

Résolution des problèmes de performances de bout en bout

muyal@renater.fr

andreu@renater.fr

Contributions

- ◆ FX Andreu, Renater
- ◆ S. Muyal, Renater
- ◆ B. Tuy
- ◆ C. Grenet
- ◆ ...

Agenda

- ◆ Introduction: Qu'est-ce qu'un problème de performance?
- ◆ Collecte des informations de base
- ◆ L'investigation pas à pas (Méthodologie)
- ◆ Travaux et projets sur le sujet et quelques exemples

Introduction: Qu'est qu'un problème de performance?

Définition

- ◆ Les pannes sont « facilement » repérées par les NOCs ou ASRs:
 - État binaire: marche/ne marche pas
 - Il existe un ensemble d'outils de supervision qui détectent et signalent automatiquement les pannes.
 - Il est souvent simple de connaître la cause de la panne: coupure électrique, coupure d'un lien, etc

- ◆ **Problème de performance:** Dégradation sans qu'il y ait coupure du service
 - ex: multimédia: voix saccadée
 - ex: Temps de réponse d'un serveur plus élevé que d'habitude

- ◆ Les problèmes de performances sont beaucoup plus difficile à résoudre qu'une panne:
 - Difficile à déceler par les outils basiques de supervision
 - Difficile de faire le lien entre les indicateurs applicatifs et les indicateurs réseau
 - Configuration préalable de seuils dans les outils de supervision en fonction des applications: long et pas évident.

Introduction: Qu'est qu'un problème de performance?

Définition

- ◆ Les problèmes de performances applicatives sont donc souvent décelés par l'utilisateur final:
 - Les dégradations sont perçues au niveau applicatif, donc par l'utilisateur final.
 - C'est l'utilisateur qui signale dans la plupart des cas à son administrateur système et réseau (ASR) les problèmes de performances.

- ◆ Une fois le problème de performance signalé, difficile de savoir
 - **où** se trouve le problème
 - **la cause/élément** qui provoque cette dégradation: On se renvoie la balle souvent!!

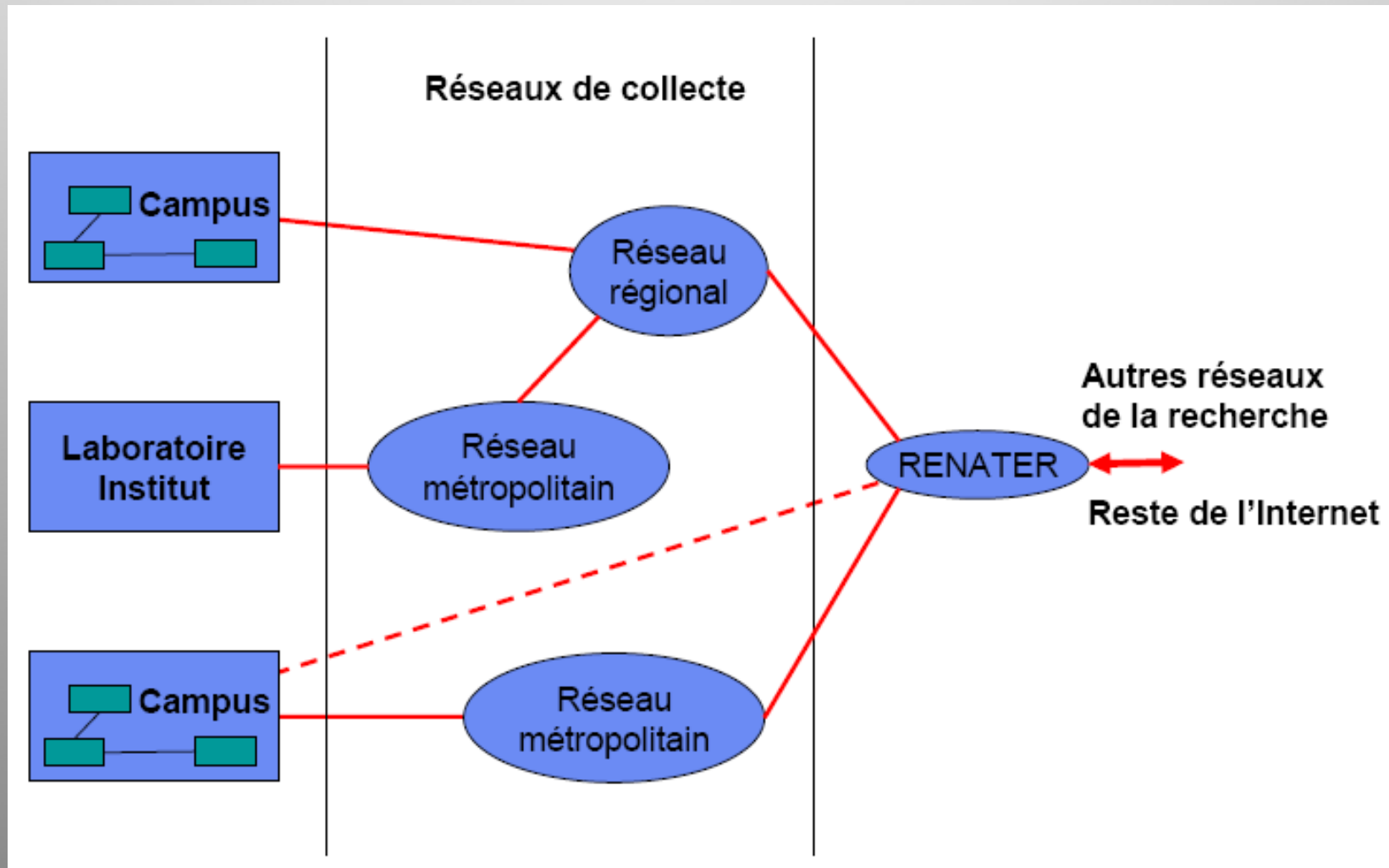
→ une investigation est souvent nécessaire

