ESCOM-IPN

Práctica 4
Analizador de Trama LLC

Redes de Computadoras

Laura Andrea Morales López

MSc. NIDIA ASUNCIÓN CORTEZ DUARTE

Abril 2018

Resumen

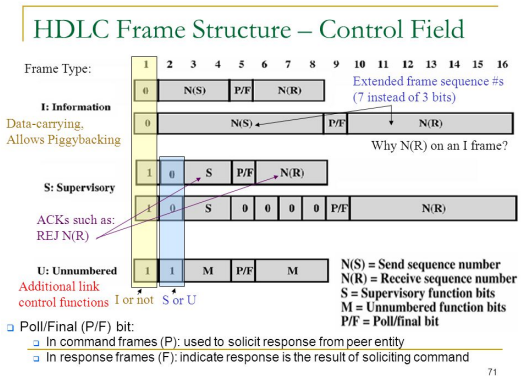
In this report we will see an analysis of LLC frames and a program to do it quickly.

Índice

1. Indroducción: Control de enlace lógico

El protocolo LLC (control lógico de enlace) es un protocolo de capa de enlace de datos derivado de HDLC, del cual hereda su campo de control, y fue estandarizado por la IEEE bajo la denominación 802.2. Igual que en HDLC se tienen tramas de información, supervisión y no numeradas distinguiéndose entre ellas por los bits menos significativos de su campo de control.

Tenemos la siguiente cabecera:



Con esta podemos analizar tramas de este tipo.

2. Problema

Realizar este tipo de análisis en muy especifico y fácil de realizar, sin embargo tiende a los errores humanos.

Como vimos en la cabecera tenemos los boques de la información y sus respectivo significados con lo cual podemos leer de alguna manera lo que nos estan diciendo las tramas.

3. Hipotesis:

Una cabecera LLC puede ser leída de manera secuencial, por lo tanto es posible programar la lectura del las mismas.

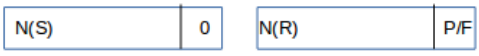
4. Software

- Librería stdlib.h

- Librería stdio.h
- Librería stdbool.h
- Librería string.h

5. Procedimiento

- Se analiza cada trama por separado
- Procesamos el campo de control que se encuenta el en byte 17.
- Checamos el tipo de trama que tendremos, para esto realizamos un & con 3.
  - Si es un 0 o un 2 entonces es de Información. En este tipo de trama la cabecera de 2 bytes es de la siguiente marera con lo que realizando unicamente un corrimiento de >> 1 podemos obtener N(s) del byte 17 o N(R) del byte 18. Y además obtenemos el P/F con un &1.



  - Si es un 1 tenemos una trama de Supervisión.

    Con esta trama podemos obtener el valor del comando realizando un corrimiento >> 2 y luego un &3.

    Despues checando el P/F podemos determinar si es necesario char el SAPd para saber si es comando o respuesta Si P/F=1 chacamos el Sap (byte 16) y si es 0 es Comando y si es 1 es Respuesta.

    El P/F lo podemos obtener realizando un & 1 en el byte 18.

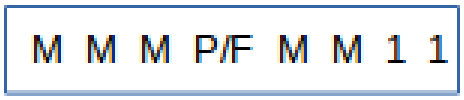
    Y el Sap con un & 1 con el byte 16.

    Con esto tenemos la posicón del comando o respuesta a usar dependiento de la siguiente tabla.

SS	Bits	Comando	Significado
0	00	RR	Receptor Listo
1	01	RNR	Receptor no listo
2	10	REJ	Rechazo
3	11	SREJ	Rechazo selectivo

- Si es un 3 tenemos una trama Sin numerar.

  Como en la de supervisión checamos el P/F y el SAP y con corrimientos y and's: $((byte17 >> 2)\&3)|((byte17 >> 3)\&28)$ podemos obtener los números MM y con la siguiente tabla podemos obtener su significado.



M	Comando	Respuesta	Sig.Com	Sig.Res
0	UI	UI	Unnumered Information	Unnumered Information
1	SIM	RIM	Set Inicialization Mode	Request Inicialization Mode
3	SARM	DM	Modo Desconectado	
4	UP		Unnumered Pull	
7	SABM			
8	DISC	RD	Disconect	Request Disconect
11	SARME			
12		UA		Unnum. Acknow.
15	SABME			
16	SNRM		SetNormalResponseMode	
17		FRMR		
19	RSET			
23	XID	XID	Ex.Station Identif.	Ex. Station Identif.
27	SNMRE			

- Mostrar la información obtenida de cada trama.

