

# Support du TP d'administration

## LAN / CISCO

IUT Aix-Marseille - INFO Aix

C. Pain-Barre, A. Meyer, F. Dumas

**i** La (re)lecture de ce document **avant** un TP d'administration LAN / CISCO est vivement recommandée pour faciliter sa progression dans le TP (et sa notation) !  
Amener une clé USB au cours du TP pour sauvegarder votre travail.

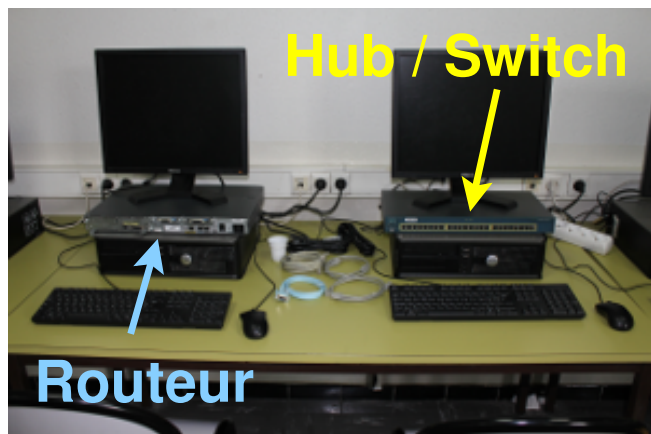
### Table des matières

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Les routeurs CISCO</b>	<b>6</b>
2.I	Ports d'administration	6
2.II	Interfaces réseau	7
2.III	Emplacements d'extension	8
2.III.A	Cartes WIC	8
2.III.B	Cartes NM	9
2.IV	Composants de la carte mère	10
2.V	Processus de démarrage des routeurs	11
<b>3</b>	<b>Logiciels du routeur : IOS et CLI</b>	<b>14</b>
3.I	Les modes principaux de la CLI	14
3.I.A	Mode utilisateur	14
3.I.B	Mode privilégié	14
3.I.C	Mode de configuration globale	14
3.I.D	Mode de configuration d'interface	15
3.I.E	Changements de mode	15
3.II	Annulation d'une commande	16
3.III	Messages d'information	16
3.IV	Pagination des messages	17
3.V	L'aide à la saisie	17
3.VI	Moyens d'accéder à la CLI	19
<b>4</b>	<b>Minicom pour accéder à la console</b>	<b>21</b>
4.I	Câblage du port console au port série	21
4.II	Le logiciel minicom	22
4.II.A	Paramétrage de minicom	22
4.II.B	Utilisation	23

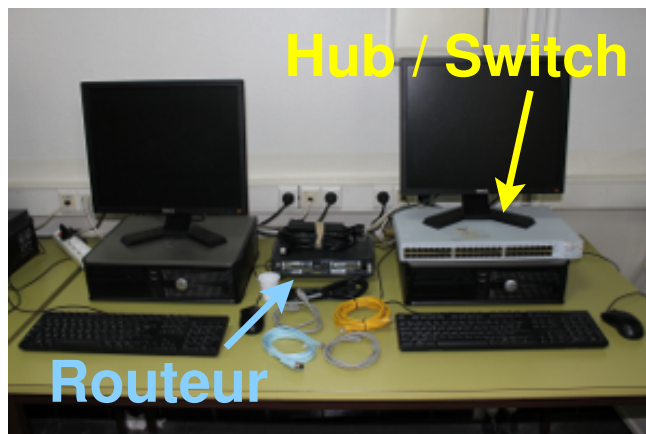
<b>5</b>	<b>VirtualBox</b>	<b>25</b>
5.I	Exécution de la machine virtuelle	26
5.II	Utilisateurs de la machine virtuelle	26
5.III	Utilisation d'un périphérique USB	26
5.IV	Partage de fichiers entre les systèmes hôte et invité	27
<b>6</b>	<b>Quelques commandes réseau sous Linux</b>	<b>28</b>
6.I	Configuration d'une interface réseau	28
6.II	Configuration statique de la table de routage	29
6.III	Configuration par DHCP	30
<b>7</b>	<b>Utilisation de TFTP</b>	<b>31</b>
7.I	Dépôt d'un fichier du routeur sur le serveur TFTP	32
7.II	Récupération sur le routeur d'un fichier du serveur TFTP	33
<b>8</b>	<b>Analyse de trafic réseau avec Wireshark</b>	<b>35</b>
8.I	Interface de Wireshark	35
8.I.A	Barres de menu et d'actions	35
8.I.B	Liste des trames	38
8.I.C	Détails des protocoles reconnus dans une trame	39
8.II	Filtres de capture et d'affichage	40
8.II.A	Expression d'un filtre de capture avec la libpcap	40
8.II.A.1	Primitives pour Ethernet	41
8.II.A.2	Primitives pour IP	41
8.II.A.3	Primitives communes à UDP et TCP	41
8.II.A.4	Cas particulier des primitives spécifiques à ICMP	42
8.II.B	Les filtres d'affichage	42
8.II.B.1	Syntaxe d'un filtre d'affichage	42
8.II.B.1.a	Filtre par protocole	42
8.II.B.1.b	Filtre selon la valeur d'un champ	43
8.II.B.1.c	Test d'un champ booléen	44
8.II.B.1.d	Connecteurs	45
8.II.B.2	Extrait des identificateurs et champs disponibles	45
8.II.B.2.a	Identificateur et champs pour Ethernet	45
8.II.B.2.b	Identificateur et champs pour IP	46
8.II.B.2.c	Identificateur et champs pour TCP	46
8.II.B.2.d	Identificateur et champs pour UDP	47
8.II.B.2.e	Identificateur et champs pour ICMP	47

# 1 Introduction

Ces TP (TP5 du module M2102 du S2, TP5 du module M3102 du S3) concrétisent l'apprentissage réseau en manipulant du matériel professionnel, notamment des routeurs CISCO et des switchs. Les étudiants travaillent en binôme avec un espace de travail similaire à ceux des prises de vues de la figure 1.



(a) Avec séries 2500, 2600 et modèle 1760



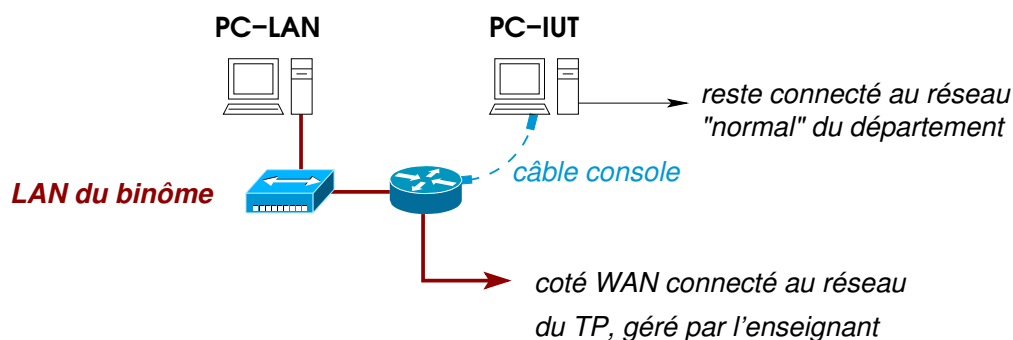
(b) Avec routeur 1720 ou 1721

**FIGURE 1** – *Présentations probables de l'espace de travail*

Chaque binôme dispose en principe de :

- 2 PC sous Linux (voir ci-dessous) ;
- 1 routeur CISCO parmi les séries 1700, 2500 et 2600, doté de plusieurs interfaces réseaux, notamment Ethernet ;
- 1 Switch (ou plus rarement un Hub), au nombre de ports variable (minimum 4) ;
- 1 câble console DB-9/RJ-45 pour connecter le port série (DB-9) d'un PC au port console du routeur (RJ-45) ;
- suffisamment de câbles Ethernet et/ou de câbles séries pour réaliser les raccordements éventuellement demandés.

**i** Au S2, les binômes câblent les équipements et réalisent des configurations (routeur, PC) pour former un LAN opérationnel connecté à Internet, en procédant à certaines analyses réseau. Au S3, les configurations sont plus poussées.



**FIGURE 2** – Environnement de travail d'un binôme : le PC-IUT reste connecté au réseau du département et servira à administrer le routeur. Le PC-LAN sera relié au LAN du binôme qui comprend un hub/switch et un routeur. Le routeur sera être connecté au réseau géré par l'enseignant (côté WAN). L'organisation exacte côté WAN est variable. Elle fait partie des détails du sujet du TP.

Les 2 PC du binôme ne jouent pas le même rôle :

- l'un des deux sera nommé le **PC-IUT**. Il doit rester connecté au réseau du département (et à Internet). Il servira notamment à administrer le routeur par l'intermédiaire du câble console à connecter au **port console** du routeur.
- l'autre PC sera nommé le **PC-LAN**. Il sera déconnecté du réseau du département et sera connecté au LAN du binôme. Sur ce PC seront exécutées une ou plusieurs machines virtuelles qui seront intégrées dans ce LAN.

La figure 2 illustre le câblage (traits rouges) réalisé pour former le **LAN** d'un binôme, constitué du PC-LAN, du switch (ou hub) et du routeur. La flèche rouge en bas à droite illustre le câblage entre le routeur du binôme et le réseau géré par l'enseignant qui mène vers Internet. Ce sera le côté **WAN** du routeur du binôme.

✍ On utilisera les termes suivants :

- le **côté WAN** est la liaison (interface du routeur) menant vers Internet. Cette partie est variable et fait partie du sujet du TP.
- le **côté LAN** est la liaison (interface du routeur) au LAN du binôme.

## La suite du document

La partie plutôt "matérielle" des routeurs est décrite dans la section 2. Pour les configurer, il faut se familiariser avec l'interface de commandes —appelée CLI pour *Command Line Interface*— de l'IOS CISCO, le système d'exploitation des routeurs CISCO. C'est l'objet de la section 3.

L'accès à la CLI du routeur peut se faire à distance via une session TELNET (voire SSH pour les plus récents). En début de TP, le routeur n'est pas configuré et est donc inaccessible par le réseau. L'accès à la CLI se fera donc dans un premier temps via son port console, connecté au port série du PC-IUT en utilisant un câble "console" à connecteurs DB-9 et RJ-45, ce qui est matérialisé par la flèche en pointillés bleus sur la figure 2. Sur le PC-IUT sous Linux, l'utilitaire **minicom** permettra de dialoguer avec le routeur via cette liaison série. La section 4 décrit ce câblage et l'utilisation de **minicom**.



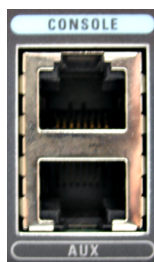
**Ce raccordement du PC-IUT au routeur n'est pas une liaison réseau ! Le PC-IUT ne sera pas intégré au LAN du binôme et restera connecté au réseau de l'IUT.**

Outre la configuration du routeur, ce TP sera aussi l'occasion d'administrer des machines virtuelles exécutées avec **VirtualBox** sur le PC-LAN, notamment pour les intégrer dans le LAN du binôme. Pour les VM sous Linux, on disposera d'outils tels que :

**ifconfig, route, ...** les commandes usuelles, rappelées dans la section 6, seront utilisées pour configurer les VM et exploiter votre LAN ;

**in.tftpd** un serveur TFTP dont l'utilisation est décrite en section 7 est opérationnel pour envoyer/recevoir des fichiers au routeur ;

**wireshark** un outil de capture et d'analyse de trames sera utilisé pour observer et interpréter une partie du trafic réseau que vous générerez. La section 8 est dédiée à ce logiciel très complet.



**FIGURE 3** – Ports console et AUX d'un 1721. Nous n'utiliserons que le port console (en haut), signalé par le texte *CONSOLE*, sur fond bleu ciel.

## 2 Les routeurs CISCO

Les routeurs dont nous disposons sont des modèles des séries 1700, 2500 et 2600. Ce sont des routeurs "professionnels", destinés aux entreprises. Ce genre de routeurs disposent d'interfaces réseaux intégrées, de différents types. Les routeurs des séries 1700 et 2600 sont modulaires. Ils disposent en plus d'emplacements pour des interfaces réseaux additionnelles. L'ensemble de ces interfaces permettent de router les datagrammes IP sur différents réseaux.

Nous commencerons par décrire les ports d'administration et les interfaces de nos routeurs avant d'aborder leur fonctionnement.

### 2.1 Ports d'administration

Nos routeurs CISCO, comme de nombreux équipements réseau, ne possèdent ni clavier ni moniteur. Ils ne peuvent donc être administrés que via un équipement externe.

Pour cela, ils disposent de deux ports dédiés à son administration, dont une vue (sur un 1721, l'un des modèles de la série 1700) est présentée dans la figure 3 :

- le **port console** à connecteur RJ-45, servant à l'administrer via une liaison série.
- le **port auxiliaire** (AUX) à connecteur RJ-45, destiné en principe à être connecté à un modem pour une administration à distance.

 Nous n'utiliserons pas le port AUX. Seul le port console nous sera utile.