- ◆ Objectifs pour les ASR:
 - Être proactif
 - Déceler les problèmes de performances avant l'utilisateur final (probablement une investigation sera tout de même nécessaire pour résoudre le problème)
 - Pendant la résolution du problème de performance, les utilisateurs pourront être informés que le service est dégradé

Introduction:

- ◆ Dans un problème de performance, un ensemble d'éléments peut être concerné
 - Les extrémités
 - Utilisateurs
 - Applications
 - Systèmes d'exploitation
 - Matériel: carte réseau, autres (disques durs, carte mère, etc)
 - Le réseau local
 - LAN, VLANs, WLANs
 - Les réseaux
 - de campus
 - de collecte
 - Nationaux, expl : RENATER et autres réseaux de la recherche
 - Internationaux : GEANT2, OpenTransit, points d'échanges
 - autres opérateurs

Introduction: Qu'est qu'un problème de performance?: Les réseaux pouvant être impliqués



Introduction: Qu'est qu'un problème de performance? Perception de l'utilisateur

- ◆ Les performances perçues par l'utilisateur sont différentes des performances réseau:
 - Application multimédia: + sensible au temps de réponse
 - Application transfert: + sensible au débit
 - Combinaison de ces critères
- ◆ Temps de réponse applicatif
 - Le temps de réponse applicatif \neq temps de réponse réseau
- ◆ Disponibilité/Fiabilité d'un service \neq Disponibilité réseau
- ◆ Applications multimédia: Le MOS (Mean Opinion Score): permet de qualifier le flux audio:

| Mean Opinion Score (MOS) | | |
|--------------------------|---------------|------------------------------|
| MOS | Qualité | Dégradation |
| 5 | Excellente | Imperceptible |
| 4 | Bonne | Perceptible mais pas gênante |
| 3 | Moyenne | Légèrement gênante |
| 2 | Mauvaise | Gênante |
| 1 | Très Mauvaise | Très gênante |

Introduction: Qu'est-ce qu'un problème de performance?

