1 Observation : Configuration Réseau sous Linux

- Beaucoup de commandes nécessitent un compte administrateur!
- Observer le résultat des commandes : *"hostname", "echo $HOSTNAME", "uname -n"*
- Expliquer le résultat de la commande : *"lspci | grep -i net"*
- Expliquer le résultat de la commande : *"sudo lshw -C network"*
- Relever les configurations des interfaces réseau de la machine en observant le résultat de la commande précédente: *"ifconfig" ou "ip"*

2 Observation : Activité Réseau et critères d'observation

- Scruter les adresses IP présentes sur le réseau ainsi que les services proposés par ces stations à l'aide de la commande *"*nmap*"*
  - . *Utiliser l'option "*-V network/mask*", exemple :* nmap -v 172.16.150.0/24
  -

- Quelle est l'option de la commande *"tcpdump"* qui permet de lister les interfaces réseaux de la station: ….....
- Comment peut on observer toutes les trames circulant sur une interface tel que "eth0" : tcpdump ...
- Comment afficher les adresses numériques plutôt que des adresses symboliques (DNS): tcpdump ...

- Analyser les fichiers *"/etc/services"* et *"/etc/protocols"*
- Pour les questions suivantes, préciser le rôle principal d'un switch.

- Placer des critères permettant d'observer les trames réseaux correspondant :
  - au choix d'un protocole précis (UDP) : tcpdump ...
  - au choix d'un port de communication TCP précis (80) sur eth0 : tcpdump …
  - au choix d'un port de communication UDP précis (161) sur wlan0 : tcpdump …
  - au choix sur une adresse IP précise : tcpdump ...
  - au choix d'un arrêt après 20 paquets: tcpdump ...
  - à un envoi du résultat dans un fichier plutôt que l'écran: tcpdump ...
  - à l'affichage sur l'écran de la lecture du fichier créé au point précédant : tcpdump ...
  - au choix de paquets FTP venant de 192.168.nnn.x et allant vers 192.168.nnn.y : tcpdump ...
    - Tenter d'observer le login et mot de passe d'accès au serveur FTP!!!

- Créer un fichier *"eth0_http"* contenant des critères de capture : tcp and port http
- Capturer le trafic dont les critères sont stockés dans ce fichier *"eth0_http"* : tcpdump ...

Il est possible d'associer à tcpdump la commande "grep" sous Linux.

- Observer les trames en affichant leur contenu entier et en hexadécimal : tcpdump ...

  Sélectionner judicieusement des trames IP avec protocole TCP puis UDP, à récupérer dans un fichier.
  Analyser leur contenu en justifiant les zones de données de chaque protocole.
  On se contentera :
  - d'expliquer les champs principaux des couches IP, TCP, UDP et
  - de délimiter les données des couches supérieures.

3 Création d'une application couche n°5 "Session" : Manipulation de "Socket" en mode connecté TCP et non connecté UDP

Nous allons concevoir (en langage C) deux clients permettant de se connecter à des serveurs via des "sockets".
On les testera en se connecter sur des serveurs existants : "httpd" en TCP port 80, "snmpd" en UDP port 161.
Il est nécessaire, dans ce cas, de connaître précisément le protocole utilisé et surtout le format d'une requête de service du serveur choisi. N'ayant pas étudier ces protocoles (pour l'instant), vous respecterez à la lettre la constitution des requêtes qui vous aurez situées.

3.1 Exemple de la structure d'un client TCP/IP HTTP :

- Récupérer et compléter le code du *"TP Client HTTP à compléter"* sur le site http://philippe.free.fr .
  - Pour la compilation : `gcc clienthttp.c -o clienthttp`
  - On notera l'usage de la commande pour son exécution : `./clienthttp serveurHTTP ressource`
  - Pour le test, le résultat doit afficher le contenu d'une page "html" correspondant à la ressource.

3.2 Exemple de la structure d'un client UDP/IP SNMP :

- Récupérer et compléter le code du *"TP Client SNMP à compléter"* sur le site http://philippe.free.fr .
  - Pour la compilation : `gcc clientsnmp.c -o clientsnmp`
  - On notera l'usage de la commande pour son exécution : `./clientsnmp agentSNMP`
  - Pour le test le résultat doit afficher un contenu héxadecimal puis ASCII

