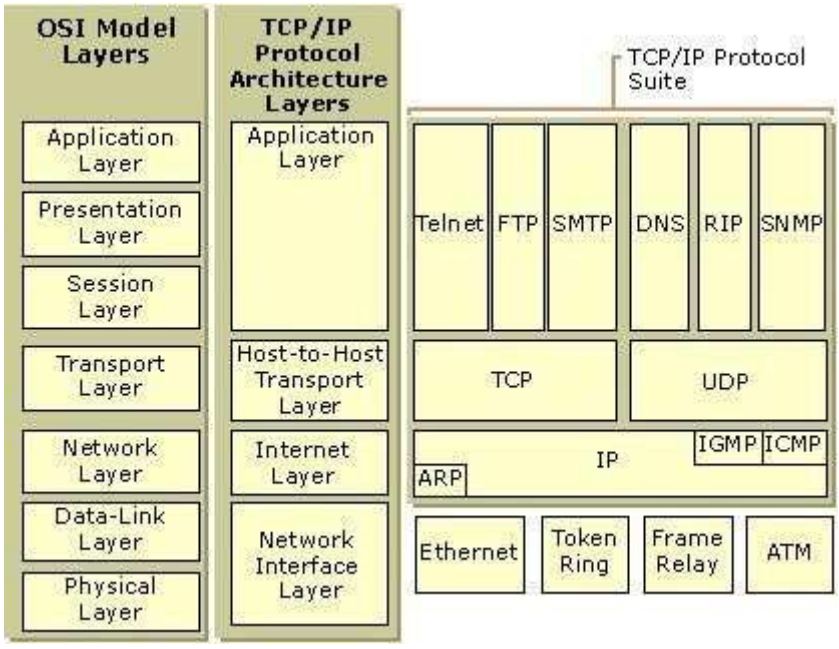
Vežba 10 - Interpretacija sadržaja paketa (1. deo)

1. TCP/IP model

Na slici 1. prikazan je međusobni odnos slojeva u OSI (Open Systems Inteconnection) modelu i u TCP/IP modelu mreže. Dok OSI model ima 7 slojeva, TCP/IP ima 4 sloja:

- Aplikativni (Application layer)
- Transportni (Transport layer)
- Internet (Internet layer)
- Sloj za pristup mreži (Network access layer , koristi se često i naziv Network interface layer)

Na desnoj strani slike 1. dati su primeri protokola koji pripadaju odgovarajućim slojevima u TCP/IP modelu.



Slika 1. TCP/IP model u poređenju sa OSI modelom mreže

Aplikacioni sloj aplikacijama obezbeđuje pristup uslugama nižih slojeva i definiše protokole koje aplikacije koriste da bi razmenile podatke. Navešćemo neke od najšire korišćenih protokola aplikacionog sloja:

- HTTP (Hyper Text Transfer Protocol) se koristi za prenos fajlova koji sačinjavaju Web stranice
- FTP (File Transfer Protocol) se koristi za prenos fajlova
- SMTP (Simple Mail Transfer Protocol) se koristi za prenos mail poruka i dodataka (eng. attachments)
- Telnet (terminal emulation protocol) se koristi za daljinsko logovanje na računare
- DNS (Domain Name System) se koristi da prevede ime hosta u IP adresu.
- RIP (Routing Information Protocol) je protokol rutiranja koji ruteri koriste za međusobnu razmenu informacija o rutiranju.