Le port console servira à administrer le routeur depuis le PC-IUT, en utilisant **minicom** depuis une fenêtre terminal. Sur cette fenêtre, nous pourrons taper des commandes pour configurer le routeur, obtenir des informations, etc. La figure 4 donne un aperçu d'un dialogue avec le routeur avec l'utilitaire **minicom**.

messages de démarrage de minicom

interactions avec la CLI

```

cpb@OP36083X: ~
Fichier  Édition  Affichage  Terminal  Onglets  Aide

Bienvenue avec minicom 2.3

OPTIONS: I18n
Compilé le Feb 24 2008, 16:35:15.
Port /dev/ttyS0

Tapez CTRL-A Z pour voir l'aide concernant les touches spéciales

Router>show users
  Line      User      Host(s)      Idle      Location
  *  0 con 0
  idle
  00:00:00

  Interface  User      Mode      Idle      Peer Address

Router>enable
Password:
Router#show interfaces description
Interface      Status      Protocol Description
Et0            admin down  down
Fa0            admin down  down
Se0            admin down  down
Router#
  
```

commande

résultat

commandes

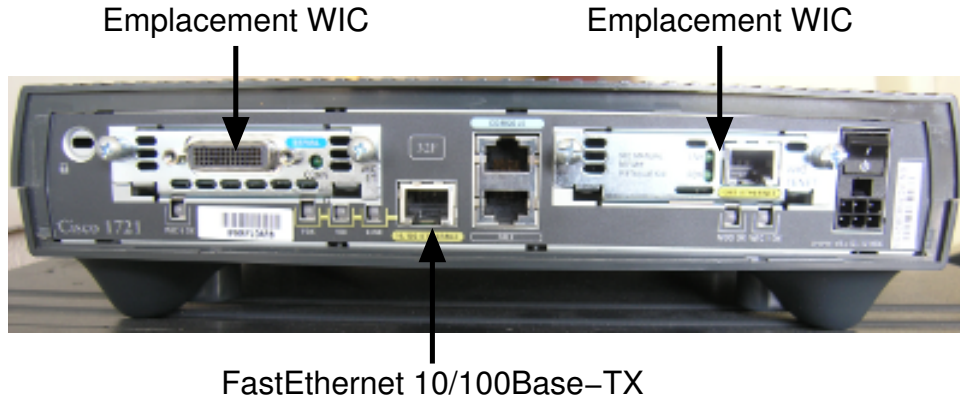
résultat

FIGURE 4 – Administration d'un routeur par le port console en utilisant minicom

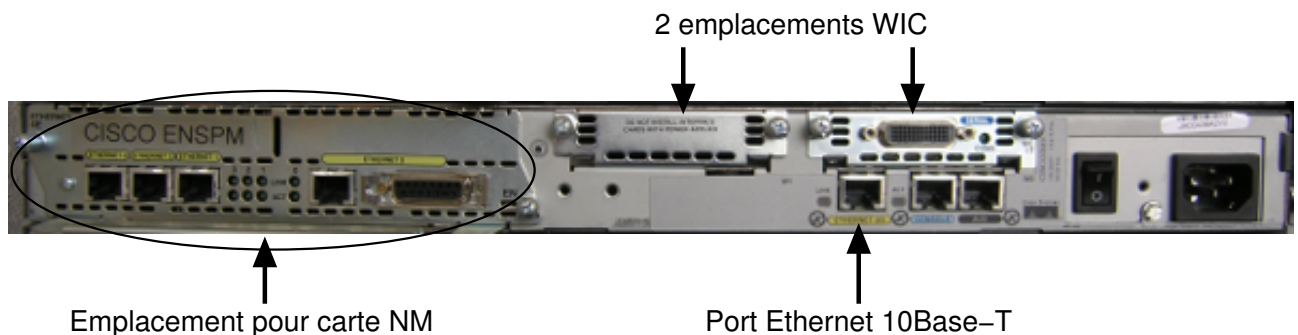
## 2.II Interfaces réseau

Les interfaces réseau et les emplacements d'extension (voir section suivante) possibles dépendent du modèle :

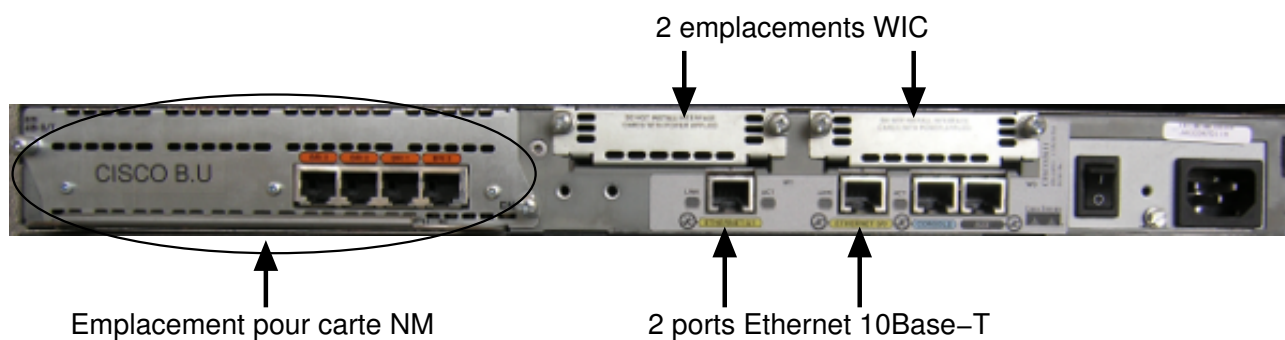
- modèles 1720 et 1721 : ils disposent d'un port FastEthernet (10/100Base-TX) et de deux emplacements (slots) pour cartes d'extension WIC (Wan Interface Card) :



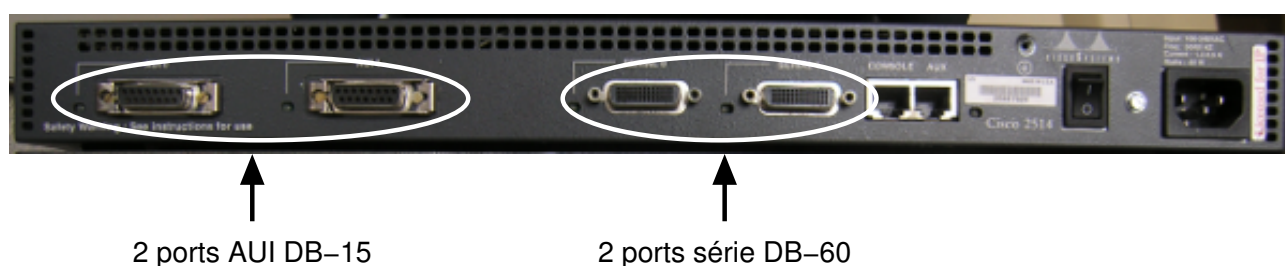
- modèle 2610 : il dispose d'un port Ethernet 10Base-T, de deux slots pour carte WIC, et d'un slot pour module réseau NM :



- modèle 2611 : il dispose de deux ports Ethernet 10Base-T, de deux slots pour carte WIC, et d'un slot pour module réseau NM :



- modèle 2514 : il dispose de 2 ports AUI (Attachment Unit Interface) Ethernet à connecteurs DB-15 et de 2 ports série à connecteurs DB-60 :



En branchant sur un port AUI un transceiver 10Base-T, on obtient une interface Ethernet 10 Mbps Half-Duplex à connecteur RJ-45 classique. Voici un exemple de ce transceiver :



❗ Selon les arrivages ; -), des routeurs d'autres types pourraient être disponibles.

## 2.III Emplacements d'extension

Les emplacements d'extension de nos routeurs sont de 2 types : pour carte WIC et pour carte NM. Les 1700 n'ont que des slots WIC, alors que les 2600 ont aussi un slot NM. Les 2500 n'ont aucun emplacement d'extension.


### 2.III.A Cartes WIC


Les cartes WIC (*WAN Interface Card*) s'enfichent dans les slots WIC du routeur et lui ajoutent une ou plusieurs interfaces d'un certain type. Voici quelques cartes WIC possibles :

- WIC-1T : 1 port série, asynchrone et synchrone (T1/E1)
- WIC-2T : 2 ports série, asynchrones et synchrones (T1/E1)
- WIC-1B-S/T : 1 port BRI S/T (Basic Rate Interface) pour Réseau Numérique à Intégration de Service
- WIC-1ADSL : 1 port ADSL
- WIC-1SHDSL : 1 port SHDSL (Symmetrical High-Bit Rate DSL)



- WIC-4ESW : Switch configurable 4 ports Ethernet (10/100BASE-TX)
- **WIC-1ENET** : carte présentée dans la figure 5 qui fournit 1 port Ethernet (10BASE-T) pouvant accepter le full-duplex ;
- ... et il y en a beaucoup d'autres

 Nos routeurs disposent de types variés de cartes mais on se servira uniquement des cartes WIC-1ENET et/ou WIC-1T et/ou WIC-2T. Certains peuvent disposer d'une carte WIC-4ESW qui fournit un switch 4 ports Ethernet.

 Généralement, les cartes WIC des 1700 sont aussi compatibles avec les routeurs des séries 2600, 3600 et 3700. Mais il y a des exceptions. Par exemple, les cartes WIC-1ENET ne sont pas supportées par les 2600.

De façon générale, avant d'investir dans une carte d'extension (WIC, NM ou autre), l'administrateur doit vérifier qu'elle est supportée par son matériel, ainsi que par la version d'IOS dont il dispose.



**FIGURE 5** – Carte WIC-1ENET fournissant un port Ethernet 10 Base-T pouvant accepter le Full-Duplex.

### 2.III.B Cartes NM

Les cartes NM (*Network Module*) sont des modules réseau plus grands que les WIC et peuvent doter le routeur de nombreuses interfaces (par exemple, Switch 16 ports Ethernet avec le NM-ESW16). Comme les WIC, il existe de nombreuses cartes NM. Nous utiliserons les cartes suivantes :

- **NM-1E** (*One-port 10BASE-T Ethernet interface*) : ce module a la particularité d'offrir le choix entre 2 ports pour une seule interface : 1 port RJ-45 et 1 port AUI. On peut utiliser n'importe lequel, mais pas en même temps.



- **NM-4E** (*4-port 10BASE-T Ethernet interface*) : ce module étend le module précédent en lui ajoutant 3 autres interfaces Ethernet 10Base-T, chacune avec un port RJ45.



❗ Certains 2600 sont équipés d'autres cartes NM (par exemple, NM BRI 4B-S/T et NM ESW-16) mais elles ne nous seront pas utiles <sup>1</sup>.

## 2.IV Composants de la carte mère

En interne, nos routeurs comportent les composants principaux suivants, dont une partie est visible sur la vue d'une carte mère d'un 1720 de la figure 6 :

- **CPU** (*Central Processing Unit*) : le microprocesseur est responsable de l'exécution du système d'exploitation du routeur. Sur les routeurs CISCO, ce système est l'**IOS** (*Internetwork Operating System*). L'IOS prend en charge les protocoles et fournit l'interface de commandes **CLI** (*Command Line Interface*) accessible via une liaison série ou une session terminal Telnet ou SSH. De la puissance du CPU dépend la capacité de traitement du routeur. Les processeurs des routeurs utilisés sont des processeurs RISC (*Reduced Instruction Set Computing*) :
  - ◇ pour les séries 1700 et 2600 : Motorola MPC860 cadencés à au plus 80 MHz
  - ◇ pour la série 2500 : Motorola 68030 cadencé à au plus 50 Mhz
- **Mémoire ROM** : mémoire permanente contenant le code pour réaliser les diagnostics de démarrage du matériel (POST, *Power On Self Test*) ainsi que le code pour le chargement (*System Bootstrap*) de l'image de l'IOS vers la RAM et la mise en route du système. Elle contient aussi un système minimal (ROMMON) et une interface de commandes simplifiée permettant de réaliser des opérations de secours : modification de la phase de boot, téléchargement par TFTP d'une image suite à une corruption de la flash, etc.
- **Mémoire Flash** (EPROM) : sous forme d'une barrette (*Flash SIMM*) ou d'une carte Flash (*Flash Card*). Elle se présente dans l'IOS comme l'unité de stockage **flash:**. Elle est utilisée pour stocker un (ou plusieurs) fichier contenant l'image de l'IOS, souvent compressée. On peut aussi y ajouter d'autres fichiers. La mémoire flash est en quelque sorte une EPROM (*Erasable Programmable Read-Only Memory*) —une ROM effaçable et programmable. Pour nos modèles, la taille de la flash n'excède pas quelques dizaines de méga-octets.
- **NVRAM** : c'est une mémoire non volatile, de petite taille (souvent 32 Ko). Elle aussi se présente dans l'IOS comme une unité de stockage **nvr:**. Elle contient la configuration, en particulier le fichier de configuration de démarrage **startup-config** et le **registre de configuration** (*configuration register*). Elle peut aussi contenir des sauvegardes de la configuration.
- **RAM** : mémoire volatile dans laquelle sont chargées toutes les informations permettant au matériel de fonctionner (tampons, tables de routage, table ARP, table de VLANs, ...). L'IOS est chargé en RAM au démarrage du matériel. Il réalise ensuite une copie du fichier `startup-config` (à partir de la NVRAM)

1. La NM ESW-16 est difficilement supportée par les 2600 : elle nécessite une importante capacité flash et RAM pour exécuter un IOS qui la supporte.

dans la RAM. Cette copie est nommée **running-config** : la configuration courante du routeur. La modification de la configuration du routeur est immédiatement appliquée mais n'affecte que la *running-config*. Elle n'est que temporaire car la RAM est effacée chaque fois que le routeur (re)boote et *a fortiori* lorsque l'alimentation est coupée. Pour rendre une modification permanente, il faut copier la *running-config* (en RAM) dans la *startup-config* (en NVRAM).

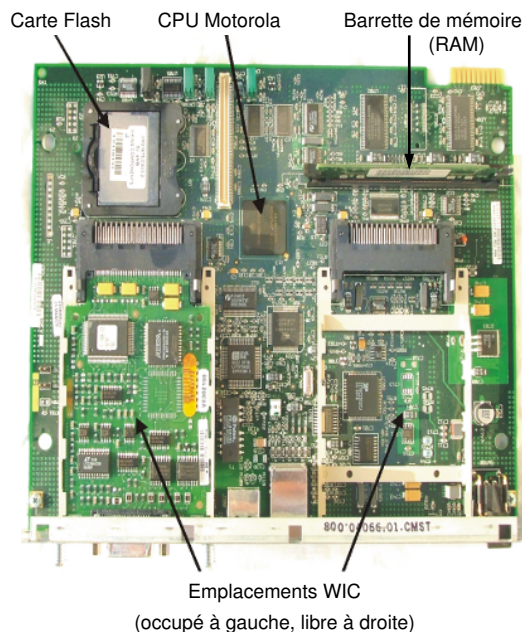



FIGURE 6 – Vue d'une carte mère d'un 1720 (série 1700)

## 2.V Processus de démarrage des routeurs

Ce processus est en partie schématisé par la figure 7. Au démarrage, le programme de la ROM (BootROM) d'un routeur CISCO va initialiser le matériel et exécuter le POST (*Power On Self Test*), effectuant les diagnostics de démarrage.

Après le POST, le **registre de configuration** est examiné pour :

- déterminer comment le routeur doit démarrer : démarrage normal ou dans un autre mode tel que ROMMON (mode monitoring de la ROM), RxBoot (IOS simplifié stocké en ROM sur d'anciens modèles, tels que les 2500) ou NetBoot (démarrage par téléchargement du système par TFTP<sup>2</sup>)
- définir les options à utiliser pendant le démarrage. Par exemple, ignorer la configuration de démarrage (*startup-config*), désactiver les messages de démarrage,...
- configurer la vitesse de la console (*baud rate*)

 Ce registre de configuration (*configuration register*) est stocké dans la NVRAM et contient une valeur sur 16 bits. Les différents bits du registre indiquent au routeur les paramètres de démarrage. Il est modifiable aussi bien en mode ROMMON qu'à travers la CLI de l'IOS. Nous serons amenés à le modifier en TP afin de prendre le contrôle du routeur et réinitialiser sa configuration.

2. *Trivial File Transfer Protocol* (TFTP) est un protocole très simple de téléchargement de fichier, ne nécessitant aucune identification. Il est souvent utilisé dans l'administration des matériels réseau. Notamment en ROMMON, RxBoot ou NetBoot car ces modes ne disposent que du système minimal contenu dans la ROM, qui a une faible capacité et peut difficilement contenir le code de protocoles plus évolués.

