**Université de Strasbourg**
**UFR Mathématique et Informatique**

**Licence 3 Informatique**
**Année 2013/2014**

# Réseaux et Protocoles

## Projet : Outil d'analyse du réseau

## 1 Introduction

Qu'ils soient dues à des pannes, à des opérations de maintenances ou à la volonté de l'opérateur d'améliorer les performances de son réseau, les changements du graphe de l'Internet sont des évènements fréquents. De tels changements induisent généralement des perturbations temporaires du trafic et peuvent mener à l'utilisation de nouvelles routes entre certains points du réseau. Dans le cadre de ce projet, vous devrez implémenter, en langage C ou C++, un outil d'analyse de route dans réseau IP, permettant de découvrir les routes utilisées et de tracer leurs éventuelles modifications.

## 2 Fonctionnement de base

Votre programme devra prendre en paramètre un nom de domaine ou une adresse IP (v4 ou v6) destination et surveiller les variations sur la route entre votre machine et la destination. Dans un premier temps, vous allez devoir implémenter, au moyen de la librairie `socket` vue en TP, les fonctionnalités de *traceroute* et de *ping* décrites ci-dessous comme deux branches indépendantes. Vous devrez ensuite les intégrer au programme principal de sorte que ce dernier se comporte comme suit ($\approx$ 5 points) :

- Lors de son exécution, votre programme devra s'assurer, au moyen d'un message ICMP request, que l'équipement distant est effectivement joignable, puis découvrir la route empruntée pour l'atteindre (*traceroute*).
- Par défaut, une sonde ICMP request devra ensuite être lancé à intervalle régulier (toutes les secondes par exemple – votre programme doit être le plus "paramétrable" possible), afin de vérifier l'intégrité de la route vers la destination. Les informations de délai et TTL, ainsi que les éventuelles erreurs, devront être extraites de ces mesures et stockées dans un fichier de log. En cas de changement de TTL et/ou, au delà d'un certain seuil de variations de délai, une nouvelle découverte de route sera à réaliser.
- Lors de l'interruption de l'exécution (sur signal de type *SIGINT* par exemple) un résumé statistique et/ou graphique de l'ensemble des informations collectées (routes avec changements éventuels, valeurs de délai, pertes, ...) devra être présenté à l'utilisateur.

Enfin, la dernière partie vous permettra d'estimer l'asyméytrie entre vos routes aller et retour dans un environnement contrôlé. A vous de produire le plus d'informations possibles.

### 2.1 Traceroute ($\approx$ 5 points)

La commande `traceroute` (`tracert` sous Windows) est un outil de diagnostic réseau permettant d'estimer la route empruntée par des paquets sur un réseau IP. Le principe de base de ce programme repose sur la modification du champ *Time-to-Live* (TTL) de l'entête IP. En effet, lorsque la valeur contenue dans ce champ, décrémentée de $1$ à chaque routeur que le paquet traverse, arrive à $0$, le routeur jette le paquet mais envoie également à l'expéditeur un message d'erreur (de type *TTL exceeded*), pour l'en informer. Ainsi, en envoyant un paquet IP avec une valeur de TTL de $1$, on recevra un message du premier routeur traversé, lequel nous permettra de connaître l'adresse IP de ce routeur. Afin de découvrir l'ensemble des routeurs utilisés pour parvenir à la destination, il faudra donc envoyer successivement des paquets avec un TTL de $1, 2, 3 \ldots$ jusqu'à arriver à la destination. On peut ensuite utiliser des fonctionnalités de *Reverse-DNS* pour retrouver le nom des routeurs à partir de leur adresse IP. Votre implémentation devra supporter trois types de sondes (TCP, UDP et ICMP) et également proposer plusieurs modes de trace (un saut après l'autre ou pas, fréquence inter-sonde, nombre de tentatives, gestion des temporisateurs, etc). Vous pourrez aussi

comparer de type de trace indirect à l'utilisation d'une option comme `ROUTE_RECORD`.

Par exemple, voici ce que l'on obtient pour le site web de l'Université de Californie, Berkeley :

```
~ $ traceroute www.berkeley.edu
traceroute to www.w3.berkeley.edu (169.229.216.200), 64 hops max, 52 byte packets
 1  routeur-espla-rc1-130-79-91-253 (130.79.91.253)
 2  * * *
 3  te1-2-nancy-rtr-021.noc.renater.fr (193.51.189.85)
 4  te0-0-0-2-paris1-rtr-001.noc.renater.fr (193.51.189.161)
 5  renater-lb1.mx1.par.fr.geant.net (62.40.124.69)
 6  ae1.mx1.lon.uk.geant.net (62.40.98.76)
 7  ae0.mx1.ams.nl.geant.net (62.40.98.81)
 8  ae2.rt1.ams.nl.geant.net (62.40.98.114)
 9  tge-0-5-0-4.211.newy.layer3.nlr.net (216.24.184.85)
10  vlan-62.clev.layer2.nlr.net (216.24.186.66)
11  vlan-63.chic.layer2.nlr.net (216.24.186.60)
12  vlan-48.kans.layer2.nlr.net (216.24.186.63)
13  vlan-46.denv.layer2.nlr.net (216.24.186.70)
14  vlan-44.albu.layer2.nlr.net (216.24.186.48)
15  vlan-45.elpa.layer2.nlr.net (216.24.186.51)
16  vlan-43.losa.layer2.nlr.net (216.24.186.73)
...
21  t5-4.inr-210-srb.berkeley.edu (128.32.255.125)
```

En observant les noms de routeurs, on peut en déduire que, lorsque l'on consulte ce site web depuis Strasbourg, les paquets de données passent par Nancy, Paris, Londres, Amsterdam, New-York, Cleveland, Chicago, Kansas City, Denver, Albuquerque, El Paso et Los Angeles, avant d'atteindre enfin Berkeley.

