Connexions Téléphoniques au Réseau Jussieu

Jacky Thibault
Centre de Calcul Recherche
Université Pierre et Marie Curie
Email: Jacky.Thibault@ccr.jussieu.fr

Version 2
31 décembre 1993

Table des matières

1 Introduction

Une nouvelle génération de modems à des prix abordables est apparue. Ce sont les `V32.bis` (couramment appelés "*14400 Bauds*").

Leur premier avantage est d'apporter une vitesse de transmission rendant bien plus confortable le travail en mode terminal ou le transfert de fichiers.

Le deuxième avantage est, qu'à cette vitesse, il devient possible d'utiliser les protocoles `IP` sur une ligne téléphonique, faisant de votre machine (micro-ordinateur, station, terminal `X` chez vous) une partie du réseau auquel elle se connecte pendant le temps de la connexion. Vous pouvez alors utiliser "*rlogin*" ou "*telnet*" pour ouvrir plusieurs sessions simultanément, voire ouvrir des fenêtres `X` ou utiliser des services comme "*xgopher*" si vous disposez d'un terminal `X`.

Le protocole `IP` (avec `TCP` au-dessus dans notre cas) ajoute un tel surcroît de trafic ("*overhead*") qu'avec des vitesses inférieures, cela est lent (à 9600 Bauds), voire prohibitif (à 2400), ou impossible (à 1200 Bauds) !

Le protocole permettant d'utiliser `IP` sur une ligne téléphonique est soit `SLIP` (ancien et frustre), soit `PPP` (récent, fiable, sophistiqué mais actuellement un peu moins répandu que `SLIP`).

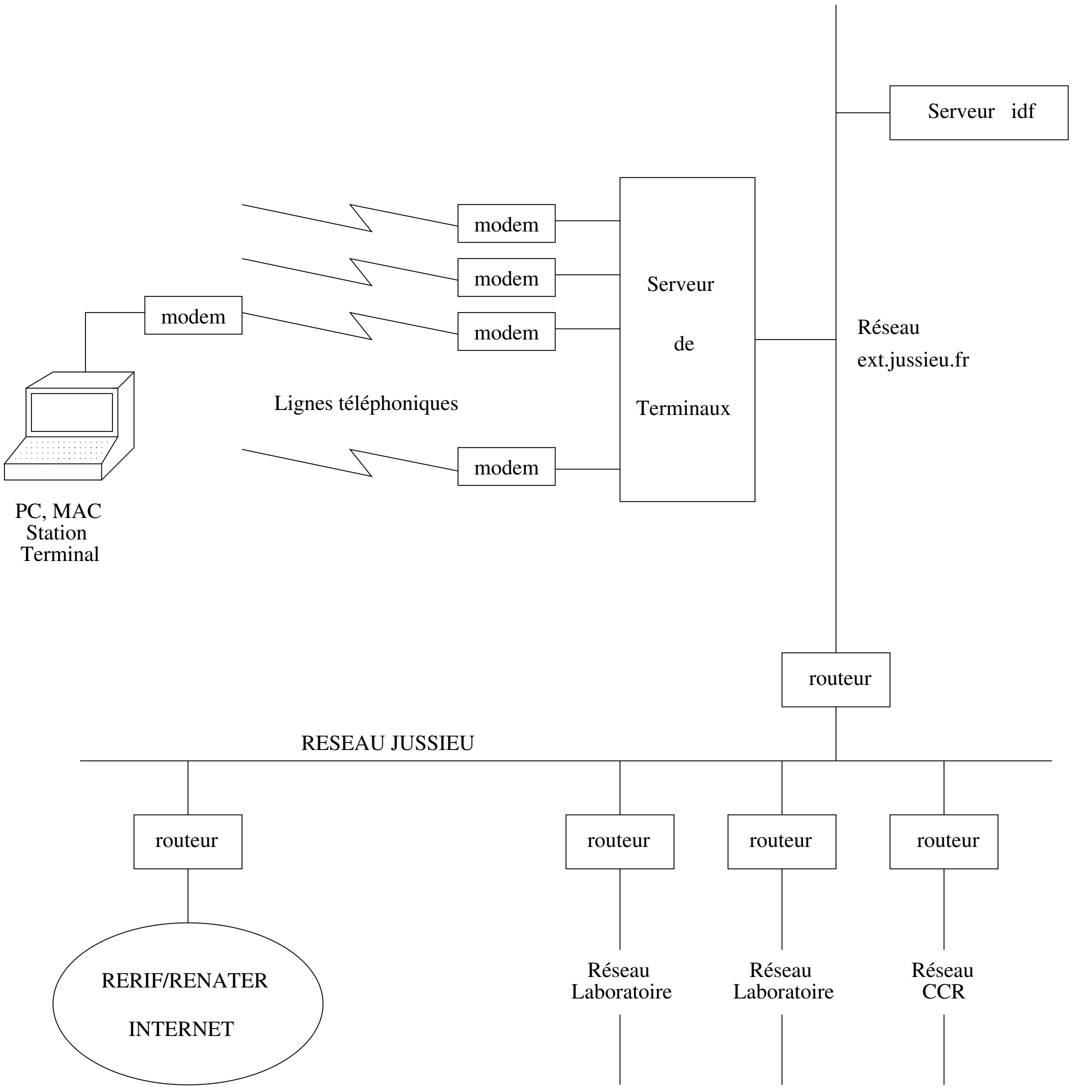
Le protocole `ARAP` permet à un Macintosh de communiquer avec un réseau `AppleTalk` en utilisant les lignes téléphoniques.[1]