Transportni sloj je zadužen da obezbedi aplikacionom sloju uslugu uspostave veze/sesije i usluge prenosa datagrama između krajnjih aplikacija. Glavni protokoli transportnog sloja su TCP (Transmission Control Protocol) i UDP (User Datagram Protocol). Kada su podaci prosleđeni transportnom sloju, TCP i UDP protokoli razbijaju podatke u manje delove – segmente. Segmenti koji stižu na odredište ne moraju biti vezani za istu aplikaciju, potrebno je imati i broj porta koji će identifikovati odredišnu aplikaciju kojoj su podaci namenjeni, ali i port izvora koji označava aplikaciju koja podatke šalje. Ova dva broja (dva porta) se stavljaju u zaglavlje segmenta. Mehanizam numerisanja (SEQ) i potvrde prijema (ACK) koristi TCP protokol, tako da se za njega kaže da pruža uslugu pouzdane isporuke podataka, dok UDP spada protokole sa nepouzdanom isporukom. U poglavljima 2.3. i 2.4. je detaljno opisana struktura zaglavlja koje UDP i TCP dodaju segmentu.

Internet sloj je zadužen za IP adresiranje, formiranje datagrama (paketa) i rutiranje. Glavni protokoli na Internet sloju su: IP, ARP i ICMP.

- IP (Internet Protocol) je protokol zadužen za IP adresiranje, rutiranje, fragmentaciju i ponovno sastavljanje paketa.
- ICMP (Internet Control Message Protocol) je zadužen za obezbeđenje dijagnostičkih funkcija u mreži i obaveštavanje o greškama koje se dese prilikom prenosa paketa.

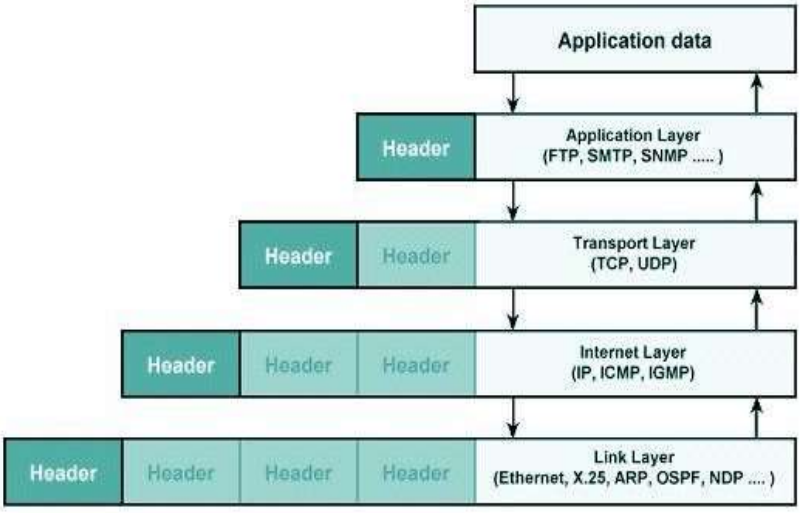
IP protokol dodaje segmentu svoje zaglavlje u kojem se nalazi IP adresa računara koji šalje poruku i IP adresa računara kome je poruka namenjena. Stvara se jedinstvena celina koja se naziva paket. Zaglavlje koje IP protokol dodaje paketu detaljno je objašnjeno u poglavlju 2.2.

Sloj za pristup mreži je odgovoran za slanje i prijem TCP/IP paketa preko mrežnog medijuma. TCP/IP je osmišljen da bude nezavisan od načina pristupa medijumu, od vrste medijuma kao i od formata okvira. Na taj način TCP/IP se koristi za povezivanje različitih tipova mreža uključujući LAN tehnologije (Ethernet i Token Ring), kao i WAN tehnologije (X.25 i Frame Relay). Kada paket stigne na najniži nivo (nivo pristupa mreži), dodaje se zaglavlje koje se sastoji od izvorne i odredišne MAC adrese, dužine, podataka i kontrolne

sume koja se dodaje na kraju. Ovaj skup podataka naziva se okvir (eng. frame). Detaljan opis okvira je dat u poglavlju 2.1.

