La couche transport de la pile IP

Chaput Emmanuel

INP TOULOUSE ENSEEIHT

2016-2017

Notes :

Plan

1. Le protocole UDP
2. Le protocole TCP
3. Connexion TCP
4. La taille des segments TCP
5. Les échanges de données TCP
6. Les mécanismes supplémentaires de TCP

Notes :

Le protocole UPD : plan

1. Le protocole UDP
   - Le datagramme UDP
   - Port, numéro de port, socket
2. Le protocole TCP
3. Connexion TCP
4. La taille des segments TCP
5. Les échanges de données TCP
6. Les mécanismes supplémentaires de TCP
7. Références bibliographiques

Notes :

Le protocole UDP

- *User Datagramm Protocol* [6]
- Service d'envoi de données
- Frontière des messages préservées
- Pas de connexion
  - Service non fiable
  - Ordre non garanti
  - Pas de contrôle de flux

Notes :

Le datagramme UDP

source port	dest port
length	checksum
data	

`source port` Numéro de port émetteur
`destination port` Numéro de port destinataire
`length` Longueur du message
`checksum` Sur pseudo-en-tête

Notes :

Port, numéro de port, socket

- Port
  - Point d'accès au service transport
  - Permet de communiquer avec l'application
- Numéro de port
  - Identifiant du port
  - Entier sur 2 octets
  - Stocké en *big endian*
- Socket
  - Triplet (adresse IP, protocole id, numéro de port)
  - Protocole de transport généralement implicite

Notes :

Les well known ports

- Les applications classiques ont un numéro officiel
  - Serveur Web : 80
  - Serveur mail : 25
  - Serveur DNS : 53
- Évite la mise en place d'un mécanisme d'annuaire dynamique
  - Comme le DNS
- Liste gérée par l'IANA
  - [2]
  - Historiquement dans une RFC

Notes :

Le protocole TCP : plan

1 Le protocole UDP

2 Le protocole TCP
- Le segment TCP
- Numérotation
- Les données urgentes
- Le flag `push`
- Les options TCP

3 Connexion TCP

4 La taille des segments TCP

5 Les échanges de données TCP

7 Références bibliographiques

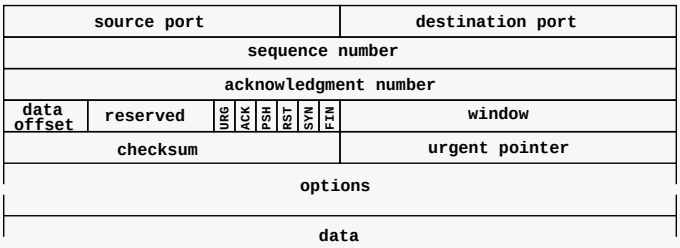
Notes :

Le protocole TCP

- *Transport Control Protocol* [7]
- Service en mode connecté
  - Mise en place de la connexion
  - Terminaison de la connexion
  - Pas d'émission hors connexion
- Service fiable
  - Flux d'octets
  - Contrôle de flux
  - Ordonné
  - Sans perte
- Protocole "complexe"
  - Fragmentation
  - Mécanisme de reprise sur perte
  - Fenêtre, timer, ...
  - Évolutions

Notes :

Le segment TCP



- Un format de message unique et fixe
  - Traitement simple
  - Piggy-backing

Notes :

Description de l'en-tête TCP

`source port` Numéro de port du destinataire
`destination port` Numéro de port de l'émetteur
`sequence number` Numéro de séquence
`acknowledgment number` Numéro d'accusé de réception
`data offset` Taille de l'en-tête en mots de 32 bits
`reserved` Non défini
`URG, ACK, PSH, RST, SYN, FIN` Flags (*voir plus loin*)
`window` Taille de fenêtre
`checksum` Contrôle sur pseudo en-tête et données
`urgent pointer` Fin des données urgentes
`options` (*voir plus loin*)

Notes :

Notion de connexion

Qu'est-ce qu'une connexion TCP ?

- Association établie pour permettre la communication entre deux entitées applicatives
- Définie par un couple de sockets

Qu'est-ce qu'une incarnation de connexion ?

