Internet et les liaisons

(/udd/bcousin/Polys/Reseaux-locaux/5.Les_liaisons.fm- 18 septembre 2003 12:17)

Plan

- Introduction
- Le protocole PPP
- Adaptation aux procédés de transmission
- LCP
- Authentification
- NCP
- Conclusion

Bibliographie

- L.Toutain, Internet et les réseaux locaux, Hermès, 1996.
- Les RFC

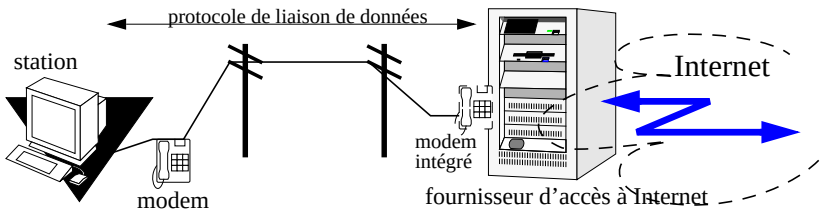
1. Introduction

Les stations peuvent être connectées à Internet au moyen de liaisons :

- liaison série
- liaison point-à-point

Et non pas au moyen d'un réseau local.

Par exemple, une connexion à Internet à travers le réseau téléphonique :



Les caractéristiques des liaisons sont très variables :

- il faut un protocole qui s'adapte

  ⇨ PPP

2. Le protocole PPP

2.1. Présentation

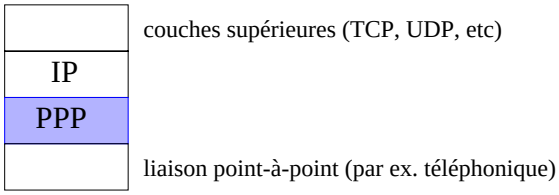
❑ Point to Point Protocol (rfc 1661)

Les services offerts :

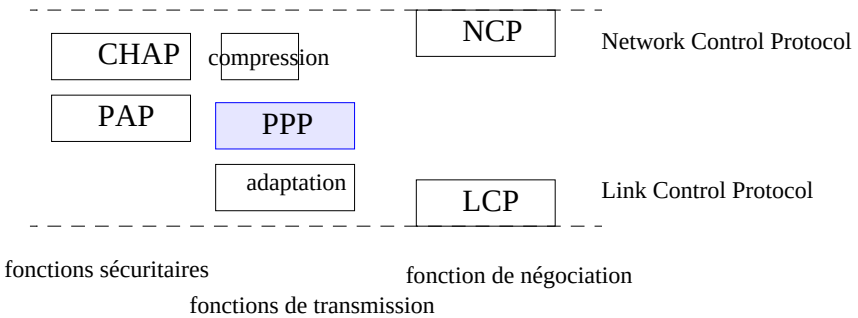
- transmission de données sur une liaison point à point
- masque l'hétérogénéité des caractéristiques de la liaison :
  - asynchrone, synchrone, téléphonique, spécialisée, HDLC, X25, RNIS, etc.
- supporte divers protocoles de niveau Réseau :
  - IP, IPX, Appletalk, IPv6, et même des trames !, etc.
- une transmission transparente vis-à-vis des codes réservés par les modems
- protection contre les erreurs
- contrôle l'accès au réseau
- négociation des différents paramètres
- mise en oeuvre de techniques d'optimisation de la transmission (compression)

2.2. Architecture

Dans Internet :

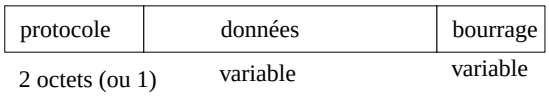


Les autres fonctions et protocoles complémentaires :



2.3. Format général du paquet PPP

paquet PPP



Protocole :

