Configuration de la QoS sur un réseau de Campus

Didier Benza
SEMIR, INRIA Sophia-Antipolis
2004, route des Lucioles BP 93 FR 06902 Sophia-Antipolis Cedex
Didier.Benza@sophia.inria.fr

Résumé

Si les techniques de Qualité de Service ou QoS sont très utilisées dans les environnements opérateurs, elles restent peu utilisées sur les réseaux de campus. Le principal usage qui en est fait actuellement est la protection des communications téléphoniques. C'est bien sûr nécessaire, mais c'est très en-dessous de ce que savent faire les équipements des réseaux de campus. Une configuration globale de la QoS permet de rendre le réseau intelligent vis-à-vis des flux qu'il achemine. Cela lui permet de réagir beaucoup mieux à des stress ou à des agressions.

Après quelques rappels sur DiffServ et l'architecture matérielle des équipements de niveau 2, nous montrerons les applications de la QoS au-delà de la protection de la voix et nous détaillerons les éléments qui doivent être configurés. Nous présenterons aussi un outil développé par les moyens informatiques de l'INRIA de Sophia qui simplifie les tâches de configuration de la QoS au niveau 3 sur un routeur Cisco.

Mots clefs

QoS, 802.1p, DSCP, DiffServ, PHB, XMLQoS, Cisco

1 Contexte

Le Centre de Recherche INRIA de Sophia-Antipolis a depuis longtemps fait le choix d'avoir une attitude protectionniste sur la gestion de la bande passante. C'est à dire, de limiter les débits en fonction du type d'usage. Cela est dû à la fois au nombre de réseaux que nous gérons, mais aussi et surtout au fait que certains de nos réseaux sont utilisés à des fins expérimentales ou servent à héberger des organismes (startups, Ercim/W3C) avec lesquels nous négocions des contrats de service. Lorsque les valeurs issues de la négociation sont fixées, nous implémentons la configuration de QoS résultante sur nos équipements de niveau 3.

Par ailleurs, nous avons changé les équipements du réseau local en 2005. L'appel d'offre se passait quelque temps avant la mise en place d'une architecture de téléphonie sur IP. Cela nous a rendu particulièrement exigeants lors de la rédaction du CCTP et vigilants sur les capacités des équipements lors de l'analyse des réponses, puisque les téléphones devaient être raccordés directement sur les équipements de niveau 2. Lors de la mise en place de la maquette qui a précédé la mise en service des nouveaux équipements, nous avons découvert l'étendue et la richesse de la configuration de la QoS au niveau 2.

2 Les services différenciés

L'architecture des Services Différenciés ou DiffServ, décrite dans la RFC 2475 [1], permet aux équipements réseaux centraux de prendre des décisions sur l'avenir d'un paquet sans qu'ils aient besoin pour cela de maintenir des tables d'états pour chaque flux. La mise à jour et la conservation de telles tables ne pourraient pas passer à l'échelle de l'Internet. Les nœuds placés à la frontière d'un domaine effectuent un travail de classification des flux, de mise en forme du trafic et de marquage des paquets. Le marquage utilise une valeur numérique (*Code Point*) qui est insérée dans l'en-tête IP. Les nœuds placés au cœur du réseau peuvent associer très rapidement des décisions d'acheminement aux flux agrégés avec un même *Code Point*. On nomme *Per Hop Behavior* (PHB), le comportement d'un nœud relatif à un *Code Point* donné. Un « Domaine DS » est un ensemble de nœuds contigus ayant le même ensemble de PHB définis. Une « Région DS » est un ensemble de domaines DS contigus, chaque domaine établissant des Spécifications de Niveau de Service avec ses voisins, comme décrit dans la RFC 3260 [2].

Afin de transporter le *Code Point* au sein de l'en-tête IP, les 6 bits de l'octet TOS du paquet IPv4 (ou de l'octet Traffic Class d'IPv6) ont été ré-utilisés [3]. Il s'agit du « champ DS ». Ces six bits portent les valeurs dites DSCP pour *Differentiated Services Code Point*. Il y a donc 64 *Code Point* différents, chacun pouvant être associé à un PHB. Par extension du PHB, la RFC 3086 [4] définit le *Per Domain Behavior* (PDB) qui représente le comportement attendu d'un domaine DS vis-à-vis de flux portant une valeur DSCP particulière.

- La valeur DSCP 0 est associée au PHB *Best Effort* (BE) ou *Default*. Le PHB BE correspond au comportement par défaut réservé à la grande majorité des paquets. Il s'agit d'un acheminement sans garantie en termes de variation de délai ou de pertes.
- Les *Class Selector Code Point* (CS) sont les valeurs numériques du champ DSCP ayant une représentation binaire de la forme xxx000. Il y a donc 8 CS différents. La RFC 3662 [5] suggère que le Class Selector 1 (001000) soit utilisé pour marquer des paquets devant recevoir une priorité plus basse que la normale (LE : *Lower Effort*). Le PHB associé à cette valeur DSCP est le suivant : un paquet LE ne devrait être transmis sur un lien que s'il est inutilisé. Cette classe est aussi nommée *Scavenger* dans la littérature, par analogie avec la classe du même nom sur Internet 2 [6].
- La RFC 2597 [7] standardise les PHB *Assured Forwarding* (AF). Ces PHB correspondent à la définition de 4 classes AF*i* où $1 \leq i \leq 4$, classées par

ordre de priorité croissante. Chaque classe dispose de 3 niveaux de *drop precedence* ou probabilité de suppression. On note un PHB AF sous la forme AF*ij* où $1 \leq j \leq 3$. Pour une classe *i* donnée, plus la valeur de *j* est grande, plus la probabilité que le paquet soit détruit, en cas de congestion, est élevée. Les nœuds qui supportent les classes AF doivent limiter la congestion sur le long terme en détruisant des paquets mais autorisent des congestions de courte durée provenant de *bursts* en retardant des paquets.

