Министерство образования и науки РФ
Новосибирский Государственный Технический Университет
Кафедра ПМт

Лабораторная работа №5

«Анализ структуры кадра/фрейма Ethernet»

по курсу «Сетевые информационные технологии»

Вариант 1

Факультет: ПМИ
Группа: ПММ-81
Студенты: Михайлов А. А.,
Санина А. А.

Преподаватель: Долозов Н. Л.

Новосибирск 2012

1. Цель работы

Спроектировать и реализовать программу, выполняющую анализ структуры кадра/фрейма технологии Ethernet.

2. Задание

Кадры представлены в виде файлов двоичного формата (отсутствует преамбула и контрольная сумма, для исходящего кадра длина может быть меньше минимальной). В программе должна быть предусмотрена возможность выбора файла.

Минимальная информация, которую должна выдавать программа должна включать:

1) Количество фреймов в файле.

2) Тип каждого фрейма.

3) IP- адреса (основную информацию заголовка IP-пакета).

4) MAC- адреса (основную информацию заголовка Кадра).

Программа должна выполнить анализ файла с именем ethernetxx, где xx номер вашей бригады.

2.1. Исходный текст программы

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  #include <winsock.h>
6
7
8
9  #define ETHERNET_HEADER_LENGTH 14
10 #define IPPACKET_HEADER_LENGTH 20
11 #define ARP_REQUEST_LENGTH     28
12 #define SPANNING_TREE_LENGTH   38
13
14 #define ETHERNET_FRAME_MIN_LEN 60
15
16
17 #pragma pack(1)                 // disable structure packing in msvs
18 struct IPPacket {
19     struct IPPacketHeader {     // Header length is 20 bytes for IPv4
20         u_char  version;                    // must be 0x45 for correct IPv4
21         u_char  service;
22         u_short length;                     // total length (headers + body)
23         u_short identification;
24         u_char  flags;                      // some flags again
25         u_char  offset;
26         u_char  livetime;
27         u_char  protocol;
28         u_short checksum;                   // not for all packet, just for headers
29         u_char  sourceIP[4];
30         u_char  destinationIP[4];
31     } header;
32
33     struct IPPacketBody {
34         void *data;
35         //IPPacketBody() : data( NULL ) { }
36     } body;
37 };
38
39
40 #pragma pack(1)                 // disable structure packing in msvs
41 struct ARPRequest {
42     u_short h_type;
43     u_short p_type;
44     u_char  h_size;
45     u_char  p_size;
46     u_short opcode;
47     u_char  senderMAC[6];
48     u_char  senderIP [4];
49     u_char  targetMAC[6];
50     u_char  targetIP [4];
51 };
52
53
54 #pragma pack(1)                 // disable structure packing in msvs
55 struct SpanningTree {
56     u_char  logical_link[3];
57     u_char  tree[35];
58 };
59
60
61 #pragma pack(1)                 // disable structure packing in msvs
62 struct EthernetFrame {
```