- identifie le protocole chargé des données
- 2 octets par défaut; un seul octet si négocié (cf option de code 7 de LCP)
- compatible avec le format étendu d'adresse d'HDLC (ISO 3309) :
  - les octets intermédiaires ont leur LSb = 0, le dernier octet a son LSb = 1
- [0***$_{16}$ - 3***$_{16}$] protocoles Réseau et [8***$_{16}$ - b***$_{16}$] protocoles NCP associés
- [4***$_{16}$ - 7***$_{16}$] protocoles spécifiques n'ayant pas de NCP
- [c***$_{16}$ - f***$_{16}$] protocoles LCP
- exemples de code du champ Protocole :
  - 0021_{16} : IP, 0029_{16} : AppleTalk, $002b_{16}$: IPX, $002d_{16}$: entête TCP/IP comprimées, $002f_{16}$: entêtes TCP/IP ne pouvant être comprimées
  - 8021_{16} : IPCP, 8029_{16} : AppleTalk, $802b_{16}$: IPX
  - $c021_{16}$: LCP, $c023_{16}$: PAP, $c223_{16}$: CHAP

Données :

- de 0 à MRU octets de données
- MRU ("Maximum Receive Unit")
  - 1500 octets par défaut
  - peut être négocié (cf option de code 1 de LCP)

Bourrage :

- permet d'obtenir un paquet de longueur fixe ou respectant une longueur minimale
- codage dépendant du protocole (cf option de code 10 de LCP)

2.4. Opérations

Contrôle de la liaison :

- utilise les paquets LCP
- test de la ligne
- négociation des options

Authentification

- utilise les paquets PAP ou CHAP

Contrôle du protocole Réseau :

- utilise les paquets NCP
- configuration d'un ou plusieurs protocoles partageant la liaison

Echange de données :

- paquets du protocole PPP

Fermeture de la liaison :

- paquet LCP ou NCP de fermeture
- perte de porteuse, temporisateur d'inactivité, intervention de l'administrateur

3. Adaptation aux procédés de transmission

3.1. Introduction

Encapsulation des paquets PPP dans des trames compatibles avec le procédé de transmission utilisé par la liaison point-à-point.

Le mode de transmission de la liaison :

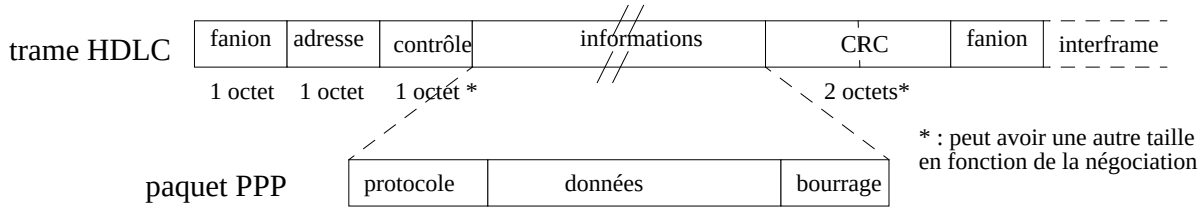
- synchrone
- asynchrone

Le service offert par la liaison :

- avec mise en oeuvre d'un mécanisme de contrôle de la fiabilité
  - rfc 1663 (compatible LAP-B : ISO 7776, appelé "numbered mode")
  - peu utilisé
- sans augmentation de la fiabilité ("connectionless")
  - rfc 1662
  - pour
    . liaison asynchrone ("XON/XOFF protocol", "start/stop protocol", etc)
    . liaison synchrone (compatible LAP-B : "unumbered")

3.2. Format général de la trame

Encapsulation dans une trame au format similaire à HDLC (ISO 4335)



Fanion :

- 01111110_2
- délimiteur de trame

Adresse :

- 11111111_2 : adresse de diffusion
- liaison point-à-point

Contrôle :

- type de la trame
- 00000011_2 : trame UI (“Unnumbered Information”)
- bit Poll/Final à zéro
- liaison suffisamment fiable (rfc 1662)
- liaison à fiabiliser (rfc 1663) - LAP_B :
  - trames I, SABM(E), UA, etc
  - format normal ou étendu ☞ SABM(E)
    - champ *contrôle* sur 1 ou 2 octets
    - numérotation modulo 8 ou 128

FCS (“Frame Check Sequence”) :

- par défaut : CCITT 16 bit CRC
- négociable : pas de FCS ou CRC de 32 bits (option 9 du LCP)

Interframe :

- soit l’adresse de la trame suivante
- soit des octets de remplissage (des fanions)

3.3. Le tramage

3.3.1 Transparence

La séquence binaire du fanion ne doit pas apparaître au sein de la trame :

- L'encodage ("stuffing")

Il existe plusieurs procédés d'encodage :