◆ Démarche pour résoudre un problème de performance par un ASR:

- Collecte d'information
 - Auprès de l'utilisateur final
 - Sur son propre réseau (celui de l'ASR)
 - Information mise à disposition par les autres réseaux

- Vérifier qu'il s'agit d'un « vrai » problème réseau
 - Problème autre qu'applicatif (Base de données, CPU/mémoire de la machine, etc)

- Investigation/tests :
 - Faire des tests de bout en bout avec les machines impliquées
 - Faire des tests à partir de systèmes dédiés
 - Suivre le protocole de test décrit dans cette présentation
 - Établir un diagnostic

Agenda

◆ Méthodologie :

– 1 - Collecte des informations de base

- Description du problème: collecte au près de l'utilisateur
- Connaître les sources d'information disponibles
- Comprendre/Identifier le problème : une description pas à pas

– 2 – Tests à effectuer (démarche + Présentation d'outils)

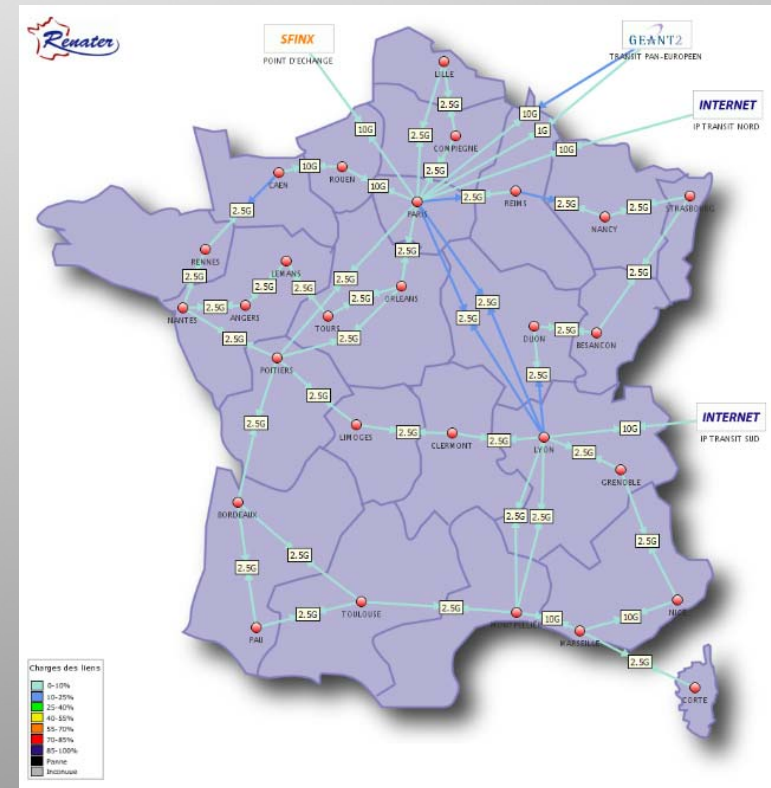
◆ Travaux et projets sur le sujet et études de cas

1. Collecte des informations de base Description du problème par l'utilisateur final

- ◆ L'ASR est contacté par l'utilisateur final généralement
- ◆ L'ASR doit collecter un maximum d'informations:
 - Fonctionnement de l'application sans dégradation (fonctionnement normal)
 - Valeurs de référence pour les applications du SI
 - Fonctionnement observé avec dégradation
- ◆ L'ASR doit s'assurer que les données collectées sont correctes:
 - L'utilisateur n'est pas forcément un expert
 - Perceptions \neq en fonction des utilisateurs
 - Attention aux unités
- ◆ Canevas/Template de collecte d'informations:
 - Connaître le contexte (utilisateur, appli, systèmes, chemin)
 - Permet de relayer facilement le problème à d'autres réseaux

1. Collecte des informations de bases Les sources d'information 1/2

- ◆ Avant de commencer les tests, beaucoup d'informations sont déjà disponibles au niveau des réseaux traversés:
 - Weathermap (carte du réseau avec le taux d'utilisation des liens)
 - Diffusion des tickets d'incident et de maintenance
 - Looking glass : Permet d'exécuter certaines commandes sur les équipements réseau