Le réseau `JUSSIEU` est maintenant équipé d'une batterie de modems `V32.bis`. Ces modems sont connectés à un nouveau serveur de terminaux permettant les connexions `SLIP`, `PPP` et `ARAP`. Ces matériels appartiennent à un nouveau sous-réseau du réseau `JUSSIEU` appelé "*ext.jussieu.fr*" sur lequel sont regroupés différentes machines offrant des services accessibles aux utilisateurs hors-campus.

Pour des raisons de sécurité, un filtrage a été mis en place : il n'est *-par exemple-* pas possible de se connecter directement sur une machine extérieure au campus à partir de ce serveur de terminaux.

[1] Supporté par la prochaine version du logiciel du serveur de terminaux. Ce papier sera alors mis à jour.

2 La Connexion



"*idf*" est une machine dédiée à des services régionaux offerts à des chercheurs (enseignement supérieur ou CNRS) hors campus (courrier électronique, impressions, listes de diffusion, etc...).

3 Les Modems

Ces modems sont appelables à partir de modems :

- `V32.bis` (14400 Bauds)
- `V32` (9600 Bauds)
- `V22.bis` (2400 Bauds)
- `V22` (1200 Bauds)
- `V21` (300 Bauds) pour certains modems.

De plus ils font du contrôle d'erreur (`V42` et `MNP4`) et de la compression (`V42.bis` et `MNP5`).

Ces modems n'étant pas compatibles `V23` (1200/75 Bauds), une autre batterie de modems est disponible pour les appels *minitel* ou *émulation minitel*.

Au moment de l'établissement de la connexion, une négociation sur la liaison a lieu si les deux modems le permettent : vitesse, correction d'erreur, compression...

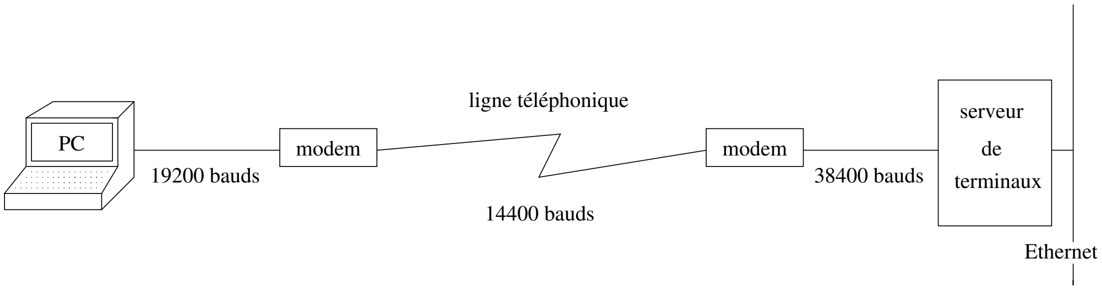
Toujours si les deux modems le permettent, pendant toute la durée de la connexion, des re-négociations peuvent avoir lieu : par exemple la vitesse de communication peut être diminuée s'il y a trop d'erreurs (on parle alors de repli sur des vitesses inférieures), ou augmentée si la qualité de la ligne la rend possible (et si on n'est pas à la vitesse maximum!).

Le standard `V32.bis` permet la négociation des vitesses 14400, 12000, 9600, 7200 et 4800 Bauds et le standard `V32`, 9600 et 4800 Bauds. Les standards `V32.bis`, `V32`, `V22.bis` et `V22` prévoient l'utilisation des protocoles de correction d'erreurs et de compression cités ci-dessus.

En `V32` et `V32.bis`, entre le poste de travail et le modem :

- le contrôle de flux doit être géré par les signaux `RS-232 RTS` (Request to Send) et `CTS` (Clear to Send), car le contrôle par le biais des caractères `XON` et `XOFF` provoque un ralentissement trop important et/ou une perte de caractères, dûe aux temps de réaction insuffisants des logiciels à ces vitesses
- la vitesse de la liaison modem-terminal est indépendante de la vitesse de communication sur la ligne téléphonique et de la vitesse de la liaison modem-serveur-de-terminaux. La compression et le contrôle de flux se chargent de réguler.