– Le code DSCP EF (pour *Expedited Forwarding),* défini dans la RFC 3246 [8], correspond au PHB attendu pour le traitement de paquets nécessitant un service de bout en bout avec peu de pertes, de latence et de gigue. Un paquet marqué avec la valeur EF doit être expédié le plus rapidement possible. Il ne doit être retardé que par l'acheminement d'autres paquets EF qui le précédent.

Un nœud de périphérie est du type *Multi-Field Classifier*, c'est-à-dire qu'il va considérer des éléments tels que le protocole, les adresses source et destination et les numéros de ports pour établir un marquage DS qui correspondra à un PHB au sein du réseau. Un profil de trafic spécifie les propriétés temporelles d'un flux sélectionné par un *classifier*. Il donne des règles qui permettent de déterminer si un paquet particulier du flux est dans le profil ou hors profil. Un flux hors profil est conditionné par *shaping* ou par *policing* :

– Le *shaping* consiste à retarder certains ou tous les paquets de façon à obliger un flux à se conformer à son profil de trafic. Il nécessite donc une file d'attente. Quand elle est pleine, les paquets sont détruits.

– Le *policing* détruit certains ou tous les paquets d'un flux afin qu'il se conforme à son profil de trafic

La RFC 4594 [9] vient de définir de façon très explicite quel marquage DSCP devrait être appliqué en fonction du type de flux. Les principales recommandations sont reproduites dans la figure 1.

Classe de Service	**DSCP**
Network Control	CS6
Telephony	EF
Signaling	CS5
Multimedia Conferencing	AF41, AF42, AF43
Real-Time Interactive	CS4
Multimedia Streaming	AF31, AF32, AF33
Broadcast Video	CS3
Low-Latency Data	AF21, AF22, AF23
OAM	CS2
High-Throughput Data	AF11, AF12, AF13
Standard	DF (CS0)
Low-Priority Data	CS1

Figure 1 - *Associations entre Classe de Service et DSCP*

La RFC 4594 [9] indique aussi des règles de marquage spécifiques pour les flux AF. Un *classifier* doit utiliser l'algorithme *Single Rate Three Color Marker* (srTCM défini dans la RFC 2697 [10]). Pour une classe AF*i,* les paquets respectant le profil de trafic (*Committed Information Rate*) devraient être marqués en AF*i*1*,* les paquets qui dépassent un premier seuil *(Committed Burst Size*) devraient être marqués AF*i*2 et ceux qui dépassent un second seuil *(Excess Burst Size),* AF*i*3*.*

3 La QoS au niveau 2 (Ethernet)

La gestion de la QoS sur des liens Ethernet se fait en utilisant le champ CoS (Class of Service) défini par la norme IEEE 802.1p et utilisé par les trames 802.1q. Le champ CoS a une longueur de 3 bits et ne permet donc la transcription que de 8 classes de services. Des règles d'association du code DSCP vers une valeur CoS appropriée sont nécessaires. Ces associations n'ont pas fait l'objet d'une normalisation.

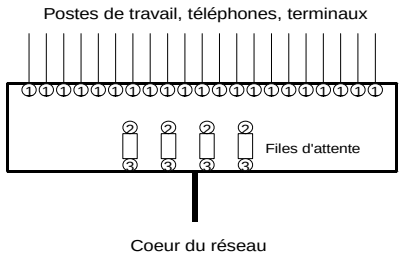

Figure 2 - *Les différents points à configurer*

La figure 2 montre 3 points où la QoS intervient dans un équipement réseau. Ces points sont décrits ci-après :

1. La classification et le *policing* : Tous les paquets issus de la classification sont marqués d'une valeur DSCP.

2. La gestion de la congestion : (*AQM : Active Queue Management*). Il s'agit de répartir les différents PHB utilisés dans les différentes files d'attente (généralement moins nombreuses) et de programmer la gestion de la congestion afin que la destruction des paquets ne soit pas aléatoire. On trouvera ici des algorithmes tels que *Weighted Tail Drop (WTD), Random Early Detection* (RED), *Weighted RED* qui préviennent la congestion en détruisant préventivement des paquets de la file. WTD permet, par exemple, d'associer une valeur DSCP ou CoS à un seuil. Une fois que ce seuil de remplissage a été atteint, les trames marquées avec cette valeur peuvent être détruites.