1. Collecte des informations de bases
Les sources d'information 2/2

◆ Les réseaux régionaux, de campus, MANs de la communauté :

– le site gt-méto :

- http://gt-metro.grenet.fr/index.php/Liste_des_r%C3%A9seaux_de_collecte

◆ Les réseaux nationaux/internationaux :

– <http://www.renater.fr>

– Liste des pages de diffusion d'information des NRENs :

- http://monstera.man.poznan.pl/wiki/index.php/Network_Weathermaps

– Autres réseaux (opérateurs privés, ISPs, GIXs) via :

- <http://stream.free.fr/index.html>

1. Collecte des informations de bases

Comprendre le problème : une description pas à pas

- ◆ Avec ces informations collectées, l'ASR essaie de déceler d'où peut provenir le problème:
 - Vérification qu'il s'agit d'un problème « réseau » et non pas d'un problème applicatif (vérification des logs applicatif et système, *restart* de l'application, *reboot* du système si nécessaire, tests du réseau local, tests du chemin complet).
 - Le plus souvent les informations collectées permettent d'écartier certaines pistes mais ne permettent pas de trouver l'origine du problème.
 - Des tests additionnels sont nécessaires...

2. Les étapes dans l'analyse d'un problème complexe de performance


- ◆ La première étude du problème n'a pas suffi, il faut faire une étude plus poussée. Il est indispensable dans cette partie d'avoir au préalable installé et étudié les outils standards de supervision et monitoring.
- ◆ La méthode :
 - Vérification approfondie :
 - Sur le(s) système(s)
 - Sur le(s) réseau(x)
 - Tester :
 - Utiliser le(s) poste(s) impacté(s) ou au plus proche
 - Utiliser des systèmes correctement paramétrés : CPU et réseau disponible
 - Utiliser les bons outils et dans le bon ordre

1. Vérification des informations aux extrémités

- ◆ Le type d'application/logiciel et son mode de fonctionnement
- ◆ Les piles protocolaires employées (RTP, TCP, UDP, IPv4, IPv6, etc)
- ◆ Les systèmes d'exploitation (version, noyau, etc.)
- ◆ Les adresses IP et les ports de niveau 4 des extrémités
- ◆ Le type de cartes réseau, processeur, mémoire vive

Vérification des informations du système: Linux

- ◆ Vérification des paramètres de la pile TCP dans le cas de problèmes de transfert longue distance et/ou très haut débit :

- Rappel « TCP tuning background » : <http://dsd.lbl.gov/TCP-tuning/background.html>
- Paramètres principaux : 
- Possibilité de « tuner » le code de l'application : <http://dsd.lbl.gov/TCP-tuning/setsockopt.html>


```
Dans /proc/sys/... :  
net.core.rmem_max = 16777216  
net.core.wmem_max = 16777216  
net.ipv4.tcp_rmem = 4096 87380 16777216  
net.ipv4.tcp_wmem = 4096 65536 16777216
```

- ◆ Vérification du mode de communication de la carte réseau :

- Utilisation du logiciel ethtool, exemple : 

```
[root@ORSAY root]# ethtool eth0  
Settings for eth0:  
Supported ports: [ TP ]  
Supported link modes: 10baseT/Half 10baseT/Full  
100baseT/Half 100baseT/Full  
Supports auto-negotiation: Yes  
Advertised link modes: 1000baseT/Full  
Advertised auto-negotiation: No  
Speed: 100Mb/s  
Duplex: Full  
Port: Twisted Pair  
PHYAD: 1  
Transceiver: internal  
Auto-negotiation: on  
Supports Wake-on: puag  
Wake-on: g  
Current message level: 0x00000002 (2)  
Link detected: yes
```

- ◆ Vérification de la MTU :