Ainsi on peut avoir :



Grâce à la compression, on peut avoir une vitesse de transfert bien supérieure à 14400 bits/seconde... C'est en général la bande passante du logiciel `IP` et/ou du port série de votre `PC` ou `MAC` qui sera le goulot d'étranglement de la transmission.

Pour tout modem compatible **`HAYES`**, il y a quelques paramètres à positionner pour que cela marche. Une fois les bons paramètres trouvés, leur sauvegarde dans la mémoire du modem ou de votre logiciel de communication permet de ne pas avoir besoin de les retrouver à chaque connexion !

4 Le Serveur de Terminaux

Un serveur de terminaux assure :

- la concentration de lignes série
- le "*passerellage*" avec le protocole du réseau local (`IP` en l'occurence)
- le routage des paquets constitués vers les machines.

Un serveur de terminaux donne donc accès au réseau local à partir de lignes série. Cet accès peut être direct (un terminal VT100 directement connecté à un des ports par exemple) ou via un modem (par ligne téléphonique par exemple). Une fois connecté au serveur on peut se "*logger*" sur un ordinateur par commandes *telnet* ou *rlogin*. Lorsque le "*login*" est effectué, le serveur de terminaux devient transparent pour l'utilisateur ; il ne fait que passer des caractères, sans les altérer, en provenance du modem vers le site destination, et vice-versa.

Ce serveur (comme les 4 autres du `CCR`) est une "`annex`".

Le système d'exploitation de l'`annex` est de type `UNIX` avec un langage de commandes (le "`CLI`" pour Command Line Interpreter) ne comportant qu'une dizaine de commandes.

Au moment de la connexion, l'`annex` affiche un mot du jour (comportant en général un avertissement sur les connexions "abusives") puis le prompt "*annex :*" (invitation à entrer une commande). On a alors ouvert une session annex.

A partir d'une session annex, on peut ouvrir jusqu'à 3 sessions telnet/rlogin simultanées.

On sort des sessions telnet/rlogin par les commandes habituelles des machines sur lesquelles on est connecté. A la sortie de la dernière session telnet/rlogin, la session annex se ferme automatiquement. Si elle ne se fait pas automatiquement, il est fortement recommandé de fermer votre session annex "proprement" en tapant la commande "*hangup*" : cela garantit une bonne fin de session (fermeture d'éventuelles sessions telnet/ rlogin "oubliées" en arrière plan, réinitialisation du port). La fermeture (automatique ou manuelle) de la session annex provoque un raccrochage téléphonique.

Si la ligne téléphonique est inactive (aucun caractère émis ou reçu), pendant plus de 10mn, il y a fermeture automatique de la (des) session(s) telnet/rlogin et de la session annex.

On peut associer une adresse `IP` (différente !) à chaque port d'une `annex`. Cette adresse est celle que devra avoir le micro s'il y a utilisation des protocoles `SLIP` et `PPP`.

La brochure "`User's guide`" de l'`annex` est disponible au guichet du `CCR`.

5 Connexion sans Protocole

Il faut d'abord régler quelques paramètres :

- de votre terminal/micro/station (vitesse ligne série maximum pour modem `V32/V32.bis`, égale au débit du modem pour `V22/V22.bis` ; 8 bits sans parité ; contrôle de flux `CTS/RTS` pour `V32/V32.bis`, `XON/XOFF` ou `CTS/RTS` pour `V22/V22.bis`...) pour le contrôleur et le logiciel de communication
- de votre modem (ligne téléphonique la plus rapide possible, compression `V42.bis`, correction d'erreurs `V42`, type de connexion qui sera negociée au moment de l'appel, contrôle de flux...) [2].

Ces configurations doivent être sauvegardées (sur le micro et/ou sur le modem) afin de ne pas avoir à les réinventer à chaque connexion.

Il n'y a plus qu'à numéroter (en général *ATDT...* ou *ATDP...*).

Si la négociation entre les 2 modems s'est bien passée, le mot du jour puis le prompt "*annex:* " apparaissent.

Vous disposez alors d'une dizaine de commandes dont :

- *telnet machine1* (pour vous "*logger*" sur "*machine1* ")
- *rlogin machine2* (pour vous "*logger*" sur "*machine2* ").

Ce nom peut être simple, partiel ou complet :

- simple : *rlogin idf*
  La machine "*idf*" est sur le même sous-réseau que l'`annex` (*ext.jussieu.fr*), le suffixe "*ext.jussieu.fr*" est ajouté automatiquement.
  Mais "*rlogin moka* " provoquera un "*host unknown* " car "*moka.ext.jussieu.fr* " n'existe pas (c'est "*moka.ccr.jussieu.fr* ")
- partiel : *telnet moka.ccr*
  Le suffixe "*jussieu.fr* " est ajouté et... ça marche
- complet : *rlogin masi.ibp.fr*
  Mais "*telnet prep.ai.mit.edu* " donnera "*network unreachable* " à cause du filtrage : on ne peut se connecter au "*reste du monde* " que via une machine identifiée et possédant un système de journalisation des événements (un "*log* ").