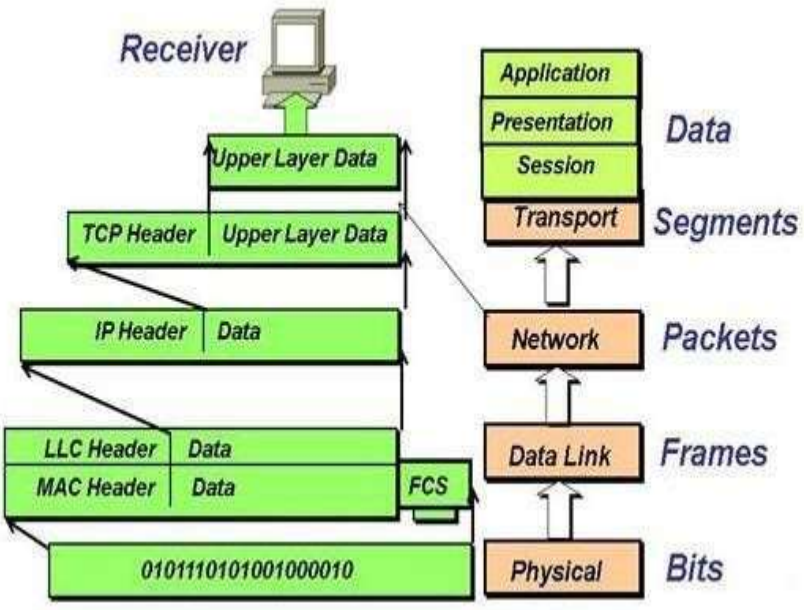
1.1 Enkapsulacija paketa

Svi aplikacioni podaci pre slanja preko mreže prolaze kroz TCP/IP protokol stek pri čemu im se na svakom sloju dodaje zaglavlje odgovarajućeg protokola. Ovaj proces se naziva **enkapsulacija podataka** i prikazan je na slici 2. Ukoliko se na sloju za pristup mreži koristi Ethernet, onda je rezultat enkapsulacije Ethernet okvir.



Slika2. Enkapsulacija podataka

Na prijemnoj strani, primljeni paketi prolaze obrnut proces (proces **dekapsulacije**) pri čemu se na svakom sloju skida zaglavlje odgovarajućeg protokola, pa se samo podaci prosleđuju protokolu višeg sloja.



Slika 3. Dekapsulacija podataka na prijemnoj strani

2. Zaglavlja protokola iz TCP/IP modela

2.1. Opis Ethernet okvira

Nivo linka (nivo za pristup mreži) dodaje svoje zaglavlje datagramu/paketu koga je primio od protokola mrežnog sloja. Tako se dobija okvir. Struktura Ethernet okvira data je da slici 4.

7 bajta	1 bajt	6 bajta	6 bajta	2 bajta	46-1500 bajta	4 bajta
Preamble	Start Frame Delimiter	Destination MAC address	Source MAC address	Length/ Type	DATA	FCS

Slika 4. Struktura Ethernet okvira

Ethernet okvir se sastoji iz:

- ***Preamble*** – Niz od 7 bajta koji imaju oblik 10101010. Oni služe prijemniku da sinhronizuje takt sa predajnikom.
- ***Start Frame Delimiter*** – Ovaj bajt ima vrednost 10101011 i prijemniku daje znak da sledi početak okvira.
- ***Destination MAC address*** – MAC adresa (48 bita) računara koji prima podatke.
- ***Source MAC address*** – MAC adresa (48 bita) računara koji šalje podatke.
- ***Length/Type*** –U Ethernet II ovo polje se tretira kao ”Type” odnosno označava koji tip podataka je u polju “data”. Ako okvir prenosi IP datagram u ovom polju biće heksadecimalna vrednost 0800. U verziji Ethernet 802.3 ovo polje označava dužinu podataka (broj bajta) koji se prenose unutar okvira.
- ***Data/Pading*** – Podaci (0-1500 bajta) dobijeni od mrežnog sloja (najčešće je to IP paket). Ako podaci koji se prenose imaju manje od 46 bajta onda se dopunjavaju sa 0 (“padding”) do minimalne dužine podataka od 46 bajta. Razlog za uvođenje minimalne dužine okvira je u mehanizmu detekcije sudara.
- ***FCS (Frame Check Sequence)*** **–** računa se pomoću 32-bitnog CRC i koristi se za detekciju grešaka u Ethernet okviru.