- exemple de decouverte de la MTU sur le chemin avec
tracpath (ou autre) 

```
tracpath 193.51.184.89  
1?: [LOCALHOST] pmtu 1500  
1: nri-a-gi1-0-0-140.cssi.renater.fr.183.51.193.in-addr.arpa (193.51.183.186) 0.447ms  
2: nri-b-g14-0-0-101.cssi.renater.fr (193.51.187.18) asymm 9 7.893ms  
3: reims-pos2-0.cssi.renater.fr (193.51.179.150) asymm 8 8.094ms  
4: nancy-pos1-0.cssi.renater.fr (193.51.179.138) asymm 7 7.898ms  
5: strasbourg-pos2-0.cssi.renater.fr (193.51.180.42) 7.903ms  
6: sonde-gip-strasbourg.cssi.renater.fr (193.51.184.89) 8.222ms reached  
Resume: pmtu 1500 hops 6 back 6
```

Vérification des informations du système: Windows XP/2000

◆ Windows XP/2000:

- Support de la RFC1323: Large window (rcv_window codé sur 30bits ~ 1Go au lieu de 64K dans Windows NT)
- Modification des valeurs de:
 - "Tcp Receive Window" =BDP (ex: 4 000 000 en octets)
 - "Window Scaling" activé
 - "Selective Acks" activé
 - "Time Stamping" activé
 - Le plus simple est d'utiliser des programmes vous permettant de modifier les valeurs prédéfinies par Windows: [DrTCP](#), [SG TCP Optimizer](#) ou [Cablenut](#)

```
# turn on window scale and timestamp option
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Tcp1323Opts=3
# set default TCP receive window size
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\TcpWindowSize=256000
# set max TCP send/receive window sizes (max you can set using setsockopt call)
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\GlobalMaxTcpWindowSize=16777216
```

Vérification des informations du système: Windows Vista

- ◆ Support de l'auto-tuning:
 - Régulation automatique des fenêtres TCP
 - Jusqu'à 16 MB maximum pour la fenêtre de réception (receive window)
 - Possibilité d'activer l'auto-tuning via l'interface netsh

```
netsh interface tcp set global autotuninglevel=disabled  
netsh interface tcp set global autotuninglevel=normal
```

- Pas de moyen d'ajuster le default TCP buffer (64 KB).
- L'algorithme d'autotuning n'est pas utilisé si le RTT est inférieur à 1 ms
 - Il peut donc y avoir un impact sur le LANso single streamTCP will be throttled on a LAN by this small default TCP buffer.

2. Les étapes dans le débogage

Rappel sur les paramètres systèmes (1/2)

- ◆ Même si la plupart du temps les systèmes d'exploitation (OS) sont correctement paramétrés par défaut, il est parfois nécessaire de « tuner » son OS pour obtenir de meilleures performances.
- ◆ Rappel : Sur un réseau haut débit et sur de longues distances, même la perte d'un seul paquet peut gravement affecter les performances. Pour TCP, le **débit maximum** est :

$$\text{Rate} < \frac{\text{MSS}}{\text{RTT}} * \frac{1}{\sqrt{p}}$$

MSS = Maximum Segment Size

RTT = Round Trip Time

P = Packet Loss probability

From Matt Mathis document

- ◆ Exemple de débit obtenu en fonction de l'implémentation de TCP et du RTT :

| Path | Linux 2.4 Un-tuned | Linux 2.4 Hand-tuned | Linux 2.6 with BIC | Linux 2.6, no BIC |
|----------------------------|-----------------------|-------------------------|-----------------------|----------------------|
| LBL to ORNL RTT = 67 ms | 10 Mbps | 300 Mbps | 700 Mbps | 500 Mbps |
| LBL to PSC RTT = 83 ms | 8 Mbps | 300 Mbps | 830 Mbps | 625 Mbps |
| LBL to IE RTT = 153 ms | 4 Mbps | 70 Mbps | 560 Mbps | 140 Mbps |

- Results = Peak Speed during 3 minute test
- Must increase default max TCP send/receive buffers
- Sending host = 2.8 GHz Intel Xeon with Intel e1000 NIC