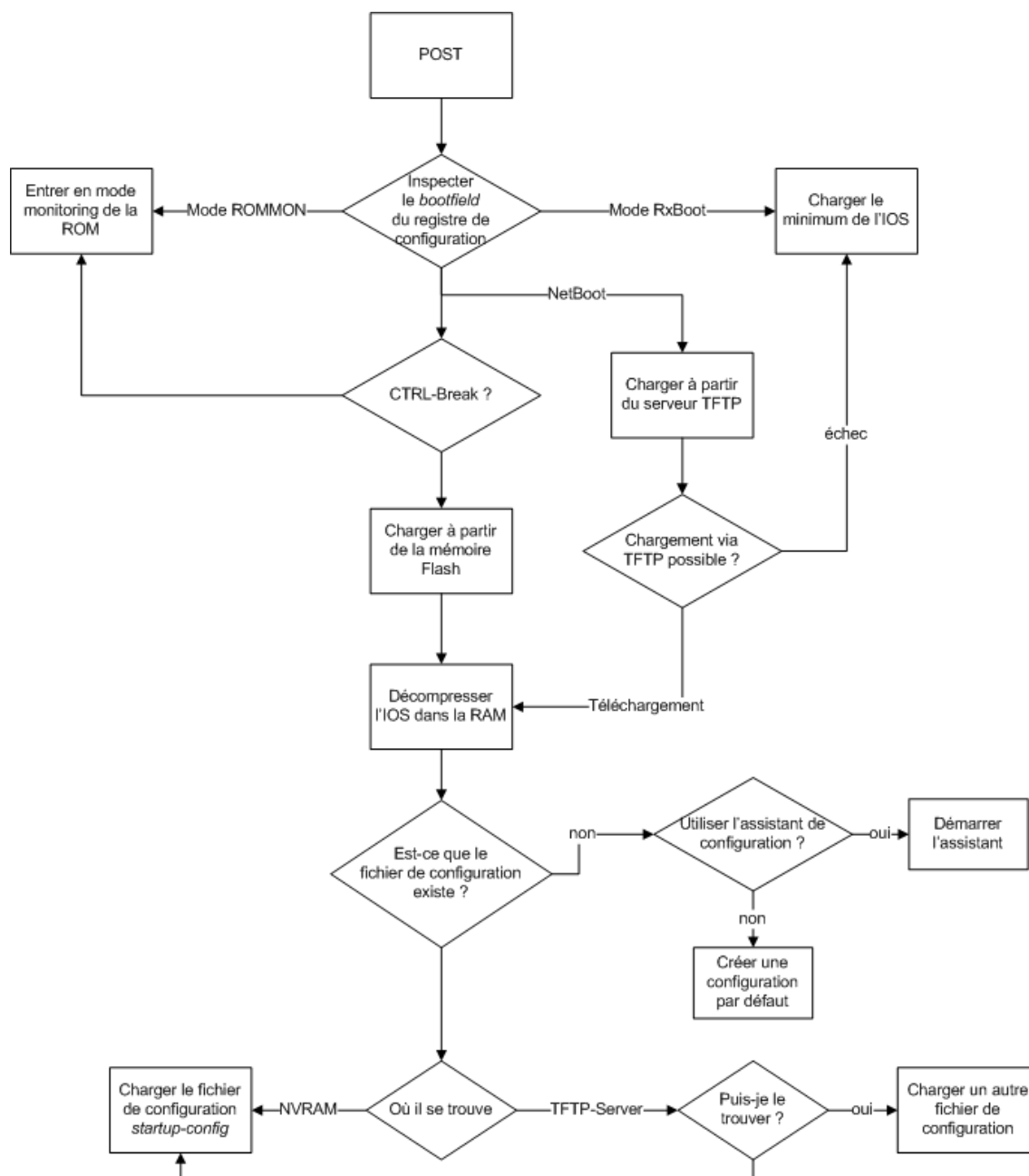


FIGURE 7 – Processus de démarrage d'un routeur CISCO

Le cas normal (valeur standard du registre de configuration) correspond à la partie centrale inférieure de la figure 7, où :

1. le BootROM charge l'image IOS stockée dans la carte flash vers la RAM (en la décompressant si nécessaire) ;
2. le BootROM cède sa place à l'IOS (le système d'exploitation effectif) qui démarre
3. Une fois initialisé, l'IOS copie la configuration stockée dans la NVRAM (`startup-config`) pour l'utiliser en RAM comme configuration courante (`running-config`). Une autre possibilité (très pratique en cas de mauvaise manipulation) est de récupérer cette configuration auprès d'un serveur TFTP, si l'administrateur prudent l'y a sauvegardé. . .
4. le routeur est alors opérationnel et actif

Les démarrages en ROMMON ou en RxBoot (obsolète) chargent un système minimal placé en ROM, permettant l'exécution de quelques commandes via une interface de commandes basique. Ces modes sont utilisés pour des opérations de secours, telles que :

- formatage/gestion de la carte flash dont le système de fichier est endommagé (suite à un incident) provoquant l'impossibilité de démarrer normalement
- téléchargement d'une image de l'IOS dans la flash (le plus souvent par TFTP)
- modification du registre de configuration pour régler de nouveaux paramètres de démarrage

**i** Si vous avez compris le rôle des fichiers `startup-config` et `running-config`, vous devriez comprendre aussi pourquoi il est conseillé de sauvegarder régulièrement la configuration actuelle qui se trouve dans la RAM, idéalement après chaque étape de configuration du routeur. Il suffit simplement de faire une copie de `running-config` vers `startup-config` pour ne pas perdre la configuration stockée dans la RAM.

Il est encore plus prudent (et recommandé) de sauvegarder la configuration en lieu sûr, en copiant la `startup-config` (ou la `running-config`) via un réseau TCP/IP sur un serveur TFTP, ce qui est très simple à réaliser sur ces routeurs. En cas de problème, par exemple panne de routeur nécessitant son remplacement, la configuration pourra être téléchargée sur le routeur.

## 3 Logiciels du routeur : IOS et CLI

La CLI (*Command Line Interface*) est l'interface entre l'utilisateur ou l'administrateur et l'IOS (*Internetwork Operating System*). Par analogie aux systèmes Unix, la CLI est un shell (assez basique), alors que l'IOS est le système d'exploitation. La CLI lit des commandes utilisateur et demande à l'IOS de réaliser les actions correspondantes. Dans un premier temps, on utilisera **minicom** pour accéder à la CLI du routeur et lui faire exécuter des commandes (voir figure 4, page 7).

### 3.1 Les modes principaux de la CLI

#### 3.1.A Mode utilisateur

Lorsque le routeur (ou le switch) CISCO démarre normalement et a chargé l'IOS, il affiche un prompt de la forme :

```
nom-du-routeur>
```

où *nom-du-routeur* est le nom d'hôte (*hostname*) du routeur et est configurable. Si le nom d'hôte n'a jamais été modifié, le prompt est :

```
Router>
```

Le symbole > indique qu'on se trouve en **mode utilisateur**, où l'on a accès à un nombre limité de commandes, *a priori* inoffensives, permettant simplement d'observer l'état du routeur. Bien que ce soit rarement le cas, il est possible de protéger l'accès à ce mode par un mot de passe.

#### 3.1.B Mode privilégié

Pour effectuer l'administration et la configuration du routeur il faut d'abord passer en **mode privilégié** (aussi appelé mode EXEC). En mode utilisateur, la commande **enable** fait passer au mode privilégié. L'inverse se fait avec la commande **disable**. On sait qu'on se trouve en mode privilégié grâce au prompt : le > est remplacé par le symbole #. En mode privilégié le prompt aura donc la forme :

```
nom-du-routeur#
```

Le mode privilégié permet notamment de copier des fichiers (manipulations de la flash, de la NVRAM et de la RAM) et d'obtenir certaines informations qui doivent rester inaccessibles à toute tierce personne, comme par exemple la configuration (*startup-config* ou *running-config*) qui contient les mots de passe (cryptés ou non).

#### 3.1.C Mode de configuration globale

Pour modifier des paramètres globaux réseaux du routeur (par exemple, activation de certains protocoles), il faut entrer en **mode de configuration globale**, accessible à partir du mode privilégié en tapant :

```
nom-du-routeur#configure terminal
```

```
Enter configuration commands, one per line. End with CNTL/Z.
```

L'accès à ce mode modifie le prompt qui devient :

```
nom-du-routeur (config) #
```

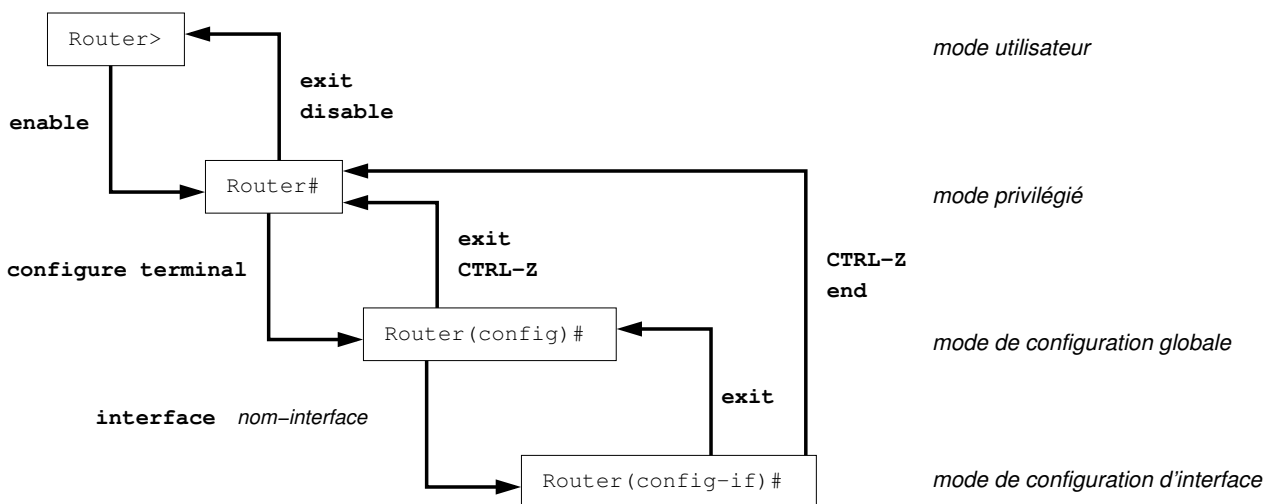


FIGURE 8 – Les principaux modes de la CLI

### 3.1.D Mode de configuration d'interface

Pour modifier la configuration d'une des interfaces réseau du routeur, il faut passer en **mode de configuration d'interface**, accessible à partir du mode de configuration globale en tapant :

```
nom-du-routeur (config) #interface nom-interface
```

où *nom-interface* est le nom qu'a donné l'IOS à l'interface qu'on veut configurer. Ce peut être **Ethernet0**, **FastEthernet0**, **Serial0**, **Ethernet0/0**, etc.

- ❗ Le nom IOS des interfaces du routeur pourra être obtenu en utilisant, en mode utilisateur ou privilégié, la commande :

```
Router>show interfaces
```

En mode de configuration d'interface, le prompt change à nouveau et l'on a accès aux commandes permettant de la configurer :

```
nom-du-routeur (config-if) #
```

### 3.1.E Changements de mode

L'accès aux modes précédents se fait dans l'ordre : utilisateur puis privilégié puis configuration globale puis configuration de l'interface.

- 🔗 Il existe d'autres modes plus spécifiques, comme par exemple le mode de configuration des adresses gérées par le serveur DHCP du routeur, qu'on utilisera aussi au cours du TP.

Pour revenir d'un mode au mode de niveau inférieur, on utilise la commande **exit** (en mode privilégié, elle a le même effet que **disable**). Pour revenir au mode privilégié à partir de n'importe quel mode auquel il mène, on peut utiliser le raccourci **CTRL+Z** ou la commande **end**.

La figure 8 résume les différents modes de la CLI de CISCO et indique à chaque fois les commandes utilisées pour passer d'un mode à un autre.

### 3.II Annulation d'une commande

Lorsque dans un mode, on a tapé et validé une commande, on pourra le plus souvent l'annuler en la faisant précéder du mot clé **no**.

#### Exemple 1

Dans le mode de configuration du terminal, la commande

```
Router (config) #ip http server
```

active un serveur Web sur le routeur, alors que

```
Router (config) #no ip http server
```

le désactive.

En mode de configuration d'interfaces, la commande

```
Router (config-if) #shutdown
```

désactive une interface réseau, alors que

```
Router (config-if) #no shutdown
```

l'active.



**i** Il sera souvent bien pratique dans ce cas d'utiliser le rappel de l'historique (voir section 3.V), surtout si la commande à annuler est un peu longue ou complexe.

### 3.III Messages d'information

Lorsqu'on administre le routeur par son port console, il arrive que le routeur affiche sur la CLI des messages d'information, par exemple lorsqu'il vient de détecter qu'une interface réseau est connectée. Ces messages commencent par % (et sont éventuellement précédés d'une date) comme :

```
%LINEPROTO-5-UPDOWN : Line Protocol on Interface FastEthernet0/0, changed state to up.
```

ou encore :

```
*Mar 1 00:00:04.259: %SYS-7-NV_BLOCK_INIT: Initialized the geometry of nvram
```

Ces messages masquent parfois le prompt de la CLI. Pour le faire réafficher, taper simplement sur **CTRL+L** (ou **Entrée** si la ligne de commande est vierge). La combinaison **CTRL+C** peut être aussi utilisée pour réafficher le prompt sans tenir compte de ce qu'on vient de taper, notamment lorsqu'on se trompe en saisissant une commande.

**💣** Lorsque vous revenez au mode privilégié, attendez quelques secondes avant de taper de nouvelles commandes car le routeur va afficher des messages d'information. Si l'on est en train de taper une commande, ces messages sont très gênants. Mais il suffit de taper **CTRL+L** pour faire réafficher le prompt et ce qu'on a déjà tapé.



### 3.IV Pagination des messages

Lorsque la CLI doit afficher plus de 25 lignes d'un coup, elle pagine cet affichage<sup>3</sup>. Dans ce cas, la dernière ligne affiche quelque chose comme :

```
--More--
```

ce qui signifie que la CLI attend pour afficher la suite. On peut alors taper :

- `Espace` pour passer à la page suivante
- `Entrée` pour voir la ligne suivante
- `q` ou `CTRL+C` pour arrêter l'affichage

**i** On remarquera qu'il n'est pas possible de revenir en arrière sur le texte affiché...

### 3.V L'aide à la saisie

La CLI dispose essentiellement de trois fonctionnalités qui facilitent la saisie de commandes : le rappel de l'historique, une aide contextuelle et le remplissage (*completion*) automatique.

#### Rappel de l'historique

Le **rappel de l'historique** est la possibilité de rappeler les dernières saisies en les faisant défiler en utilisant les flèches du haut (vers les plus anciennes) et du bas (vers les plus récentes). Chaque mode dispose de son propre historique : dans un mode donné, on ne rappelle que les commandes tapées dans ce mode. En principe, l'historique est activée par défaut. Sinon, en mode privilégié, il faut utiliser la commande :

```
Router#terminal history [nombre-de-lignes]
```

où *nombre-de-lignes* est optionnel et spécifie le nombre de lignes à retenir (par défaut 10).

**i** Le rappel de l'historique est très utile pour annuler une commande complexe.

#### Aide contextuelle

L'**aide contextuelle** nous permet de savoir quelles sont les commandes disponibles et, pour chacune, quels sont ses paramètres possibles. Elle intervient lorsque l'on tape le caractère ? :

- ? seul fait afficher la liste des commandes disponibles dans le mode courant, avec une courte description ;
- si l'on a déjà commencé à taper un mot, ? fait afficher la liste des mots attendus, commençant par ce qu'on a tapé ;
- lorsqu'on a tapé une commande, à chaque début de mot, on peut taper ? pour faire afficher les paramètres possibles à cette position ainsi qu'une courte description. Certains paramètres seront à remplacer par une valeur, notamment :
  - ◇ **A . B . C . D** devra être remplacé par une adresse IP ;
  - ◇ **WORD** devra être remplacé par un mot (suite de caractères) dont la signification dépend du contexte ;
  - ◇ **<cr>** représente le retour à la ligne, ce qui veut dire qu'on peut terminer la commande sans avoir à taper autre chose.

