Vežba 6 – Hvatanje i slanje paketa upotrebom Libpcap (Raspbian) – WinPcap (Windows) programske biblioteke

1. Izgradnja paketa

Primer programskog koda datog u nastavku ilustruje deklaraciju struktura *UDP*, *IP* i *ETH* okvira.

```
   /**
      Ethernet addresses size in bytes
   */
   #define ETH_ADDR_LENGTH       6

/*-------------------------------------------------------------------------------*/
   /**
      UDP header configuration
   */
   typedef struct {
       /**
          Source port
       */
       uintXX   udp_sport;
       /**
          Destination port
       */
       uintXX   udp_dport;
       /**
          UDP length
       */
       uintXX   udp_ulen;
       /**
          UDP checksum
       */
       uintXX   udp_sum;
   } udphdr_type;
```

```
   typedef udphdr_type udp_header;
/*--------------------------------------------------------------------------*/


/*--------------------------------------------------------------------------*/

   /**
      IP header configuration
   */
   typedef struct {
       /**
          Header length
       */
       uintX    ip_hl:4;
       /**
          Version
       */
       uintX    ip_v:4;
       /**
          Type of service
       */
       uintX    ip_tos;
       /**
          Total length
       */
       uintXX   ip_len;
       /**
          Identification
       */
       uintXX   ip_id;
       /**
          Fragment offset field
       */
       uintXX   ip_off;
       /**
          Time to live
       */
       uintX    ip_ttl;
       /**
          Protocol
       */
       uintX    ip_p;
       /**
          Checksum
       */
       uintXX   ip_sum;
       /**
          Source address
       */
       uintXX   ip_src;
       /**
          Destination address
       */
       uintXX   ip_dst;
   } iphdr_type;

   typedef iphdr_type ip_header;

/*--------------------------------------------------------------------------*/
```

```
/*-------------------------------------------------------------------------*/

       /**
          Ethernet (MAC, HW) header configuration
       */
       typedef struct {
           /**
              Destination host address
           */
           uintX    eth_dhost[ETH_ADDR_LENGTH];
           /**
              Source host address
           */
           uintX    eth_shost[ETH_ADDR_LENGTH];
           /**
              IP? ARP? RARP? etc
           */
           uintXX   eth_type;
       } ethhdr_type;


       typedef ethhdr_type eth_header;

/*-------------------------------------------------------------------------*/
```

Zadatak:

Zameniti odgovarajućim brojevima ekstenzije tipova polja (X, XX) u odgovarajućim zaglavljima *UDP*, *IP* i *ETH* okvira.

```
/*-------------------------------------------------------------------------*/
       typedef unsigned char         uint8;         /*       0 .. 255          */
       typedef unsigned short        uint16;        /*       0 .. 65535        */
       typedef unsigned int          uint32;        /*       0 .. 4294967295   */
/*-------------------------------------------------------------------------*/
```

2. Hvatanje paketa

Zadatak:

Umesto funkcije ***pcap_loop*** (deklaracija i opis parametara opisani ispod) za hvatanje paketa iz primera sa prethodne vežbe koristiti funkciju ***pcap_next_ex.*** Datu funkciju upotrebiti unutar funkcije za prijem paketa *_pcap_receive* koja se proziva unutar *main* funkcije u unapred zadatom vremenskom intervalu.

- ***int pcap_loop (pcap_t * p,***
  int cnt,
  pcap_handler callback,

u_char * user

);

❖ Kao parametri funkcije navode se:

- ***p*** - je pcap struktura koja se koristi kao povratna vrednost funkcije ***pcap_open_offline*** *ili* ***pcap_open_live***.
- ***cnt*** - broj paketa koji želimo uhvatiti. Za neograničen broj paketa koriste se negativne vrednosti.
- ***callback*** - funkcija koja će biti zadužena za prijem svakog pojedinačnog paketa.
- ***user*** - stanje sesije.

Ukoliko želimo izbeći *callback* funkciju, nad kojom ne postoji potpuna kontrola izvršavanja, pojedinačni paket je moguće uhvatiti pomoću funkcije ***pcap_next_ex***.

- ***int pcap_next_ex (pcap_t * p,***
  struct pcap_pkthdr ** pkt_header,
  const u_char ** pkt_data
  );

Ova funkcija se koristi za preuzimanje narednog dostupnog paketa, zaobilazeći *callback* metod.