2. Les étapes dans le débogage

Les Tests

- ◆ Lors d'un problème de performance réseau, le protocole de tests à suivre est le suivant :
 - Test de connectivité :
 - Pb : s'assurer que le poste distant est bien connecté au réseau et correctement configuré (adresse IP, masque, filtre ICMP, mode duplex...)
 - Ok : passer au point suivant
 - Test du chemin :
 - Pb : généralement un problème sur le chemin peut être une route asymétrique ou un problème de MTU
 - Ok : passer au point suivant
 - Test de la bande passante disponible (avec recherche de pertes) :
 - En UDP :
 - Pb : par exemple, un débit plus faible que celui attendu, c'est un problème de congestion
 - Ok : passer au point suivant
 - En TCP :
 - Pb : si le test UDP est correct cela provient de la configuration TCP d'un des hôtes.
 - Test applicatif ?? : Si possible

Vérification de la connectivité et du chemin :

◆ Ping :

- Utilisé pour tester la disponibilité d'un hôte
- Algorithme : utilisation du protocole ICMP
 - Envoi d'un paquet « echo request » avec:
 - Un identifiant par processus ping
 - Un numéro de séquence par echo request
 - L'hôte cible répond avec un ICMP echo reply
 - *Le résultat* : RTT, TTL, et numéro de séquence.

◆ Traceroute :

- Utilisé pour découvrir le chemin jusqu'à la destination.
- Algorithme : utilisation de ICMP ou UDP et le champ TTL de l'en-tête IP
 - Envoi d'un paquet avec TTL=1
 - Le premier routeur renvoi un ICMP time exceeded.
 - Envoi d'un paquet avec le TTL à 2 vers la cible, le deuxième routeur répond.
 - On continue jusqu'à la destination en incrémentant le TTL ou jusqu'au TTL max.

◆ Inconvénient des deux méthodes lorsque :

- Les chemins aller et retour sont asymétriques
- l'ICMP est filtré
- Les paquets ICMP ne sont pas traités par les routeurs comme les autres paquets

2. Les étapes dans le débogage

Vérification de la bande passante

- ◆ Plusieurs méthodes de mesure de la bande passante
- ◆ Outils qui estiment la bande passante théorique (taille des liens)
 - *pchar* ou *pathchar*
 - Ils permettent d’obtenir la taille du lien entre deux équipements de routage et donnent ainsi une estimation de la bande passante théorique saut par saut. Ils **sont très peu intrusifs** mais ne sont pas toujours suffisants pour résoudre des problèmes de performances haut débit (mode solo).
 - Ces outils sont parfois peu fiables mais fournissent un indice de confiance.
- ◆ Outils qui mesurent la bande passante disponible entre deux hôtes au niveau UDP ou TCP en générant du trafic jusqu’à saturation du plus petit des liens traversés.
 - *Iperf*
 - Ces outils sont très intrusifs mais ils permettent d’obtenir une meilleure mesure de la bande passante disponible pour une application entre deux hôtes (mode client-serveur).

2. Les étapes dans le débogage

Vérification de la bande passante disponible – exemple

◆ *La démarche suivie*

- Après vérification des systèmes d'exploitation (avec *tuning* des paramètres TCP si besoin), de la connectivité et du chemin emprunté, nous allons déterminer le débit maximum que nous allons atteindre en UDP et en TCP (avec un flux).
- La première phase consiste en un test *Iperf* UDP avec le paramètre *-b* en indiquant le débit souhaité, par exemple pour un débit de 90Mb/s mettez sur la ligne de commande : *-b90M* . C'est lors de cette phase que le serveur *Iperf* indiquera s'il y a du déséquilibrage et/ou des pertes de paquets pouvant être dus à une congestion. Si le débit est atteint sans aucune alerte, la partie réseau empruntée est capable de fournir un tel débit lors d'un test en TCP si les systèmes sont correctement paramétrés.
- Lors de la deuxième phase, nous vérifions que nous atteignons le même débit en TCP. Si le débit n'est pas atteint et que le test UDP est valide, il est fortement conseillé de vérifier les paramètres systèmes des hôtes (charges, *tuning* TCP). C'est à cette deuxième phase que les traces suivantes correspondent.