3. Ceci parce que le routeur doit pouvoir être administré par un terminal texte 80 colonnes, 25 lignes.

## Exemple 2

En mode privilégié, si l'on tape :

```
Router#?
Exec commands:
  access-enable      Create a temporary Access-List entry
  access-profile     Apply user-profile to interface
  access-template    Create a temporary Access-List entry
  ...
```

↪ alors la liste des commandes du mode privilégié est affichée. Comme tous les affichages, elle est paginée si elle excède 25 lignes

```
Router#pi?
ping
```

↪ une seule commande commence par pi, c'est **ping**

```
Router#ping ?
WORD  Ping destination address or hostname
ip     IP echo
ipv6   IPv6 echo
tag    Tag encapsulated IP echo
<cr>
```

↪ **<cr>** indique que les paramètres sont optionnels, **WORD** doit être remplacé par une adresse ou un nom d'hôte, alors que les autres possibilités sont à taper

```
Router#ping 139.124.187.4 ?
<cr>
```

↪ **<cr>** indique qu'il n'y a plus rien à entrer



## Remplissage automatique

Le **remplissage automatique** permet de se contenter de ne taper que les débuts des mots lorsqu'il n'y a pas d'ambiguïté. Par exemple, en mode privilégié taper :

```
Router#conf t
```

est synonyme de **configure terminal** car il n'y a qu'une seule commande commençant par **conf** et uniquement **terminal** comme paramètre de cette commande qui commence par **t**. De même, lorsqu'on tape un début de mot, on peut le faire compléter par la CLI s'il n'y a pas d'ambiguïté en tapant sur Tabulation. Dans l'exemple précédent, taper **conf**Tabulation fait compléter par **configure**.

### 3.VI Moyens d'accéder à la CLI

Nos routeurs CISCO, comme de nombreux équipements réseau, ne possèdent pas de clavier ni de moniteur. Ils ne peuvent donc être administrés que via un équipement externe.

❗ Ces routeurs, comme de nombreux routeurs et switch de type "grand public" ou SOHO (*Small Office Home Office*), peuvent activer un serveur HTTP permettant leur administration via une interface Web. Le point faible de ce type d'interface graphique est qu'elle n'est conçue que pour les cas d'utilisation les plus fréquents et qu'elle ne permet pas un réglage fin de la configuration.

Si la configuration souhaitée n'est pas prévue dans l'interface<sup>4</sup>, il devient nécessaire de configurer le routeur à travers son interface de commandes en ligne (la CLI sur l'IOS CISCO). Cette interface n'est généralement pas disponible sur le matériel destiné au "grand public", alors que les administrateurs des routeurs "professionnels" privilégient son utilisation.


Pour accéder à l'interface en ligne de commandes CLI, l'administrateur a plusieurs possibilités :


- se connecter par SSH : cela suppose que le routeur est opérationnel au niveau TCP/IP, que la version de l'IOS prenne en charge SSH et que le routeur ait été configuré pour permettre cet accès ;
- se connecter par TELNET : cela suppose que le routeur est opérationnel au niveau TCP/IP et ait été configuré pour permettre cet accès ;
- se connecter à distance par l'intermédiaire d'un équipement (généralement un modem) relié au port AUX : cela suppose que l'administrateur ait mis en place cette liaison et configuré le routeur pour permettre cet accès ;
- **se connecter au port "console" du routeur** : ce port est une interface série qui, comme son nom l'indique, est destinée à l'administration du routeur. C'est l'interface de secours, toujours opérationnelle (sauf en cas de défaillance grave matérielle). En contrepartie, il faut avoir accès physiquement au routeur, alors que les possibilités précédentes permettent un accès à distance.

Le port console est indispensable dans de nombreuses situations :

- le matériel est dans son état "sortie d'usine", c-à-d. sans configuration réseau. La console est donc le seul moyen d'accéder à l'interface d'administration du routeur. L'administrateur pourra ensuite le configurer et permettre son administration à distance ;
- le matériel a été configuré et intégré dans un réseau, mais suite à une erreur de configuration, le dispositif n'est plus joignable via le réseau. Il n'est pas rare en effet qu'en modifiant la configuration, l'administrateur finisse par s'interdire l'accès distant au routeur (par exemple en activant un filtre TCP/IP qui bloque le protocole utilisé pour l'administration à distance) ;
- le mot de passe pour accéder à l'interface d'administration du matériel a été oublié ou le matériel acheté est d'occasion avec un mot de passe inconnu. L'interface console permettra de ré-initialiser le mot de passe.

4. Par exemple, la configuration d'un accès DSL via IPoATM (*IP over ATM*) pour "Free dégroupé" sur un routeur CISCO 827H ou 1720


 Le port console est souvent présent sur du matériel réseau qualifié de "professionnel" (souvent cher et seulement disponible chez les fournisseurs de matériel réseau pour les entreprises), alors qu'il ne l'est pas sur du matériel destiné au "grand public" (souvent bon marché et assez bas de gamme). Notamment, le matériel Linksys (filiale de CISCO pour le grand public), n'en possède généralement pas.

 **Il existe une procédure permettant de prendre le contrôle d'un routeur dont le mot de passe est inconnu. Quiconque peut l'appliquer s'il a accès au port console du routeur. C'est pour cela qu'il est extrêmement important que les routeurs aient également une sécurité physique pour empêcher tout accès non autorisé, par exemple en les installant dans des locaux fermés et sécurisés.**

## 4 Minicom pour accéder à la console

Le routeur n'étant pas configuré dans un premier temps, il nous faudra en prendre le contrôle et l'administrer. Dans cette situation, la seule possibilité est d'utiliser le port d'administration du routeur : son **port console**.

Ce port console permet d'y connecter la **console** du routeur : un terminal informatique constitué d'un clavier et d'un écran textuel. Les terminaux en mode texte présentent les sorties uniquement sous forme textuelle, et disposent simplement d'un clavier pour les entrées. Un exemple de terminal texte qui fut répandu en France est le Minitel, relié aux serveurs par l'intermédiaire de la ligne téléphonique. Dans le monde informatique, le VT100 de DEC est devenu un standard de fait, avec un affichage de 25 lignes sur 80 colonnes.

 La console est le seul terminal permettant d'administrer le routeur dans ses différents modes de démarrage (ROMMON, RxBoot, NetBoot ou Normal). Pour prendre le contrôle du routeur, il faudra le démarrer en mode ROMMON.


La console (le terminal informatique) sera émulée par le logiciel **minicom** exécuté sur le PC-IUT sous Linux. Pour cela, le port console du routeur doit être relié au port série du PC-IUT en utilisant un câble console DB-9/RJ-45. Sur la fenêtre exécutant **minicom**, nous pourrions interagir avec le routeur dans ses différents modes de démarrage, en particulier ROMMON et normal. Ce dernier nous donnera accès à la CLI (voir figure 4, page 7).

### 4.1 Câblage du port console au port série

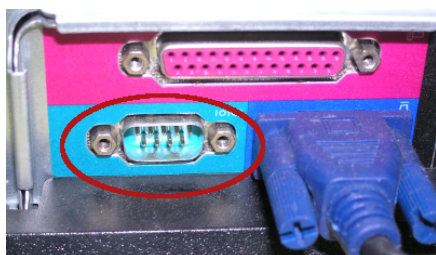
Le port (ou interface) série d'un PC est de type RS-232 avec un connecteur mâle DB-9 (9 broches). RS-232 (aussi appelé EIA RS-232C ou V.24) est une norme standardisant un port de communication de type série.

Sous Linux, ces ports sont généralement désignés par le fichiers spéciaux `/dev/ttyS0`, `/dev/ttyS1`, etc. Sous MS-DOS et Windows, ils sont désignés par les noms **COM1**, **COM2**, etc. (ce qui leur a valu le surnom de "ports COM").

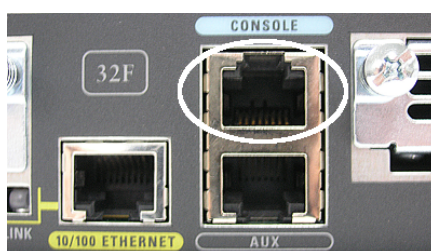
Le port RS-232 est fréquemment utilisé dans l'industrie pour connecter différents appareils électroniques (automate, appareil de mesure, carte mères, PDA, etc.).

 Le port série disparaît progressivement des PC au profit des ports USB. Il existe néanmoins des adaptateurs USB/Série.

Sur le routeur, le port console est aussi un port série mais avec un connecteur RJ-45 (8 broches). Nous les connecterons par l'intermédiaire du **câble console DB-9/RJ-45** (de couleur bleu clair pour les câbles de marque CISCO).



Port série DB-9 du PC

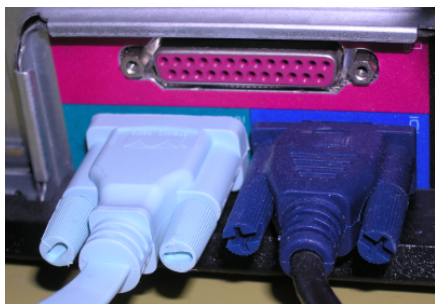


Port console RJ-45 d'un 1700



Câble console

Le câblage consiste simplement à enficher le connecteur DB-9 femelle du câble console dans le port série du PC, et son connecteur RJ-45 dans le port console :



Port série



Port console



Câblage final

## 4.II Le logiciel minicom

**minicom** est logiciel de communication "terminal" (terminal en mode texte) qui permet de se connecter à un équipement distant en utilisant un modem ou une interface série. Dans notre cas, il nous permettra d'administrer le routeur via la liaison série à son port console. Sur les systèmes Windows, **HyperTerminal** mais aussi **Putty** peuvent jouer ce rôle.

Plus précisément, **minicom** est un émulateur de terminal, aussi appelé console virtuelle (ou terminal virtuel) qui émule le fonctionnement d'un terminal informatique.

**i** **minicom** a bien d'autres fonctions mais qui ne seront pas utiles dans notre contexte.

### 4.II.A Paramétrage de minicom

Le port console du routeur est par défaut configuré pour fonctionner à 9600 baud, en transmettant 8 bits de données sans bit de parité mais avec un bit stop.

**i** Les paramètres de communication du port console peuvent être modifiés en modifiant le registre de configuration du routeur.

Il est nécessaire de configurer **minicom** pour utiliser les paramètres de communication suivants :

- Port série à utiliser : **/dev/ttyS0**
- Vitesse de modulation : **9600 baud**
- Nombre de bits de données : **8**
- Nombre de bits de parité : **aucun**
- Nombre de bits stop : **1**
- contrôle de flux matériel : **oui** (inutile)
- contrôle de flux logiciel : **aucun**

Le paramétrage de **minicom** se fait interactivement en tapant sur un terminal Linux la commande :

```
$ minicom -s
```

qui affiche le menu de configuration :

```
+-----[configuration]-----+
| Noms de fichiers et chemins
| Protocoles de transfert
| Configuration du port série
| Modem et appel
| Ecran et clavier
| Enregistrer config. sous dfl
| Enregistrer la configuration sous...
| Sortir
| Sortir de Minicom
+-----+-----+
```

On s'en servira durant le TP.

**i** Si l'on n'est pas root, on peut modifier mais pas sauver la configuration du terminal.

## 4.II.B Utilisation

Une fois configuré, en choisissant "*Sortir*", on se retrouve directement sur le terminal virtuel :

```
Bienvenue avec minicom 2.3

OPTIONS: I18n
Compilé le Feb 24 2008, 16:35:15.
Port /dev/ttyS0

Tapez CTRL-A Z pour voir l'aide concernant les touches spéciales
```

**i** Si root a modifié et sauvé la configuration, alors on peut lancer le terminal virtuel en exécutant **minicom** sans option :

```
$ minicom
```

Si on allume le routeur, on voit apparaître à l'écran les premiers messages du démarrage du routeur :

```
System Bootstrap, Version 12.2(7r)XM2, RELEASE SOFTWARE (fc1)
TAC Support: http://www.cisco.com/tac
Copyright (c) 2003 by cisco Systems, Inc.
```


Ce que le routeur envoie est affiché et ce que l'on tape est transmis au routeur (voir figure 4, page 7).

Dans certains cas, nous aurons besoin d'agir sur **minicom**. Par exemple, pour quitter **minicom**, il faut taper **CTRL+A** puis **X** (la casse n'a pas d'importance) et confirmer la sortie.

La combinaison **CTRL+A** est la **séquence d'échappement**, permettant d'entrer des commandes **minicom**. Outre la sortie, voici quelques commandes d'intérêt :

- **CTRL+A** puis **Z** afficher un menu présentant les commandes disponibles (voir figure 9) ;

- `CTRL+A` puis `F` envoyer un "*break*" sur la liaison série

 L'émission du *break* est indispensable pour prendre le contrôle du routeur.

- `CTRL+A` puis `O` afficher le menu de configuration.

```
-----  
                          Résumé des commandes de Minicom  
                          Les commandes peuvent être appelées par CTRL-A <touche>  
                          Fonctions principales          Autres fonctions  
Répertoire.....D  Exécuter un script.G | Effacer l'écran...C  
Envoyer des fichiersS  Recevoir des fichiers | Configurer Minicom.O  
Paramètres de comm.P  Ajouter LF.....A | Suspendre minicom..J  
Capture act/désact.L  Raccrocher.....H | Sortir et ràz.....X  
Envoyer « break »..F  Initialiser modem..M | Quitter sans ràz...Q  
Réglages du term...T  Exécuter Kermit...K | Touches du curseur.I  
Coupure des lignes.W  local Echo on/off..E | Help screen.....Z  
Paste file.....Y      | revenir en arrière.B  
  
Sélectionnez une commande ou tapez Entrée pour revenir.  
  
Ecrit par Miquel van Smoorenburg 1991-1995  
Quelques additions de Jukka Lahtinen 1997-2000  
i18n par Arnaldo Carvalho de Melo 1998  
-----
```

**FIGURE 9** – *Menu des commandes minicom*



## 5 VirtualBox

**VirtualBox** est un outil libre de **virtualisation** développé par Sun Microsystems (rachetée depuis par Oracle), dont le site Web officiel est <http://www.virtualbox.org/>. Des versions existent pour différentes plateformes : Windows, Linux, Macintosh, et OpenSolaris. Sur ces systèmes —appelés  *systèmes hôtes*—, il permet de créer et d'exécuter une (ou plusieurs) machine virtuelle —appelée  *système invité*—sur laquelle on peut installer et démarrer un système d'exploitation (Windows, Linux ou autre).