```
63      struct EthernetFrameHeader {
64          u_char  destinationMAC[6];
65          u_char  sourceMAC[6];
66          u_short etherType;       // must be 0x0800 for IPv4
67      } header;
68
69      union EthernetFrameBody {
70          IPPacket         ip_packet;
71          ARPRequest       arp_request;
72          SpanningTree     spanning_tree;
73      } body;
74  };
75
76
77
78
79  int getFileSize( FILE *input )
80  {
81      int fileSizeBytes;
82      fseek(input, 0, SEEK_END);
83      fileSizeBytes = ftell(input);
84      fseek(input, 0, SEEK_SET);
85
86      return fileSizeBytes;
87  }
88
89
90
91
92  char *MACToStr( u_char *addr, char *buf ) {
93      sprintf( buf, "%x:%x:%x:%x:%x:%x", addr[0], addr[1], addr[2], addr[3], addr[4], addr[5] );
94      return buf;
95  }
96
97
98
99
100 char *IPToStr( u_char *addr, char *buf ) {
101     sprintf( buf, "%d.%d.%d.%d", addr[0], addr[1], addr[2], addr[3] );
102     return buf;
103 }
104
105
106
107 bool isSpanningTree( u_char *addr ) {
108     return ( addr[3] == 0x00 ) && ( addr[4] == 0x00 ) && ( addr[5] == 0x00 );
109 }
110
111
112
113
114 int main( int argc, char **argv )
115 {
116     if ( argc < 2 ) {
117         printf( "Usage : %s dump.bin\n", argv[0] );
118         return EXIT_FAILURE;
119     }
120
121     char filename[FILENAME_MAX];
122     //strcpy( filename, "I:/ethers02.bin" );
123     strcpy( filename, argv[1] );
124
125     char buf[32];        // need to keep formatted strings
126
127     FILE *f = fopen( filename, "rb" );
128     if ( !f ) {
129         printf( "Cannot open file %s\n", filename );
130         return EXIT_FAILURE;
131     }
132     u_long file_size = getFileSize( f );
133
134     EthernetFrame frame;
135
136     u_long current_position;
137     u_long i;
138     for ( current_position = 0, i = 1; current_position < file_size; current_position = ftell( f ), ++i ) {
139         printf( "================================================================\n" );
140         printf( "frame %d found at %d :\n", i, current_position );
141         fread( &frame, ETHERNET_HEADER_LENGTH, 1, f );
142         u_short etherType = ntohs(frame.header.etherType);
143         printf( "    Source MAC : (%s)\n", MACToStr( frame.header.sourceMAC, buf ) );
144         printf( "    Destination MAC : (%s)\n", MACToStr( frame.header.destinationMAC, buf ) );
145         printf( "    EtherType/Length : 0x%04x\n", etherType );
146         if ( etherType == 0x0800 ) {          // IP
147             fread( &(frame.body.ip_packet), IPPACKET_HEADER_LENGTH, 1, f );
148             printf( "    Type : IP Packet\n" );
149             printf( "    Source IP : (%s)\n", IPToStr( frame.body.ip_packet.header.sourceIP, buf ) );
150             printf( "    Destination IP : (%s)\n", IPToStr( frame.body.ip_packet.header.destinationIP, buf
                    ) );
151             u_short length      = ntohs( *(short*)(&frame.body.ip_packet.header.length) );
152             u_short body_length = length - IPPACKET_HEADER_LENGTH;
153             body_length = max( body_length, ETHERNET_FRAME_MIN_LEN - ETHERNET_HEADER_LENGTH -
                    IPPACKET_HEADER_LENGTH );
154             printf( "    Total frame length : %d\n", body_length + ETHERNET_HEADER_LENGTH +
```

```
                IPPACKET_HEADER_LENGTH );
155             fseek( f, body_length, SEEK_CUR );
156         } else if ( etherType == 0x0806 ) {     // ARP
157             fread( &(frame.body.arp_request), ARP_REQUEST_LENGTH, 1, f );
158             printf( "    Type : ARP Request\n" );
159             printf( "    Sender MAC : (%s)\n", MACToStr( frame.body.arp_request.senderMAC, buf ) );
160             printf( "    Sender IP : (%s)\n",  IPToStr( frame.body.arp_request.senderIP, buf ) );
161             printf( "    Target MAC : (%s)\n", MACToStr( frame.body.arp_request.targetMAC, buf ) );
162             printf( "    Target IP : (%s)\n",  IPToStr( frame.body.arp_request.targetIP, buf ) );
163             u_short body_length = ETHERNET_FRAME_MIN_LEN - ETHERNET_HEADER_LENGTH - ARP_REQUEST_LENGTH;
164             printf( "    Total frame length: %d\n", body_length + ETHERNET_HEADER_LENGTH +
                    IPPACKET_HEADER_LENGTH );
165             fseek( f, body_length, SEEK_CUR );
166         } else if ( isSpanningTree( frame.header.destinationMAC ) ) {
167             fread( &(frame.body.spanning_tree), SPANNING_TREE_LENGTH, 1, f );
168             printf( "    Type : Spanning tree\n" );
169         } else {
170             printf( "    Type : Unknown packet\n" );
171             goto end;
172         }
173     }
174 end:
175     printf( "================================================================\n" );
176     printf( "Proceed: %d/%d bytes of source\n", current_position, file_size );
177
178     return EXIT_SUCCESS;
179 }
```

2.2. Результат анализа