3. L'ordonnancement : C'est la façon dont le commutateur choisit, parmi les paquets en attente dans les différentes files, celui qu'il va acheminer. On trouvera ici des algorithmes tels que *Shaped Round Robin* (SRR), *Weighted Round Robin* (WRR) ou *Deficit WRR* (DWRR). SRR, par exemple, fonctionne en deux modes : *shaped* ou *shared*. Une file d'attente servie en mode *shaped* dispose d'une bande passante garantie et bornée. Une file d'attente servie en mode *shared* est assurée de pouvoir disposer d'au moins un certain pourcentage de la bande passante restante. Si une file en mode *shared* est vide, les autres files disposent temporairement de plus de bande passante. Cet algorithme permet donc d'éviter que les flux prioritaires monopolisent un lien. Le contrôle du profil de bande passante est généralement assuré dans les équipements avec un algorithme *Leaky Bucket* ou *Token Bucket.* Ce dernier est représenté dans la figure 3 sous une version imagée. Des jetons tombent dans un seau toutes les 1/r secondes et le seau ne peut en contenir qu'un nombre limité au total. Quand un paquet de N octets arrive, chaque octet du paquet est associé à un jeton. S'il y a assez de jetons dans le seau, le paquet peut passer et il emporte avec lui N jetons.
S'il n'y en a pas assez, le paquet est considéré comme non conforme. srTCM, par exemple, utilise 2 *token buckets*.
Les commutateurs disposent d'un nombre variable de files d'attente en entrée et en sortie. Les structures de files d'attente des équipements Cisco sont décrites dans la documentation avec des expressions du type *1PxQyT* où *1P* indique qu'il y a une file prioritaire, *xQ* représente *x* files d'attentes non prioritaires et *yT* représente *y* seuils par file d'attente.

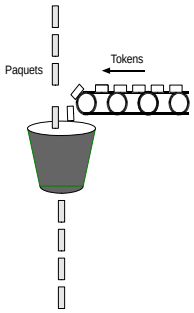

Figure 3 - *L'algorithme du token bucket*

4 La QoS sur un réseau de campus

Une architecture de QoS ne permet pas à un réseau manifestement sous-dimensionné de bien fonctionner, il doit être adapté au trafic acheminé. Mais le ratio habituel entre le débit des ports de collecte et celui des liens montant est trop faible dans les réseaux actuels pour qu'ils soient non bloquant. À plusieurs reprises, ces dernières années, on a vu des vers qui se répandent très rapidement et génèrent des quantités considérables de trafic. Slammer [11] générait 54 000 scans par seconde pendant son pic d'expansion. Le nombre de machines infectées doublait toutes les 8,5 secondes et le ver générait des flux UDP à la vitesse maximum de la carte réseau. Slammer ne visait pas un port très populaire (sinon Internet aurait été bloqué en quelques minutes) mais réussissait à infecter une machine avec un seul paquet. Le cœur du ver était une minuscule boucle qui générait un paquet de 404 octets vers une adresse IP aléatoire. Une machine infectée connectée à 100 Mbits/s pouvait envoyer 30 000 paquets par seconde. Une prochaine génération de ver, au lieu de se propager aléatoirement, pourrait viser l'adressage du réseau de la machine infectée. Qui peut dire qu'un ver de ce type ne sera pas lancé dans les années qui viennent ?

Par ailleurs, on constate une intégration croissante du réseau dans le Système d'Information : agrégation de ports vers des serveurs et configuration du bon algorithme de répartition de charge, configuration de *Virtuals Servers* au niveau des *Data Center,* etc. Le réseau est configuré au plus près des applications, il constitue une part importante de l'infrastructure du SI. Il est nécessaire de doubler les liens, de doubler les alimentations et les CPU des équipements réseaux, mais ces mesures ne protègent que de pannes matérielles, elles ne peuvent rien contre une attaque virale généralisée. L'architecture DiffServ permet d'augmenter globalement la disponibilité des services jugés indispensables par l'organisme.

Une fois rédigée la liste, nécessairement courte, des services indispensables, l'équipe réseau doit mettre en place une configuration globale de QoS chargée de protéger les flux correspondant à ces services. Elle doit créer un domaine DS au sein de l'établissement. Cette mise en place doit se faire en prenant en compte, en même temps, le niveau 3 et le niveau 2.

Afin d'illustrer notre propos, nous allons prendre un exemple concret d'implémentation d'une politique de QoS au niveau 2. Il ne s'agit pas de l'architecture mise en place sur le réseau de Sophia, mais plutôt d'une présentation des possibilités des équipements.

On choisit de :

- garantir le fonctionnement de l'ensemble des postes téléphoniques ;
- garantir le fonctionnement des équipements de visioconférence ;
- protéger les accès à quelques applications client/serveur interactives (missions, bons de commandes, demandes d'interventions aux moyens informatiques...).

Pour des raisons opérationnelles évidentes, nous ajoutons à cette liste d'applications proposée par la direction de l'organisme, la protection du trafic OAM[1] (SNMP, ssh sur les équipements, etc). Bien sûr, cette liste de services est un exemple, un autre site pourra choisir de protéger la messagerie, un serveur Web ou des sauvegardes.

Nous allons caractériser chaque trafic (type de trafic et profil de bande passante par port) et utiliser la figure 1

[1]OAM : *Operation Administration and Management*

pour déterminer quel PHB utiliser. Les valeurs de CoS sont choisies arbitrairement quand aucune norme n'existe.