- par octet : caractère d'échappement
  - plutôt utilisé par les transmissions asynchrones
- par bit : insertion de bits
  - plutôt utilisé par les transmissions synchrones

3.3.2 Encodage par octet

Un caractère d'échappement doit être placé devant tous caractères spéciaux

- 0x7d

Les caractères spéciaux

- le fanion
- le caractère d'échappement

- n'importe quel caractère défini par l'ACCM ("Asynchronous Control Character Map")
  - L'ACCM est négociée (option 2 de LCP)

Exemple :

- à transmettre : 0x7d
- transmis : 0x7d 0x5d (0x7d ^ 0x20)

3.3.3 Encodage par bit

Insertion systématique d'un bit à 0 après 5 bits consécutifs à 1

- augmentation de la longueur de la trame
- un nombre quelconque de bits
- procédé standard d'HDLC

Exemple :

- à transmettre : 01111001.01111110.01111100.11111111.11000000_2
- transmis : 01111001.011111010.011111000.111110111.110000000_2

4. Le protocole LCP

4.1. Introduction

3 types de fonctions LCP :

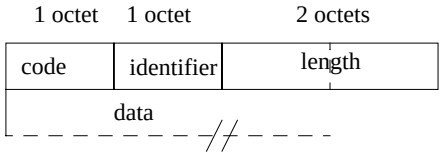
- configuration de la liaison :
  - établissement et négociation des options
  - les paquets : config-req, config-ack, config-nack, config-rej
  - ces paquets utilisent toujours le format de paquet par défaut
- libération de la liaison
  - les paquets : term-req, term-ack
- maintenance de la liaison :
  - les paquets : code-rej, proto-rej, echo-req, echo-reply, disc-req
  - paquets supplémentaires : d'identification, de temps de vie restant de la session

Chaque paquet LCP est encapsulé dans un paquet PPP

- le champ Protocol du paquet PPP = 0xc021

RFC 1570 et 1661

4.2. Format général des paquets LCP



Paquet LCP

Code :

- identifie le type du paquet
- par exemple : 1 = Configure-Request

Identifier :

- associe les demandes avec leurs réponses

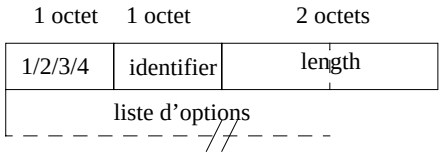
Length :

- longueur (totale) en octets du paquet LCP
- permet d'éliminer les octets de bourrage du paquet PPP

Data :

- dépend du type du paquet

4.3. La configuration



Paquet LCP de configuration

Une configuration pour chacun des deux sens, indépendamment.

☞ liaison configurée de manière assymétrique

Demande de configuration :

- champ Code = 1 : Configuration.Request
- la listes des options à négocier (valeurs différentes des valeurs par défaut)

Accord de configuration

- champ Code = 2 : Configuration.Ack
- accord avec toutes les options négociées

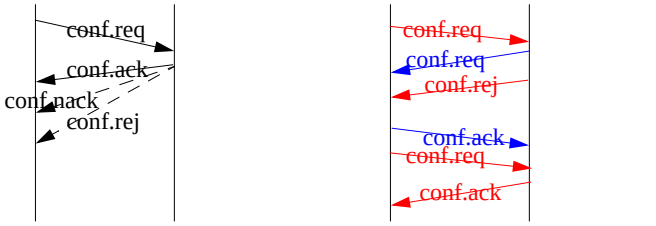
Désaccord de configuration

- champ Code = 3 : Configuration.Nack
- désaccord avec certaines valeurs des options proposées

Refus de configuration

- champ Code = 4 : Configuration.Reject
- lorsque certaines options ne sont pas négociables
- retourne les valeurs acceptables

Scénarios :

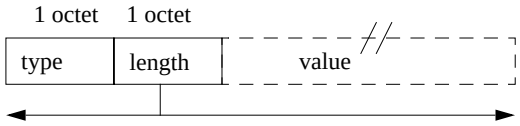


Dans les 2 sens avec re-négociation

4.4. Les options

Utilisées au sein de la liste d'options des paquets de configuration

Format TLV



Option des paquets LCP

Type = 1 : négociation du MRU ("Maximum Receive Unit")

- par défaut 1500 octets
- données sur 2 octets : taille maximum des paquets PPP

Type = 2 : négociation de la liste des caractères spéciaux

- données sur 4 octets (l'ACCM):
  - chaque bit identifie un des 32 premiers caractères
  - si le bit est positionné le caractère doit être transcodé

Type = 3 : négociation du protocole d'authenfication

- données sur 3 octets :
  - identifie le protocole (2 octets) : 0xc023 = PAP, 0xc223 = CHAP

  - identifie l'algorithme de chiffrement (1 octet) : 0x5 = MD5.