- Une instance d'une connexion
- Caractérisée par une durée de vie
- Deux incarnations ne peuvent se chevaucher (temporellement)

Notes :

La numérotation

- Numérotation octet par octet
  - Numérotation absolue
  - Modulo 2^{32}
- Numéro de séquence initial synchronisé à l'établissement
  - Minimiser les interactions entre incarnations
  - Problèmes de sécurité [**?**]
- Accusé de réception selon même numérotation
  - Numéro du prochain octet attendu
  - Acquittement implicite de tout ce qui précède

Notes :

Les données urgentes

- Le bit `URG` valide la présence du pointeur `Urgent Pointer`
- `Urgent Pointer` pointe sur le premier octet de données après les données urgentes
- Forme de "out-of-band data"
- Utilisation non spécifiée
  - Telnet [9] l'utilise pour le *synch* (`Ctrl-C`)

Notes :

Le flag push

- But : délivré immédiatement
  - Pas de concaténation
- Sur l'émetteur
  - Émission immédiate sur demande de l'application
  - Information piggy-backée
- Sur le récepteur
  - Fourniture immédiate à l'application

Notes :

Les options TCP

- Utilisation négociée à l'établissement de la connexion
  - Le client propose l'utilisation de l'option
  - Le serveur valide si il comprend et si il accèpte
  - Option éventuellement utilisée dans les données
- Outil d'évolution du protocole
  - Extensions
  - Levées de limites
  - Conserve l'interopérabilité
- Limité en taille

Notes :

Format des options TCP

`Kind` Quelle option ?

`Length` Taille de l'option

`Data` Contenu (éventuel) de l'option

Padding

`End Of Options` (kind 0) matérialise la fin des options

`No Option` (kind 1) insérée entre deux options pour alignement

Notes :

La connexion TCP : plan

- Les well known ports

3 Connexion TCP
- Établissement
  - Ouverture simultanée
- Terminaison d'une connexion TCP
- Le flag reset
- Diagramme d'états TCP
  - Les accusés de réception
  - Les accusés de réception retardés
  - Choix du timer de ré-émission
  - L'algorithme de Nagle
  - Gel de la taille de fenêtre
  - L'option window scale
  - L'option `timestamp`

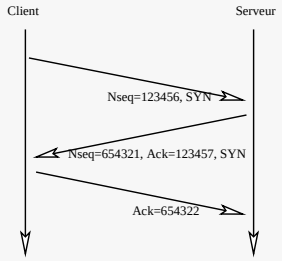
Notes :

Connexion TCP

- Établissement
  - En trois temps
  - Négociation de paramètres
- Terminaison
  - En trois ou quatre temps
  - Semi-fermeture possible
- Mécanismes de réinitialisation
  - En cas de dysfonctionnement

Notes :

Établissement d'une connexion TCP



- Synchronisation des numéros de séquence
  - Segments SYN (`SYN = 1`)
  - Pas (plus) de données
- Négociation des options

Notes :

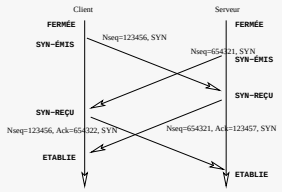
Un exemple

```
graham.4123 > albert.telnet:
   S 3608234478:3608234478(0) win 5840
   <mss 1460,sackOK,timestamp 60120005 0,nop,wscale 0> (DF)
albert.telnet > graham.4123:
   S 2429582946:2429582946(0) ack 3608234479 win 5840
   <mss 1460,nop,nop,sackOK,nop,wscale 0> (DF)
graham.4123 > albert.telnet:
   . ack 1 win 5840 (DF)
```

- Le bit `SYN` consomme un octet

Notes :

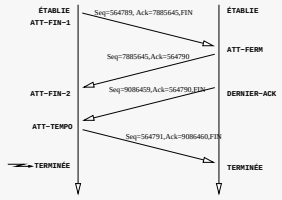
Ouverture simultanée



- Théoriquement possible
  - Une seule connexion établie
- En pratique impossible
  - Assymétrie client/serveur

Notes :

Terminaison d'une connexion TCP



- Quatre phases
  - Possiblement trois
- Fermeture unilatérale possible
  - Communication simplexe
  - Rarement utilisé

Notes :