- La téléphonie est une application critique qui ne tolère ni perte, ni latence, ni gigue. Nous utiliserons le PHB EF (46). La valeur CoS associée sera 5 selon le standard IEEE 802.1D [12]. On limite la bande passante maximum à 128 Kbits/s.
- Nous marquerons la signalisation avec le PHB CS5 (40) et une CoS de 3. La signalisation qui accompagne une communication téléphonique ne dépasse pas 32 Kbits/s.
- Les applications que nous devons protéger sont du type *low-latency* : un humain est devant l'écran et attend une réponse. Nous utiliserons donc le PHB AF21 (18) et la valeur CoS 2. Pour cet exemple, le profil normal de trafic sera de 5 Mbits/s. Dans la réalité, chaque application peut disposer d'un profil différent.
- La visioconférence disposera du PHB AF41 (34) et de la valeur CoS 4. La visioconférence utilise au maximum 2 Mbits/s par équipement.
- Le trafic OAM utilisera le PHB CS2 (16) et une CoS de 7. Son profil est de 1 Mbits/s.
- Tous les autres paquets émis par le poste de travail seront marqués avec le PHB BE (0), nous utiliserons la valeur CoS de 0. Ils disposeront d'un profil de 5 Mbits/s.

Les équipements de collecte, utilisés dans l'exemple, sont des Cisco Catalyst 3750 et 3750-E. Voici le détail des fonctionnalités en chacun des 3 points indiqués dans la figure 2 :

1. Les C3750 sont des *Multi-Field Classifiers*, ils peuvent pratiquer du *policing* ou du *shaping* en entrée et remarquer ou détruire les paquets qui ne respectent pas leur profil. L'architecture matérielle ne permet pas la mise en place de srTCM. En contrepartie, nous marquerons les paquets dépassant leur profil dans la classe Scavenger. À la liste des PHB décidés précédemment, nous ajoutons donc la classe Scavenger avec le PHB LE (8) et une CoS de 1.

2. Les C3750 disposent de files d'attente du type 1P3Q3T qui peuvent aussi être configurées sous la forme 4Q3T si on souhaite renoncer à la *priority queue*. L'algorithme AQM est *Weighted Tail Drop* (WTD), il faut donc établir des associations entre les valeurs CoS ou DSCP, d'une part et des seuils dans des files, d'autre part.

3. L'algorithme d'ordonnancement utilisé sur les C3750 est SRR.

Programmation d'un commutateur de collecte

Nous activons la QoS au niveau global du commutateur :

```
mls qos
```

À partir de cet instant, tout paquet entrant sur le commutateur est systématiquement remarqué avec le PHB BE et une CoS de 0 par le commutateur.

Nous mettons en place les associations DSCP vers CoS et CoS vers DSCP que nous avons définies :

```
mls qos map dscp-cos 0 to 0
mls qos map dscp-cos 8 to 1
mls qos map dscp-cos 18 to 2
mls qos map dscp-cos 40 to 3
mls qos map dscp-cos 34 to 4
mls qos map dscp-cos 46 to 5
mls qos map dscp-cos 16 to 7
mls qos map cos-dscp 0 8 18 40 34 46 47 16
```

Nous mettons en place les ACL permettant de classifier les paquets. S'agissant d'ACL tout à fait classiques, nous ne les détaillerons pas. Pour l'exemple, voici ce que pourrait être l'ACL de classification des flux OAM :

```
ip access-list extended management
 permit udp any any eq 161
 permit tcp any any eq 161
 permit tcp any r.r.r.0 0.0.0.255 eq ssh
```

On définit ensuite des *class-maps* utilisant les ACL que nous venons de définir. Là encore, nous n'écrirons qu'une seule *class-map*, pour l'exemple :

```
class-map match-all management
 match access-group name management
```

Nous programmons le comportement global de remarquage DSCP pour les paquets dépassant leur profil pour les PHB BE, CS2, AF21, AF41, CS5 vers le PHB LE :

```
mls qos map policed-dscp 0 16 18 34 40 to 8
```

L'action *policed-dscp-transmit* provoquera un remarquage du paquet suivant la configuration programmée avec la commande ci-dessus. On crée deux *policy-map*, chargées de marquer et policer le trafic. Une pour la téléphonie, qui sera appliquée sur les ports raccordant des téléphones et une pour les autres ports. Si on utilise les *voice-vlan*, qui permettent de cascader un PC derrière un téléphone, on peut se contenter d'une seule *policy-map* pour les deux VLANs. Dans la *policy-map* créée pour le trafic de téléphonie, on profile à 32 Kbits/s les flux qui ne sont pas de la voix ou de la signalisation et on marque ces paquets avec le PHB LE. Ce qui dépasse le profil est détruit. Chaque ligne *police x y* configure un *token bucket* de débit *x* bits/s admettant des *bursts* de *y* octets.

```
policy-map telephonie
 class telephonie
  set dscp 46
  police 128000 8000 exceed-action drop
 class signalisation
  set dscp 40
  police 32000 8000 exceed-action
       policed-dscp-transmit
 class class-default
  set dscp 8
  police 32000 8000 exceed-action drop

policy-map data
 class prioritaire
  set dscp 18
  police 5000000 32000 exceed-action
       policed-dscp-transmit
 class visioconf
  set dscp 34
  police 2000000 128000 exceed-action
       policed-dscp-transmit
 class management
  set dscp 16
  police 1000000 32000 exceed-action
       policed-dscp-transmit
 class class-default
  set dscp 0
  police 5000000 32000 exceed-action
       policed-dscp-transmit
```

