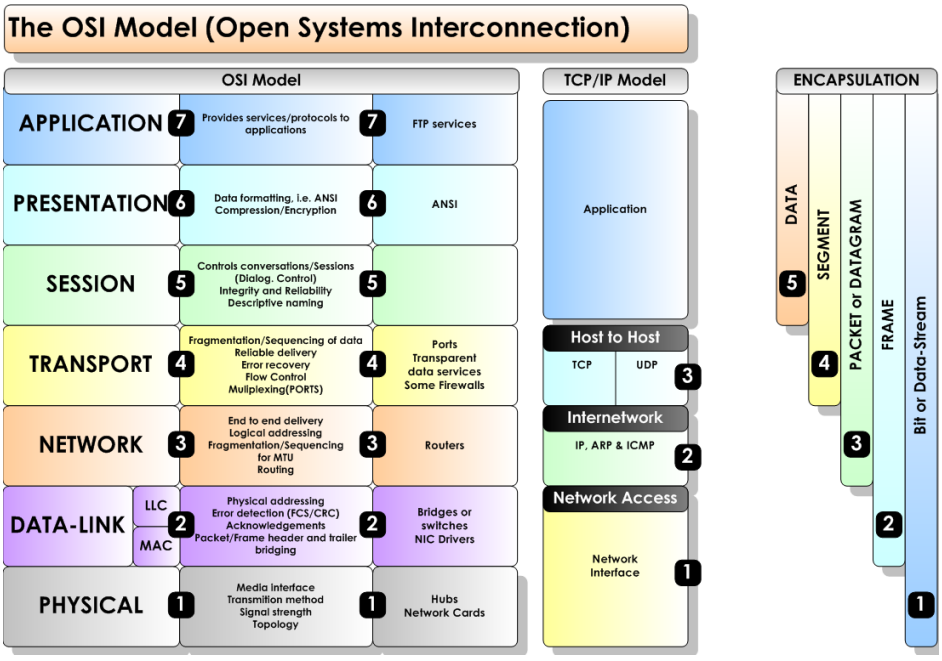
Vežba 6 – Interpretacija sadržaja mrežnih paketa upotrebom Libpcap (Raspbian) –WinPcap (Windows) programske biblioteke

1. TCP/IP model

TCP/IP je razvijen kao otvoren standard, što znači da ga svako može slobodno koristiti. Dok OSI (Open Systems Inteconnection) model ima 7 slojeva, TCP/IP model sadrži 4 sloja:

- Aplikativni (*Application layer*)
- Transportni (*Transport layer*)
- Internet (*Internet layer*)
- Mrežni (*Network access layer*)

Na slici 1. prikazan je međusobni odnos slojeva u OSI modelu i u TCP/IP modelu mreže.



Slika 1. Usporedba OSI i TCP/IP modela mreže

Aplikacija sa ***aplikativnog*** sloja, koja interpretira podatke i prikazuje informacije u razumljivom obliku, koristi protokole za slanje i primanje podataka kroz mrežu. Kako bi se podaci poslali na odredišnu adresu, nižem sloju (***transportnom***) je potrebno poslati broj port-a koji će osigurati da će podaci stići do odgovarajućeg servisa koji će ih dalje obrađivati.

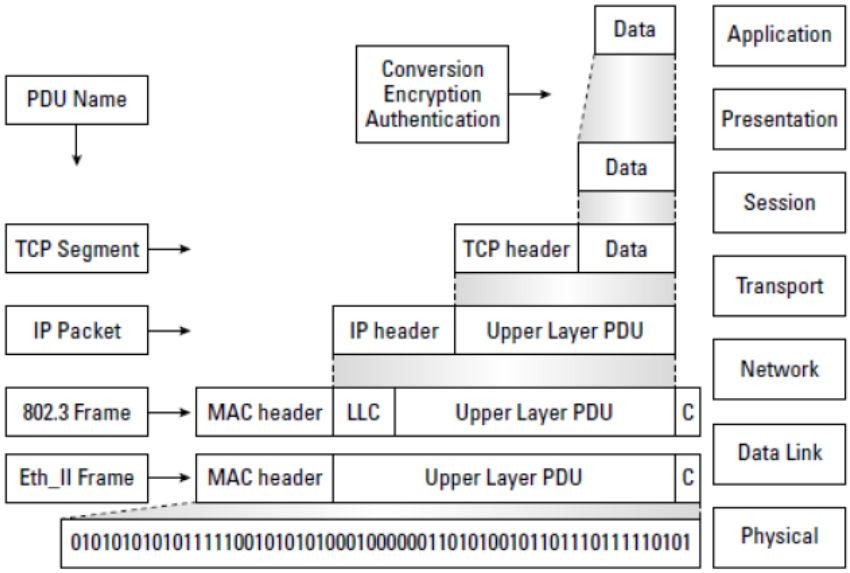
Kada su podaci prosleđeni transportnom sloju, TCP i UDP protokoli razbijaju podatke u manje delove – ***segmente***. Pošto nije sigurno na koji način će segmenti stizati do odredišta, koristi se mehanizam numerisanja segmenata (Sequence number). Segmenti koji stižu ne moraju biti vezani za istu aplikaciju, potrebno je imati i broj porta, tako da se u izvoru određuje i odredišni (aplikativni) i izvorni port (od pošaljioca). Mehanizam numerisanja (SEQ) i potvrde prijema (ACK) koristi TCP protokol, tako da se za njega kaže da je pouzdan, dok UDP spada u nepouzdane protokole.