Kada koristimo WinPcap biblioteku unutar uhvaćenog okvira nije prisutno FCS polje, zato što se ono uklanja nakon što mrežni adapter izvrši CRC proveru okvira. Pošto mrežni adapter odbacuje pakete sa pogrešnim CRC poljem, *WinPcap* nije u mogućnosti da hvata takve pakete. Takođe, mrežni adapter uklanja i preambulu za sinhronizaciju (*Preamble*) i polje koje označava početak okvira (*Start frame delimiter*). Na slici 5 dat je Ethernet okvir koji nam se proseđuje sa mrežnog adaptera pomoću WinPcap biblioteke.

6 bajta	6 bajta	2 bajta	46-1500 bajta
Destination MAC address	Source MAC address	Length/Type	DATA

Slika 5. Struktura Ethernet okvira koji nam prosledi WinPcap funkcija za hvatanje paketa

Prvo proverimo da li protokol na sloju za pristup mreži odgovara Ethernet protokolu. Ovaj podatak nam daje funkcija `pcap_datalink()`. Deklaracija ove funkcije je:

```
int pcap_datalink(pcap_t *p);
```

Dakle, za prosleđeni deskriptor adaptera funkcija `pcap_datalink()` vraća oznaku tipa (protokola) na sloju za pristup mreži (link layer). U slučaju Ethernet protokola vraća simboličku konstantu `DLT_EN10MB`.

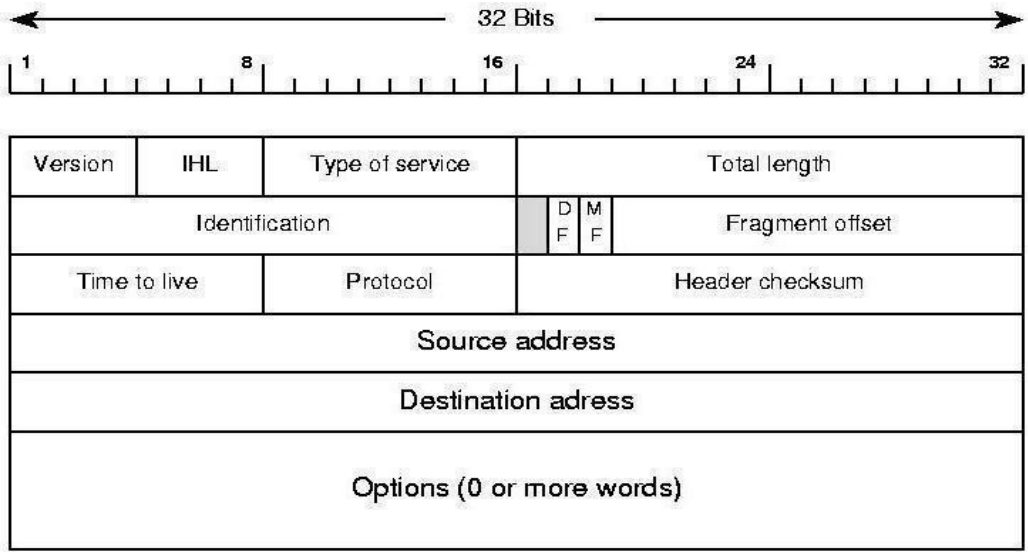
```
// Check the protocol type of link layer
if (pcap_datalink(device_handle) != DLT_EN10MB) // DLT_EN10MB oznacava Ethernet
{
	printf("\nThis program works only on Ethernet networks.\n");
	// Free the device list
	pcap_freealldevs(alldevs);
	return -1;
}
```