Primer korišćenja:

```
/*-------------------------------------------------------------------------*/

        pcap_t *handle = NULL;
        struct pcap_pkthdr *header;
        const unsigned char *pkt_data;

        int res = pcap_next_ex( handle, &header, &pkt_data);
/*-------------------------------------------------------------------------*/
```

3. Slanje paketa

Paket koji je pripremljen za slanje šalje se na mrežu pomoću funkcije ***pcap_sendpacket***.

- ***int pcap_sendpacket (pcap_t * p,***
  u_char * buf,

int size

);

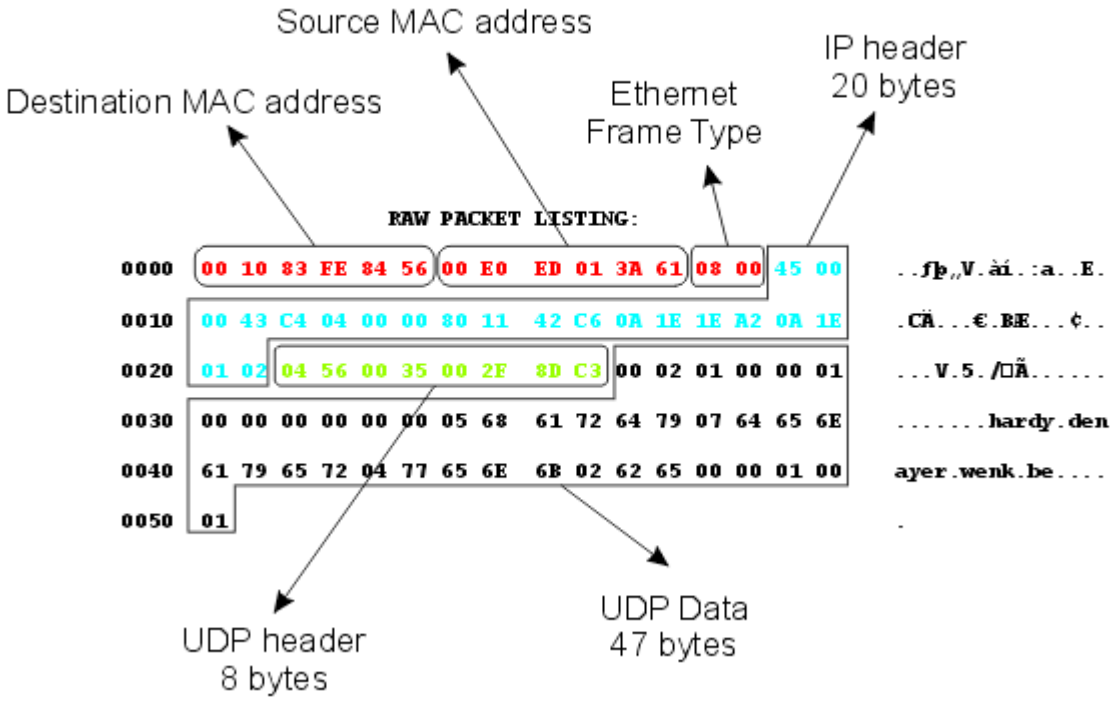
- ***p*** - interfejs koji će se koristiti za slanje paketa.
- ***buf*** - sadrži podatke o paketu za slanje (uključujući i različita zaglavlja protokola).
- ***size*** – veličina bafera odnosno veličina paketa za slanje. MAC CRC ne treba da bude uključen u okviru paketa, jer je transparentno izračunat i dodat od strane rukovaoca mrežnog adaptera. Povratna vrednost funkcije je 0 ako je paket uspešno poslat, -1 inače.

Zadatak:

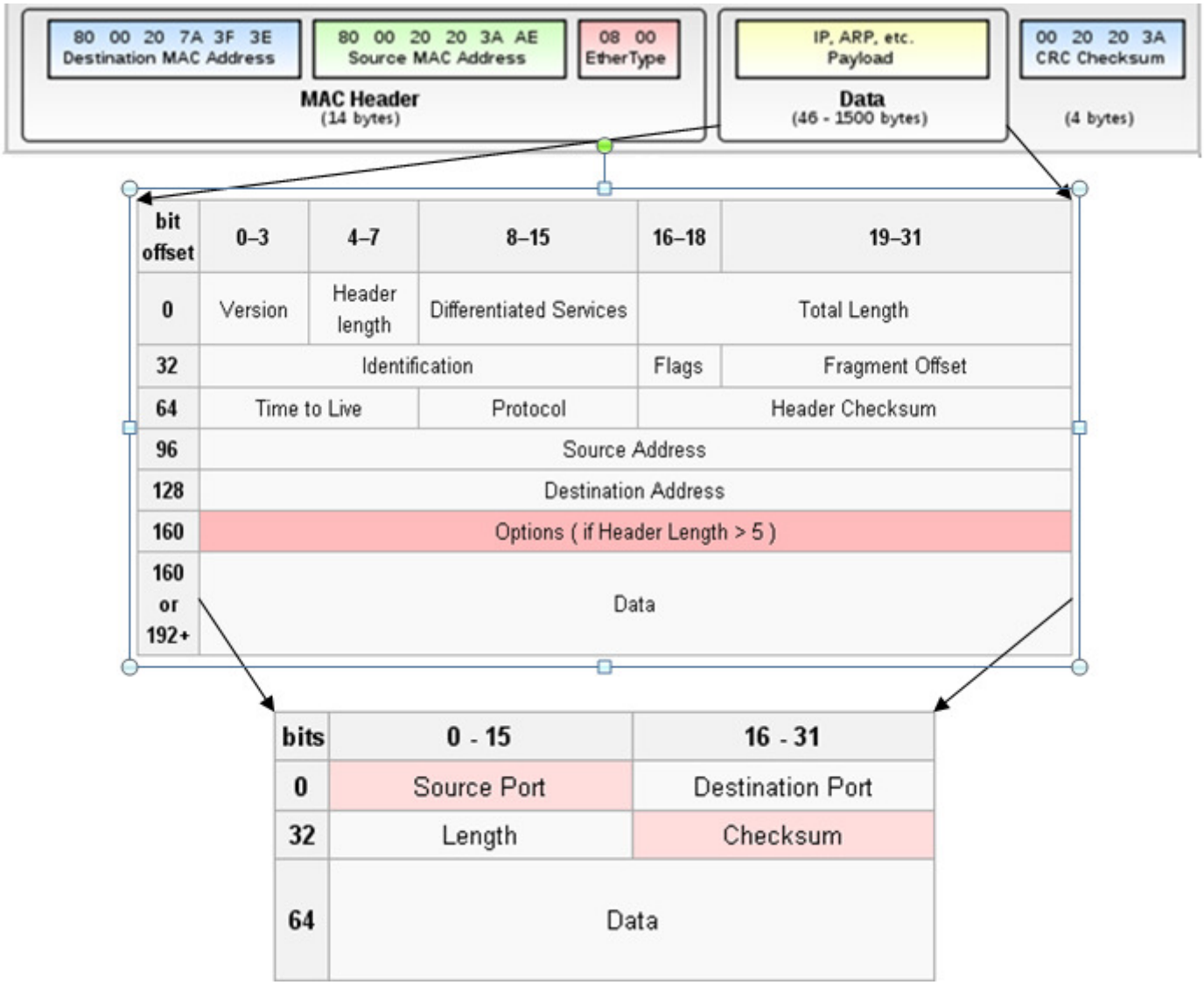
U primeru sa prethodne vežbe koristiti funkciju ***pcap_sendpacket*** za slanje paketa.

Implementirati sledeće test–scenarije u okviru projekta sa prethodne vežbe za prosleđivanje paketa na drugi računar u lokalnoj mreži upotrebom ***pcap_sendpacket*** funkcije:

- Promena ETH adrese odredišta
- Promena IP adrese odredišta
- Promena UDP porta odredišta
- Promena UDP korisničkih podataka



Slika 1. Primer asembliranog paketa



Slika 2. Enkapsulacija

4. Interpretacija primljenih paketa

Zadatak:

U okviru funkcije za prijem paketa *_pcap_receive*, implementirati interpretaciju svakog primljenog paketa i na konzolu ispisati vrednosti pojedinačno svakog polja *UDP*, *IP* i *ETH* zaglavlja.

- Posmatrati i analizirati razmenu paketa u Wireshark-u.

DODATNI ZADACI:

- Pronalaženje korisnika sa najviše saobraćaja adresiranog na odredišni računar u mreži.
- Identifikovanje protokola i aplikacija koje se trenutno koriste.
- Određivanje prosečnog broja paketa u sekundi, prosečnog broja bajtova u sekundi ili ukupnog saobraćaja adresiranog na odredišni računar u mreži.
- Prikaz svih korisnika usluga adresiranih na odredišni računar u okviru mreže.
- Određivanje prosečne dužine paketa kojeg koristi aplikacija za prenos podataka na mreži.