On applique les deux politiques aux interfaces. Disons que les 12 premiers ports raccordent des téléphones.

```
interface range FastEthernet1/0/1-12
 service-policy input telephonie

interface range FastEthernet1/0/13-48
 service-policy input data
```

Nous organisons l'affectation des différents PHB dans les 4 files d'attentes du port *uplink*. Nous avons défini 7 PHB différents, certains PHB vont donc devoir partager la même file d'attente. Les seuils et l'utilisation de l'algorithme WTD permettent d'organiser cette coexistence et de favoriser les PHB les plus prioritaires. Nous appliquons grossièrement la règle suivante : nous affectons le trafic de téléphonie à la file 1 (configurée en file prioritaire), les trafics prioritaires à la file 2, le trafic BE à la file 3 et le trafic Scavenger à la file 4. Nous utilisons les seuils afin d'introduire une préférence pour les flux prioritaires dans une file partagée. Nous programmons les associations DSCP->(file, seuil) de façon cohérente avec les associations CoS->(file, seuil).

```
mls qos srr-queue output cos-map queue 1
      threshold 3  5
mls qos srr-queue output cos-map queue 2
      threshold 3  3
mls qos srr-queue output cos-map queue 2
      threshold 2  4
mls qos srr-queue output cos-map queue 2
      threshold 2  2
mls qos srr-queue output cos-map queue 2
      threshold 1  7
mls qos srr-queue output cos-map queue 3
      threshold 3  0
mls qos srr-queue output cos-map queue 4
      threshold 3  1

mls qos srr-queue output dscp-map queue 1
      threshold 3  46
mls qos srr-queue output dscp-map queue 2
      threshold 3  40
mls qos srr-queue output dscp-map queue 2
      threshold 2  34
mls qos srr-queue output dscp-map queue 2
      threshold 2  18
mls qos srr-queue output dscp-map queue 2
      threshold 1  16
mls qos srr-queue output dscp-map queue 3
      threshold 3  0
mls qos srr-queue output dscp-map queue 4
      threshold 3 8
```

Nous devons maintenant programmer les niveaux exacts des seuils. Le seuil n°3 est implicitement toujours à 100% de la taille de la file. Nous avons utilisé ce seuil pour les PHB EF, BE et LE qui disposent respectivement de la totalité des files 1, 3 et 4, nous n'avons donc pas besoin de programmer les seuils de ces files. Nous avons affecté les PHB AF21, AF41, CS2 et CS5 aux différents seuils de la file 2. Nous devons programmer la valeur des seuils qui seront utilisés. Cela se fait en programmant un *queue-set*. Deux *queue-sets* sont disponibles sur le commutateur, chaque interface étant affectée à l'un ou l'autre. Nous programmons les seuils suivants :

- 50% pour le seuil n° 1 (associé aux paquets OAM)
- 90% pour le seuil n°2 (associé aux applications prioritaires et à la visioconférence)

```
mls qos queue-set output 1 threshold 2 50 90
100 100
```

Enfin, nous devons programmer la façon dont l'algorithme SRR va vider chaque file vers le lien. On choisit de configurer la file 1 en mode *shaped* pour le trafic de téléphonie. On limite ce trafic à 1% de la bande passante sur le lien (1/100ème). Les autres files sont positionnées en mode *shared* et se partageront la bande passante restante. On choisit de répartir la bande passante restante de la façon suivante :

- 70% pour le trafic prioritaire
- 25% pour le trafic BE
- 5% pour le trafic Scavenger

```
Interface gigabitethernet1/0/1
 srr-queue bandwidth shape 100 0 0 0
 srr-queue bandwidth share 1 70 25 5
 queue-set 1
 priority-queue out
```

La dernière ligne active le fonctionnement de la file 1 en file prioritaire : la file est inspectée après chaque expédition. Si un paquet attend, il passera toujours le premier.

Programmation du cœur de réseau

Nous avons vu comment configurer les équipements de collecte afin qu'ils classent, policent et marquent le trafic entrant sur le réseau. Le cœur de réseau ne fait pas de marquage, il fait confiance aux valeurs DSCP et CoS qu'il reçoit des commutateurs de collecte. Il agit comme un *Behavior Aggregate Classifier* défini en [3]. Par contre, il faut programmer les associations (DSCP, CoS) vers les files d'attente, comme nous l'avons fait pour les liens montants des commutateurs de collecte. Cette programmation étant rigoureusement identique, dans le principe, à celle des commutateurs de collecte, nous ne la détaillerons pas. Dans le cas du Cisco Catalyst 6500, qui est l'équipement de cœur que nous utilisons, chaque carte d'interface dispose d'une architecture qui lui est propre. Par exemple, les cartes 4 ports 10 Gbits disposent d'une architecture 1P7Q8T pour les files (1 file prioritaire, 7 files d'attentes non prioritaires et 8 seuils par file) et l'algorithme d'ordonnancement est DWRR, alors que les cartes 24 et 48 ports 1 Gbits ont une architecture 1P3Q8T/DWRR. La programmation du *Control Plane Policing* (CoPP), sur les équipements de cœur ou de bordure, afin de limiter l'accès des trafics indésirables vers leur CPU est au-delà du périmètre de ce document. La lecture du document *Deploying Control Plane Policing* [13] permettra toutefois d'obtenir toutes les informations nécessaires.