Nakon potvrde da naši paketi dolaze sa Ethernet mreže, postupak izdvajanja zaglavlja nam je olakšan. Naime Ethernet (MAC) zaglavlje na paketu iznosi tačno 14 bajta i ukoliko nam njegovi podaci nisu od interesa možemo ga lako preskočiti pri parsiranju primljenog paketa.

```
// Ethernet header
typedef struct ethernet_header{
    unsigned char dest_address[6];       // Destination address
    unsigned char src_address[6];        // Source address
    unsigned short type;                 // Type of the next layer
}ethernet_header;
```

2.2. Opis IP zaglavlja

IP paket se sastoji iz IP zaglavlja i podataka koji su dobijeni od protokola sa transportnog sloja (TCP ili UDP segmenta). Struktura IP zaglavlja prikazna je na slici 6.



Slika 6. Zaglavlje IP paketa

IP zaglavlje bez opcionih okteta ima minimalnu dužinu od 20 bajta i sastoji iz sledećih polja:

- **Version** – Verzija IP protokola koji je generisao datagram.
- **Internet header length** – dužina IP zaglavlja izražena u umnošcima 32-bitnih reči. Kako IP zaglavlje ima minimalno 20 bajta, najmanja vrednost ovog polja može biti 5 (20 bajta = 20 x 8 bita = 5 x 4 x 8 bita = 5 x 32-bitnih reči), dok sa sa opcionim poljima zaglavlje može dostići veličinu od 24 bajta.
- **Type of service** – Ukazuje na željeni kvalitet usluge, odnosno specificira željeni način tretiranja datagrama tokom njegovog prenosa kroz mrežu. Ovo polje je veličine 1 bajt, a postavljanjem određenih bita na vrednost 0/1 definiše se kako postupati sa paketom u smislu prioriteta slanja, kašnjenja, propusne moći mreže i pouzdanosti slanja.
- **Total length –** Dužina celokupnog datagrama (izražena u bajtima) – uključuje IP zaglavlje i podatke.
- **Identification** – Identifikacioni broj koji je isti za sve fragmente koji potiču od istog datagrama. Pomoću njega je omogućeno prijemnoj strani da ponovo sastavi originalni datagram.
- **Flags –** 3 kontrolna bita od kojih se dva koriste u postupku fragmentacije.
  - Bit 0 je rezervisan i ima vrednost 0.
  - Bit 1 ima oznaku DF (Don't Fragment) i opisuje da li je fragmentacija dopuštena (0 – dopuštena fragmentacija, 1 – nije dopuštena).
  - Bit 2 ima oznaku MF (More Fragments) i opisuje status pristiglog fragmenta (0 –

poslednji fragment, 1 – ima još fragmenata). Kada se koristi fragmenatcija datagrama, svi fragmenti osim zadnjeg imaju ovo polje setovano na 1. Kada nema fragmentacije, tada se šalje ceo datagram odjednom i ovo polje ima vrednost 0.

- **Fragment offset –** Ako je MF setovan, onda "Fragment offset" pokazuje poziciju tekućeg fragmenta unutar originalnog datagrama. Prvi fragment ima "offset" 0.
- ***Time to live*** – Određuje maksimalnu dužinu "opstanka" datagrama u mreži, u smislu broja rutera kroz koje će proći. Na izvornom računaru se postavlja na početnu vrednost (između 15-30). Svaki ruter duž putanje datagrama do odredišta umanjuje vrednost TTL polja za 1. Ako u nekom ruteru vrednost TTL polja dostigne 0, taj datagram se odbacuje (smatra se da predugo kruži kroz mrežu), a izvorišni računar se obaveštava o tome.
- ***Protocol* –** Ovo polje sadrži oznaku protokola sledećeg sloja kome pripadaju podaci u polju "data". Primeri: 1 za ICMP protocol, 6 za TCP i 17 za UDP.
- ***Header checksum*** – Kontrolna suma IP zaglavlja.
- ***Source address*** – IP adresa izvorišnog računara.
- ***Destination address*** –IP adresa odredišnog računara.
- ***Options* –** Opciono polje datagrama.