Un exemple

Fermeture en trois temps

```
albert.telnet > graham.4128:
   F 187:187(0) ack 110 win 5840 (DF) [tos 0x10]
graham.4128 > albert.telnet:
   F 110:110(0) ack 188 win 5840 (DF)
albert.telnet > graham.4128:
   . ack 111 win 5840 (DF) [tos 0x10]
```

- Le bit `FIN` consomme un octet

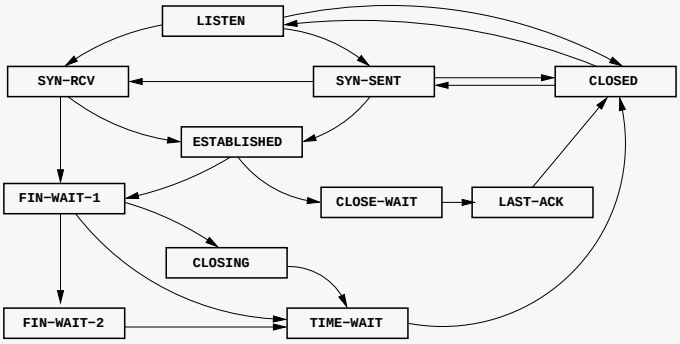
Notes :

Le flag reset

- Réinitialisation de la connexion
  - Cas des connexions “semi ouvertes”
- Tentative d’ouverture d’une connexion déjà ouverte
  - Suite crash
- Réception de données sur une connexion inconnue
- Accusé de réception de données non émises

Notes :

Diagramme d’états TCP



Notes :

Un exemple

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN
tcp        0      0 147.127.18.113:8080     0.0.0.0:*               LISTEN
tcp        0      0 147.127.18.113:1057     207.200.86.65:80        TIME_WAIT
tcp        1      0 147.127.18.113:1036     216.40.34.70:80         CLOSE_WAIT
tcp        0      0 147.127.18.113:143      147.127.18.98:1053      ESTABLISHED
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 147.127.18.113:23       147.127.18.98:1123      ESTABLISHED
tcp        0      0 147.127.18.113:23       147.127.18.98:1122      ESTABLISHED
```


Notes :

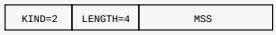
La taille des segments TCP : plan

- Les well known ports
- Ouverture simultanée

4 La taille des segments TCP
- L'option MSS
- TCP et le PMTUD
  - Les accusés de réception
  - Les accusés de réception retardés
  - Choix du timer de ré-émission
  - L'algorithme de Nagle
  - Gel de la taille de fenêtre
  - L'option window scale
  - L'option `timestamp`

Notes :

L'option MSS



- Négociée à l'établissement de la connexion
  - [8]
- Obtenue grâce à la taille maximale des paquets IP
  - `MSS = MTU - sizeof(IPHEADER) - sizeof(IPOPTIONS) - sizeof(TCPHEADER)`
- Minimise les risques de fragmentation IP
  - Sur les réseaux d'accès

Notes :

TCP et le PMTUD

- TCP peut utiliser PMTUD
  - Raffiner le MSS
  - Obligatoire avec IPv6
- Efficacité dans les données
  - Segments SYN : 40 octets
- Problèmes de déploiement
  - Filtrage de ICMP

Notes :

Un exemple (1)

Établissement de la connexion

```
arp who-has routeur tell lune
arp reply routeur is-at 00:01:02:ab:c9:9e
IP (tos 0x0, ttl  64, id 57757, offset 0, flags [DF], length: 60)
   lune.1028 > terre.5001: S [tcp sum ok] 0:0(0) win 5840
   <mss 1460,sackOK,timestamp 20409 0,nop,wscale 0>
IP (tos 0x0, ttl  63, id 0, offset 0, flags [DF], length: 60)
   terre.5001 > lune.1028: S [tcp sum ok] 0:(0) ack 1 win 5792
   <mss 1460,sackOK,timestamp 229964 20409,nop,wscale 0>
```


Notes :

Un exemple (2)

Premières émissions de données

