TD N° 2: protocoles SLIP et HDLC

Rémy Grünblatt – remy@grunblatt.org

23 octobre 2019

1 Protocole SLIP

Le protocole SLIP (pour *Serial Line Internet Protocol*) est un protocole d'encapsulation qui permet à deux ordinateurs reliés entre eux par une liaison série d'échanger des paquets IP, en point-à-point. Il est défini dans la RFC 1055 (Requests For Comments) et est souvent surnommé « Rick Adams' SLIP », du nom de son créateur et en raison du grand nombre de protocoles portant aussi le nom de SLIP.

On considère la topologie suivante :

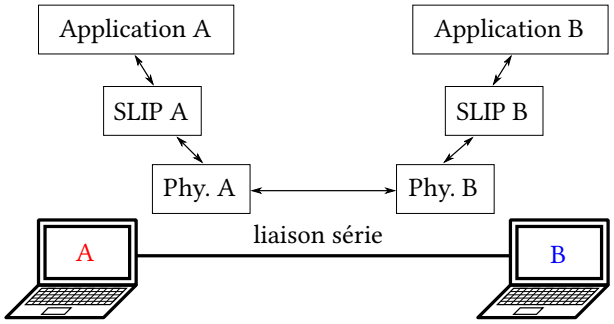

FIGURE 1 – Représentation d'une connection via SLIP

1. À côté de chaque flèche de la figure 1, spécifier le type de signal ou de données échangé entre les deux entités séparées par la flèche.
2. On suppose que la couche SLIP A reçoit la séquence suivante depuis la couche *Phy. A*, exprimée en hexadécimal, c'est-à-dire en base 16 :
   0x6C 0xC0 0x6E 0x6F 0x75 0x76 0x65 0x6C 0x6C 0x65 0x73 0x20 0x64 0x75 0xDB 0xDC 0x20 0xC0 0xAB 0xC0 0x6D 0x6F 0x6E 0x64 0x65 0xC0
   Quelles sont les données transmises par la couche *SLIP A* à la couche *Application A* ? Pourquoi ?
3. *Application A* souhaite transmettre les données suivantes :
   0x63 0x65 0x63 0x69 0x20 0x63 0x65 0x74 0xC0 0x20 0x75 0x6E 0x20 0x74 0x65 0x74 0x65 0xC0
   Quelles sont les données transmises par la couche *SLIP A* à la couche *Phy. A* ?
4. La couche *SLIP B* reçoit ces données. Quelles données passera-t-elle à *Application B* ?
5. On donne cet extrait de la RFC définissant SLIP :

   ```
   SLIP DRIVERS

      The following C language functions send and receive SLIP packets.
      They depend on two functions, send_char() and recv_char(), which send
      and receive a single character over the serial line.
   ```

```c
#define END 0300
#define ESC 0333
#define ESC_END 0334
#define ESC_ESC 0335

void send_packet(char *p, int len) {
  send_char(END);
  while (len--) {

```

```
10      switch (*p) {
11
12        case END:
13          send_char(ESC);
14          send_char(ESC_END);
15          break;
16
17        case ESC:
18          send_char(ESC);
19          send_char(ESC_ESC);
20          break;
21
22        default:
23          send_char(*p);
24      }
25      p++;
26    }
27    send_char(END);
28  }
29
30  int recv_packet(char *p, int len) {
31    char c;
32    int received = 0;
33    while (1) {
34      c = recv_char();
35
36      switch (c) {
37
38        case END:
39          if (received)
40            return received;
41          else
42            break;
43
44        case ESC:
45          c = recv_char();
46          switch (c) {
47
48            case ESC_END:
49              c = END;
50              break;
51
52            case ESC_ESC:
53              c = ESC;
54              break;
55          }
56
57        default:
58          if (received < len)
59            p[received++] = c;
60      }
61    }
62  }
```

6. Commenter le code et ré-exprimer les algorithmes d'envoi et de réception de données en français.

2 Protocole HDLC

Le protocole HDLC (*High Level Data Link Control*) est un protocole de la couche *liaison de données* du modèle OSI, c'est-à-dire la couche *liaison* (*link*) du modèle TCP/IP. Le protocol PPP s'inspire fortement de HDLC. Les trames binaires utilisées par le protocole HDLC possèdent le format suivant :

Figure 2 – Format d'une trame HDLC.