Definisaćemo strukture koje će nam služiti za čuvanje podataka o IP adresi i IP zaglavlju paketa.

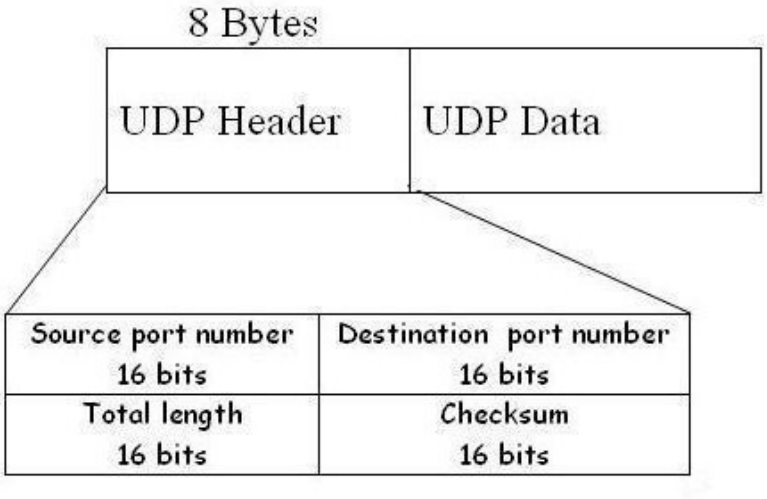
```
// IPv4 header
typedef struct  ip_header{
    unsigned char header_length :4;       // Internet header length (4 bits)
    unsigned char version :4;             // Version (4 bits)
    unsigned char tos;                    // Type of service
    unsigned short length;                // Total length
    unsigned short identification;        // Identification
    unsigned short  fragm_fo;             // Fragment flags and offset (13 + 3 bits)
    unsigned char ttl;                    // Time to live
    unsigned char next_protocol;          // Protocol of the next layer
    unsigned short checksum;              // Header checksum
    unsigned char src_addr[4];            // Source address
    unsigned char dst_addr[4];            // Destination address
    unsigned int options_padding;         // Option + Padding
}ip_header;
```

2.3. Opis UDP zaglavlja

UDP segment se sastoji od UDP zaglavlja i UDP podataka. Ovi podaci su dobijeni od nekog protokola sa aplikacionog sloja (npr. DNS protokola).

UDP zaglavlje ima fiksnu dužinu (8 bajta) i ima sledeća polja:

- ***Source port*** – port koji identifikuje aplikaciju na izvorištu.
- ***Destination port*** – port koji identifikuje aplikaciju na odredištu.
- ***Total length*** – ukupna veličina korisničkog datagrama (zaglavlje i podaci).
- ***Checksum*** – Kontrolna suma, koja se koristi za detekciju grešaka u poruci.



Slika 7. Izgled UDP segmenta

Struktura za čuvanje podataka o UDP zaglavlju je data u nastavku.

```
 //UDP header
typedef struct udp_header{
    unsigned short src_port;        // Source port
    unsigned short dest_port;       // Destination port
    unsigned short datagram_length; // Length of datagram (UDP header and data)
    unsigned short checksum;        // Header checksum
 }udp_header;
```

2.4. Opis ICMP zaglavlja

ICMP (*Internet Control Messaging Protocol*) je protokol sloja mreže koji obezbeđuje pojedine mehanizme oglašavanja. U suštini, ICMP obezbeđuje povratnu spregu u vezi problema u komunikaciji u okruženju. Primeri upotrebe ICMP-a su kada paket ne može da stigne do odredišta, ruter nema dovoljnu veličinu bafera da bi ga prosledio (korisnik da bi ga primio) ili kada ruter može da uputi poruku da postoji kraća ruta. U većini slučajeva, ICMP poruka se šalje kao odgovor na paket, bilo da ga šalje ruter koji se nalazi na putu datagrama ili host koji je odredište.