L'`annex` conserve en mémoire pendant un certain temps le nom et le numéro `IP` des machines avec lesquelles il y a eu une connexion récente. Vous pouvez tenter de taper un nom "court" : elle utilise (comme pour les commandes) la chaine minimale unique.

D'autres commandes sont disponibles telles que "`SLIP`" et "`PPP`". Dans ces cas, vous passez explicitement le port dans le protocole correspondant. Vous devez alors revenir sur votre micro ou station (sans couper la ligne téléphonique) et lancer une application utilisant ce protocole.
Voir les paragraphes suivants et/ou le "`User's guide`" de l'`annex`.

[2] Heureusement, la plupart des paramètres ont une valeur "*par défaut* " qui convient très bien !

6 IP sur Lignes Téléphoniques

6.1 SLIP (*RFC 1055*) : Serial Line IP

Ce protocole est extrêmement simple : les paquets `IP` sont mis sur la ligne série avec un octet de séparation `END-SLIP` (code *0xC0*). Un deuxième code "*spécial SLIP*" est le `ESC-SLIP` (code *0cDB*). Si un de ces deux codes apparaît dans la trame, il est transformé en *0xDB 0xDC* pour le premier et en *0xDB 0xDE* pour le second. La transformation inverse est faite à l'autre bout de la ligne. C'est tout.

Il n'y a donc

- pas de détection d'erreurs
- pas de négociation (adresse `IP`, taille de paquets, protocole transporté...)
- pas de sécurité, pas de détection de boucle...

Il est donc adapté pour des connexions où tous les paramètres entre les 2 machines sont fixés d'avance et non sujets à modifications fréquentes (cas d'une ligne spécialisée).

Dans notre cas, l'`annex` a une adresse `IP` par port (par ligne téléphonique). Sur chaque port, elle attend la connexion d'une machine ayant le numéro `IP` associé. Comme on y accède via une ligne téléphonique groupée, on ne sait jamais sur quel port on va arriver, donc jamais le numéro `IP` qu'il faut avoir pour communiquer en `SLIP` avec ce port...

Cela est pourtant possible en "*manuel*" : il faut se connecter sans protocole, passer le port en `SLIP`, récupérer le numéro `IP` du port (affiché par la commande *slip*), reporter ce numéro dans le fichier de configuration `IP` sur son micro et lancer `IP` sur `SLIP`. C'est donc lourd...

6.2 PPP (*RFC 1331* à *1334*) : Point to Point Protocol

Ce protocole est un "vrai" protocole : les paquets `IP` sont encapsulés en `HDLC` (*High Level Datal Link Control*, simplifié dans `PPP` pour l'efficacité). Il y a détection d'erreur sur chaque paquet. `PPP` permet d'encapsuler autre chose que de l'`IP` dans les trames `HDLC`...

La négociation initiale (lors de l'établissement de la ligne, juste après la négociation "*matérielle*" entre les 2 modems) porte sur la taille des trames, la compression propre à `HDLC` (simplification de la trame), le traitement des caractères de contrôle du protocole transporté, l'identification (sécurité), les détections de boucles...

Si le protocole transporté est `IP`, il y a négociation de l'adresse `IP` et de la compression `VJ` (voir paragraphe suivant).

Tout cela au prix de quelques secondes pendant lesquelles les `PPP` échangent des paquets du genre "*Je peux faire cela, et toi ?*".

Ensuite, selon le résultat des négociations, `PPP` ajoute entre 4 et 10 octets à chaque paquet.

6.3 La Compression des Headers TCP/IP (*RFC 1144*)

Cette compression est plus connue sous le nom "*Van Jacobson*" ou `VJ` du nom du chercheur de Berkeley qui a proposé cette compression.

Elle consiste à supprimer au départ et à remettre à l'autre bout de la ligne téléphonique un certain nombre d'octets identiques ou prévisibles dans chaque paquet d'une liaison point à point. On passe donc de 40 octets (entête TCP=20 + entête IP=20) à entre 3 et 16 octets d'entête. Plus la ligne est basse-vitesse, plus il est important de grapiller de la bande passante...

Cette compression est négociable en `PPP` seulement. En `SLIP`, il faut un "*driver*" différent (souvent appelé `CSLIP`). Côté **`annex`**, pas de problème : selon les premiers paquets reçus de la ligne téléphonique, elle se met en protocole `SLIP` ou `CSLIP`.

6.4 Choix de PPP

L'utilisation compliquée (voire impossible) de `SLIP` avec des lignes téléphoniques groupées, la plus grande souplesse/robustesse de `PPP` font que c'est ce protocole qui a été retenu pour les connexions directes avec protocole. L'"*overhead*" de `PPP` par rapport à `SLIP` n'est pas significatif à partir de 9600 Bauds.