6. Resultados

Tenemos los siguientes resultados:

```
Trama 1    ToT 3    T-Unnumerated Comando                SABME
Trama 2    ToT 3    T-Unnumerated Respuesta              UA
Trama 3    ToT 4    T-Supervisión              N(R):0               RR
Trama 4    ToT 4    T-Supervisión              N(R):0               RR
Trama 5    ToT 18   T-Informacion N(s):0    N(r):0
Trama 6    ToT 18   T-Informacion N(s):0    N(r):1
Trama 7    ToT 4    T-Supervisión              N(R):1               RR
Trama 8    ToT 4    T-Supervisión              N(R):1               RR
Trama 9    ToT 172  T-Información N(s):1    N(r)1
Trama 10   ToT 4    T-Supervisión              N(R):2               RR
Trama 11   ToT 95   T-Información N(s):1    N(r)2
Trama 12   ToT 4    T-Supervisión              N(R):2               RR
Trama 13   ToT 145  T-Informacion N(s):2    N(r):2
Trama 14   ToT 4    T-Supervisión              N(R):3               RR
Trama 15   ToT 70   T-Informacion N(s):2    N(r):3
Trama 16   ToT 4    T-Supervisión              N(R):3               RR
Trama 17   ToT 126  T-Información N(s):3    N(r)3
Trama 18   ToT 4    T-Supervisión              N(R):4               RR
Trama 19   ToT 4    T-Supervisión              N(R):4               RR
Trama 20   ToT 126  T-Informacion N(s):4    N(r):4
Trama 21   ToT 4    T-Supervisión              N(R):5               RR
Trama 22   ToT 4    T-Supervisión              N(R):5               RR
Trama 23   ToT 18   T-Información N(s):5    N(r)5
Trama 24   ToT 4    T-Supervisión              N(R):6               RR
Trama 25   ToT 139  T-Unnumerated Sin PF
Trama 26   ToT 53   T-Informacion N(s):6    N(r):5
Trama 27   ToT 53   T-Informacion N(s):6    N(r):7
Trama 28   ToT 18   T-Información N(s):7    N(r)6
Trama 29   ToT 4    T-Supervisión              N(R):8               RR
Trama 30   ToT 18   T-Informacion N(s):8    N(r):6
Trama 31   ToT 4    T-Supervisión              N(R):9               RR
Trama 32   ToT 3    T-Unnumerated Comando                DISC
Trama 33   ToT 3    T-Unnumerated Respuesta              UA
```

7. Conclusión

Realizar este tipo de programa evita tener que realizar operaciones que tienden a error humano, tales como olvidar un bit o colocar alguno demás, ademas de así mejorar el tiempo en que se realizaría.

Como se vió mecanizarlo es sencillo por su orientación a bits con ello podemos analizar muchas tramas de manera rápida.