Segmenti se dalje isporučuju sledećem nižem sloju (***internet***), koji pomoću IP protokola segmentu dodaje svoje zaglavlje u kojem se nalazi izvorna IP adresa računara i odredišna IP adresa dobijena od aplikacije. Stvara se jedinstvena celina koja se naziva ***paket*** ili ***datagram***.

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 niži nivo (***mrežni***), 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*** (frame).

2. Enkapsulacija

Nezavisnost funkcionisanja između različitih slojeva omogućena je enkapsulacijom podataka na svakom nivou i definisanjem jasnih interfejsa za komunikaciju između nivoa. Procesom enkapsulacije dodaju se zaglavlja informacija na podatke višeg sloja. *Protocol Data Unit* (PDU) predstavlja jediničnu formu podataka koju koristi protokol.



Slika 2. Enkapsulacija paketa

3. Zaglavlja protokola iz TCP/IP modela

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 3.

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 3. 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 računara koji prima podatke (48 bita).
- ***Source MAC address*** – MAC adresa računara koji šalje podatke (48 bita).
- ***Length*** – 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 CRC i koristi se za detekciju grešaka u Ethernet okviru.

Kada koristimo WinPcap/libpcap 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/libpcap* 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 4 dat je Ethernet okvir koji nam se prosleđuje sa mrežnog adaptera pomoću WinPcap/libpcap biblioteke.

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

Slika 4. Struktura Ethernet okvira koji nam prosledi WinPcap/libpcap 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*.

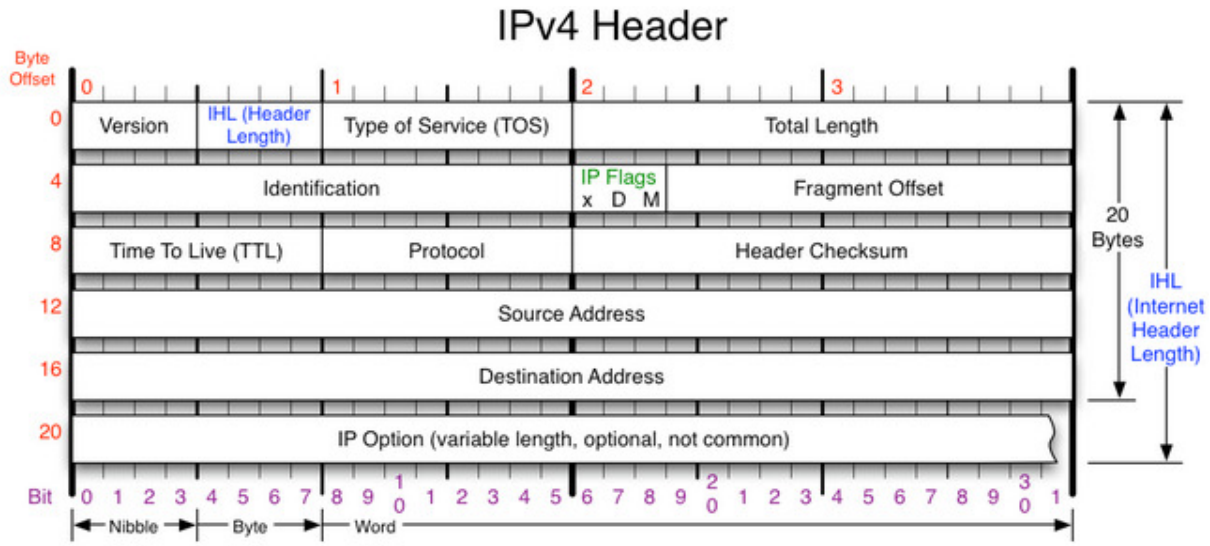
```
// Check the protocol type of link laye
if (pcap_datalink(device_handle) != DLT_EN10MB) // DLT_EN10MB oznacava Ethernet
{
	printf("\nThis program works only on Ethernet networks.\n");

	// 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;
```