Type = 4 : négociation du protocole de surveillance de la liaison

- données sur 2 octets identifie le protocole :
  - 0xc025 = "Link Quality Report"

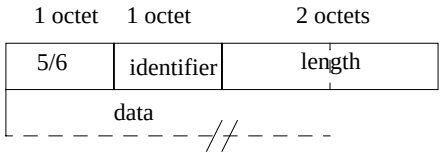
Type = 5 : négociation du nombre magique

- valeur du nombre magique sur 4 octets
- permet de détecter les rebouclages

Autres options :

- 7 : compression du champ *protocole* dans les paquets PPP
- 8 : compression des champs *adresse* et *contrôle* de la trame
- 9 : procédé de calcul du champ FCS
- 10 : bourrage auto-descriptif
- 11 : transmission en mode fiable (LAP-B)
- 13 : rappel automatique ("call back")
- 15 : trames composites (une trame contient plusieurs paquets)

4.5. Libération de la liaison



Paquet LCP de libération

Demande de libération :

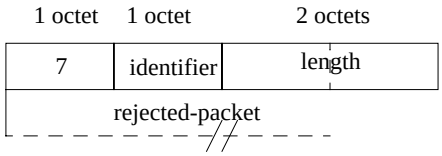
- champ Code = 5 : Terminaison.Request
- les octets de données ne sont pas interprétés

Accord de libération

- champ Code = 6 : Terminaison.Ack
- les octets de données ne sont pas interprétés

4.6. Trames d'administration

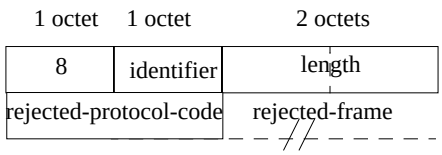
Rejet de paquet



Paquet LCP de rejet de paquet

- champ Code = 7 : Code.Reject
- à la suite de la réception d'une paquet LCP ayant un code inconnu
- le champ données contient le (début du) paquet LCP rejeté

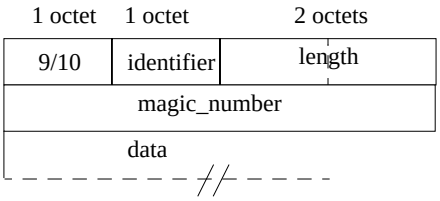
Rejet de trame



Paquet LCP de rejet de trame

- champ Code = 8 : Protocol.Reject
- à la suite de la réception d'une trame ayant un champ "protocol" inconnu
- le champ de données contient cette valeur (2 octets)
- et (un extrait de) la trame rejetée

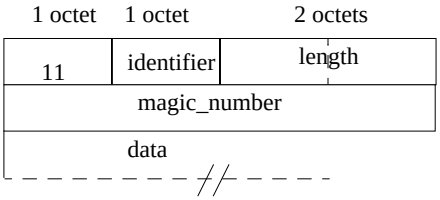
Echo



Paquet LCP d'écho

- champ Code = 9 et 10 : Echo.Request et Echo.Reply
- lors de la réception d'un paquet Echo.Request un paquet Echo.Reply est retourné
- test de la liaison : rebouclage, détermination de la qualité de la liaison
- magic-number :
  - permet d'identifier la présence d'un rebouclage
  - par défaut 0,
  - valeur négociée par l'option correspondante lors de la configuration
- le champ données peut contenir des données quelconques

Destruction



Paquet LCP à détruire

- champ Code = 11 : Discard.Request
- lors de la réception d'un paquet Discard.Request, le paquet est détruit.
- test de la liaison : déboguage, évaluation de performance
- magic-number :
  - permet d'identifier la présence d'un rebouclage
  - par défaut 0,
  - valeur négociée par l'option correspondante lors de la configuration

