rate
810000bit ceil 810000bit
```

Tout d'abord, un nouveau filtre est ajouté par les deux premières commandes :

- La 1ère commande «ip6tables -t mangle -I POSTROUTING 1 -o tun0 -d 2001:660:6602:103:218:8bff:fea7:579c -m multiport -p UDP --dports 2502 -j MARK --set-mark 1 » utilise l'outil ip6tables qui permet de gérer le filtrage de paquets IPv6. L'option « -t » indique la table de correspondance des paquets sur laquelle la commande doit opérer : dans notre cas, le terme « mangle » indique que l'on va effectuer une modification spéciale des paquets. L'option « -I POSTROUTING 1 » permet d'ajouter une règle à une position donnée, ici, la règle POSTROUTING (pour modifier les paquets lorsqu'ils sont sur le point de sortir) à la position 1 (pour dire que c'est la 1ère règle qui sera appliquée aux paquets). Ensuite, l'option « -o tun0 » indique l'interface de sortie qui envoie les paquets. L'option « -d 2001:660:6602:103:218:8bff:fea7:579c » indique que la règle s'applique sur les paquets ayant pour destination l'adresse IPv6 indiquée. Les options suivantes « -m

multiport -p UDP --dports 2502 » indique que l'on va chercher les correspondances avec un ou plusieurs ports pour les paquets utilisant le protocole UDP, dans notre cas, le port de destination 2502. Si le paquet correspond aux critères précédents, l'option « -j MARK --set-mark 1 » indique qu'il faut activer la valeur de marquage Netfilter associée à ces paquets, ici 1.

- La 2ème commande, « tc filter add dev tun0 protocol ipv6 parent 1:0 prio 1 handle 1 fw flowid 1:100 », permet d'ajouter un filtre (filter add), attaché à l'interface tun0 (dev tun0) et ayant pour parent, l'objet d'identifiant 1:0 (parent 1:0). Le protocole correspondant à ce filtre est IPv6 (protocol ipv6) et la priorité du filtre est 1 (prio 1), la valeur maximale. Ensuite, l'option « handle 1 fw flowid 1:100 » indique que l'on veut transmettre les paquets ayant pour valeur de marquage Netfilter 1 (ceux marqués par la commande ip6tables précédentes) à la classe d'identifiant 1:100.

Une fois que le filtre a été créé, des modifications doivent être réalisées sur la taille des files. Dans le cas considéré, le débit de l'application vidéo pour laquelle on veut effectuer une réservation dans la file EF est d'environ 180 kbit/s. Les deux commandes suivantes sont alors nécessaires pour la file EF:

- « tc class change dev tun0 parent 1:1 classid 1:10 htb rate 180100bit ceil 990400bit prio 1 » qui augmente le paramètre rate de la classe 1:10 de 180000 ; le paramètre ceil étant limité par la valeur du paramètre rate de sa classe parente, il ne peut être augmenté.
- « tc class change dev tun0 parent 1:10 classid 1:100 htb rate 180100bit ceil 180100bit » qui augmente les paramètres rate et ceil de la classe 1:100 de 180100.

Ensuite, les mêmes types de commande doivent être réalisés mais, cette fois-ci, pour diminuer de 180kbit la taille de la file BE, qui correspond aux classes d'identifiant 1:30 et 1:500 (voir Figure 1).

Conclusion

Cet exemple permet donc de comprendre comment TC peut être utilisé pour la gestion des files d'attente dans le cas où le modèle DiffServ est utilisé. Ainsi, différents niveaux hiérarchiques peuvent être définis pour classifier des flux en fonction de leurs caractéristiques (ou exigences). Ainsi, les flux vidéo et audio peuvent être configurés pour utiliser le service EF, les applications de type streaming peuvent utiliser le service AF tandis que les téléchargements et le « web browsing » seront redirigés vers la classe BE. Cependant, si TC permet de créer des *queuing discipline*, des classes ou encore des filtres, il est important de souligner qu'il ne permet pas le marquage de paquet. Ainsi, pour ce faire, nous avons ici choisi d'utiliser l'outil ip6tables, spécifique aux paquets IPv6.

Bibliographie

[1] Linux Advanced Routing & Traffic Control HOWTO, http://lartc.org/howto/