Outils utilisés dans TPs réseau

Ce document est un manuel d'utilisation de tous les outils qui sont employés dans les TPs. Simples scripts écrits en Korn-Shell ou utilitaires plus complets écrits en C, ils sont tous accessibles par le répertoire **/root/bin** des stations.
Les outils sont présentés par ordre alphabétique.

Wireshark : capture et analyse de trames sur un réseau

Wireshark permet de capturer et d'analyser les trames véhiculées sur les réseaux sur lesquels les stations sont connectées (en l'occurrence, des réseaux Ethernet).
Une capture peut être lancée en cliquant le bouton **start** du menu capture. Une nouvelle fenêtre apparaît permettant de spécifier des paramètres de la capture :

- Choix de l'interface Ethernet
- Filtre à appliquer au moment de la capture (protocole, nombre de paquet à capturer ...)
  Voici quelques exemples de filtres de capture :
  - dest ether 08.00.08.15.ca.fe
    Adresse destination ethernet
  - src host 192.168.0.10
    Adresse source IP
  - host 192.168.0.10 and not tcp port 80
    Adresse source ou destination IP et port TCP différent de 80

La sauvegarde d'une capture peut se faire sous deux formats :
- Simple fichier texte : Option **print** du menu File.
- Fichier au format Wireshark (lisible seulement depuis Wireshark) : Option save du menu File.

---

check-route : contrôle des tables de routage

check-route affiche toutes les 5 secondes le contenu des tables de routage d'une station. Pour interrompre **check-route**, il suffit de taper sur **Ctrl-C**.

---

udpmt et tcpmt : mesure de débit utile sur un réseau

udpmt et **tcpmt** permettent de calculer le débit au niveau applicatif d'une communication entre deux machines soit en utilisant le protocole UDP ou TCP.
L'utilitaire **udptarget** (respectivement pour **tcpmnt tcptarget**) doit être lancé auparavant sur la machine destination.

Syntaxe :
udpmt [-V] [-h] [-p port] [-s taille_paquet] [-n nb_paquet] [serveur]
tcpmt [-h] [-p port] [-s taille_paquet] [-n nb_paquet] [serveur]

tcptarget [-p port] ou udptarget [-p port]

- **-V** : mode verbose. Dans ce mode **udpmt** affiche (en mode verbose **-V**) un rapport regroupant différentes statistiques :
  - l'instant d'émission (en ms depuis 0h)
  - le nombre de paquets émis durant l'intervalle
  - le nombre totale de paquets émis
  - le débit mesuré durant l'intervalle, en kbit/s
  - le débit moyen sur les 10 dernieres secondes
  - le débit moyen depuis le début
- **-n nb_paquet** : donne le nombre de paquet du test par defaut, udpmt ne s'arrête pas
- **-s taille_paquet** : indique la taille des paquets (données de niveau application) à émettre (1472 octets par défaut)
- **-p port** : permet de changer le numéro de port par défaut sur lequel le serveur attend (Par exemple pour générer plusieurs trafics vers une même machine).

udptarget affiche les débits constatés en réception.

pong : interception de paquets ICMP Echo Request

pong est un utilitaire qui permet de signaler toute réception de paquets ICMP de type Echo request. Ces paquets sont habituellement générés par l'utilitaire standard **ping** (la station qui en reçoit doit normalement retourner un paquet ICMP de type Echo reply à l'émetteur).
A chaque fois qu'un tel paquet est reçu, **pong** le signale par le message suivant : *Echo request from....*
Ce message précise le nom ou l'adresse de la machine d'où vient le paquet, le numéro du processus (**pid**) qui l'a envoyé, et l'adresse de la machine à qui il est destiné (normalement, c'est une des adresses de la station elle-même).

Pour arrêter le traitement de **pong**, il suffit de faire Ctrl C

send_icmp : Emission de paquets ICMP

send_icmp permet de construire et d'envoyer des paquets de type ICMP sur un réseau à destination d'une machine donnée.

Les paquets ICMP envoyés sont construits à partir du contenu de fichiers ASCII, dans lequel ils sont décrits octets par octets. Ces octets sont habituellement écrits sous format décimal, mais vous pouvez les écrire en hexadécimal en précisant le caractère **x** avant chaque valeur.
Les octets doivent être séparés par un ou plusieurs *espaces*, *tabulations* ou *retour à la ligne*. Voici un exemple syntaxiquement correct de ce que **send_icmp** s'attend à analyser :

```
   23 xA0     100     244     xff
1 2 3 5 0 98
```

send_icmp ne peut envoyer qu'un seul paquet ICMP à la fois. Il a deux paramètres : le nom ou l'adresse de la station à qui doit être envoyé le paquet, et le nom du fichier dans lequel est stocké le paquet en question.

Dans les TPs, **send_icmp** n'est utilisé que pour envoyer des paquets ICMP de type Echo request. Le champ checksum doit être initialement nul (le checksum est situé dans le troisième et le quatrième octet du paquet). C'est **send_icmp** qui remplit automatiquement ce champ en fonction des valeurs des autres octets du paquet.