2. Les étapes dans le débogage

Vérification de la bande passante disponible

◆ Traces (serveur puis client):

```
[root@LYON netwarrior]# ./iperf -s -i 5
```

```
-----  
Server listening on TCP port 5001  
TCP window size: 64.0 KByte (default)  
-----
```

```
[ 6] local 193.51.184.113 port 5001 connected with 193.51.183.225 port 39518  
[ ID] Interval   Transfer   Bandwidth  
[ 6] 0.0- 5.0 sec 52.1 MBytes 87.4 Mbits/sec  
[ 6] 5.0-10.0 sec 52.7 MBytes 88.4 Mbits/sec  
[ 6] 0.0-10.0 sec 105 MBytes 87.9 Mbits/sec
```

```
netflow1:/home/maintenance# ./iperf -c 193.51.184.113 -i 5
```

```
-----  
Client connecting to 193.51.184.113, TCP port 5001  
TCP window size: 16.0 KByte (default)  
-----
```

```
[ 3] local 193.51.183.225 port 39518 connected with 193.51.184.113 port 5001  
[ ID] Interval   Transfer   Bandwidth  
[ 3] 0.0- 5.0 sec 52.3 MBytes 87.8 Mbits/sec  
[ 3] 5.0-10.0 sec 52.6 MBytes 88.2 Mbits/sec  
[ 3] 0.0-10.0 sec 105 MBytes 87.8 Mbits/sec
```

2. Les étapes dans le débogage

Vérification de la bande passante

- ◆ Lorsqu'on ne dispose pas de serveur à l'autre extrémité, on peut utiliser des outils qui ne fonctionnent pas en mode Client/Serveur :
 - Ex Pchar :
 - *Burst* successifs saut par saut en faisant varier la taille des paquets: On déduit en récupérant le RTT le débit du lien.
 - Tests assez long car logiciel peut « intrusif »
 - Se base sur des messages ICMP: Problème si filtrage
 - $r2 = 0.991958$ → indice de confiance par lien.

```
pchar to 193.51.184.89 (193.51.184.89) using UDP/IPv4
Using raw socket input
Packet size increments from 32 to 1500 by 32
46 test(s) per repetition
32 repetition(s) per hop
0: 193.51.183.185 (netflow-nri-a.cssi.renater.fr)
  Partial loss: 0 / 1472 (0%)
  Partial char: rtt = 0.134549 ms, (b = 0.000090 ms/B), r2 = 0.991958
                stddev rtt = 0.000992, stddev b = 0.000001
  Partial queueing: avg = 0.000078 ms (864 bytes)
  Hop char: rtt = 0.134549 ms, bw = 88449.851910 Kbps
  Hop queueing: avg = 0.000078 ms (864 bytes)
1: 193.51.183.186 (nri-a-gi1-0-0-140.cssi.renater.fr.183.51.193.in-addr.arpa)
  Partial loss: 0 / 1472 (0%)
  Partial char: rtt = 7.637975 ms, (b = 0.000085 ms/B), r2 = 0.654697
                stddev rtt = 0.008588, stddev b = 0.000009
  Partial queueing: avg = 0.000143 ms (864 bytes)
  Hop char: rtt = 7.503426 ms, bw = --,--- Kbps
  Hop queueing: avg = 0.000064 ms (0 bytes)
2: 193.51.187.18 (nri-b-g14-0-0-101.cssi.renater.fr)
  Partial loss: 0 / 1472 (0%)
  Partial char: rtt = 7.643269 ms, (b = 0.000068 ms/B), r2 = 0.273070
                stddev rtt = 0.015285, stddev b = 0.000017
  Partial queueing: avg = 0.000155 ms (864 bytes)
  Hop char: rtt = 0.005294 ms, bw = --,--- Kbps
  Hop queueing: avg = 0.000012 ms (0 bytes)
....
5: 193.51.180.42 (strasbourg-pos2-0.cssi.renater.fr)
  Partial loss: 292 / 1472 (19%)
  Partial char: rtt = 7.625242 ms, (b = 0.000227 ms/B), r2 = 0.999402
                stddev rtt = 0.001030, stddev b = 0.000001
  Partial queueing: avg = 0.000033 ms (864 bytes)
  Hop char: rtt = 0.038397 ms, bw = 80400.838878 Kbps
  Hop queueing: avg = -0.000016 ms (0 bytes)
6: 193.51.184.89 (sonde-gip-strasbourg.cssi.renater.fr)
  Path length: 6 hops
  Path char: rtt = 7.625242 ms r2 = 0.999402
  Path bottleneck: 40644.270594 Kbps
  Path pipe: 38740 bytes
  Path queueing: average = 0.000033 ms (864 bytes)
  Start time: Fri Aug 10 10:48:41 2007
  End time: Fri Aug 10 11:38:53 2007
```