```
IP (tos 0x0, ttl  64, id 57760, offset 0, flags [DF], length: 1500)
   lune.1028 > terre.5001: . 25:1473(1448) ack 1 win 5840
   <nop,nop,timestamp 20409 229964>
IP (tos 0xc0, ttl  64, id 21840, offset 0, flags [none], length: 576)
   routeur > lune:
   icmp 556: terre unreachable - need to frag (mtu 512) for
      IP (tos 0x0, ttl  63, id 57760, offset 0, flags [DF], length: 1500)
         lune.1028 > terre.5001: . 25:1473(1448) ack 1 win 5840
         <nop,nop,timestamp 20409 229964>
IP (tos 0x0, ttl  64, id 57761, offset 0, flags [DF], length: 1500)
   lune.1028 > terre.5001: . 1473:2921(1448) ack 1 win 5840
   <nop,nop,timestamp 20409 229964>
IP (tos 0xc0, ttl  64, id 21841, offset 0, flags [none], length: 576)
   routeur > lune:
   icmp 556: terre unreachable - need to frag (mtu 512) for
      IP (tos 0x0, ttl  63, id 57761, offset 0, flags [DF], length: 1500)
         lune.1028 > terre.5001: . 1473:2921(1448) ack 1 win 5840
         <nop,nop,timestamp 20409 229964>
```


Notes :

Un exemple (3)

Mise en place de la nouvelle valeur

```
IP (tos 0x0, ttl  64, id 49440, offset 0, flags [none], length: 552)
   lune.1028 > terre.5001: . 25:525(500) ack 1 win 5840
   <nop,nop,timestamp 20410 229964>
IP (tos 0x0, ttl  64, id 49441, offset 0, flags [none], length: 552)
   lune.1028 > terre.5001: . 525:1025(500) ack 1 win 5840
   <nop,nop,timestamp 20410 229964>
IP (tos 0x0, ttl  64, id 49442, offset 0, flags [none], length: 500)
   lune.1028 > terre.5001: . 1025:1473(448) ack 1 win 5840
   <nop,nop,timestamp 20410 229964>
IP (tos 0x0, ttl  63, id 25325, offset 0, flags [DF], length: 52)
   terre.5001 > lune.1028: . [tcp sum ok] 1:1(0) ack 525 win 6432
   <nop,nop,timestamp 229964 20410>
IP (tos 0x0, ttl  63, id 25326, offset 0, flags [DF], length: 52)
   terre.5001 > lune.1028: . [tcp sum ok] 1:1(0) ack 1025 win 7500
   <nop,nop,timestamp 229964 20410>
IP (tos 0x0, ttl  63, id 25327, offset 0, flags [DF], length: 52)
   terre.5001 > lune.1028: . [tcp sum ok] 1:1(0) ack 1473 win 8500
   <nop,nop,timestamp 229964 20410>
```


Notes :

Les échanges de données TCP : plan

- Les well known ports
- Ouverture simultanée

5 Les échanges de données TCP
- Gestion de la fenêtre sur le récepteur
  - Les accusés de réception
  - Les accusés de réception retardés
- Gestion de la fenêtre sur l'émetteur
  - Choix du timer de ré-émission
  - L'algorithme de Nagle
  - Gel de la taille de fenêtre
  - L'option window scale
  - L'option `timestamp`

Notes :

Les échanges de données TCP

- Flux d'octets segmenté
  - Numérotation des segments
  - Octet par octet
- Buffer de réception
  - Stockage des segments hors séquence
  - Contrôle de flux
- Buffer d'émission
  - Stockage des segments non acquittés
  - Contrôle de flux et reprise sur perte

Notes :

Numérotation, notations

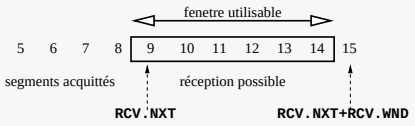
Caractéristiques d'un émetteur
- `SND.UNA` Premier octet non acquité
- `SND.NXT` Prochain octet à émettre
- `SND.WND` Taille de la fenêtre (d'émission)

Caractéristique d'un récepteur
- `RCV.NXT` Prochain octet attendu
- `RCV.WND` Taille de la fenêtre (de réception)

Caractéristiques d'un segment
- `SEG.ACK` Prochain octet attendu par le récepteur
- `SEG.LEN` Nombre d'octets du segment
- `SEG.SEQ` Numéro du premier octet du segment

Notes :

Gestion de la fenêtre sur le récepteur



- Évolution de `RCV.NXT`
  - Réception d'un segment en séquence
- Évolution de `RCV.WND`
  - Réception d'un segment dans la fenêtre hors séquence
  - Consommation de données par l'application
- Accusé de réception (immédiat ou piggy-backé)
  - Réception d'un segment en séquence
  - `SEG.ACK` reçoit `RCV.NXT`
  - `SEG.WND` reçoit `RCV.WND` (*aka* `awnd`)
- `RCV.NXT + RCV.WND` ne doit jamais décroître

Notes :

Acceptabilité d'un segment

SEG.LEN	RCV.WIN	Acceptable si ...
0	0	SEG.SEQ = RCV.NXT
0	> 0	RCV.NXT $\leq$ SEG.SEQ $<$ RCV.NXT+RCV.WND
> 0	0	Non recevable
> 0	> 0	RCV.NXT $\leq$ SEG.SEQ $<$ RCV.NXT+RCV.WND ou RCV.NXT $\leq$ SEG.SEQ+SEG.LEN-1 $<$ RCV.NXT+RCV.WND

Notes :

Les accusés de réception

- Un segment : un accusé de réception
  - Segment court
- Possibilité de piggy-backing
- Accusé de réception cumulatif
  - Numéro du prochain octet attendu
- Validé par le bit `ACK`

Notes :

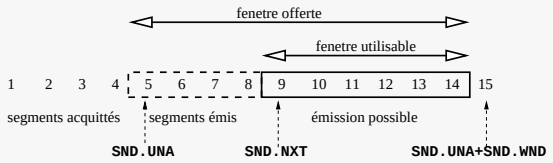
Les accusés de réception retardés

- Ne pas accuser réception immédiatement
- Objectifs
  - Utiliser le piggy-backing
  - Minimise les risques de SWS (voir plus loin)