5. L'authentification

5.1. Présentation

Avant d'établir définitivement une session avec une entité distante, il convient de vérifier si cette entité est bien celle qu'elle annonce être :

- protocole d'authenfication

Principe :

- les entités partagent un secret :
  - ☞ le mot de passe

Problème :

- lors de la transmission sur le réseau le mot de passe est visible
- par exemple : le protocole PAP ("password authentification protocol")
  - un simple protocole en 2 phases "two-way handshake"

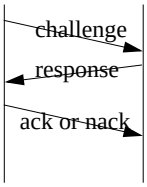
Solution :

- échange de messages chiffrés
- le protocole CHAP ("challenge authenfication protocol")

5.2. Le protocole CHAP

☞ rfc 1334

Fonctionne en 3 phases ("three-way handshake")



Principe de fonctionnement :

- Les 2 entités disposent d'un même procédé de chiffrement secret :
  - procédé paramétré par une clef commune et secrète.
  - par exemple : l'algorithme MD5
- la première entité soumet à l'autre un message ("challenge")
- la seconde retourne le message après chiffrement ("response")
- la première compare le message chiffré reçu et le chiffrement local du message puis en informe l'entité distante :
  - s'ils sont identiques l'authentification est réussie ("ack"),

- sinon elle a échoué (“nack”)

Remarques :

- l’existence de la fonction inverse n’est pas nécessaire !
- le message soumis doit être différent à chaque fois pour éviter le rejeu
- l’authentification peut être renouvelée pendant la phase de transfert des données

5.3. Les paquets CHAP

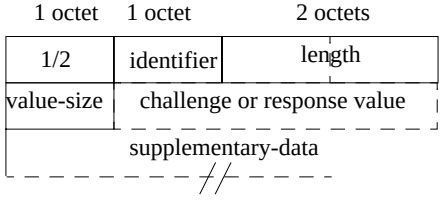
La fonction d’authentification est sélectionnée grâce à l’option de type 3 des paquets de configuration du protocole LCP

Les paquets CHAP sont transmis dans des paquets PPP dont le champ *protocol* vaut 0xc223.

Le format des paquets CHAP est similaire à celui des paquets LCP.

4 types de paquets CHAP :

- Dépendent du champ *code* (premier octet) :
  - 1 = Challenge



Paquet CHAP de challenge ou de réponse

  - 2 = Response
  - 3 = Success
  - 4 = Failure

6. Les protocoles NCP

6.1. Présentation des NCP

Chaque protocole de niveau Réseau a un protocole de configuration (NCP) adapté :

- IP ☞ IPCP (rfc 1332)
- Appletalk ☞ ATCP (rfc 1378)
- IPX ☞ IPXCP (rfc 1552)

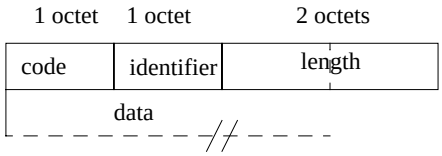
La forme que prennent les paquets NCP dépend du protocole de niveau Réseau employé.

La fonction de configuration de niveau Réseau ressemble à celle de niveau Liaison.

☞ IPCP

6.2. Introduction à IPCP

Le format général des paquets IPCP est le même que celui de LCP



Paquet IPCP

Les mêmes IPCP paquets sont utilisés :

- avec les mêmes codes :
  - config-req [1], config-ack [2], config-nack [3], config-rej [4], term-req [5], term-ack [6], code-rej [7].
  - mais seulement ceux-là.
- pour la même utilisation
- avec le même format particulier pour chaque type de paquets

Seules les options changent !

6.3. Les options des paquets IPCP

Le format général des options de IPCP est celui des options de LCP :

☞ TLV

Option de type=1 : configuration des adresses

- obsolète car difficile d'assurer la convergence
- cf. type=3

Option de type=2 : choix de l'algorithme de compression

- identificateur de l'algorithme (2 octets) :
  - 0x002d : Van Jacobson compressed TCP/IP
- le champ *données* dépend de l'algorithme choisi :
  - pour l'algorithme de Van Jacobson, 2 octets :
    . nombre de connexions pouvant avoir leurs entêtes simultanément compressées
    . présence du numéro de connexion : obligatoire (0) ou non (1)