Par virtualisation, on entend que **VirtualBox** est capable de simuler, auprès du système invité, l'existence d'un ordinateur réel avec son processeur, sa mémoire ainsi que ses périphériques (vidéo, clavier, souris, réseau, etc.). Les caractéristiques de cette machine virtuelle sont paramétrables : taille de la RAM, nombre d'interfaces réseau, etc. Pour le système invité, tout se passe comme s'il avait vraiment affaire à un ordinateur ayant ces caractéristiques. C'est **VirtualBox** qui se charge d'utiliser les ressources du système hôte afin de faire fonctionner la machine virtuelle.

Dans notre cas, nous utiliserons **VirtualBox** afin de créer et d'exécuter un système **Linux Debian/Lenny** sur lequel on pourra se loger en tant que root et l'administrer. Nous pourrons aussi démarrer d'autres types de systèmes. Sur la figure 10, on peut voir la machine virtuelle tourner sur la fenêtre de droite qui a les caractéristiques principales suivantes :

- 1 processeur avec 800 Mo de RAM
- 64 Mo de mémoire vidéo ;
- 1 disque dur de 4 Go ;
- 1 carte réseau Ethernet d'adresse MAC **08:00:27:71:gg:bb** où *gg* et *bb* sont des nombres décimaux sur 2 chiffres représentant respectivement votre numéro de groupe et votre numéro de binôme. Cette carte correspond à l'interface **eth0** sur le système invité ;
- 1 port série **/dev/ttyS0**

Il sera possible de partager des fichiers entre les systèmes hôte et invité. Il sera aussi possible d'utiliser des périphériques USB sur les 2 systèmes, mais pas en même temps.

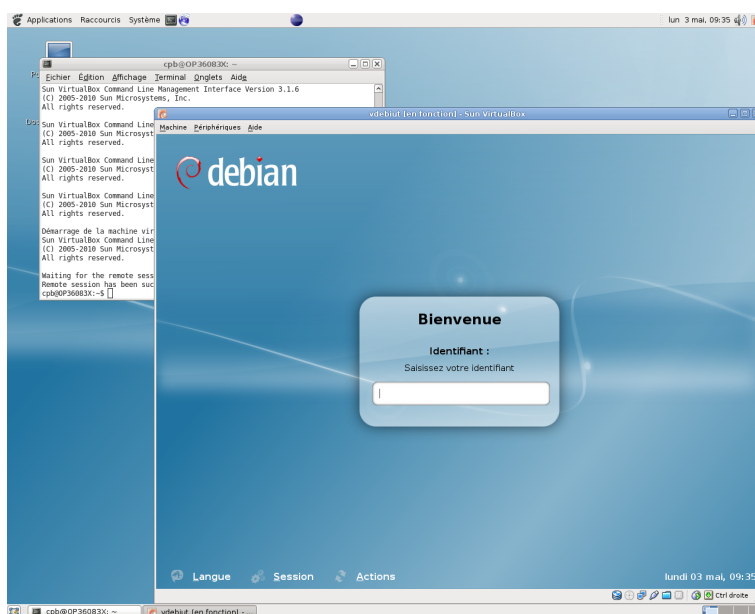


FIGURE 10 – À droite, une machine virtuelle Linux Debian/Lenny

## 5.I Exécution de la machine virtuelle

Le système invité est déjà prêt à l'emploi. Nous avons automatisé la création et l'exécution de la machine virtuelle correspondante. Dans le cas normal, vous aurez simplement à exécuter le script **mkdebborg.bash** sur un terminal de la machine (hôte), d'entrer votre numéro de groupe et de binôme lorsque cela vous est demandé (pour la génération de l'adresse MAC) d'attendre un petit peu et d'effectuer quelques clics de confirmation.

### Exemple 3

Ci-dessous, un exemple de création/exécution automatique de la machine virtuelle du binôme 13 du groupe 2 :

```
$ mkdebborg.bash
Nom de la VM demandée : vdebborg (Debian avec bridge)
Votre numero de groupe de première année (1 à 5) ? 2
Votre numéro de binôme ? 13
L'adresse MAC de cette VM sera : 08:00:27:71:02:13
Construction de la VM 'vdebborg'
Création du répertoire de la VM
Copie de l'image disque (patienter quelques minutes...)
 302MO 0:00:09 [19,6MO/s] [=> ] 8% ETA 0:01:42
...
```

⇒ On voit ici sur le message apparaissant en italiques que l'adresse MAC de l'interface réseau de la machine virtuelle sera *08:00:27:71:02:13*



**i** Si besoin, la méthode de création et d'exécution manuelle de la machine virtuelle vous sera communiquée.

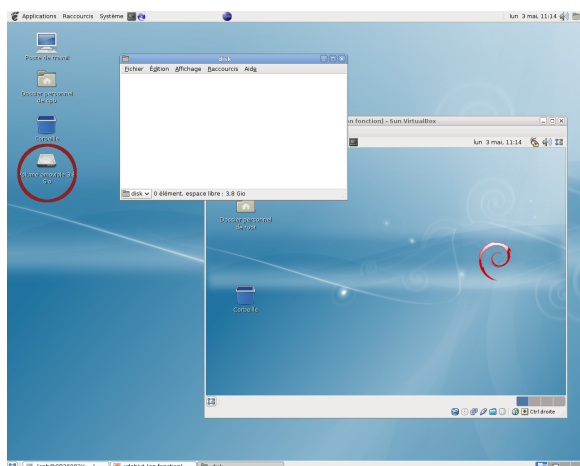
## 5.II Utilisateurs de la machine virtuelle

Le mot de passe du root de la machine virtuelle est **<re>zo++**. Si besoin, l'utilisateur normal **toto** a aussi été créé. Son mot de passe est **++re<zo>**.

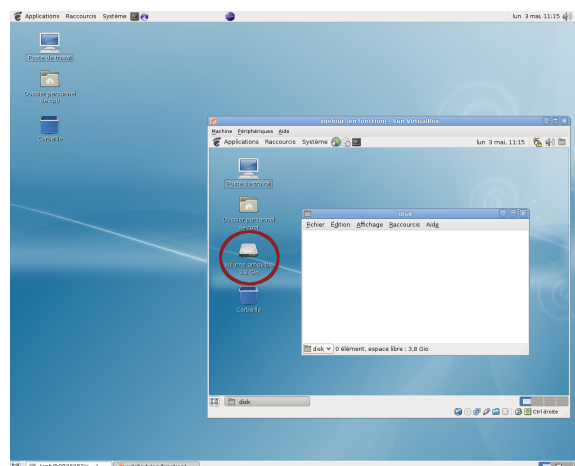
## 5.III Utilisation d'un périphérique USB

Lorsqu'on connecte un périphérique USB sur l'ordinateur, il est reconnu et utilisé en priorité par le système hôte. Par exemple, lorsqu'on connecte une clé USB, elle apparaît comme "Volume amovible" sur le bureau du système hôte comme on le voit sur la figure **11(a)**.

Or, vous aurez parfois à sauvegarder des fichiers du système invité sur clé USB. Pour cela, suite à l'apparition de la clé sur le système hôte, cliquer sur le menu *Périphériques* en haut de la fenêtre **VirtualBox**, puis sélectionner *Périphériques USB* et choisir la clé à utiliser (ne pas sélectionner la souris ni le clavier). La clé va alors disparaître du système hôte pour apparaître sur le bureau du système invité, comme on le voit sur la figure **11(b)**.



(a) périphérique USB sur le système hôte



(b) périphérique USB sur le système invité

FIGURE 11 – Différentes possibilités pour un périphérique USB

## 5.IV Partage de fichiers entre les systèmes hôte et invité

La machine virtuelle est aussi configurée pour avoir accès au répertoire `vmshared` situé sur le bureau de l'utilisateur de la machine hôte et, ce, en lecture/écriture/exécution. Ceci se fait par l'intermédiaire d'un **partage** géré par VirtualBox et que l'on a nommé **hostshared** dans la machine virtuelle. Pour utiliser ce partage, il faut réaliser un montage de ce dernier sur un répertoire du système invité, qui deviendra une sorte de raccourci vers le répertoire `~/Bureau/vmshared` du système hôte.

### Exemple 4

Supposons que l'on veuille que le répertoire `~/Bureau/vmshared` de la machine hôte apparaisse comme le répertoire `hote` dans le répertoire d'accueil de l'utilisateur `root`. Dans un terminal du système invité, il suffit de taper les commandes suivantes :

```
# mkdir ~/hote
# mount -t vboxsf hostshared ~/hote
```

⇒ le montage a été réalisé : `~/hote` est maintenant la même chose que le répertoire `~/Bureau/vmshared` du système hôte

```
# ls ~/hote
#
```

⇒ *a priori* vide dans un premier temps, on peut y récupérer/placer des fichiers (et des répertoires) qui sont/seront placés votre répertoire `~/Bureau/vmshared` du système hôte.

Lorsqu'on n'a plus besoin de ce partage, on peut le démonter en tapant :

```
# umount ~/hote
```

□

## 6 Quelques commandes réseau sous Linux

Cette section présente un rapide résumé de quelques commandes réseau étudiées dans divers TP, auxquels vous pourrez vous référer éventuellement.

### 6.1 Configuration d'une interface réseau

La configuration d'une interface réseau sous Linux se fait en utilisant la commande **ifconfig**. Dans notre cas, le synopsis suivant suffira.

#### Synopsis

```
ifconfig [-a | interface]
ifconfig interface [adresse-ip] [netmask masque] [mtu mtu] [hw ether adresse-mac]
```

La première forme sert à la consultation de la configuration courante. Si *interface* n'est pas indiquée, seule la configuration des interfaces actives (UP) est affichée. Les interfaces inactives (DOWN) sont aussi affichées si on utilise **-a**. Sinon, seule la configuration de *interface* est affichée.

La deuxième forme permet une configuration élémentaire de l'*interface* indiquée, où :

- *adresse-ip* est l'adresse IPv4 à lui attribuer ;
- **netmask** *masque* précise le masque de sous-réseau à utiliser ;
- **mtu** *mtu* précise le MTU (en octets). Il n'est généralement pas utile de le modifier ;
- **hw ether** *adresse-mac* permet d'utiliser une autre adresse MAC (Ethernet) que celle de la carte réseau.

#### Exemple 5

Supposons qu'on veuille donner à l'interface **eth0** l'adresse 150.151.152.1/16. La commande correspondante est la suivante :

```
# ifconfig eth0 150.151.152.1 netmask 255.255.0.0
```

On peut vérifier ensuite la configuration :

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:71:02:13
          inet  adr:150.151.152.1  Bcast:150.151.255.255  Masque:255.255.0.0
          adr inet6: fe80::a00:27ff:fe71:213/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:15046 errors:0 dropped:0 overruns:0 frame:0
          TX packets:290 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:1814063 (1.7 MiB)  TX bytes:60474 (59.0 KiB)
          Interruption:10 Adresse de base:0xd020
```




## 6.II Configuration statique de la table de routage

Rappelons que la table de routage se configure avec **route**, dont le synopsis suivant devrait suffire.

### Synopsis

```
route [-n]
route add -net adresse-destination netmask masque gw routeur [dev interface]
route del -net adresse-destination netmask masque gw routeur [dev interface]
```

La première forme permet d'afficher la table de routage. L'option **-n** désactive la résolution inverse DNS, ce qui est parfois bien pratique. La seconde permet d'ajouter une route passant par le routeur *routeur* vers la destination *adresse-destination* de masque *masque*. Si besoin, l'interface à utiliser pour cette route peut être indiquée avec **dev interface**. La dernière forme permet de supprimer une route.

 Il n'est pas nécessaire d'ajouter les routes directes de l'hôte (réseaux auxquels est connecté). En effet, la configuration de ses interfaces avec **ifconfig** provoque automatiquement leur ajout dans la table.

### Exemple 6

Supposons que nous voulons ajouter une route vers la destination `139.124.110.0/24` qui passe par le routeur `150.151.152.250` et **une route par défaut** qui passe par le routeur `150.151.152.153`. Dans un premier temps, on peut vérifier que l'utilisation de **ifconfig** précédemment a bien automatiquement modifié la table de routage :

```
# route -n
Table de routage IP du noyau
Destination      Passerelle      Genmask          Indic Metric Ref      Use Iface
150.151.0.0      0.0.0.0         255.255.0.0     U      0      0      0 eth0
```

On ajoute ensuite la route vers `139.124.187.0/24` :

```
# route add -net 139.124.110.0 netmask 255.255.255.0 gw 150.151.152.250
```

puis la route par défaut :

```
# route add -net 0.0.0.0 netmask 0.0.0.0 gw 150.151.152.153
```

Puis on peut vérifier en affichant la table et en pingant une station :

```
# route -n
Table de routage IP du noyau
Destination      Passerelle      Genmask          Indic Metric Ref      Use Iface
150.151.0.0      0.0.0.0         255.255.0.0     U      0      0      0 eth0
139.124.110.0    150.151.152.250 255.255.255.0   UG     0      0      0 eth0
0.0.0.0          150.151.152.153 0.0.0.0         UG     0      0      0 eth0
# ping 139.124.110.4
PING 139.124.187.4 (139.124.110.4) 56(84) bytes of data.
64 bytes from 139.124.110.4: icmp_seq=1 ttl=64 time=0.276 ms
64 bytes from 139.124.110.4: icmp_seq=2 ttl=64 time=0.485 ms
64 bytes from 139.124.110.4: icmp_seq=3 ttl=64 time=0.408 ms
64 bytes from 139.124.110.4: icmp_seq=4 ttl=64 time=0.363 ms
[CTRL]+[C]
--- 139.124.187.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2998ms
rtt min/avg/max/mdev = 0.276/0.383/0.485/0.075 ms
```

□

### 6.III Configuration par DHCP

La commande Linux permettant d'utiliser un client DHCP pour configurer une interface est **dhclient**. Dans notre cas, le synopsis suivant suffira.

#### Synopsis

```
dhclient interface
```

où *interface* est le nom de l'interface que l'on veut configurer par DHCP. La commande active un client DHCP qui cherchera un serveur DHCP pour configurer automatiquement l'interface spécifiée.

#### Exemple 7

Voici un exemple d'utilisation de cette commande sous Linux pour paramétrer l'interface **eth1** :