```
13:57:30.405145 IP client.51152 > serveur.telnet:
    Flags [P.], seq 44:45, ack 178, win 108, length 1
13:57:30.834413 IP serveur.telnet > client.51152:
   Flags [P.], seq 178:179, ack 45, win 5840, length 1
13:57:30.834444 IP client.51152 > serveur.telnet:
   Flags [P.], seq 45:47, ack 179, win 108, length 2
13:57:31.417571 IP serveur.telnet > client.51152:
   Flags [P.], seq 179:180, ack 47, win 5840, length 1
13:57:31.457102 IP client.51152 > serveur.telnet:
   Flags [.], ack 180, win 108, length 0
```


Notes :

Gestion de la fenêtre en émission



- Fenêtre glissante
- Évolution de `SND.UNA` sur accusé de réception
  - Recevable si $\texttt{SND.UNA} < \texttt{SEG.ACK} \leq \texttt{SND.NXT}$
- Évolution de `SND.NXT` sur demande de l'application
- Évolution de `SND.WND` sur réception d'un segment
  - `SND.WND` reçoit `SEG.WND`

Notes :

Choix du timer de ré-émission

- Plus délicat que HDLC
  - Gigue non bornée
- Mesure expérimentale du RTT
  - `SRTT` reçoit α `* SRTT + (1-`α`) * RTT`
- Time de ré-émission calculé en fonction
  - `RTO` reçoit β `* SRTT`

Notes :

Les mécanismes supplémentaires de TCP : plan

6 Les mécanismes supplémentaires de TCP
- Le timer keepalive
- Le syndrome de la fenêtre stupide
- Accusés de réception sélectifs
- TCP face aux réseaux haut-débit

Notes :

Le timer keepalive

- Si l'application est muette
  - Aucun segment ne circule
  - Aucune remontée de dysfonctionnement
- Introduction d'un timer
  - Émission régulière de "sondes"
  - Segment vide
- Problème applicatif
- Mécanisme controversé [1]

Notes :

Le syndrome de la fenêtre stupide

- Émission de *tinygrams*
- Production trop lente sur l'émetteur
  - Algorithme de Nagle
- Consommation trop lente sur le récepteur
  - Gel de la taille de la fenêtre

Notes :

Algorithme de Nagle

- Certaines applications génèrent de petits messages
  - Telnet
  - Apparition de tinygrams
- Principe : limiter les petits segments
  - Interdits tant que la fenêtre n'est pas vide
- [5]

Notes :

Algorithme de Nagle : exemple