2. Les étapes dans le débogage

Analyse des paquets

- ◆ Analyse de flux TCP avec tcpdump/wireshark
 - Avantage de pouvoir faire des captures au niveau des deux extrémités (identification des pertes, de boitiers intermediaires qui remarquent certaines options, etc)
 - Graphiques avec les pertes, retransmission, numéro de séquences, etc

2. Les étapes dans le débogage

Description des problèmes possibles

- ◆ Problèmes de performances dus à des pertes de paquets (détection possible avec *Iperf* sur des flux UDP):
 - Vérifier sur le poste (*ifconfig*, *ethtool*)
 - Vérifier sur le réseau local (*weathermap*, connection sur les équipements)
 - Alerter le réseau de collecte (si les pertes ne sont pas identifiées aux endroits précédents ou si la perte est détectée par les *pchar-like* sur un réseau tiers) après vérification des tickets d'incidents et de maintenances dudit réseau s'ils sont publics.

- ◆ Problèmes de performance sans perte de paquets, sans déséquilibrage, avec un débit UDP acceptable → il y a de très forte chance que ce soit un problème de configuration du système

Que faire :

- ◆ Un problème de performance de bout en bout a été détecté mais n'est pas résolu : que faire ?
 - Alerter le réseau de collecte qui analysera le problème et pourra éventuellement le faire « remonter » ...

Quelques exemples de problèmes de performances: LAN

◆ Half/Full duplex et Auto-négociation

- *Mismatch* entre deux équipements
- Recommandation
 - Pour un parc homogène et récent: Activer l'auto-négociation
 - Pour un parc hétérogène ou vétuste: Configuration manuelle

◆ Collisions:

- Utilisation de Hubs
- Recommandation:
 - Dans la mesure du possible faire évoluer son parc vers un réseau commuté

Quelques exemples de problèmes de performances: WLAN

- ◆ Performance avec du WIFI bien moindre et sensible à beaucoup plus de paramètres aléatoires:
 - CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) permet d'éviter les collisions.
 - Mais le média est tout de même partagé par l'ensemble des utilisateurs

Quelques exemples de problèmes de performances: LAN/Campus

- ◆ Problème de performance au niveau du LAN: Boîtiers intermédiaires
 - Pare-feux
 - Boîtiers de performances: Lisseur de trafic, etc
 - Boîtiers NAT
 - Proxies/mandataires
 - VPNs
- ◆ Ces boîtiers ajoutent un temps de traitement qui peut entraîner des ralentissements, surtout s'ils ne sont pas traités par le matériel

3. Les Travaux et projets sur le sujet

- ◆ Gt-metro
 - <http://gt-metro.grenet.fr/>

- ◆ PERT: Performance Enhancement Response Team
 - Structure au niveau européen pour traiter les problèmes de performance
 - Pert-KB (base de connaissance du “Performance Enhancement Response Team” – PERT): <http://kb.pert.geant2.net/PERTKB/>

- ◆ Perfsonar
 - Mesures de performance interdomaine: <http://www.perfsonar.net/>

- ◆ Liens spécifiques sur comment faire des transferts haut-débit et/ou longue distance :
 - <http://dsd.lbl.gov/TCP-tuning/links.html>
 - <http://dsd.lbl.gov/TCP-tuning/TCP-Tuning-Tutorial.pdf>
 - <http://2007.jres.org/planning/slides/63.pdf>

Questions??