1. Le fanion de début et le fanion de fin indiquent les deux extrêmités de la trame : ils sont constitués de la chaîne binaire 0111 1110.
2. L'adresse de destination identifie l'ETTD destinataire
3. Le champ Commande contient toutes les informations nécessaires à l'interprétation de la trame.
4. Suivant le rôle de la trame, le champ Données contient une chaîne binaire ou non.
5. Le Contrôle d'erreur est réalisé par un code CRC basé qur le polynôme générateur G(x)=x 16 + x 12 + x 5 + 1
6. Le traitement de la transparence binaire est réalisé comme suit : une phase de lecture systématique de la trame est réalisée avant émission pour insérer un 0 après cinq 1 consécutifs. Lors de la réception, ces 0 insérés artificiellement sont supprimés.

Il y a 3 types de trames spécifiés par le protocole HDLC, selon le contenu du champ *Commande* :

1. **Trame d'information** :
   Ce type de trame est employé pour le transport de données binaires ; c'est le seul type de trame pour lequel le champ de *Données* n'est pas vide.

1 bit	3 bits	1 bit	3 bits
0	N(S)	P/F	N(R)

Figure 3 – Format du champ Information - Trame d'information

   N(s) est le numéro de la trame émise, qui sera utilisé pour recomposer les données en cas d'un échange entre les trames HDLC réalisé par la couche inférieure. Le bit P/F peut être interprété de 4 manières différentes, selon que la trame provient de l'initiateur de l'échange P ou de son correspondant F :

Source de la trame	P/F	Signification
Trame provenant de l'initiateur de l'échange	P=0	L'initiateur n'attend pas de réponse à cette trame
	P=1	L'initiateur attend une réponse à cette trame
Trame transmise à l'initiateur de l'échange	F=0	Le secondaire n'a pas terminé d'émettre des trames à l'initiateur
	F=1	Le secondaire a terminé ses envois de trames à l'initiateur

   Enfin, le champ N(R) permet d'acquitter toutes les trames dont le numéro est inférieur à N(R).
2. **Trame de supervision de données** : Ce type de trame est utilisée pour gérer l'échange ; le champ *données* est donc vide.

1 bit	1 bit	2 bits	1 bit	3 bits
1	0	SS	P/F	N(R)

Figure 4 – Format du champ Information - Trame de supervision de données

   Les 2 bits SS informent le destinataire de certaines requêtes de l'émetteur :

SS	Opération (Abbréviation)	Signification
00	Receive Ready (RR)	Acquittement de toutes les trames de numéro inférieur à N(R) et en attente.
01	Reject (REJ)	Demande de rejet de toutes les trames de numéro supérieur à N(R)
10	Receive Not Ready (RNR)	Demande d'une suspension des envois de trames après la trame de numéro N(R)
11	Selective Reject (SREJ)	Rejet de la trame N(R) et demande de retransmission de celle-ci

3. **Trame de supervision de la liaison** : Ce type de trame a pour rôle de gérer la mise en place et le relâchement de la connexion. Nous n'entrerons pas dans cet exercice plus dans le détail du fonctionnement de ce type de trame.

1 bit	1 bit	2 bits	1 bit	3 bits
1	1	MM	P/F	MMM

Figure 5 – Format du champ Information - Trame de supervision de la liaison

Un ETTD B est contacté sur l'un de ses ETCD par un autre ETTD A pour un échange de données. Au cours de cet échange, B reçoit la chaîne binaire suivante :

01111110 010111110 00101000 10111110 10101000011101011 01111110
01111110 010111110 10100101 1110010100110000 01111110
01111110 010111110 10001101 1100110011100101 01111110

1. Extraire les trames contenues dans cette chaîne binaire ;
2. Réaliser le traitement de la transparence binaire sur ces trames, comme effectué par B ;
3. Déterminer le type de chacune des trames ;
4. Déduire des résultats obtenus une vision globale du déroulement de l'échange en cours.