7 Connexion Directe en PPP

L'aspect réglage du modem, du contrôleur et du logiciel de communication est le même que pour une connexion sans protocole. Le logiciel est ici le "*driver*" `PPP`.

Votre logiciel `PPP` doit être configuré "*sans adresse* `IP`" afin d'accepter de s'en voir imposer une par l'`annex` (voir plus loin).

Une fois la connexion `PPP` établie entre le poste de travail et l'`annex`, celle-ci est transparente. L'`annex` n'est plus alors qu'une passerelle entre votre micro/station et le réseau `JUSSIEU`.

On peut faire *telnet*, *ftp*, *smtp*, *pop*, *news*, etc...

Un certain nombre de paramètres `IP` doivent être connus de votre micro ou station avant connexion (ils ne font pas partie de la négociation) :

- le *netmask* du réseau `JUSSIEU`
- le *broadcast* du sous-réseau *ext.jussieu.fr*
- l'adresse du routeur de sortie du sous-réseau *ext.jussieu.fr* vers le réseau `JUSSIEU`
- l'adresse du serveur de noms *dns.jussieu.fr*.

8 Pour les PC

`PARAGRAPHE SUJET A MODIFICATION`
`(tests en cours)`

Selon que vous utilisez `DOS` ou `WINDOWS`, que vous possédez ou non un logiciel `TCP/IP` (payant ou gratuit)... il y a plusieurs cas de figure :

8.1 DOS ou WINDOWS sans TCP/IP

Services : terminal et transport de fichiers

⟶ *kermit*: utilisation aisée pour se connecter à une machine en tant que terminal et y disposer de toutes les commandes (par exemple, pour la messagerie : *mail ou elm*).

⟶ *xmodem* : protocole de transport de fichiers plus simple que la même fonction de kermit.

8.2 DOS + une pile TCP/IP (SLIP et/ou PPP) gratuite :

⟶ `NCSA/telnet`

- `PPP`
- Services de base : *telnet*, *ftp*.

⟶ `ka9q`

- `PPP`
- Services de base : *telnet*, *ftp*.

8.3 WINDOWS + une pile TCP/IP (SLIP et/ou SLIP) gratuite :

⟶ `Trumpet/winsock`

- `SLIP`
- Services de base : *telnet*, *ftp*
- Messagerie : `PCeudora`
- nécessite une macro-commande pour numéroter et lancer `IP` sur `SLIP`.

8.4 DOS + une pile TCP/IP (SLIP et/ou PPP) payante :

⟶ PC/TCP

- PPP
- Services de base : *telnet*, *ftp*
- Messagerie : intégrée
- nécessite une macro-commande pour numéroter et lancer IP sur PPP.

8.5 WINDOWS + une pile TCP/IP (SLIP et/ou PPP) payante :

⟶ PC/TCP

- PPP
- Services de base : *telnet*, *ftp*
- Messagerie : **PCeudora**
- nécessite une macro-commande (*kermit*) pour numéroter et lancer IP sur PPP.

REMARQUES :

- les services tels que les clients *gopher*, *wais*... sont restreints au réseau JUSSIEU quand ils existent.
- PC/TCP permet l'intégration de tous les services IP
  (services de base, messagerie, *news*...)
- NUPop (messagerie) pourrait fonctionner avec PC/TCP (à vérifier)
  avec sa propre pile TCP/IP, NUPop offre "*en plus*" : *telnet*, *ftp*, *gopher*... sur SLIP.

SOLUTIONS :

⟶ On peut se procurer au CCR :

1. *kermit*, *xmodem*
2. **NCSA/telnet** et **ka9q** (DOS, gratuit)
3. **NUPop** (DOS, gratuit)
4. **Trumpet/winsock** (WINDOWS, gratuit)
5. **PCeudora** (WINDOWS, gratuit)
6. **PC/TCP** (licence).

9 Pour les MAC

`PARAGRAPHE SUJET A MODIFICATION`
`(tests en cours)`

Selon que vous possédez ou non un logiciel `TCP/IP` (payant ou gratuit), il y a plusieurs cas de figure :

9.1 Sans TCP/IP :

Services : terminal et transport de fichiers

⟶ *kermit*: utilisation aisée pour se connecter à une machine en tant que terminal et y disposer de toutes les commandes (par exemple, pour la messagerie : *mail ou elm*).

⟶ *xmodem* : protocole de transport de fichiers plus simple que la même fonction de kermit.

9.2 Avec TCP/IP (SLIP et/ou PPP) gratuit :

⟶ `NCSA/telnet` 2.5

- `SLIP` et `PPP` (?)
- Services de base : *telnet*, *ftp*
- Messagerie : `eudora` avec la "*Communication ToolBox*"
- A tester.

9.3 Avec TCP/IP (SLIP et/ou PPP) payant :

⟶ `MacTcp`

- `PPP` avec `MacPPP`
- Services de base : *telnet*, *ftp*, `eudora`
- Services supplémentaires : *news*
- Services restreints : *gopher*, *mosaïc*...