Dans le cas où des protocoles de routage transitent entre le cœur de réseau et le routeur WAN, ces flux (TCP) doivent absolument être protégés avec le PHB CS6.

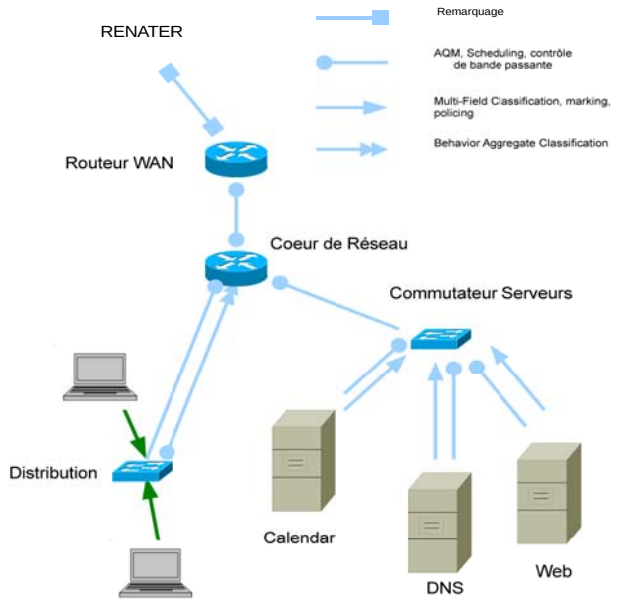

Figure 4 - *Vue globale des liens protégés*

La figure 4 schématise les différents points qui sont programmés. La figure montre des serveurs et le commutateur les distribuant, dont la configuration n'est pas évoquée dans cet article. La configuration du commutateur pour des serveurs est différente de celle pour des postes clients dans le sens où on marque les flux qui sortent des serveurs, mais ils ne sont pas policés de la même façon. Par contre, on peut limiter le trafic *Scavenger* à destination des serveurs à des valeurs raisonnables afin qu'ils ne soient pas affectés par du trafic viral.

Actions au niveau de l'accès WAN

L'accès WAN est une ressource critique, souvent fortement dégradée lors d'attaques. C'est aussi le point de contact entre deux domaines DS, celui de l'opérateur et celui du campus. Il faut donc mettre en place des règles de remarquage et de policing. La plupart des sites universitaires sont raccordés à des réseaux régionaux ou métropolitains, chacun d'eux ayant son offre de Services Différenciés. Pour l'exemple, utilisons les choix de Renater qui offre un modèle pour l'offre des réseaux régionaux. Le document Classes de Services de RENATER [14] liste l'offre disponible pour les ayant droits :

- Less than Best Effort, DSCP LE (8), profil non limité ;
- Best Effort, DSCP BE (0), profil non limité ;
- Better than Best Effort, DSCP AF41 (34) profil limité à 10% de l'agrément.

La téléphonie, la signalisation et les flux OAM n'ont pas vocation à sortir du campus. On voit dans la figure 5 ce que pourraient être les règles de remarquage pour les autres flux.

Flux	**Campus**	**RENATER**
Visioconférence	AF41	AF41
Prioritaires	AF21	AF41
BE	CS0	CS0
LE	CS1	CS1

Figure 5 - *Exemple de règles de remarquages*

Nous allons détailler le remarquage dans le sens Renater vers Campus, nous ne détaillerons pas l'autre sens, qui est similaire. Nous devons programmer une règle de remarquage pour les paquets AF41, en mettant en place un marquage en fonction de la destination, en effet c'est le seul moyen de sélectionner les flux de visioconférence dans les flux prioritaires et de leur appliquer un marquage différent. Nous devons aussi faire du policing afin de limiter les flux à des valeurs raisonnables. Disons que nous programmons 10 Mbits/s pour la totalité des visioconférences et 2 Mbits/s maximum pour chacune. Pour les flux prioritaires, nous choisissons 10 Mbits/s au total et 5 Mbits/s par flux. On remarque LE les paquets qui dépassent légèrement la limite et on détruit les paquets qui la dépassent trop. On ne détaille pas les ACL qui correspondent aux destinations. Voici un exemple de *class-map* pour les flux à destination des équipements de visioconférence marqués avec le PHB AF41 :

```
class match-all vers-visio
 match dscp af41
 match access-group name vers-visio
```

On définit la politique en entrée de RENATER. Chaque ligne *police x y z...* configure un srTCM :

```
policy-map deRENATER
 class vers-visio
  police 10000000 810000 1280000
       conform-action set-dscp-transmit af41
       exceed-action set-dscp-transmit cs1
       violate-action drop
  police flow 2000000 200000 conform-action
       set-dscp-transmit af41
       exceed-action drop
 class vers-prioritaire
  police 10000000 810000 1280000
       conform-action set-dscp-transmit af21
       exceed-action set-dscp-transmit cs1
       violate-action drop
  police flow 5000000 400000 conform-action
       set-dscp-transmit af21
       exceed-action drop
 class bestEffort
  set dscp default
 class class-default
  set dscp cs1

interface FastEthernet1
 service-policy input deRENATER
```

Il ne serait pas déraisonnable de positionner des profils de trafic, dans la politique ci-dessus, pour les trafics LE et BE et d'intercepter les paquets marqués avec le DSCP AF41 qui ne seraient pas réellement destinés à des applications prioritaires.