```
14:55:29.197854 IP ln.1030 > tr.telnet: P 0:1(1) ack 0 win 40544
14:55:29.289499 IP tr.telnet > ln.1030: P 1:2(1) ack 1 win 5792
14:55:29.289570 IP ln.1030 > tr.telnet: . ack 2 win 40544
14:55:29.758491 IP ln.1030 > tr.telnet: P 1:2(1) ack 2 win 40544
14:55:29.849355 IP tr.telnet > ln.1030: P 2:3(1) ack 2 win 5792
14:55:29.849434 IP ln.1030 > tr.telnet: . ack 3 win 40544
...
14:59:27.872777 IP ln.1030 > tr.telnet: P 16:17(1) ack 3635 win 49232
14:59:28.869437 IP tr.telnet > ln.1030: P 3635:3636(1) ack 17 win 5792
14:59:28.869535 IP ln.1030 > tr.telnet: P 17:20(3) ack 3636 win 49232
14:59:29.869350 IP tr.telnet > ln.1030: P 3636:3637(1) ack 20 win 5792
14:59:29.869429 IP ln.1030 > tr.telnet: P 20:23(3) ack 3637 win 49232
14:59:30.869299 IP tr.telnet > ln.1030: P 3637:3639(2) ack 23 win 5792
14:59:30.888561 IP tr.telnet > ln.1030: P 3639:5087(1448) ack 23 win 5792
14:59:30.888774 IP ln.1030 > tr.telnet: . ack 5087 win 52128
14:59:31.879378 IP tr.telnet > ln.1030: P 5087:5450(363) ack 23 win 5792
14:59:31.910609 IP ln.1030 > tr.telnet: . ack 5450 win 52128
```


Notes :

Gel de `awnd`

- Autre cause de tinygram
  - Ouverture trop faible de la fenêtre de réception
- Solution
  - Ne pas ouvrir trop tôt, trop peu
- Mise en œuvre déjà évoquée
  - Accusés de réception retardés

Notes :

Accusés de réception sélectifs

Négociation à l'établissement

KIND=4	LENGTH=2

Utilisation dans les données

	KIND=4	LENGTH
offset de début de bloc		
offset de fin de bloc		

...

Notes :

Option Window Scale

KIND=3	LENGTH=3	SHIFT.CNT

- Accroissement de la taille de fenêtre [3] [4]
- Émission
  - `SEG.WND = RCV.WND >> rcv.scale`
- Réception
  - `SND.WND = SEG.WND << snd.scale`

Notes :

Timestamp

- Permettre une meilleure mesure du RTT [4]
- Estampille temporelle véhiculée dans les options
  - Négociée à l'établissement
- Estampille renvoyée avec l'accusé de réception
  - Estampille du premier en cas d'accusé retardé
  - Estampille du dernier en séquence en cas de perte
  - Du segment retransmis en cas de perte
- Valeur utilisée
  - Estampille du dernier segment reçu en séquence non encore acquitté

Notes :

Protection Against Wrapping Sequence number

- Constat
  - 2^{32} octets, ce n'est pas si vaste
- Risque de rebouclage
  - Réinitialisation de la connexion
- Principe : utiliser les timestamp
- [4]

Notes :

[1] Internet Engineering Task Force.
RFC 1122 : Requirements for internet hosts – communication layers.
Technical report, IETF, October 1989.

[2] IETF, http ://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml.

Service Name and Transport Protocol Port Number Registry.

[3] V. Jacobson and R. Braden.
RFC 1072 : Tcp extensions for long-delay paths.
Technical report, IETF, October 1988.

[4] V. Jacobson, R. Braden, and D. Borman.
TCP extensions for high performance.
RFC 1323, Internet Engineering Task Force, May 1992.

[5] John Nagle.
RFC 896 : Congestion control in ip/tcp internetworks.
Technical report, IETF, January 1984.

Notes :

[6] J. Postel.
RFC 768 : User datagram protocol specification.
Technical report, IETF, August 1980.

[7] J. Postel.
RFC 793 : Transmission control protocol.
Technical report, IETF, 1981.

[8] J. Postel.
TCP maximum segment size and related topics.
RFC 879, Internet Engineering Task Force, November 1983.

[9] J. Postel and J. Reynolds.
Telnet protocol specification.
RFC 854, Internet Engineering Task Force, May 1983.

Notes :