`REMARQUES :`

- les services tels que les clients *gopher*, *wais*... sont restreints au réseau `JUSSIEU` quand ils existent.
- si on dispose de la licence `MacTcp`, l'installation de `MacPPP` (gratuit) est très simple et, une fois les paramètres de connexion enregistrés, il n'y a plus qu'à "*cliquer*"...
- les *telnet* et *ftp* sont plus conviviaux sur `MacTcp` que sur `NCSA/telnet`...

`EN COURS D'ETUDE :`

- `MacX` qui émule un terminal `X`
- `Zterm` (*shareware*) qui implémente *xmodem*, *ymodem* et *zmodem*
- `MacLayers` qui permet d'avoir plusieurs fenêtres `MAC` sur le même port série.
- un `TCP/IP` à prix d'amis.

`SOLUTIONS :`

⟶ On peut se procurer au `CCR` :

1. *kermit*, *xmodem*
2. `eudora` avec `MacTcp` pour `eudora` (gratuit en version 1.4.1)
   les versions 2.x sont payantes sans "*plus*" significatif
3. `NCSA/telnet` version 2.5 (gratuit)
   la version 2.6 est annoncée
4. `MacTcp` (licence Campus `JUSSIEU` : version 1.1.1)
5. *ftp* : *fetch* version 2.1.1 (nécessite `MacTcp`)
6. *news* : `InterNews`, `TheNews` et `Newswatcher` (nécessitent `MacTcp`)
7. *gopher* : `Turbogopher` (nécessite `MacTcp`)
   voir première remarque
8. *mosaïc* (nécessite `MacTcp`)
   voir première remarque.

10 Pour les Stations ou Micros UNIX

`PARAGRAPHE SUJET A MODIFICATION`
`(tests en cours)`

`UNIX` offre en standard (ou facile à installer) :

1. des commandes de connexion en mode terminal + transport de fichiers: *tip*, *kermit*, *xmodem*...
2. `TCP/IP` avec tous les services reposant sur cette couche : *telnet*, *ftp*, *smtp*, *news*...
   Si `SLIP` est quelquefois fourni, il faut se procurer `PPP`.
   Il y a des versions "*domaine public*" dont l'installation n'est pas forcément aisée...

`REMARQUES :`

- les services tels que les clients *gopher*, *wais*... sont restreints au réseau `JUSSIEU` quand ils existent.
- `PINE` est un logiciel "*domaine public*" qui, installé sur une station `UNIX`, permet d'accéder à une boite aux lettres distante en utilisant les protocoles `IMAP` et `SMTP`, sans avoir besoin de configurer *sendmail*. On retrouve alors sur le client `UNIX` les mêmes fonctionnalités que sur `PC` ou `MAC` (à vérifier).

`EN COURS D'ETUDE :`

- un gérant de fenêtres *shell* sur un port série (comme `MacLayers`)
- une automatisation du rapatriement des courriers et des news en mode "*batch*".

`SOLUTIONS :`

On peut, par *ftp anonyme*, facilement récupérer une version récente de *kermit* et *xmodem*.

11 Pour les PC sous Linux, FreeBSD et NetBSD

Plusieurs systèmes de type `UNIX` sont disponibles gratuitement. Il s'agit de `Linux`, `FreeBSD` et `NetBSD`. Ces systèmes sont tous disponibles par *ftp anonyme* sur le serveur *ftp.ibp.fr* dans les répertoires */pub/linux*, */pub/FreeBSD* et */pub/NetBSD*.

`Linux` ne supporte actuellement que le protocole `SLIP` alors que `FreeBSD` et `NetBSD` supportent à la fois `SLIP` et `PPP`.

11.1 Mise en œuvre de SLIP sous Linux

11.1.1 Prérequis

La mise en œuvre de `SLIP` sous `Linux` nécessite l'installation préalable des commandes de gestion réseau *ifconfig* et *route*. Il est également nécessaire de disposer des commandes *ftp*, *rlogin* et *telnet* au minimum. Il est conseillé d'utiliser un noyau de version supérieure ou égale à `0.99pl14`.

11.1.2 Installation de SLIP

Pour pouvoir utiliser `SLIP`, le noyau doit avoir été compilé avec l'option "*SLIP (serial line) support*" lors du *make config*.

Il faut ensuite utiliser la commande *dip* (*Dialup IP*) qui se charge de l'établissement de la connexion avec l'`annex`, configure l'interface `SLIP` et ajoute les routages nécessaires. Le programme source de cette commande peut être trouvé sur le serveur *ftp.ibp.fr* dans le répertoire */pub/annex/linux*. Le fichier *dip.tar.gz* contient les programmes source ainsi qu'un fichier de commandes appelé *annex.dip* qui automatise la connexion.