Opis IP zaglavlja

Internet protokol (IP) je usmeravajući protokol za prenos datagrama. IP zaglavlje koje se dodaje kreiranom segmentu sadrži informacije potrebne da se novonastali paketi – datagrami usmere prema odredištu. Struktura IP zaglavlja prikazana je na slici 5.



Slika 5. Zaglavlje Ipv4 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.
- ***IHL (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.
  Podešavanje bita:
  - ***Biti 0-2 – prioritet***
    111 – Kontrola od strane mreže
    110 – Kontrola unutar mreže
    101 – CRITIC/ECP
    100 – Kratkotrajno opterećenje
    011 – Brzo
    010 – Direktno
    001 – Prioritet
    000 – Uobičajen postupak
  - ***Bit 3- kašnjenje***
    0 – Normalno kašnjenje
    1 – Malo kašnjenje
  - ***Bit 4 – propusna moć***
    0 – Normalna propusnost
    1 – Velika propusnost
  - ***Bit 5 – pouzdanost***
    0 – Normalna pouzdanost
    1 – Velika pouzdanost
  - ***Biti 6-7 – za buduću upotrebu***
- ***Total length –*** Dužina 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 postavljeno na vrednost 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.

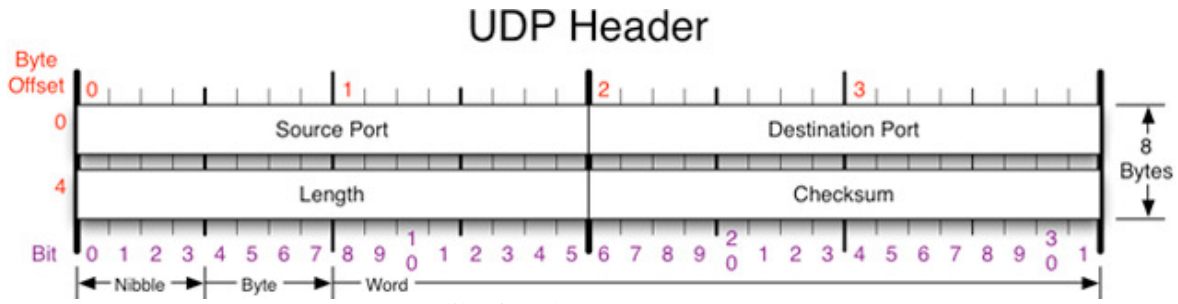
Data – podaci koji se šalju.

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_flags :3; // Flags (3 bits)
    unsigned short fragm_offset :13; // Fragment offset (13 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;
```

Opis UDP zaglavlja

UDP segment se sastoji od UDP zaglavlja i UDP podataka.



Slika 6. Izgled UDP segmenta

Zaglavlje ima fiksnu dužinu (8 bajta) i ima sledeća polja:

- ***Source port*** – prolaz koji identifikuje aplikaciju na izvorištu.
- ***Destination port*** – prolaz koji identifikuje aplikaciju na odredištu.
- ***Total length*** – ukupna veličina korisničkog datagrama (zaglavlje i podaci).
- ***Checksum*** – opciono polje. Koristi se za detekciju greške.

Data – podaci koji se šalju.

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;
```

4. 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 (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*.

```
// Retrieve 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 (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.

```
// Retrieve the position of the UDP header
int length_bytes = ih->header_length * 4; // header length is calculated
```

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).

```
// using words (1 word = 4 bytes)
udp_header* uh = (udp_header*) ((unsigned char*)ih + length_bytes);
```

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->dst_port);
```

ZADATAK

U prilogu vežbe data je implementacija programa koji omogućava presretanje na mrežnoj kartici paketa koji implementiraju Ethernet, IPv4 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 zaglavlja. (**0.5 bodova**)
2. Odrediti poziciju na kojoj se nalazi početak sadržaja koji pripada aplikativnom sloju. (**0.5 bodova**)
3. Saznati kolika je veličina podataka koji pripada aplikativnom sloju. (**1 bod**)
4. Omogućiti ispis aplikativnih podataka u sirovom obliku (po bajtima) koristeći heksadecimalni zapis. (**1 bod**)

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.