## 2.2 Ping ($\approx$ 5 points)

La commande `ping` est un outil habituellement utilisé pour mesurer la connectivité entre deux équipements sur un réseau IP. Sous sa forme la plus simple, il utilise le protocole ICMP afin d'envoyer, vers une machine ou un routeur, un message de type *ECHO_REQUEST*. Si le réseau est opérationnel et que l'équipement destination est configuré pour supporter de telles requêtes, il répondra alors par un message *ECHO_REPLY*. Les pertes ainsi que la durée séparant l'envoi de la requête et la réception de la réponse, appelée *round-trip time* (RTT), sont enregistrées afin d'évaluer la qualité du canal de communication. Votre programme devra être capable d'envoyer des sondes ICMP, UDP et TCP et interpréter les différents type de messages reçus pour observer s'il existe ou non des différences significatives entre types de sondes.

Ainsi, en utilisant la commande `ping` sur le même serveur que précédemment, on obtient :

```
~ $ ping www.berkeley.edu
PING www.w3.berkeley.edu (169.229.216.200): 56 data bytes
64 bytes from 169.229.216.200: icmp_seq=0 ttl=40 time=197.524 ms
64 bytes from 169.229.216.200: icmp_seq=1 ttl=40 time=197.559 ms
64 bytes from 169.229.216.200: icmp_seq=2 ttl=40 time=197.695 ms
64 bytes from 169.229.216.200: icmp_seq=3 ttl=40 time=197.693 ms
64 bytes from 169.229.216.200: icmp_seq=4 ttl=40 time=197.539 ms
^C
--- www.w3.berkeley.edu ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 197.524/197.602/197.695/0.076 ms
```

Ces résultats indiquent que les paquets IP ont mis approximativement 200ms pour faire l'aller-retour Strasbourg - Berkeley et, sachant que la valeur de TTL initiale était de 64, que 24 routeurs ont été traversés (lors du trajet retour).

### 2.3 Contrôle des deux extrémités ($\approx$ 5 points)

Si vous disposez de deux machines, vous êtes capables de contrôler indépendamment les messages aller et retour et ainsi de proposer votre propre protocole de mesure. Vous pourrez par exemple construire facilement des ping entièrement UDP ou TCP. Dévellopez un programme vous permettant de produire des informations uni-directionnelles afin de mieux cerner l'évolution des routes empruntées entre les deux machines que vous contrôlez. Ce programme devra associer les résultats obtenus de part et d'autre afin de rassembler tous les résultats possibles.

## 3 Fonctionnalités supplémentaires (points bonus)

Les fonctionnalités décrites ci-dessous sont **optionnelles** et ne devront être abordées que lorsque la base de votre programme fonctionnera parfaitement.

### 3.1 Reverse-DNS

L'envoi répété de messages contenant une valeur de TTL de plus en plus grande permet de découvrir, au moyen des messages d'erreurs, les adresses IP des routeurs utilisés pour atteindre la destination. Cependant, des adresses seules sont peu utiles lorsque l'on cherche à tracer sur une carte la route empruntée. Heureusement, les noms d'hôte (*hostname*) des routeurs contiennent généralement des informations relatives à leur localisation géographique. Cette extension consiste à récupérer ces noms au moyen de requêtes DNS inversées, permettant de faire le lien entre une adresse IP et le nom d'hôte correspondant.

### 3.2 Paris-traceroute

Afin de mieux répartir la charge du trafic sur leur réseau, les opérateurs peuvent utiliser des techniques d'équilibrage de charge (*load balancing*). De tels mécanismes permettent aux flux de données d'utiliser différentes routes pour parvenir à une destination donnée, et ce même si elles proviennent de la même source. Bien que l'intérêt pour l'opérateur soit assez évident, ce phénomène peut sensiblement compliquer la découverte des routes. Pour cette extension, vous devrez vous baser sur les explications données sur le site `www.paris-traceroute.net` afin de rendre votre implémentation de *traceroute* capable de découvrir les routes alternatives et minimiser les erreurs d'un `traceroute` standard. Comment se comporte les mécanismes de répartition de flux selon le type de sonde et les éventuelles options utilisées ?

# 4 Rendu fonctions de base : le 20 décembre 2013

Ce projet est à réaliser en **groupe de deux à trois personnes**. Vous devrez rendre, sous forme d'une archive au nom des membres du groupe, à la fois le code source en langage C ou C++ ainsi qu'un rapport détaillant vos choix d'implémentation, les mécanismes utilisés ainsi que les fonctionnalités de vos programmes. Tout retard sera sanctionné mais vous avez jusqu'au 17 janvier pour nous rendre la partie portant sur les fonctionnalités supplémentaires.

## 4.1 Code

Votre programme devra impérativement respecter les règles suivantes :

- être rédigé selon les règles de bonne pratique en vigueur ;
- ne contenir aucune fuite de mémoire ;
- proposer un Makefile permettant de lancer la compilation automatiquement, en utilisant les options de compilations suivantes :
  ```
  -std=gnu99 -pedantic -Werror -W -Wall -Wextra -Wmissing-declarations
  -Wmissing-prototypes -Wredundant-decls -Wshadow -Wbad-function-cast -Wcast-qual
  ```
- être intégralement commenté (chaque fonction, argument et retour) selon la norme utilisée par *doxygen*.

## 4.2 Rapport

Le projet devra également être accompagné d'un court rapport (10 pages max) décrivant :

- la démarche adoptée et les difficultés techniques rencontrées ;
- les choix en termes protocolaires ;
- la répartition du travail au sein du groupe (fonctions implémentées, etc).

**Toute forme de plagiat sera sanctionnée.**