L'installation de *dip* est assez simple : il faut vérifier que le fichier *pathnames.h* contient les bons chemins vers les commandes *ifconfig* et *route* puis taper *make*.

11.1.3 Utilisation de DIP

Afin de créer une connexion `SLIP`, il suffit de vérifier que le fichier de commandes *annex.dip* contient les bons paramètres (notamment la chaîne d'initialisation du modem et le numéro de téléphone appelé) puis de taper la commande "*dip annex.dip*".

Vous êtes ensuite connectés à l'`annex` en protocole `SLIP`.

La fin de la session `SLIP` est provoquée en tuant le processus *dip* par la commande "*killall -HUP dip*".

11.2 Mise en œuvre de SLIP et PPP sous FreeBSD et NetBSD

Les systèmes `FreeBSD` et `NetBSD` ne disposent pas de commande *dip* permettant d'automatiser la connexion (comme c'est le cas pour le système `Linux`). La démarche est un peu plus contraignante.

11.2.1 Prérequis

Sous `FreeBSD` et `NetBSD`, il est nécessaire d'utiliser un noyau avec les options suivantes :

options "COM_BIDIR"
pseudo-device sl
pseudo-device ppp

De plus, les ports série doivent être gérés par le "*driver*" `sio` et non `com`. Le fichier de configuration du noyau doit donc contenir les définitions suivantes :

device sio0 at isa? port "IO_COM1" tty irq 4 vector siointr
device sio1 at isa? port "IO_COM2" tty irq 3 vector siointr

au lieu de :

device com0 at isa? port "IO_COM1" tty irq 4 vector comintr
device com1 at isa? port "IO_COM2" tty irq 3 vector comintr

De plus, vous devez créer les fichiers spéciaux représentant vos ports série par les commandes :

mknod /dev/cua00 c 28 128
mknod /dev/cua01 c 28 129

Enfin, vous devez activer les ports série lors du démarrage du système. Cela est effectué en plaçant les commandes suivantes à la fin du fichier */etc/rc.local* :

comcontrol /dev/tty00 bidir
comcontrol /dev/tty01 bidir
stty -f /dev/tty00 crtscts
stty -f /dev/tty01 crtscts

11.2.2 Déclaration du modem

Le fichier */etc/remote* doit contenir une ligne de déclaration de votre modem. Cette déclaration définit un nom logique, le port série sur lequel il est connecté, la vitesse de transmission et la parité utilisée.

A titre d'exemple, la déclaration suivante définit le nom logique "*modem*" connecté sur le deuxième port série (*cua01*) à une vitesse de transmission de 38400 bauds sans parité :

modem:dv=/dev/cua01:br#38400:pa=none

11.2.3 Etablissement d'une connexion SLIP

Il faut tout d'abord établir une connexion avec l'`annex`. Ceci est effectué en utilisant la commande *tip*, par exemple : *tip modem*.

Cette commande vous ouvre une communication avec le modem. Vous pouvez ensuite faire appeler l'`annex` par le modem (par une commande du style *ATDT numéro-de-téléphone*) puis taper la commande *slip* lorsque vous êtes connectés sur l'`annex`.

Le reste de la connexion est normalement établi à l'aide des commandes *slattach*, *ifconfig* et *route*. Afin de simplifier la connexion, un fichier de commande appelé *start-slip* est disponible sur le serveur *ftp.ibp.fr* dans le répertoire */pub/annex/bsd*. Ce fichier de commandes prend comme argument l'adresse `IP` attribuée par l'`annex` lors du lancement de la connexion `SLIP`.

La fin de la connexion est provoquée par l'exécution du fichier de commandes *stop-slip* disponible au même endroit que *start-slip*.

11.2.4 Etablissement d'une connexion PPP

Il est plus facile d'établir une connexion `PPP` qu'une connexion `SLIP`. En effet, les commandes *pppd* et *chat* sont suffisamment évoluées pour gérer automatiquement la numérotation ainsi que l'attribution d'adresses `IP`.

Trois fichiers de commande sont disponibles sur le serveur *ftp.ibp.fr* dans le répertoire */pub/annex/bsd* :

- *start-ppp* : établissement de la connexion `PPP`
- *route-ppp* : mise en place des routages nécessaires après établissement de la connexion `PPP`
- *stop-ppp* : fin de la connexion `PPP`.

Après avoir mis à jour le numéro de téléphone dans le fichier *start-ppp*, la connexion est établie en lançant *start-ppp* puis *route-ppp* lorsque le système affiche un message signalant que l'interface `ppp0` est connectée. La fin de la connexion est provoquée par *stop-ppp*.

11.2.5 Raccrochage du modem

La plupart du temps, la fin d'une connexion `SLIP` ou `PPP` ne provoque pas de raccrochage automatique du modem. Il convient alors de couper manuellement le modem.