Option de type=3 : configuration de l'adresse IP utilisée sur la liaison

- le champ *données* contient l'adresse IP proposée (4 octets)
  - 0.0.0.0 : c'est une demande d'obtention d'adresse

6.4. Compression des entêtes TCP/IP

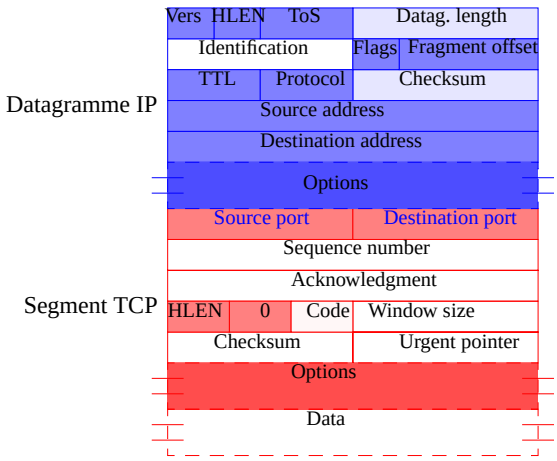
6.4.1 Introduction

Procédé de compression défini par le rfc 1144

- générique
- utile pour les liaisons à faible débit
  - sur liaison point à point utilisant PPP, par ex.

Deux paquets successifs d'une même connexion TCP possèdent des champs identiques :

- transmission des champs différents
  ☞ moins de champs
- transmission de la valeur de la différence
  ☞ des champs plus courts



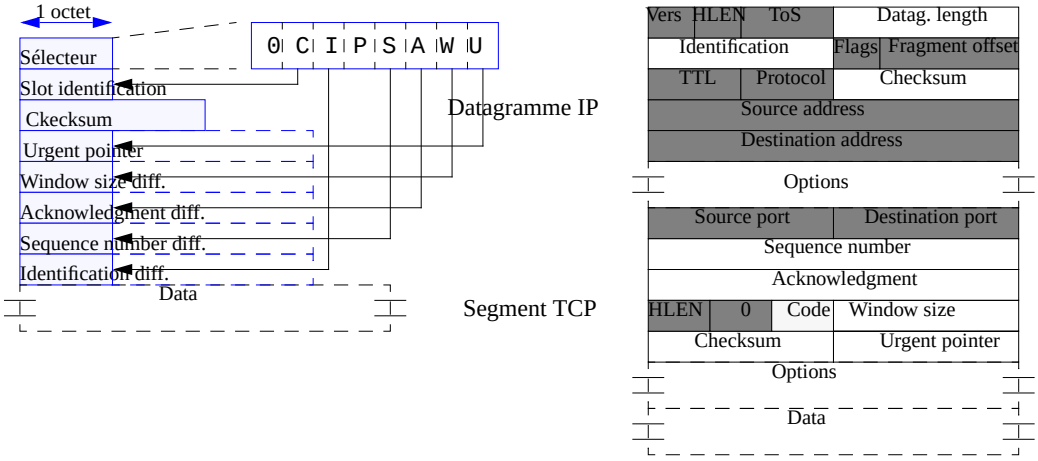
Hypothèses nécessaires à la compression :

- Compression uniquement des segments TCP :
  - ☞ non compression des paquets UDP
- Pas de compression s'il y a une nouvelle partie variable dans l'entête :
  - soit options IP,
  - soit options TCP
- Pas de compression si le paquet TCP est spécial:
  - ☞ bits RST, SYN ou FIN du champ *Code* positionnés
- Pas de compression si les champs constants sont modifiés :
  - Vers, HLEN, ToS, Flags, Fragment Offset (pas de fragmentation=0), TTL
  - Protocol, Destination and Source addresses, Destination and Source ports
- Les champs *checksum*, *datagram length* d'IP sont redondants donc reconstituables.

6.4.2 Format de l'entête compressée

Le paquet compressé a un format variable :

- Seuls l'octet *Sélecteur* et le champ *Checksum* sont toujours présents



6.4.3 Les différents champs de l'entête compressée

Sélecteur

- Indique la présence ou l'absence du champ optionnel correspondant :
  - C, I, S, A, W, U
- Bit "Push" du champ *code* du paquet TCP :
  - P (idem bit "ack" : A, bit "Urg": U)

Slot identification

- Identifie le contexte (un par sens de transmission) où sont conservées les données nécessaires à la compression/décompression
- Lorsque la compression ne peut pas avoir lieu, le champ *protocol* du datagramme IP contient cet identificateur. Le champ *protocol* du paquet PPP vaut 0x002f.