Dans cet exemple, nous avons donc implémenté 7 *Per Domain Behavior* différents permettant de répondre aux demandes de haute disponibilité. Nous avons établi des règles de remarquage préservant ces flux au niveau de l'opérateur. Dans un cadre de fonctionnement normal, sans congestion, la configuration que nous avons mise en place aura pour seul effet d'assurer l'acheminement du trafic de

téléphonie avec une latence et une variation de délai minimum. Mais dans le cas d'un stress ou d'une agression, comme une attaque virale, le trafic indésirable sera marqué dans la classe Scavenger et il impactera donc moins le trafic légitime.

5 XMLQoS

Depuis quelques années déjà, nous avons décidé d'implémenter une configuration de QoS sur le routeur C6500 chargé du routage interne. Cette configuration décrit très précisément une matrice des débits autorisés pour les flux entre chaque réseau et de débit maximum par réseau. Mais la maintenance au quotidien de cette configuration est vite devenue ardue.

Pour comprendre le problème, admettons par exemple que nous ayons 3 réseaux reliés à l'Internet A = 10.1.0.0/16, B = 10.2.0.0/16 et C = 10.3.0.0/16. On veut mettre en place du *policing* pour les flux entre A et B à une valeur X, pour les flux qui vont de A vers l'Internet à une valeur Y et pour les flux qui vont de B vers l'Internet à une valeur Z. On ne veut pas policer les flux qui vont vers C.

Comment décrit-on l'Internet ? Ce n'est ni A, ni B, ni C.

Voilà la configuration qui résulte de cet exemple simple :

```
ip access-list extended from_A_to_any
 deny ip any 10.2.0.0 0.0.255.255
 deny ip any 10.3.0.0 0.0.255.255
 permit ip 10.1.0.0 0.0.255.255 any

class-map match-any from_A_to_any
 match access-group name from_A_to_any

ip access-list extended from_A_to_B
 permit ip 10.1.0.0 0.0.255.255 10.2.0.0
       0.0.255.255

class-map match-any from_A_to_B
 match access-group name from_A_to_B

policy-map policy_A_out
 class from_A_to_any
  police flow Y y conform-action transmit
       exceed-action drop
 class from_A_to_B
  police flow X x conform-action transmit
       exceed-action drop

interface interfaceA
 service-policy input policy_A_out

ip access-list extended from_B_to_any
 deny ip any 10.1.0.0 0.0.255.255
 deny ip any 10.3.0.0 0.0.255.255
 permit ip 10.2.0.0 0.0.255.255 any

class-map match-any from_B_to_any
 match access-group name from_B_to_any

ip access-list extended from_B_to_A
 permit ip 10.2.0.0 0.0.255.255 10.1.0.0
       0.0.255.255

class-map match-any from_B_to_A
 match access-group name from_B_to_A

policy-map policy_B_out
 class from_B_to_any
  police flow Z z conform-action transmit
       exceed-action drop
 class from_B_to_A
  police flow X x conform-action transmit
       exceed-action drop

interface interfaceB
 service-policy input policy_B_out
```

On voit que pour cet exemple très simple, la configuration est déjà longue et que la combinatoire allonge les ACL. C'est très rapidement difficile à maintenir lorsque le nombre de réseaux croît. Chaque modification exige de repasser sur toutes les ACL et de chercher son impact avec les risques d'erreurs qui en découlent.

Nous avons donc décidé de rédiger nos règles simplement, via un *abstraction layer* basé sur XML. Nous utilisons une application pour générer le code Cisco final. L'exemple précédent se rédige dans notre langage de la façon suivante :

```
<config>
 <networks>
  <network name="A">
   <interface name="interfaceA"/>
   <is>10.1.0.0 0.0.255.255</is>
  </network>
  <network name="B">
   <interface name="interfaceB"/>
   <is>10.2.0.0 0.0.255.255</is>
  </network>
  <network name="C">
   <interface name="interfaceC"/>
   <is>10.3.0.0 0.0.255.255</is>
   <noqos opt="yes"/>
  </network>
 </networks>
 <flowmatrix>
  <flow source="A" destination="B" bw="X"
       burst="x"/>
  <flow source="A" destination="any" bw="Y"
       burst="y"/>
  <flow source="B" destination="A" bw="X"
       burst="x"/>
  <flow source="B" destination="any" bw="Z"
       burst="z"/>
 </flowmatrix>
</config>
```

Au-delà de la différence de lisibilité évidente entre les deux codes, la figure 6 montre une comparaison du nombre de lignes de code IOS et XMLQoS nécessaires pour créer une configuration proche de celle ci-dessus, en augmentant régulièrement le nombre de réseaux (contrôle de la bande passante globale par VLAN en plus).

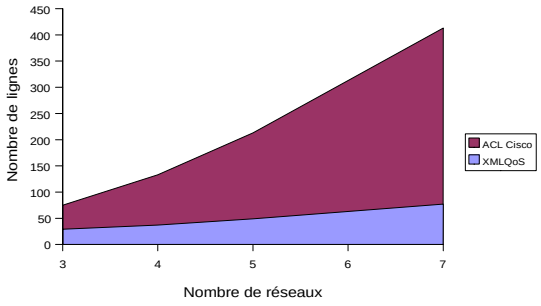

Figure 6 - *Nombre de lignes comparé XMLQoS/Cisco*

XMLQoS est composé d'un script PERL et d'une DTD XML. Nous utilisons la forge INRIA pour son développement et l'application est téléchargeable à l'adresse suivante : http://xmlqos.gforge.inria.fr/.