```
# dhclient eth1
Internet Systems Consortium DHCP Client V3.1.3
Copyright 2004-2009 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/eth0/00:21:9b:df:db:f9
Sending on   LPF/eth0/00:21:9b:df:db:f9
Sending on   Socket/fallback
DHCPDISCOVER on eth1 to 255.255.255.255 port 67 interval 5
DHCPOFFER from 139.124.187.10
DHCPREQUEST on eth1 to 255.255.255.255 port 67
DHCPACK from 139.124.187.10
bound to 139.124.187.34 -- renewal in 23348 seconds.
```

- ⇒ le serveur 139.124.187.10 a répondu et a attribué l'adresse 139.124.187.34 à cette interface. Il faudra renouveler le bail dans 23348 secondes (près de 7 heures). D'autres paramètres réseaux ont aussi été obtenus auprès du serveur.



## 7 Utilisation de TFTP

TFTP (*Trivial File Transfer Protocol*) est un protocole de transfert de fichiers, très simple, reposant sur UDP. Il est défini dans la RFC 1350. Ses fonctionnalités sont très limitées, notamment comparé à FTP. Il se limite au transfert de fichier uniquement, sans authentification ni cryptage. TFTP suit le modèle client-serveur. Le port UDP réservé aux serveurs TFTP est le port 69.

Dans les grandes lignes, le transfert de fichier entre un client et un serveur TFTP suit la procédure suivante :

1. Le client demande au serveur (port UDP 69) le transfert d'un fichier dont il indique la référence. TFTP n'offrant pas de possibilité de se déplacer dans des répertoires ni d'obtenir une liste de fichiers disponibles, le client doit connaître la référence du fichier :
  - qu'il veut récupérer s'il s'agit d'une requête de lecture (message RRQ) ;
  - qu'il veut déposer sur le serveur s'il s'agit d'une requête d'écriture (message WRQ).
2. Si le serveur accepte le transfert, il répond au client en utilisant un autre port qui sera désormais utilisé pour communiquer avec le client (envoyer/recevoir le fichier)
3. TFTP utilisant UDP, la bonne marche du transfert doit être assurée par le client et le serveur. Ils utilisent une succession de messages DATA (contenant une partie du fichier) et ACK (accusant réception d'une partie), selon un protocole de type envoyer et attendre : une partie du fichier n'est envoyée que si la partie précédente a été correctement reçue (acquittée).

On utilisera régulièrement les fonctionnalités de client TFTP du routeur afin de déposer/récupérer des fichiers de configuration sur le serveur TFTP (de la machine virtuelle) du PC-LAN. Le serveur TFTP sur le PC-LAN est **in.tftpd** (se trouvant dans `/usr/sbin/`). Il est activé au démarrage de la machine virtuelle, ce que l'on peut vérifier sur un terminal en tapant :

```
# netstat -lun
Connexions Internet actives (seulement serveurs)
Proto Recv-Q Send-Q Adresse locale Adresse distante Etat
...
udp          0      0 0.0.0.0:69      0.0.0.0:*
...
```

où l'on peut remarquer que le port 69 d'UDP est bien utilisé et que ce serveur est joignable en utilisant n'importe laquelle des adresses IP du système invité.

**i** Comme souvent pour un service assez rarement utilisé comme TFTP, c'est en réalité le **super-serveur inetd** qui écoute sur ce port. Dès qu'un datagramme parvient sur ce port, **inetd** exécute **in.tftpd** pour le traiter et tout se passera comme si **in.tftpd** avait été actif dès le départ. C'est une solution économe en processus. On peut toutefois modifier cette configuration et installer un serveur TFTP qui fonctionne en mode **standalone** : sans dépendre de **inetd** et activé dès le départ.

Dans le fichier `/etc/inetd.conf` contenant une partie de la configuration de **inetd**, on voit les paramètres de configuration du serveur TFTP :

```
# cat /etc/inetd.conf
# /etc/inetd.conf: see inetd(8) for further informations.
#
# Internet superserver configuration database
...
# <service_name> <sock_type> <proto> <flags> <user> <server_path> <args>
...
tftp          dgram        udp          wait        nobody /usr/sbin/tcpd /usr/sbin/in.tftpd /srv/tftp
...
```

où :

- **tftp** est le nom du service TFTP dans le fichier `/etc/services`, auquel est associé le port UDP 69
- **dgram** indique qu'il fonctionne en mode datagramme
- **udp** indique qu'il repose sur UDP
- **wait** indique qu'une fois activé, le serveur s'occupera de la totalité des datagrammes parvenant sur le port 69
- **nobody** est le nom d'utilisateur du (processus) serveur
- le reste est la commande exécutant le serveur :
  - ◊ `/usr/sbin/tcpd` est un *wrapper* : un programme permettant de filtrer les accès aux serveurs en interdisant ceux provenant des hôtes spécifiés dans `/etc/hosts.deny` sauf ceux de `/etc/hosts.allow`. Ces fichiers sont vides sur la machine virtuelle ;
  - ◊ `/usr/sbin/in.tftpd` est le code (exécutable) du serveur TFTP lui-même
  - ◊ `/srv/tftp` est le répertoire de travail du serveur TFTP, qui contiendra les fichiers envoyés/reçus. La référence du fichier, indiquée par le client, doit être relative à ce répertoire.

Le transfert d'un fichier par TFTP peut se faire du client vers le serveur (dépôt du fichier) et inversement (récupération du fichier).

## 7.1 Dépôt d'un fichier du routeur sur le serveur TFTP

Pour pouvoir déposer un fichier dans le répertoire de travail (`/srv/tftp`) du serveur **in.tftpd**, il faut préparer le serveur à accueillir le fichier. En effet, aucune sécurité n'est prévue par TFTP. Pour limiter un peu les risques, **in.tftpd** requiert que le fichier à déposer existe déjà (même vide) et qu'il ait le droit d'écriture dessus. Dans notre configuration, **in.tftpd** a pour identité `nobody`.

La procédure à suivre est la suivante :

1. Créer dans `/srv/tftp` le fichier (même vide) ayant le nom que le fichier à déposer portera sur le serveur ;
2. Modifier les permissions de ce fichier pour que le serveur ait le droit d'écriture dessus. Par la même occasion, on lui accordera aussi le droit de lecture, utile pour un éventuel transfert du serveur vers le routeur (client) ;
3. Le serveur est maintenant prêt, on peut lancer la commande sur le routeur pour déposer le fichier.

### Exemple 8

Supposons que le serveur TFTP soit opérationnel sur la station d'adresse `150.151.152.1` et que l'on veuille y déposer la configuration courante (`running-config`) du routeur en l'appelant `config-routeur.txt`.

On prépare d'abord le serveur pour accueillir le fichier `config-routeur.txt` :

```
# touch /srv/tftp/config-routeur.txt
# chmod a+rw /srv/tftp/config-routeur.txt
# ls -l /srv/tftp/config-routeur.txt
-rw-rw-rw- 1 root root 0 mai  3 13:30 /srv/tftp/config-routeur.txt
```

- ⇒ création du fichier cible vide, et ajout des droits de lecture/écriture pour tout le monde sur ce fichier. La dernière commande sert juste à vérifier que tout est ok



Sur le routeur et en mode privilégié, on peut exécuter la commande demandant le transfert. Cette commande demande quelques informations :

```
Router#copy running-config tftp:config-routeur.txt
Address or name of remote host []? 150.151.152.1
Destination filename [config-routeur.txt]? 
!!
709 bytes copied in 4.792 secs (148 bytes/sec)
```

⇒ c'est tout : le dépôt du fichier a réussi

Sur le serveur, on peut maintenant voir que le fichier n'est plus vide :

```
# ls -l /srv/tftp/config-routeur.txt
-rw-rw-rw- 1 root root 709 mai  3 13:35 /srv/tftp/config-routeur.txt
# cat /srv/tftp/config-routeur.txt

!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
...
```



## 7.II Récupération sur le routeur d'un fichier du serveur TFTP

L'opération est encore plus simple que la précédente car on n'a pas à préparer le serveur s'il a déjà le droit de lecture sur le fichier demandé.

### Exemple 9

On poursuit l'exemple précédent, mais cette fois on récupère le fichier `config-routeur.txt` et on le sauve dans la NVRAM sous le nom `running.old` :

```
Router#copy tftp:config-routeur.txt nvram:running.old
Address or name of remote host []? 150.151.152.1
Destination filename [running.old]? 
Accessing tftp://150.151.152.1/config-routeur.txt...
Loading config-routeur.txt from 150.151.152.1 (via FastEthernet0): !
[OK - 709 bytes]

709 bytes copied in 9.396 secs (75 bytes/sec)
```

⇒ le fichier a été récupéré avec succès

```
Router#dir nvram:
Directory of nvram:/

 27  -rw-          596          <no date>  startup-config
 28  ----           5          <no date>  private-config
  1  -rw-           0          <no date>  ifIndex-table
  2  ----          12          <no date>  persistent-data
  3  -rw-          709          <no date>  running.old