```
================================================================
frame 1 found at 0 :
    Source MAC : (0:90:27:a1:36:d0)
    Destination MAC : (0:2:16:9:fa:40)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (195.62.2.11)
    Destination IP : (62.167.64.216)
    Total frame length: 153
================================================================
frame 2 found at 153 :
    Source MAC : (0:2:16:9:fa:40)
    Destination MAC : (0:90:27:a1:36:d0)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (81.181.78.206)
    Destination IP : (195.62.2.11)
    Total frame length: 66
================================================================
frame 3 found at 219 :
    Source MAC : (0:2:16:9:fa:40)
    Destination MAC : (0:90:27:a1:36:d0)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (81.181.78.206)
    Destination IP : (195.62.2.11)
    Total frame length: 86
================================================================
frame 4 found at 305 :
    Source MAC : (0:90:27:a1:36:d0)
    Destination MAC : (0:2:16:9:fa:40)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (195.62.2.11)
    Destination IP : (81.181.78.206)
    Total frame length: 66
================================================================
frame 5 found at 371 :
    Source MAC : (0:90:27:a1:36:d0)
    Destination MAC : (0:2:16:9:fa:40)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (195.62.2.11)
    Destination IP : (81.181.78.206)
    Total frame length: 810
================================================================
frame 6 found at 1181 :
    Source MAC : (0:2:16:9:fa:40)
    Destination MAC : (0:8:2:8f:da:6e)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (205.188.9.82)
    Destination IP : (195.62.2.42)
    Total frame length: 252
================================================================
frame 7 found at 1433 :
    Source MAC : (0:2:16:9:fa:40)
    Destination MAC : (0:90:27:a1:36:d0)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (81.181.78.206)
    Destination IP : (195.62.2.11)
    Total frame length: 218
================================================================
```

```
frame 8 found at 1651 :
    Source MAC : (0:90:27:a1:36:d0)
    Destination MAC : (0:2:16:9:fa:40)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (195.62.2.11)
    Destination IP : (81.181.78.206)
    Total frame length: 66
================================================================
frame 9 found at 1717 :
    Source MAC : (0:8:2:8f:da:6e)
    Destination MAC : (ff:ff:ff:ff:ff:ff)
    EtherType/Length : 0x0806
    Type : ARP Request
    Sender MAC : (0:8:2:8f:da:6e)
    Sender IP : (195.62.2.42)
    Target MAC : (0:0:0:0:0:0)
    Target IP : (195.62.2.1)
    Total frame length: 52
================================================================
frame 10 found at 1777 :
    Source MAC : (0:2:16:9:fa:40)
    Destination MAC : (0:8:2:8f:da:6e)
    EtherType/Length : 0x0806
    Type : ARP Request
    Sender MAC : (0:2:16:9:fa:40)
    Sender IP : (195.62.2.1)
    Target MAC : (0:8:2:8f:da:6e)
    Target IP : (195.62.2.42)
    Total frame length: 52
================================================================
frame 11 found at 1837 :
    Source MAC : (0:8:2:8f:da:6e)
    Destination MAC : (0:2:16:9:fa:40)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (195.62.2.42)
    Destination IP : (205.188.9.82)
    Total frame length: 60
================================================================
frame 12 found at 1897 :
    Source MAC : (0:2:16:9:fa:40)
    Destination MAC : (0:90:27:a1:36:d0)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (62.167.64.216)
    Destination IP : (195.62.2.11)
    Total frame length: 60
================================================================
frame 13 found at 1957 :
    Source MAC : (0:50:8b:95:40:a8)
    Destination MAC : (ff:ff:ff:ff:ff:ff)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (195.62.2.14)
    Destination IP : (195.62.2.63)
    Total frame length: 92
================================================================
frame 14 found at 2049 :
    Source MAC : (0:50:8b:95:40:a8)
    Destination MAC : (ff:ff:ff:ff:ff:ff)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (195.62.2.14)
    Destination IP : (195.62.2.63)
    Total frame length: 92
================================================================
frame 15 found at 2141 :
    Source MAC : (0:2:16:9:fa:40)
    Destination MAC : (0:90:27:a1:36:d0)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (168.95.1.14)
    Destination IP : (195.62.2.11)
    Total frame length: 216
================================================================
frame 16 found at 2357 :
    Source MAC : (0:90:27:a1:36:d0)
    Destination MAC : (0:2:16:9:fa:40)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (195.62.2.11)
    Destination IP : (168.95.192.14)
    Total frame length: 102
================================================================
frame 17 found at 2459 :
    Source MAC : (0:4:4d:8a:b0:d5)
    Destination MAC : (1:80:c2:0:0:0)
    EtherType/Length : 0x0026
    Type : Spanning tree
================================================================
frame 18 found at 2511 :
    Source MAC : (0:2:16:9:fa:40)
    Destination MAC : (0:90:27:a1:36:d0)
```

```
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (81.181.78.206)
    Destination IP : (195.62.2.11)
    Total frame length: 210
================================================================
frame 19 found at 2721 :
    Source MAC : (0:90:27:a1:36:d0)
    Destination MAC : (0:2:16:9:fa:40)
    EtherType/Length : 0x0800
    Type : IP Packet
    Source IP : (195.62.2.11)
    Destination IP : (217.71.128.77)
    Total frame length: 90
================================================================
Proceed: 2811/2811 bytes of source
```

3. Ответы на контрольные вопросы к лабораторной работе

Все контрольные вопросы проработаны, затруднений не вызвали.