Structure d'une trame TCP

<table>
<tr><th colspan="3">Bits 0</th><th colspan="12">15 16</th><th>31</th></tr>
<tr><td colspan="16">En-tête TCP située après l'en-tête IP</td></tr>
<tr><td colspan="8">Port source (16 bits)</td><td colspan="8">Port cible (16 bits)</td></tr>
<tr><td colspan="16">Numéro de séquence (32bits)</td></tr>
<tr><td colspan="16">Numéro d'acquittement (32bits)</td></tr>
<tr><td colspan="2">Offset(4b)</td><td colspan="2">Réservé(4b)</td><td>C</td><td>E</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td colspan="4">Taille ou longueur (16 bits)</td></tr>
<tr><td colspan="8">Somme de contrôle (16 bits)</td><td colspan="8">Pointeur urgent (16bits)</td></tr>
<tr><td colspan="16">Options/remplissage (?)</td></tr>
<tr><td colspan="16">Zone de données</td></tr>
</table>

- **Port source**: de l'application TCP qui a expédié la trame. Voir "/etc/services"
- **Port cible**: de l'application TCP qui doit traiter la trame. Voir "/etc/services"
- **Numéro de séquence** : numérote le premier octet de données du flux global. Généré d'une manière aléatoire, il désigne une trame doublon si le même numéro et sur 2 trames.
- **Numéro d'acquittement** :tous les paquets de données sont acquittés jusqu'au numéro indiqué moins un. Valable si l'indicateur ACK est actif (cf ci-dessous). Parfois, il n'est pas modifié à chaque paquet.
- Offset des données : longueur de l'en-tête TCP en blocs de 32bits. Vaut au minimum 5 et donc 20 octets.
- Réservé: Pour extensions futures. Vaut 0000
- URG : URGent pointer flag : désigne un paquet urgent et prioritaire. Utilisé lors de l'interruption de processus. Si ce bit est actif, alors le champ Urgent Pointer est utilisé (cf ci-dessous).
- ACK : si 1 alors le numéro d'acquittement est valide.
- PSH : PuSH flag :Si actif alors la couche TCP en réception doit immédiatement transmettre le segment à la couche application correspondante (ex application terminale). Sinon TCP tamponne les données.
- RST : ReSeT flag : indique qu'une erreur s'est produite. Soit échec ou interruption de connexion.
- SYN : SYNchronization flafg : utilisé dans un procédure three-way hadshake. Indique au destinataire qu'une liaison doit être établie. La transmission des données débute après synchronisation.
- FIN : FINal flag : La fin de la connexion est ainsi indiquée lorsque la station a transmis toutes ses données.
- **Longueur** : Afin d'éviter un débordement du tampon du destinataire, celui ci transmet dans cd champs la quantité maximale de données que l'expéditeur peut transmettre sans confirmation (windowing).
- **Somme de contrôle** : vérification de la zone TCP de la trame = complément à 1 sur 16 bits de la somme de tous les compléments à 1 des mots 16 bits de la zone TCP
- Pointeur Urgent : valide si URG est actif. Cette valeur désigne la fin des données de priorité d'un segment TCP. Permet d'indiquer, avec le numéro de séquence, l'emplacement des données.

`Options/remplissage : propose : - un type d'option (1octet) , - une longueur(1octet), - éventuellement des données de l'option, - un remplissage de zéros pour respecter la structure de l'en-tête TCP (multiple de 32bits)`

Structure d'une trame UDP : Les ports de communication

Les spécifications précisent qu'un datagramme UDP doit se subdiviser en blocs de 32 bits. Leur structure de base ressemble à celle des paquets TCP. Ils se distinguent toutefois par l'absence de certains champs, notamment *Séquence et Acknowledge.*

<table>
<tr><th colspan="2">En-tête UDP située après l'en-tête IP</th></tr>
<tr><td>Port source (16 bits)</td><td>Port cible (16 bits)</td></tr>
<tr><td>Taille ou longueur (16 bits)</td><td>Somme de contrôle (16 bits)</td></tr>
<tr><td colspan="2">Zone de données</td></tr>
</table>

- **Port source**: de l'application UDP qui a expédié la trame. Voir "/etc/services"
- **Port cible**: de l'application UDP qui doit traiter la trame. Voir "/etc/services"
- **Longueur**: ici figure la longueur du paquet UDP. Nous pouvons qualifier ce champ de fonction de contrôle, permettant de vérifier si la trame complète a été reçue. Sa valeur minimale est de 8 octets, c'est-à-dire la taille de l'en-tête UDP. Dans ce cas, aucune donnée utile n'est transmise.
- **Somme de contrôle**: ce champ permet de vérifier tant l'en-tête UDP que le pseudo-en-tête et les données. Il assure un minimum de sécurité à la transmission de l'en-tête lui-même pour qu'il ne soit pas exposé totalement sans défense. Cette somme de contrôle permet de vérifier si le paquet UDP a été correctement transmis ou non.
- **Zone de données** (variable): cette zone contient les données.