29688 bytes total (25963 bytes free)
```

↪ et figure bien dans la NVRAM.

**i** On aurait pu tout aussi bien le copier directement à la place de la `running-config` ou de la `startup-config`.



## 8 Analyse de trafic réseau avec Wireshark

Au cours du TP, il vous sera parfois demandé d'analyser et de commenter le trafic réseau. Pour cela, vous devrez utiliser l'outil libre d'analyse de trafic réseau **wireshark**. Des versions pour de très nombreuses plateformes sont disponibles à partir de son site Web officiel (<http://www.wireshark.org>). Nous utiliserons la version installée sur la machine virtuelle, que nous avons déjà utilisée en TP.

**Wireshark** a de nombreuses fonctionnalités parmi lesquelles :

- la capture de trames circulant sur le réseau. Cette opération nécessite **les droits d'administration** pour placer l'interface réseau en **mode promiscuous** afin de garder toutes les trames qui y parviennent. La capture est réalisée via la bibliothèque **libpcap** (sous Linux) ou son équivalent sous d'autres plateformes. Les trames capturées peuvent être sauvegardées dans un fichier au format PCAP (ou l'un des multiples formats proposés), pour une analyse ultérieure. Des filtres de captures permettent de limiter les trames qui seront capturées. Nous y reviendrons dans la section 8.II ;

**i** Notons qu'on ne peut capturer une trame que si elle parvient à l'interface réseau. L'utilisation d'un switch plutôt qu'un hub limite les possibilités d'écoute du réseau. Cependant, certains switch permettent d'activer une fonction de monitoring sur un port, sur lequel sera renvoyé toutes les trames arrivant au switch.

- l'analyse des trames capturées ou chargées depuis un fichier. L'analyse ne s'arrête pas au niveau 2, mais continue aux niveaux supérieurs tant que **wireshark** reconnaît les protocoles (et il en reconnaît des centaines). Par exemple, si une trame Ethernet contient un datagramme IP, il sera aussi analysé. De même, si le datagramme IP contient un segment TCP, il sera aussi analysé. Enfin, si ce segment contient un message HTTP, il sera aussi analysé ;
- l'affichage des informations des protocoles d'une trame donnée ;
- le suivi de "dialogues" qui lie entre elles les trames d'un même dialogue, et qui permet d'établir de nombreuses statistiques sur le trafic et les discussions, y compris les débits et temps de réponse, des graphes, etc.

On comprend que c'est un des outils privilégiés des pirates. Notamment, il permet de capturer les identifiants et mots de passe transmis en clair (sans cryptage) sur le réseau où il est utilisé.

Mais c'est aussi un outil indispensable pour les administrateurs réseaux car il aide à la détection des causes d'un dysfonctionnement réseau (trames perdues/dupliquées, temps de réponse anormalement élevés, etc.) ou d'un éventuel trafic suspect.

### 8.1 Interface de Wireshark

**Wireshark** est souvent installé avec un ensemble d'outils en ligne de commande, mais il est principalement utilisé via son interface graphique. Sur notre machine virtuelle, c'est la commande **wireshark** qu'on peut lancer depuis le menu *Applications* → *Internet* → *Wireshark*.

Cette interface (voir figure 12) permet de réaliser toutes les opérations dont vous aurez besoin. Elle est composée principalement de 4 parties qui seront décrites dans les sections suivantes.

#### 8.1.A Barres de menu et d'actions

Dans la zone marquée ① de la figure 12, se trouve en haut la barre de menu qui permet d'accéder à toutes les fonctions. En particulier, le menu *File* permet de sauver/charger tout ou partie d'une capture, le menu *Capture*

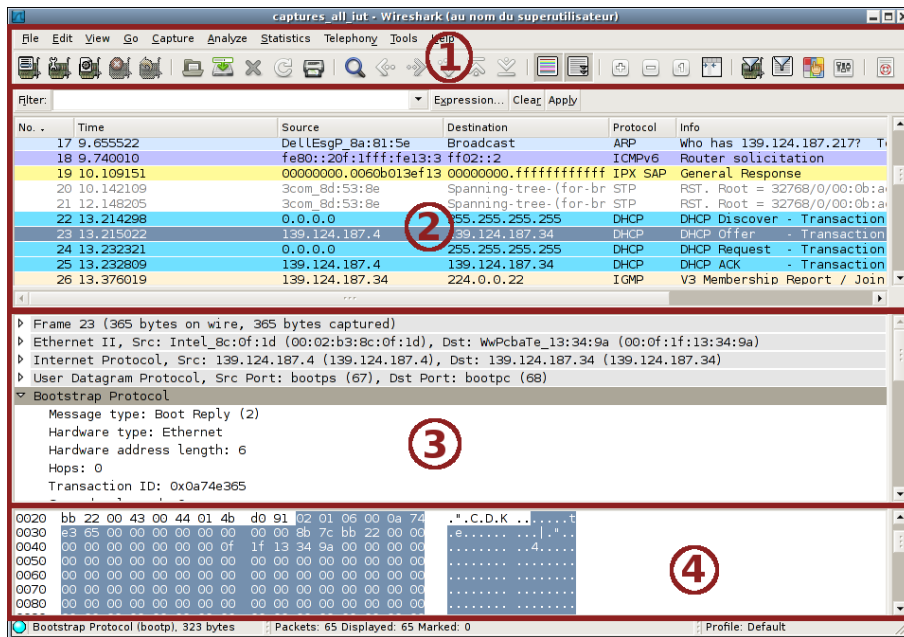


FIGURE 12 – Interface de Wireshark

permet de démarrer/arrêter/planifier une capture en précisant les interfaces à utiliser, et le menu *Analyse* permet de créer des filtres d'affichage.

Sous la barre de menu, on trouve une barre d'actions rapides (voir figure 13) regroupant les fonctionnalités les plus utilisées :

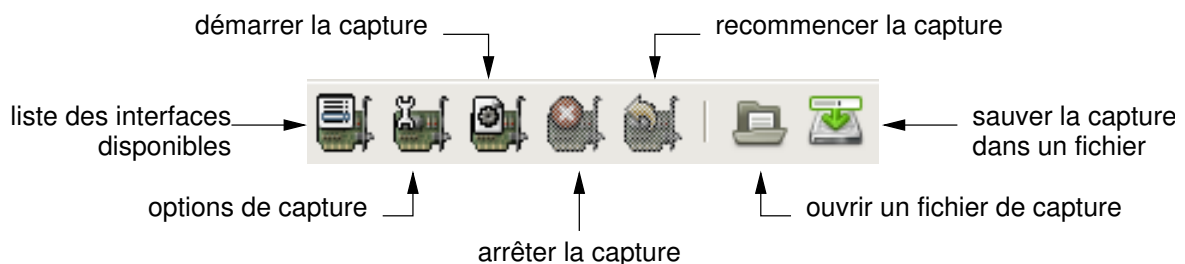


FIGURE 13 – Extrait de la barre d'actions


-  correspond au menu *Capture* → *Interfaces*. Ce bouton ouvre une boîte de dialogue (figure 14) montrant la liste des interfaces disponibles. On peut choisir sur quelle interface démarrer la capture en



FIGURE 14 – Boîte de dialogue listant les interfaces

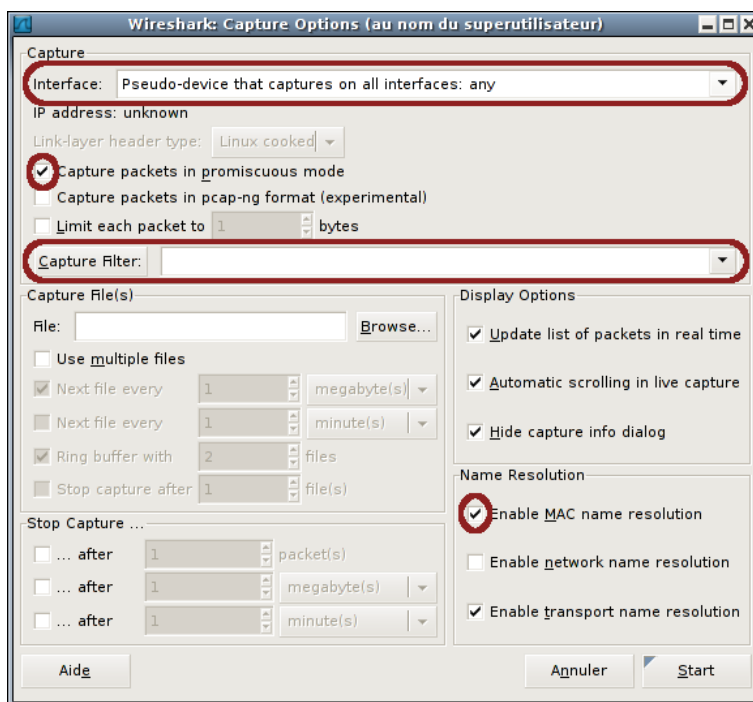


FIGURE 15 – Boîte de dialogue des options de capture

cliquant sur le bouton *Start*, ou définir des options de capture en cliquant sur *Options*. Notons que pour capturer des trames sur toutes les interfaces disponibles, il faut utiliser l'interface virtuelle **any**.

- correspond au menu *Capture* → *Options* et ouvre une boîte de dialogue (figure 15) permettant de configurer la capture. Si, en principe, la majorité des options ont une valeur par défaut satisfaisante, certaines méritent une attention particulière :
  - ◇ dans la zone *Interface* une liste déroulante permet de choisir l'interface sur laquelle la capture doit se faire (ou **any** pour toutes les interfaces) ;
  - ◇ la case "*Capture packets in promiscuous mode*" **doit être cochée** pour capturer toutes les trames qui arrivent à l'interface, sinon seules celles dont la destination correspond à l'interface seraient capturées ;
  - ◇ la zone *Capture Filter* permet de limiter la capture aux trames satisfaisant un filtre. Une liste de filtres prédéfinis est proposée dans la liste déroulante et le bouton *Capture Filter* permet de définir un filtre à partir d'expressions. Nous y reviendrons dans la section 8.II.A ;
  - ◇ la case *Enable MAC name resolution*, si cochée, active la résolution d'adresses MAC. Notamment, pour Ethernet, cela fait afficher `Broadcast` plutôt que l'adresse `ff:ff:ff:ff:ff:ff`. De plus, les 3 premiers octets des adresses Ethernet codant le constructeur, ils sont remplacés par le constructeur en question. Par exemple, l'adresse `00:18:8b:1e:b6:cc` sera affichée `Dell_1e:b6:cc`. Cette fonctionnalité ne rend pas forcément l'affichage plus lisible...
- correspond au menu *Capture* → *Start* et commence la capture avec les options définies. Les trames capturées sont analysées en temps réel. Celles respectant le filtre d'affichage sont affichées au fur et à mesure dans la zone ② de la figure 12, et la première trame affichée est détaillée dans les zones ③ et ④. Ces zones sont décrites dans les sections 8.I.B et 8.I.C.
- n'est cliquable que lorsqu'une capture est en cours. Il correspond au menu *Capture* → *Stop* et arrête la capture en cours.

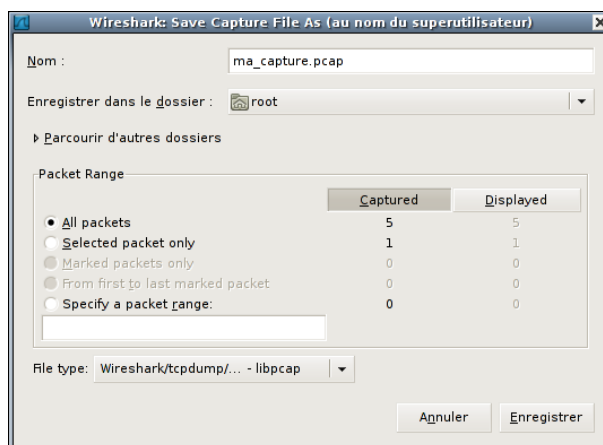





FIGURE 16 – Boîte de dialogue de sauvegarde de trames

-  n'est cliquable que lorsqu'une capture est en cours. Il correspond au menu *Capture* → *Restart* et a pour effet d'effacer les trames capturées jusque là et de continuer la capture.
-  correspond au menu *File* → *Open* et permet de charger un fichier contenant une capture (efface les trames courantes).
-  correspond au menu *File* → *Save* et permet de sauver les trames selon quelques critères. Cela ouvre la boîte de dialogue de la figure 16 dans laquelle on peut indiquer si l'on veut sauver les trames capturées ou affichées, sélectionnées, marquées ou un intervalle de trames. Le format du fichier par défaut est PCAP.

## 8.I.B Liste des trames

La liste des trames capturées et respectant le filtre d'affichage figure dans la zone ② de la figure 12, reprise dans la figure 17. En haut de cette zone se trouve le filtre d'affichage spécifié. En dessous, figure la liste des trames respectant les critères de ce filtre, à raison d'une trame par ligne. Les lignes sont colorées en fonction des protocoles reconnus dans les trames. Les colonnes affichées sont configurables via le menu *Edit* → *Preferences*. Par défaut, il y en a 6 : le numéro de la trame capturée, la date de capture depuis son début, sa source, sa destination, le protocole reconnu de plus haut niveau, et une présentation humainement compréhensible de l'information transportée pour ce protocole.

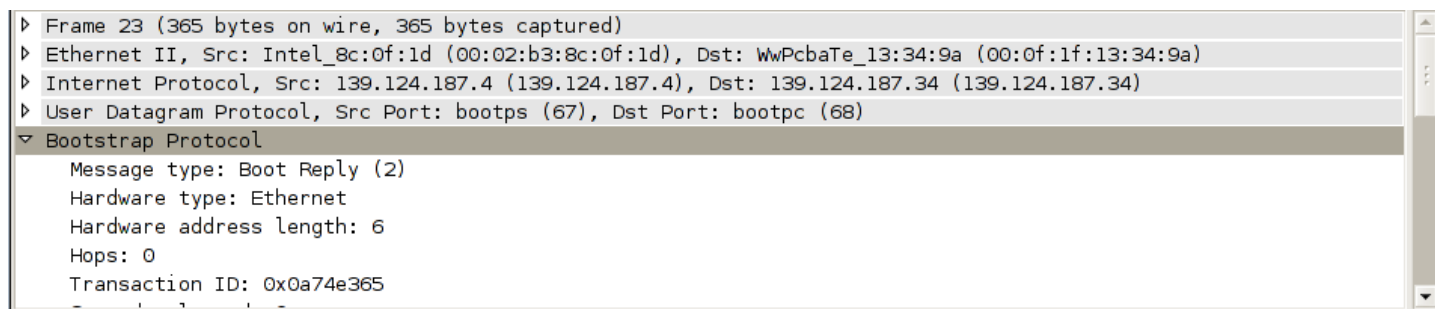
No. -	Time	Source	Destination	Protocol	Info
17	9.655522	DellEsgP_8a:81:5e	Broadcast	ARP	Who has 139.124.187.217? T
18	9.740010	fe80::20f:1fff:fe13:3	ff02::2	ICMPv6	Router solicitation
19	10.109151	00000000.0060b013ef13	00000000.ffffffffffff	IPX SAP	General Response
20	10.142109	3com_8d:53:8e	Spanning-tree-(for-br	STP	RST. Root = 32768/0/00:0b:a
21	12.148205	3com_8d:53:8e	Spanning-tree-(for-br	STP	RST. Root = 32768/0/00:0b:a
22	13.214298	0.0.0.0	255.255.255.255	DHCP	DHCP Discover - Transaction
23	13.215022	139.124.187.4	139.124.187.34	DHCP	DHCP Offer - Transaction
24	13.232321	0.0.0.0	255.255.255.255	DHCP	DHCP Request - Transaction
25	13.232809	139.124.187.4	139.124.187.34	DHCP	DHCP ACK - Transaction
26	13.376019	139.124.187.34	224.0.0.22	IGMP	V3 Membership Report / Join

FIGURE 17 – Filtre d'affichage et liste des trames correspondantes

### 8.I.C Détails des protocoles reconnus dans une trame

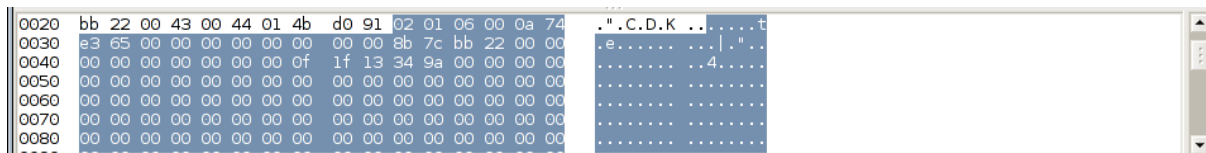
En sélectionnant une trame dans la liste précédente, comme la trame 23 sur la figure 17, les informations qui en ont été extraites sont présentées dans la zone ③ de la figure 12, reprise dans la figure 18. Dans cette zone, les informations apparaissent sous la forme d'une arborescence. En cliquant sur le symbole ▷ en début d'une ligne on la développe sur plusieurs lignes afin d'en avoir plus de détails. Au contraire, en cliquant sur ▽ on réduit l'affiche de l'information à une seule ligne. En effectuant un clic droit sur une information, on accède à un menu permettant notamment de la copier, ou de la sauver dans un fichier.

Sur la figure 18, on voit que dans la trame 23 sont reconnus les protocoles Ethernet-V2, IP, UDP et BOOTP (en réalité DHCP), dont les informations peuvent être détaillées.



**FIGURE 18** – Affichage des détails sur les protocoles reconnus dans une trame

Les octets de la trame sélectionnée sont affichés dans la zone ④ de la figure 12, reprise dans la figure 19.



**FIGURE 19** – Mise en évidence des octets d'une information

Cette zone est constituée de 3 parties :

- au centre, les octets de la trame sont affichés en hexadécimal à raison de 2 séries de 8 octets par ligne (soit 16 octets en tout, écrits avec 32 digits hexadécimaux) ;
- à droite, ces octets sont affichés sous la forme de caractères ASCII, en remplaçant les caractères non imprimables par un point (.)
- à gauche, un nombre en hexadécimal donne le numéro du premier octet de la ligne.

Lorsqu'on sélectionne une information dans la zone ③, les octets qui la constituent sont mis en évidence dans la zone ④. Sur les figures 18 et 19, l'information sélectionnée est le message DHCP et les octets qui le constituent sont mis en évidence. Inversement, lorsqu'on clique sur des octets (hexadécimaux ou ASCII) de la zone ④, l'information correspondante est développée dans la zone ③.

❗ Notons qu'en sélectionnant un protocole tel qu'Ethernet, IP, UDP, TCP, qui transporte un message d'un protocole de niveau supérieur, seul son en-tête est mis en évidence, puisque les données sont des informations de cet autre protocole.

## 8.II Filtres de capture et d'affichage

L'utilisation conjointe de filtres de capture et de filtres d'affichage a son intérêt :

- le **filtre de capture** permet de limiter le volume de trames capturées. Bien qu'il soit en principe plutôt conseillé de ne pas filtrer les trames à capturer pour ne pas passer à côté de problèmes éventuels, cela peut se traduire par une trop grande quantité de données à enregistrer, surtout si la capture dure longtemps ;
- le **filtre d'affichage** restreint l'affichage des trames à celles satisfaisant le filtre, les autres étant invisibles. Il permet de se concentrer sur un sous-ensemble des trames capturées, ce qui rend l'analyse de celles-ci plus confortable. Au cours de l'analyse d'une capture, on peut utiliser successivement plusieurs filtres d'affichage selon les informations que l'on cherche et que l'on découvre.

Ce qui peut paraître surprenant et ennuyeux est que la syntaxe des filtres de capture est différente de celle des filtres d'affichage. Cela est dû au fait que **Wireshark** utilise la bibliothèque **libpcap** pour la capture de trames qui ne reconnaît qu'un certain langage pour ses filtres. Ce langage n'est pas forcément très ergonomique mais doit être utilisé par de nombreux outils basés sur **libpcap**, tels que **tcpdump**.

Pour définir un filtre des trames à afficher d'une capture (réalisée, en cours ou chargée), il faut utiliser le langage propre à **Wireshark**, qui est à la fois plus ergonomique et plus puissant que celui de la **libpcap**.

### 8.II.A Expression d'un filtre de capture avec la libpcap

Un filtre de capture **libpcap** est exprimé avec des **primitives**, des **connecteurs** et les parenthèses. La syntaxe complète de ces filtres est décrite dans le manuel de **pcap-filter** (**man 7 pcap-filter**).

Nous nous contenterons d'une syntaxe allégée, avec des primitives simples. Dans ce contexte, un filtre suit la grammaire suivante :