Checksum

- Champ obligatoire
- Les entêtes compressées sont plus sensibles aux erreurs

Urgent pointer

- On code la valeur et pas la différence

- La présence du champ est conditionné par la validation du bit U (=1) du champ *sélecteur*

Window size difference

- Code la différence du champ entre 2 paquets successifs
- Les valeurs négatives sont représentées en complément à 2
- Si la largeur de la fenêtre est constante, le champ n'existe pas et le bit W vaut 0.

Acknowledgment number et Sequence number difference

- Code la différence du champ entre 2 paquets successifs
- Comme la capacité d'encodage est limitée ($<2^{16}$) alors que la différence peut être grande (2^{32}) :
  - les paquets contenant des champs dont la différence est trop importante ou négative ne sont pas compressés

Identification difference

- Code la différence du champ entre 2 paquets successifs
- Si le bit I vaut 0, le champ n'est pas présent et sa valeur est obtenue après incrémentation de la valeur précédente

6.4.4 Codage des valeurs des champs optionnels de l'entête compressée

La valeur du champ (Urgent pointer) lorsqu'elle est habituellement petite ou la différence entre les deux valeurs successives (les autres champs optionnels)

On code la différence entre les valeurs successives du même champ :

- les petites valeurs sont plus fréquentes

Procédé :

- si la valeur < 256 ☞ codage sur un octet
- si la valeur >= 256 ou nulle ☞ codage sur 3 octets,
  - le premier octet vaut 0,
  - les deux suivants, la valeur
- Par exemple :
  - codage de 0 ☞ 0x000000
  - codage de 2 ☞ 0x02

6.5. Empilement

L'utilisation de la fonction de compression de l'entête est sélectionnée lors de la configuration par l'option 2 du protocole IPCP.

Toutefois la compression de l'entête n'est pas toujours applicable.

Les paquets PPP, en fonction de la valeur de leur champ *Protocol*, contiennent :

- 0x0021 = un datagramme IP à ne pas comprimer
  - la fonction de compression n'a pas été sélectionnée
  - ou le datagramme ne contient pas un segment TCP
- 0x002d = une entête comprimée
  - la fonction de compression a été sélectionnée
  - et le datagramme contient un segment TCP compressible
- 0x002f = une entête à comprimer mais non-compressible
  - la fonction de compression a été sélectionnée
  - mais le datagramme contient un segment TCP non-compressible :
    . par exemple : il y a des options ou un champ qui devrait être fixe ne l'est pas.

6.6. Optimisation

Si l'on observe le trafic, on constate que, fréquemment :

- la valeur champ *Sequence number* augmente de la taille du dernier paquet
- la valeur champ *Acknowledgment number* (1^{er} cas) soit augmente de la taille du dernier paquet ($2^{ème}$ cas) soit est inchangée
- les valeurs des champs *Window size* et *Urgent Pointer* ne sont pas modifiées

On décide de coder ces 2 cas (ne nécessitant pas de champs optionnels) par des configurations spécifiques des bits du champ *Sélecteur*.

- 0x0b = SWU (1^{er} cas)
- 0x0f = SAWU ($2^{éme}$ cas)

☞ L'entête est alors réduite à uniquement 3 octets !

Si ces configurations spécifiques du *Sélecteur* doivent être utilisées :

- on transmet une entête non-compressée

7. Conclusion

Le protocole PPP est adapté à la transmission de datagramme IP sur des liaisons point à point

Le protocole PPP est un protocole générique, il s'adapte :

- au procédé de transmission (fiable ou non, synchrone ou asynchrone)
- aux différents protocoles de niveau Réseau

Utilise des protocoles de configuration de niveau Liaison (LCP) et de niveau Réseau (NCP : IPCP)

Peut être associé à un protocole d'authentification CHAP

Adaptation à la transmission : inspirée du protocole HDLC (ISO)

Technique de compression des entêtes TCP/IP : adaptation aux lignes à faible débit

PPP est utilisé pour le pontage dans les réseaux locaux

- transport de trames Ethernet au sein des paquets PPP
- administration des algorithmes de pontage à l'aide de NCP