Iako je ICMP na istom sloju kao i IP u TCP/IP arhitekturi, on je u stvari korisnik IP-a. ICMP poruka se prvo napravi pa se onda predaje IP-u koji spaja poruku sa IP zaglavljem i onda prenosi rezultujući paket na već uobičajeni način. Pošto se ICMP poruke šalju kao IP paketi, njihova isporuka nije garantovana niti je njihovo korišćenje pouzdano.

ICMP zaglavlje ima sledeća polja:

- ***Type*** (8 bita): Označava tip ICMP poruke.
- ***Code*** (8 bita): Koristi se da bi se specificirali parametari poruke koja se može kodovati sa jednim ili više bita.
- ***Checksum*** za proveru (16 bita): Suma za proveru ICMP poruke. Isti algoritam se koristi i u sumi za proveru IP-a.
- ***Data*** (32 bita): Služi za različite namjene kao što su prenos IP adrese u poruci za "*icmp redirect*", ili podjelu na dva *short* dela ***identifier*** i ***sequence*** .

bit 0-7	8-15	16-31
Tip poruke	kod	Suma za proveru (Provera bitskih grešaka)
Parametri		

Slika 8. Izgled ICMP segmenta

```
//ICMP header
typedef struct icmp_header {
      unsigned char type;
      unsigned char code;
      unsigned short checksum;
      unsigned char data[4];
} icmp_header;
```

3. Interpretacija paketa

Zadatak interpretacije paketa je da se u sirovom zapisu paketa prepozna zaglavlja protokola od kojih je paket sačinjen, kako bi se omogućila analiza sadržaja presretenog paketa. Za pristup poljima određenog protokola koriste se unapred definisane strukture `ethernet_header`, `ip_header`, `udp_header`... čiji raspored bita u potpunosti odražava sadržaj zaglavlja protokola koji struktura implementira. Nakon prepoznavanja početka zaglavlja, poljima zaglavlja pristupa se pomoću polja strukure.

Interpretacija paketa biće prikazana na primeru programa koji štampa sažeti prikaz paketa koji su koristili UDP protokol na datoj mreži, izdvajajući i tumačeći UDP zaglavlje i IP zaglavlje svakog uhvaćenog paketa.

Ako se za hvatanje paketa koristi funkcija `pcap_loop()` onda se za svaki novi paket poziva callback funkcija `packet_handler()` koja je data u nastavku.

```
// Callback function invoked by libpcap for every incoming packet
void packet_handler(unsigned char *param, const struct pcap_pkthdr
*packet_header, const unsigned char *packet_data)
{
    unsigned short src_port, dst_port;

    // Retrive the position of the ethernet header
    ethernet_header* eh = (ethernet_header*)packet_data;

    // Check the type of ethernet data
    if (ntohs(eh->type) != 0x0800)) // Ipv4 = 0x0800
       return;

    // Retrieve the position of the IP header
    ip_header* ih = (ip_header*) (packet_data + sizeof(ethernet_header));

    // Check the type of ip data
    if (ih->next_protocol != 0x11) // UDP = 0x11
       return;

    // Retrieve the position of the UDP header
    int length_bytes = ih->header_length * 4; // header length is calculated
                                              // using words (1 word = 4 bytes)
    udp_header* uh = (udp_header*) ((unsigned char*)ih + length_bytes);

    // Convert from network byte order to host byte order
    src_port = ntohs(uh->src_port);
    dst_port = ntohs(uh->dst_port);

    // Print ip addresses and udp ports
    printf("%d.%d.%d.%d : %d -> %d.%d.%d.%d : %d\n", ih->src_addr[0], ih-
        >src_addr[1], ih->src_addr[2], ih->src_addr[3], src_port,
        ih->dst_addr[0], ih->dst_addr[1], ih->dst_addr[2], ih->dst_addr[3],
        dst_port);
}
```