```

filtre ::= [not] primitive { (and|or) [not] primitive }
primitive ::= ( filtre )
primitive ::= l'une des primitives ci-dessous
  
```

#### Exemple 10

Voici un petit aperçu de filtres possibles :

- **ether host 00:21:ab:28:10:c5**  
ne retient que les trames dont l'adresse MAC source ou destination est 00:21:ab:28:10:c5
- **tcp port 23 and host 10.0.0.5**  
ne retient que le trafic TELNET (car le port TCP 23 est *a priori* réservé aux serveurs TELNET) impliquant l'hôte 10.0.0.5 (en tant que serveur ou client)
- **tcp port 23 and not src host 10.0.0.5**  
ne retient que le trafic TELNET, et ne provenant pas de l'hôte 10.0.0.5
- **host 10.0.0.5 and not (port 80 or port 25)**  
ne retient que le trafic hors HTTP et SMTP, impliquant l'hôte 10.0.0.5
- **host 10.0.0.5 and icmp[icmptype] == icmp-echo**  
ne retient que les messages ICMP de type ECHO REQUEST dont la source ou la destination est 10.0.0.5

□

Il existe des primitives pour tous les protocoles reconnus par la **libpcap**. Nous nous limiterons à un sous-ensemble des primitives relatives aux protocoles qui nous occupent.



### 8.II.A.1 Primitives pour Ethernet

Ces primitives ne peuvent être vraies que pour les trames Ethernet, FDDI, Token Ring et Wifi.

- **ether host** *ehost*  
vrai si l'adresse MAC source ou destination de la trame Ethernet est *ehost*
- **ether dst** *ehost*  
vrai si l'adresse MAC destination de la trame Ethernet est *ehost*
- **ether src** *ehost*  
vrai si l'adresse MAC source de la trame Ethernet est *ehost*
- **ether proto** *protocol*  
vrai si le champ *EtherType* de la trame Ethernet est *protocol*, qui peut être un nombre ou un nom tel que **ip**, **ip6**, **arp**, **rarp**,... Attention toutefois car ces noms sont aussi des mots clés de primitives et il faut les protéger avec un *backslash*

### 8.II.A.2 Primitives pour IP

Ces primitives ne peuvent être vraies que si la trame transporte un datagramme IP. Cela rend implicite l'un des critères **ether proto ip** et **ether proto ip6**.

- **host** *host*  
vrai si l'adresse source ou destination (IPv4/v6) du datagramme est *host*
- **dst host** *host*  
vrai si l'adresse destination (IPv4/v6) du datagramme est *host*, qui peut être une adresse ou un nom d'hôte
- **src host** *host*  
vrai si l'adresse source (IPv4/v6) du datagramme est *host*
- **dst net** *net* [**mask** *mask*]  
vrai si l'adresse destination (IPv4/v6) du datagramme fait partie du réseau *net* de masque *mask*. Si le masque n'est pas précisé, *net* peut être exprimé par un seul octet (masque 255.0.0.0), deux octets *x.y* (masque 255.255.0.0) ou trois octets *x.y.z* (masque 255.255.255.0)
- **src net** *net* [**mask** *mask*]  
vrai si l'adresse source (IPv4/v6) du datagramme fait partie du réseau *net*
- **net** *net* [**mask** *mask*]  
vrai si l'adresse source ou destination (IPv4/v6) du datagramme fait partie du réseau *net*
- **ip proto** *protocol*  
vrai s'il s'agit d'un datagramme IPv4 dont le champ *Protocole* est *protocol*, qui peut être un nombre ou un nom tel que **icmp**, **udp**, **tcp**,... Attention toutefois car ce sont aussi des mots clés de primitives et il faut les protéger avec un *backslash* !

### 8.II.A.3 Primitives communes à UDP et TCP

Ces primitives ne peuvent être vraies que si la trame transporte un datagramme UDP ou un segment TCP (inclus dans un datagramme IP).

- [**udp|tcp**] **port** *port*  
vrai si le port source ou destination du segment TCP/datagramme UDP est *port*. Le mot clé optionnel du début (**udp** ou **tcp**) précise s'il doit s'agir d'un datagramme UDP ou d'un segment TCP. En son absence, les deux conviennent. *port* peut être un nombre ou un nom figurant dans le fichier */etc/services*. Si un nom est utilisé et qu'il est spécifique à un TCP ou à UDP, seuls les paquets correspondant à ce protocole seront retenus

- `[udp|tcp] dst port port`  
vrai si le port destination est *port*
- `[udp|tcp] src port port`  
vrai si le port source du segment TCP/datagramme UDP est *port*
- `[udp|tcp] portrange port1-port2`  
vrai si le port source ou destination du segment TCP/datagramme UDP est compris entre *port<sub>1</sub>* et *port<sub>2</sub>*
- `[udp|tcp] dst portrange port1-port2`  
vrai si le port destination du segment TCP/datagramme UDP est compris entre *port<sub>1</sub>* et *port<sub>2</sub>*
- `[udp|tcp] src portrange port1-port2`  
vrai si le port source du segment TCP/datagramme UDP est compris entre *port<sub>1</sub>* et *port<sub>2</sub>*

#### 8.II.A.4 Cas particulier des primitives spécifiques à ICMP

Ces primitives ne peuvent être vraies que si la trame transporte un message ICMP. La syntaxe est un peu différente des primitives précédentes.

- `icmp`  
vrai si la trame transporte un message ICMP
- `icmp[icmptype] (==|!=) valeur`  
vrai si le type du message ICMP vaut (==) ou est différent de (!=) *valeur*, où *valeur* peut être un nombre ou un mot clé tel que `icmp-echo` (ECHO REQUEST) ou `icmp-echo-reply`
- `icmp[icmpcode] (==|!=) valeur`  
vrai si le code du message ICMP vaut (==) ou est différent de (!=) *valeur*, où *valeur* peut être un nombre ou un mot clé

**i** Ainsi qu'il a été dit, il existe de nombreuses autres primitives, et la possibilité d'exprimer des conditions plus fines/complexes mais cela sort du cadre de ce TP. Voir le manuel de **pcap-filter** pour des exemples.

## 8.II.B Les filtres d'affichage

### 8.II.B.1 Syntaxe d'un filtre d'affichage

Les filtres d'affichage de **wireshark** sont plus "orienté objets" que les filtres de capture de **libpcap**. Ce sont des critères sur des protocoles ou sur les valeurs des champs de protocoles. Le critère ne peut être vrai que si la trame contient effectivement un PDU du protocole spécifié et dont les champs satisfont les conditions exprimées.

Un critère est exprimé en indiquant le nom du protocole suivi éventuellement d'un champ et de la condition qu'il doit vérifier. Le nom du protocole doit correspondre à l'identificateur que reconnaît<sup>5</sup> **wireshark**, par exemple, `eth`, `ip`, `arp`, `icmp`, `udp`, `tcp`, `http`, `smtp`, `bootp`, `dns`...

#### 8.II.B.1.a Filtre par protocole

Si l'on ne veut afficher que les messages d'un certain protocole pour lequel un identificateur existe, il suffit d'indiquer cet identificateur.


5. En réalité, **wireshark** utilise des dissecteurs qui sont des plugin reconnaissant les protocoles et fournissant les moyens d'exprimer des conditions les concernant.

## Exemple 11

Voici quelques exemples de filtres utilisant seulement des identificateurs de protocoles :

- **arp**  
n'affiche que les trames véhiculant un datagramme ARP
- **tcp**  
n'affiche que les trames véhiculant un segment TCP (lui-même contenu dans un datagramme IP)
- **http**  
n'affiche que les trames véhiculant du trafic HTTP



 Notons que les (PDU des) protocoles ne sont recherchés (et reconnus) que dans les conditions normales d'utilisation. Notamment, un datagramme ARP n'est recherché que si le champ *EtherType* de la trame vaut `0x0806`. De même, les messages HTTP ne sont recherchés que dans les segments TCP dont le port source ou destination vaut 80.

### 8.II.B.1.b Filtre selon la valeur d'un champ

Pour exprimer des conditions sur les champs des PDU, on utilise la forme :

*protocole . champ*


où *champ* est l'identificateur du champ (ou de la propriété) à tester. La section 8.II.B.2 présente un extrait des champs disponibles. La liste complète des champs est accessible via l'URL <http://www.wireshark.org/docs/dfref/>.

Le champ peut être comparé à une valeur en utilisant l'un des opérateurs `==`, `!=`, `<`, `<=`, `>`, `>=`. Le type d'opérateur permis ainsi que les valeurs à comparer dépendent du champ.

## Exemple 12


Voici quelques exemples de filtres selon la valeur d'un champ :

- **eth.addr == 00:1f:4b:c8:05:e0**  
n'affiche que les trames Ethernet dont l'adresse MAC source ou destination est `00:1f:4b:c8:05:e0`

 On peut remarquer que **eth.addr** est une propriété dans la mesure où ce champ n'existe pas en réalité et représente une alternative entre deux champs réels.


- **eth.type == 0x0800**  
n'affiche que les trames Ethernet transportant un datagramme IP
- **arp.opcode == 1**  
n'affiche que les trames véhiculant une **requête** ARP

- `ip.hdr_len > 20`  
n'affiche que les trames véhiculant des datagrammes IP comportant des options (en-tête de taille supérieure à 20 octets)

 On peut remarquer que `ip.hdr_len` contient la taille réelle de l'en-tête, c'est à dire la valeur du champ *HLEN* du datagramme, multipliée par 4

- `ip.src == 10.0.0.5`  
n'affiche que les trames véhiculant un datagramme IP avec pour adresse source 10.0.0.5
- `ip.src != 10.0.0.0/8`  
n'affiche que les trames véhiculant un datagramme IP ne provenant pas du réseau 10.0.0.0/8
- `http.request.method == "POST"`  
n'affiche que les trames véhiculant des requêtes HTTP de type POST



 Comme on le voit dans le dernier exemple, on peut aussi tester des sous-champs pour certains protocoles.

### 8.II.B.1.c Test d'un champ booléen


Lorsque le champ a une valeur booléenne, il n'y a pas de comparaison à faire : il faut seulement le nommer (comme on le ferait pour une variable booléenne). Seules seront affichées les trames pour lesquelles ce champ existe et est vrai.

#### Exemple 13

Voici quelques exemples de filtres avec un champ booléen :

- `ip.flags.df`  
n'affiche que les trames véhiculant un datagramme IP dont le bit *Don't Fragment* est positionné
- `tcp.flags.syn`  
n'affiche que les trames véhiculant un établissement de connexion TCP
- `tcp.checksum_bad`  
n'affiche que les trames véhiculant un segment TCP dont le checksum est incorrect



 Comme on le voit dans le dernier exemple, certains "champs" booléens sont fictifs et expriment une qualité sur la valeur d'un champ effectif.


### 8.II.B.1.d Connecteurs

Enfin, on peut construire des expressions composées en utilisant les connecteurs ! (négation), && (et), || (ou), ainsi que les parenthèses.

#### Exemple 14

- `eth.src == 00:1f:4b:c8:05:e0 && arp.opcode == 1`  
n'affiche que les trames dont l'adresse MAC source est `00:1f:4b:c8:05:e0` et contenant une requête ARP
- `ip.src != 10.0.0.0/8 && ! http`  
n'affiche que les trames véhiculant un datagramme IP ne provenant pas du réseau `10.0.0.0/8` et qui ne concernent pas du trafic HTTP
- `ip && ! (ip.addr == 10.0.0.5)`  
n'affiche que les trames contenant des datagrammes IP dont ni la source ni la destination est `10.0.0.5`.



 Dans le dernier exemple, il aurait été incorrect de remplacer le test `! (ip.addr == 10.0.0.5)` par seulement `ip.addr != 10.0.0.5`, car cette dernière expression est vraie si l'une ou l'autre des adresses source ou destination est différente de `10.0.0.5`. Aussi, l'expression `! (ip.addr == 10.0.0.5)` seule est insuffisante car elle serait vraie aussi pour les trames ne contenant pas de datagramme IP (le champ `ip.addr` n'existe pas, donc l'égalité est fautive, et sa négation est vraie) qui seraient donc aussi affichées.

## 8.II.B.2 Extrait des identificateurs et champs disponibles

### 8.II.B.2.a Identificateur et champs pour Ethernet

L'identificateur de protocole `eth` qualifie les trames Ethernet-V2 et IEEE 802.3. La liste complète de ses champs est disponible à l'URL <http://www.wireshark.org/docs/dfref/e/eth.html>.

Extrait des champs susceptibles de nous intéresser :

- `eth.addr` (6-byte hardware address)  
adresse MAC source ou destination de la trame
- `eth.dst` (6-byte hardware address)  
adresse MAC destination de la trame
- `eth.src` (6-byte hardware address)  
adresse MAC source de la trame
- `eth.type` (unsigned 16-bit integer)  
champ `EtherType` (uniquement pour Ethernet-V2)
- `eth.len` (unsigned 16-bit integer)  
longueur de la trame (uniquement pour IEEE 802.3)

 Leur signification est indiquée dans le document d'URL <http://infodoc.iut.univ-aix.fr/~cpb/enseignement/reseaux/docs/ethernet/ethernet.pdf>.

### 8.II.B.2.b Identificateur et champs pour IP

L'identificateur de protocole **ip** qualifie les trames contenant un datagramme IPv4. La liste complète de ses champs est disponible à l'URL <http://www.wireshark.org/docs/dfref/i/ip.html>.

Extrait des champs susceptibles de nous intéresser :

- **ip.addr** (*IPv4 address*)  
adresse IPv4 source ou destination du datagramme
- **ip.dst** (*IPv4 address*)  
adresse IPv4 destination du datagramme
- **ip.src** (*IPv4 address*)  
adresse IPv4 source du datagramme
- **ip.proto** (*unsigned 8-bit integer*)  
champ *Protocole* du datagramme
- **ip.hdr\_len** (*unsigned 8-bit integer*)  
taille réelle de l'en-tête du datagramme (valeur du champ HLEN multipliée par 4)
- **ip.len** (*unsigned 16-bit integer*)  
taille totale du datagramme

❗ Leur signification est indiquée dans le document d'URL <http://infodoc.iut.univ-aix.fr/~cpb/enseignement/reseaux/docs/ip/ip.pdf>.


### 8.II.B.2.c Identificateur et champs pour TCP

L'identificateur de protocole **tcp** qualifie les trames contenant un segment TCP. La liste complète de ses champs est disponible à l'URL <http://www.wireshark.org/docs/dfref/t/tcp.html>.

Extrait des champs susceptibles de nous intéresser :

- **tcp.port** (*unsigned 16-bit integer*)  
Port source ou destination du segment
- **tcp.dstport** (*unsigned 16-bit integer*)  
Port destination du segment
- **tcp.srcport** (*unsigned 16-bit integer*)  
Port source du segment
- **tcp.hdr\_len** (*unsigned 8-bit integer*)  
taille réelle de l'en-tête du segment (valeur du champ LET multipliée par 4)
- **tcp.len** (*unsigned 16-bit integer*)  
taille totale du segment
- **tcp.flags.syn** (*boolean*)  
vrai si le bit SYN est positionné (établissement de connexion)
- **tcp.flags.ack** (*boolean*)  
vrai si le bit ACK est positionné (accusé de réception)
- **tcp.flags.fin** (*boolean*)  
vrai si le bit FIN est positionné (fin de connexion)

- **tcp.flags.reset** (*boolean*)  
vrai si le bit RST est positionné (fin brutale de connexion)

 Leur signification est indiquée dans les transparents du cours consacré à TCP.

### 8.II.B.2.d Identificateur et champs pour UDP

L'identificateur de protocole **udp** qualifie les trames contenant un datagramme UDP. La liste complète de ses champs est disponible à l'URL <http://www.wireshark.org/docs/dfref/u/udp.html>.

Extrait des champs susceptibles de nous intéresser :

- **udp.port** (*unsigned 16-bit integer*)  
Port source ou destination du datagramme
- **udp.dstport** (*unsigned 16-bit integer*)  
Port destination du datagramme
- **udp.srcport** (*unsigned 16-bit integer*)  
Port source du datagramme
- **udp.length** (*unsigned 16-bit integer*)  
taille totale du datagramme

### 8.II.B.2.e Identificateur et champs pour ICMP

L'identificateur de protocole **icmp** qualifie les trames contenant un message ICMP. La liste complète de ses champs est disponible à l'URL <http://www.wireshark.org/docs/dfref/i/icmp.html>.

Extrait des champs susceptibles de nous intéresser :

- **icmp.code** (*unsigned 8-bit integer*)  
code du message
- **icmp.type** (*unsigned 8-bit integer*)  
type du message pour le code donné
- **icmp.ident** (*unsigned 16-bit integer*)  
identificateur du message (présence dépendant du message)

 Leur signification est indiquée dans le document d'URL <http://infodoc.iut.univ-aix.fr/~cpb/enseignement/reseaux/docs/icmp/icmp.pdf>.