12 Pour les Terminaux X (ou émulations)

`PARAGRAPHE SUJET A MODIFICATION`
`(tests en cours)`

Un terminal `X` offre souvent une connexion série, et parfois les protocoles `SLIP` et/ou `PPP`. Si cela est le cas, le protocole `X` peut "*passer*" par téléphone. Il est inenvisageable (ce serait insupportablement long) de télécharger le code du serveur et les polices de caractères par cette voie, il faut absolument les avoir disponibles en `ROM` ou `Flash PROM`, ainsi que suffisamment de mémoire `RAM` pour minimiser l'utilisation de la ligne.

Dans l'état actuel des terminaux `X`, le rafraîchissement des fenêtres autres que textuelles est assez lent, même avec un modem à haute vitesse (au moins 9600 Bauds). En revanche, il est prévu dans un proche avenir un nouveau protocole ("*low bandwidth* `X`") qui compresse très efficacement les échanges d'informations sur la ligne. Sans nul doute, le terminal `X` à la maison deviendra alors un outil très agréable.

`EN COURS D'ETUDE :`

- un terminal `X` avec `PPP`.

13 Le Protocole ARAP

`PARAGRAPHE SUJET A MODIFICATION`
`(tests prévus)`

La prochaine version de système de l'`annex` supportera le protocole `ARAP` (*Apple Remote Access Protocol*). `ARAP` permet à un `MAC` de communiquer avec un autre `MAC` ou un réseau `AppleTalk` via une ligne téléphonique. Un utilisateur `ARAP` peut ainsi accéder à tous les services disponibles de la zone à laquelle il se connecte (partage de fichiers, messagerie, bases de données, imprimantes...) comme s'il était physiquement rattaché au réseau en `LocalTalk` ou `EtherTalk`.

`AppleTalk` sur `PPP` est prévu dans les versions suivantes : il semble que ce soit le protocole qui monte...

Des informations plus précises seront données ici au fur et à mesure que ces services seront offerts.

`REMARQUES` : Le logiciel `ARA` pour `MAC`

- nécessite le Système 7
- peut utiliser tout type de modem
- coûte environ 1000 Francs.

14 La Sécurité

Les mesures de sécurité sont un compromis entre le "risque informatique" et la facilité d'utilisation des systèmes. Ce compromis peut évoluer dans le temps en fonction des risques. Le risque d'intrusion "téléphonique" n'est pas négligeable à cause de l'anonymat de l'appelant. L'installation d'un nouveau mécanisme de sécurité se ressent presque toujours au niveau de l'ergonomie de l'utilisation d'un système.

La sécurité de l'`annex` est actuellement la suivante : vous pouvez vous connecter aux machines du réseau `JUSSIEU`

- sur lesquelles vous avez un droit d'accès (*login/mot de passe*)
- si le laboratoire autorise ce type d'accès

mais vous ne pouvez établir de liaisons directement vers l'extérieur du campus

- explicitement (*telnet*, *ftp*...)
- implicitement (*gopher*, *wais*...).

Il n'est en effet pas possible de mettre en service un serveur de terminaux avec connectivité `Internet` "*complète*". Je cite :

> *Accordingly,* `Internet` *sites SHOULD NOT have "general use" accounts, or "open" (without password) terminal servers that can access the rest of the* `Internet`*.*

Donc, pour le moment, il faut se connecter à une machine du réseau `JUSSIEU` puis, de là, accéder au monde extérieur. Ainsi, il y a une trace sur la machine "*passerelle*".

Cela oblige -*par exemple*- à faire des ftp en deux étapes pour ramener un logiciel des USA sur son `PC`.

Il se pourrait que cette sécurité soit insuffisante : on pourra alors être obligé d'installer des mécanismes plus contraignants :

- réseau "*coupe-feu*"
- *login/mot de passe* sur le serveur de terminaux
- services/machines dédiées
- ...

L'application de tout ou partie de ces solutions entraînera des contraintes supplémentaires pour l'utilisateur. Aussi, elles ne seront mises en place que si

- le risque a été sous-estimé ou est en cours d'augmentation,
- la politique générale de sécurisation du réseau `JUSSIEU` les rendent nécessaires.

15 Renseignements Pratiques

Les numéros téléphoniques d'appel, les paramètres de connexion, les paramètres `IP` sont donnés lors de l'inscription au `CCR` et accessibles "*en ligne*" sur les machines du `CCR`.

Merci à :

Rémy CARD (Remy.Card@ibp.fr) pour l'aspect `PC` sous `Linux`, `FreeBSD` et `NetBSD`
Pierre DAVID (Pierre.David@prism.uvsq.fr) pour l'aspect stations `UNIX` et terminaux `X`
Christian HASCOET (hascoet@ccr.jussieu.fr) pour l'aspect `PC`
Joël MARCHAND (jma@ariana.polytechnique.fr) pour l'aspect `MAC` et terminaux `X`
Gilles MISSONNIER (missonnier@gps.jussieu.fr) pour l'aspect `MAC`.

Me contacter pour tout renseignement complémentaire.

jt