Na početku svakog presretnog paketa koji se prosledi funkciji za analizu paketa `packet_handler` nalazi se Ethernet zaglavlje, čija dužina je fiksna i iznosi 14 bajta. Da bismo mogli pristupiti određenom polju iz Ethernet zaglavlja potrebno je kastovati tip ulaznog argumenta `packet_data` u tip `ethernet_header*`.

```
// Retrive the position of the ethernet header
ethernet_header* eh = (ethernet_header*)packet_data;
```

Ukoliko želimo analizirati samo IPv4 pakete, potrebno je proveriti vrednost polja iz Ethernet zaglavlja koje nosi informaciju o tipu protokola koji pripada sledećem sloju.

```
// Check the type of ethernet data
if (ntohs(eh->type) != 0x0800)) // Ipv4 = 0x0800
   return;
```

IP zaglavlje se nalazi odmah iza Ethernet zaglavlja. Poziciju IP zaglavlja u primljenom paketu dobijamo kada na pokazivač packet_data (koji sadrži adresu prvog bajta podataka u primljenom Ethernet okviru) dodamo dužinu Ethernet zaglavlja. Na ovaj način preskačemo iščitavanje Ethernet zaglavlja.

```
// Retrieve the position of the IP header
ip_header* ih = (ip_header*) (packet_data + sizeof(ethernet_header));
```

Iz IP zaglavlja ćemo izdvojiti IP adresu pošiljaoca i IP adresu primaoca, kako bismo ih ispisali na ekran.

Poziciju UDP zaglavlja je potrebno izračunati, jer IP zaglavlje nema fiksnu (unapred poznatu) dužinu. Dužina IP zaglavlja dostupna je preko polja `ip_header` strukture `header_length`. Kako se dužina IP zaglavlja izražava kao broj 32-bitnih reči, ovaj broj je potrebno pomožiti sa 4 kako bismo dobili dužinu zaglavlja izraženu u bajtima.

```
int ip_len = ih->header_length * 4;
```

Poziciju UDP zaglavlja dobijamo kada na adresu početne pozicije IP zaglavlja dodamo dužinu IP zaglavlja (dakle bajte (polja) koja ćemo preskočiti da bi došli do početka UDP zaglavlja).

```
udp_header* uh = (udp_header*) ((unsigned char*)ih + ip_len);
```

Pomoću pokazivača na UDP zaglavlje, iz UDP zaglavlja iščitavamo vrednost porta pošiljaoca i vrednost porta primaoca.

```
// convert from network byte order to host byte order
src_port = ntohs( uh->src_port );
dst_port = ntohs( uh->dest_port );
```

Zadatak

U prilogu vežbe data je implementacija programa koji omogućava presretanje na mrežnoj kartici paketa koji implementiraju Ethernet, IPv4, ICMP i UDP protokol. Dat je uvid u sirovi zapis paketa po pojedinačnim OSI slojevima i omogućena analiza Ethernet i IPv4 zaglavlja preko ispisa vrednosti polja koja se nalaze u njihovim zaglavljima.

Nakon razumevanja postojeće implementacije programa potrebno je omogućiti ispis sadržaja UDP zaglavlja i podataka koji pripadaju aplikativnom sloju.

1. Omogućiti ispis svih polja koja se nalaze unutar UDP i ICMP zaglavlja.
2. Odrediti poziciju na kojoj se nalazi početak sadržaja koji pripada aplikativnom sloju.
3. Saznati kolika je veličina podataka koji pripada aplikativnom sloju.
4. Omogućiti ispis aplikativnih podataka u sirovom obliku (po bajtima) koristeći heksadecimalni zapis.