8. Código

```
/*############################  TRAMA ANALIZER  ############################
* author Laura Andrea Morales* version 0.1
* team CompilandoConocimiento* date 4/03/2018
* compile "gcc TramaLLC.c -o TramaLLC"* run "./TramaLLC "
*/
#include <stdlib.h>
#include <stdio.h>
#include<stdbool.h>
#include <string.h>
#define C1 16
#define C2 17
#define NoTramas 33
typedef unsigned int uint;
typedef unsigned char byte;
typedef unsigned int sint;

char Supervision[4][5]={"RR","RNR","REJ","SREJ"};
char
     UnumerC[33][7]={"UI","SIM","-","SARM","UP","-","-","SABM","DISC","-","-","SARME","-","--","-","SABME","SNRM","----"
char UnumerR[33][7]={"UI","RIM","-","DM","-","-","-","-","RP","-","-","-","
     UA","-","-","----","----","FRMR","-","-----","","","","XID"};


char resultado[100000]={};

bool Analizer(byte T[], int i){
    sint ToT=0;
    ToT=(ToT<<8)|T[12];
    ToT|=T[13];
    sprintf(resultado+strlen(resultado),"\n      Trama %-3d ",i+1);
    sprintf(resultado+strlen(resultado)," ToT %-4d ",ToT);

    if(ToT<1500){
        //sprintf(resultado+strlen(resultado),"Tipo %d ",T[C1]&0b11);
        switch(T[C1]&0b11){
            case 0:
        //  printf("Entre a caso 0\n");
            sprintf(resultado+strlen(resultado),"T-Informacion N(s):%-5dN(r):%-5d
     ",T[C1]>>1,T[C2]>>1);
            break;

            case 1:
        //  printf("Entre a caso Trama de Supervisión\n");
            //printf("%d\n",(T[C1]>>2)&3 );

            sprintf(resultado+strlen(resultado),"T-Supervisión      N(R):%-5d %s
     ",T[C2]>>1,Supervision[(T[C1]>>2)&3]);
            break;

            case 2:
        //  printf("Entre a caso Trama de Información\n");
            sprintf(resultado+strlen(resultado),"T-Información N(s):%-5dN(r)%-5d
     ",T[C1]>>1,T[C2]>>1);
            break;

            case 3:
            //printf("Entre caso 3 Unnummered\n");
            if (((T[C1]>>4)&1))
            {
            //  printf("Checa Sap\n");
                if (T[15]&1)
                {
                    //printf("Respuesta %d",(T[C1]>>2)&3)|((T[C1]>>3)&28);
```

```
                    sprintf(resultado+strlen(resultado),"T-Unnumerated Respuesta %15s
    ",UnumerR[((T[C1]>>2)&3)|((T[C1]>>3)&28)]);
                }
                else if (!(T[15]&1))
                {
                    //printf("Comando %d",(T[C1]>>2)&3)|((T[C1]>>3)&28);
                    sprintf(resultado+strlen(resultado),"T-Unnumerated Comando    %13s
    ",UnumerC[((T[C1]>>2)&3)|((T[C1]>>3)&28)]);
                }

            }
            else sprintf(resultado+strlen(resultado),"T-Unnumerated Sin PF");


            break;



        }
        return 1;
    }
    else return 0;
}

bool AnalizarTrama(byte T[][250]){


    for (int i = 0; i < NoTramas; ++i)
    {
        if(!(Analizer(&T[i][0],i))){

            return 0;
        }
    }
    return 1;

}


int main(int argc, char const *argv[]){

byte T[33][250]={
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x03,0xf0,0xf0,
0x7f,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 1
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x03,0xf0,0xf1,
0x73,0x81,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 2
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf0,
0x01,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00},//Trama 3
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
0x01,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 4
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0,
0x00,0x01,0x0e,0x00,0xff,0xef,0x19,0x8f,0xbc,0x05,0x7f,0x00,0x23,0x00,0x7f,0x23}, //Trama 5
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x12,0xf0,0xf0,0x00,0x03,0x0e,0x00,0xff,0xef,0x19,0x8f
    //Trama 6
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,0x01,0x03,0x00,0x00,0x00,0x00,0x00,0x00
    //Trama 7
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,0x01,0x03,0x00,0x00,0x00,0x00,0x00,0x00
    //Trama 8
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0xac,0xf0,0xf0,0x02,0x02,0x0e,0x00,0xff,0xef,0x16,0x04
    //Trama 9
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,0x01,0x04,0x00,0x00,0x00,0x00,0x00,0x00
    //Trama 10
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x5f,0xf0,0xf0,0x02,0x04,0x0e,0x00,0xff,0xef,0x16,0x0c
    //Trama 11
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1,
0x01,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 12

{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x91,0xf0,0xf0,
0x04,0x04,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23}, //Trama 13
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
0x01,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 14
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x46,0xf0,0xf0,
0x04,0x06,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23}, //Trama 15
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1,
0x01,0x06,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 16
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x7e,0xf0,0xf0,0x06,0x06,0x0e,0x00,0xff,0xef,0x16,0x0c
    //Trama 17
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
0x01,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 18
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1,
0x01,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 19
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x7e,0xf0,0xf0,
0x08,0x08,0x0e,0x00,0xff,0xef,0x16,0x0c,0x00,0x00,0x28,0x00,0x28,0x00,0x7f,0x23}, //Trama 20
{0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
0x01,0x0a,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 21
{0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x04,0xf0,0xf1,
0x01,0x0a,0x0e,0x00,0xff,0xef,0x19,0x8f,0xbc,0x05,0x7f,0x00,0x23,0x00,0x7f,0x23}, //Trama 22
```

```
138 {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0,
139 0x0a,0x0b,0x0e,0x00,0xff,0xef,0x14,0x00,0x00,0x00,0x28,0x00,0x00,0x00,0x7f,0x23}, //Trama 23
140 {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
141 0x01,0x0d,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 24
142 {0x03,0x00,0x00,0x00,0x01,0x00,0x04,0xac,0x44,0x4d,0x02,0x00,0x8b,0xf0,0xf0,0x03,0x2c,0x00,0xff,0xef,0x08,0x00,0x00
        //Trama 25
143 {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x35,0xf0,0xf0,
144 0x0c,0x0a,0x0e,0x00,0xff,0xef,0x16,0x04,0x00,0x00,0x00,0x00,0x28,0x00,0x7f,0x23}, //Trama 26
145 {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x35,0xf0,0xf0,
146 0x0c,0x0e,0x0e,0x00,0xff,0xef,0x16,0x04,0x00,0x00,0x00,0x00,0x28,0x00,0x7f,0x23}, //Trama 27
147 {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0,
148 0x0e,0x0d,0x0e,0x00,0xff,0xef,0x14,0x00,0x00,0x00,0x28,0x00,0x00,0x00,0x7f,0x23}, //Trama 28
149 {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
150 0x01,0x11,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 29
151 {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x12,0xf0,0xf0,
152 0x10,0x0d,0x0e,0x00,0xff,0xef,0x18,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x7f,0x23}, //Trama 30
153 {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x04,0xf0,0xf1,
154 0x01,0x13,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 31
155 {0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x03,0xf0,0xf0,
156 0x53,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Trama 32
157 {0x00,0x02,0xb3,0x9c,0xdf,0x1b,0x00,0x02,0xb3,0x9c,0xae,0xba,0x00,0x03,0xf0,0xf1,
        0x73,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}}; //Trama 33
    };
158
159
160
161     if(AnalizarTrama(T)){
162         printf("%s\n",resultado);
163     }
164
165     return 0;
166 }
```

Referencias

[1] Nidia Cortez. *Redes de Computadoras* ESCOM, 2018.