L'hébergement sur la forge rendant aisé le travail collaboratif, toute personne désirant contribuer à l'évolution de XMLQoS est la bienvenue.

6 Bilan et perspectives

L'utilisation de XMLQoS nous épargne énormément de temps et de travail de configuration. Elle nous permet d'éviter au quotidien des erreurs de manipulation. Cette syntaxe simplifiée permet de masquer la complexité de la configuration et facilite les modifications de celle-ci.

Le script actuel correspond à nos besoins. Toutefois le travail sur la syntaxe XML et le script peut encore être poursuivi :

- le script ne génère des *policy-maps* qu'en entrée sur les interfaces, pas en sortie ;
- il ne permet pas de manipuler le champ DSCP, ni de définir un srTCM ;
- il ne permet pas de prendre en compte plusieurs types d'équipements (en fonction de leurs possibilités).

La configuration en production sur le réseau de l'INRIA Sophia utilise une configuration de QoS matérielle de niveau 2 depuis maintenant 2 ans. Sur ces deux années d'exploitation, la téléphonie n'a jamais été impactée par un problème survenu sur le réseau et la qualité des communications est la même que celle qui existait sur l'infrastructure dédiée précédente. Lors de la phase de maquette, nous avons généré des flux UDP de 2 gbits/s vers des liens de 100 Mbits/s sans impact sur une communication téléphonique empruntant le même lien.

Nous avons utilisé les techniques de QoS au niveau 2 afin de rendre prioritaires certains VLANs par rapport à d'autres. Sur la période d'observation, aucun événement majeur n'a déclenché l'activation de la QoS. Nous envisageons une évolution de l'architecture de sécurité dont l'une des briques sera la mise en place d'une architecture proche de celle que nous venons de décrire.

L'utilisation de QoS au niveau 3 nous a rendu de précieux services en limitant les conséquences d'évènements accidentels, par exemple :

- la chute d'une des 3 implantations mondiales du W3C, celle hébergée au MIT et la réaffectation des connexions sur les 2 sites restants (INRIA Sophia et Université de Keio), la limitation programmée de la bande passante à la valeur négociée avec le consortium a évité que toute la bande passante de l'institut soit utilisée ;
- la limitation de la bande passante allouée à chaque visiteur et globalement à l'ensemble du réseau visiteur a évité des propagations rapides de virus ;
- la programmation par un doctorant d'un algorithme de gestion de processus distribués et le lancement d'un test avec des commandes wget visant le rapatriement de fichiers ISO aurait pu complètement utiliser la bande passante du site si les flux n'avaient pas été limités.

Pour chaque application visible de l'Internet, il est possible de mettre en place un profil de trafic normal. Le trafic légitime n'est pas impacté par ce profil et en cas d'attaque, le trafic abusif est limité à un seuil tolérable. Ce mécanisme vient efficacement compléter l'ouverture d'ACL et participe à la Politique de Sécurité du Système d'Information.

Bibliographie

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, *An Architecture for Differentiated Services*, RFC 2475, Décembre 1998

[2] D. Grossman, *New Terminology and Clarifications for Diffserv*, RFC 3260, Avril 2002

[3] K. Nichols, S. Blake, F. Baker, D. Black, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*, RFC 2474, Décembre 1998

[4] K. Nichols, B. Carpenter, *Definition of Differentiated Services Per Domain Behaviors and Rules for their Specification* , RFC 3086, Avril 2001

[5] R. Bless, K. Nichols, K. Wehrle, *A Lower Effort Per-Domain Behavior (PDB) for Differentiated Services*, RFC 3662, Décembre 2003

[6] Stanislav Shalunov, Benjamin Teitelbaum, *QBone Scavenger Service (QBSS) Definition*, Mars 2001

[7] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, *Assured Forwarding PHB Group*, RFC 2597, Juin 1999

[8] B. Davie, A. Charny, J.C.R. Bennet, K. Benson, J.Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, D. Stiliadis, *An Expedited Forwarding PHB (Per-Hop Behavior),* RFC 3246, Mars 2002

[9] J. Babiarz, K. Chan, F. Baker, *Configuration Guidelines for DiffServ Service Classes*, RFC 4594, Août 2006

[10] srTCM J. Heinanen, R. Guerin, *A Single Rate Three Color Marker* , RFC 2697, Septembre 1999

[11] Jerome H. Saltzer, *Slammer : An urgent wake-up call*, Février 2003

[12] IEEE, *802.1D, Media Access Control (MAC) Bridges*, Table G-2, p 244, http://standards.ieee.org/getieee802/download/802.1D-2004.pdf, Juin 2004

[13] Cisco Systems, *Deploying Control Plane Policing, White Paper*, http://www.cisco.com/application/pdf/en/us/guest/products/ps6642/c1244/cdccont_0900aecd804fa16a.pdf, 2005

[14] RENATER, *Classes de Services*, http://www.renater.fr/spip.php?article389, Juin 2007