Vežba 5 – Hvatanje i filtriranje paketa upotrebom Libpcap (Raspbian) – WinPcap (Windows) programske biblioteke

1. Otvaranje adaptera

Nakon dobavljanja liste mrežnih adaptera, neophodno je otvoriti neki od dostupnih mrežnih adaptera da bi se moglo otpočeti hvatanje mrežnog saobraćaja. Funkcija *pcap_open_live()* služi za otvaranje adaptera nakon čega se paketi mogu uživo hvatati na izabranom adapteru.

```
pcap_t* pcap_open_live (const char* device_name, int packet_length, int promisc, int timeout,
char* error_buffer);
```

Funkcija:	Opis:
pcap_open_live	Funkcija za otvaranje adaptera radi hvatanja paketa uživo sa mreže.
Parametri:	**Opis:**
const char *device_name	Ime adaptera koji se otvara.
int packet_length	Specificira maksimalnu veličinu paketa u bajtima koji će biti sačuvan u bafer i prosleđen korisničkoj aplikaciji. Ako je *packet_length* dužina manja od veličine paketa, samo prvih *packet_length* bajta će biti uhvaćeno i prosleđeno kao podaci paketa. Ukoliko želimo uvek uhvatiti ceo paket bez obzira na njegovu veličinu, potrebno je izabrati maksimalnu veličinu paketa od 65536 bajta.
int promisc	Nenulta vrednost označava slobodan (eng. *Promiscous*) režim adaptera koji omogućava hvatanje bilo kog paketa koji stigne do mrežnog adaptera bez obzira kome je paket namenjen (čita pakete namenjene/adresirane i drugim hostovima).

	U normalnom režimu (kada je *promisc* postavljen na 0) adapter hvata (čita) samo one pakete koji su adresirani za njega. Obratiti pažnju da režim rada mrežnih adaptera može biti promenjen od strane druge aplikacije.
int timeout	Označava vreme u milisekundama za koje se hvataju paketi (iščitavaju iz adaptera), čak iako ništa nije pristiglo u adapter. Nakon isteka zadatog vremena svi pristigli paketi se iščitavaju u jednoj operaciji iz kernela OS. Ako se postavi na vrednost 0, onda će se funkcija izvršiti tek kada paket stigne na adapter. Ako se postavi na vrednost -1, onda će se funkcija iščitavanja izvrši odmah.
char * error_buffer	Bafer koji sadrži opis greške ukoliko se ona desi pri pozivu funkcije.
Povratna vrednost:	Opis:
pcap_t*	Ako se funkcija uspešno izvrši vraća deskriptor adaptera otvorenog za hvatanje paketa na mreži. U suprotnom, vraća se vrednost NULL a u parametru *error_buffer* se nalazi poruka o grešci.

Primer:

```
/*-------------------------------------------------------------------------*/
pcap_t* adhandle;

/* Open the adapter */
if ((adhandle = pcap_open_live(d->name,   // name of the device
                         65536,           // portion of the packet to capture.
                                          // 65536 grants that the whole packet
                                          // will be captured on all the MACs.
                            1,            // promiscuous mode (nonzero is promiscuous)
                          1000,           // read timeout
                         errbuf           // error buffer
    )) == NULL)
{
       fprintf(stderr,"\nUnable to open the adapter.
                        %s is not supported by WinPcap\n");

       /* Free the device list */
       pcap_freealldevs(alldevs);
       return -1;
}
/*-------------------------------------------------------------------------*/
```

2. Hvatanje paketa

Nakon što se adapter otvori, na njemu se mogu hvatati paketi sa mreže pomoću callback funkcije *pcap_loop()*. Ova funkcija će da se izvrši tek kad uhvati onoliko paketa koliko je zadato u njenom parametru *max_packets*.

int pcap_loop (pcap_t* device_handle, int max_packets, pcap_handler callback, unsigned char * param);

Funkcija:	Opis:
pcap_loop	Funkcija za hvatanje paketa.
Parametri:	**Opis:**
pcap_t *device_handle	*pcap_t* je deskriptor adaptera koji je otvoren za hvatanje paketa. Ova struktura se dobija kao povratna vrednost funkcija *pcap_open_live()* i *pcap_open_offline()*. Ime adaptera koji se otvara.
int max_packets	Broj paketa koje želimo uhvatiti. Nulta i negativna vrednost *max_packets* znači da će se petlja vrteti beskonačno (ili bar dok se ne desi greška), odnosno nećemo ograničiti broj paketa koje hvatamo.
pcap_handler callback	Funkcija koja je zadužena za prijem svakog pojedinačnog paketa.
unsigned char * param	Stanje sesije.
Povratna vrednost:	**Opis:**
int	Ako se funkcija uspešno izvrši vraća 0, kao znak da je dostigla iščitavanje zadatog broja paketa. U slučaju greške vraća se vrednost -1. Biće vraćena vrednost -2 u slučaju da je petlja prekinuta pomoću funkcije *pcap_breakloop()* pre nego što je prvi paket uhvaćen.

```
typedef struct pcap pcap_t;
```

Struktura *pcap* predstavlja deskriptor adaptera koji je otvoren za hvatanje paketa. Ova struktura je nedostupna korisniku, a njenim sadržajem se upravlja pomoću funkcija u *wpcap.dll*.

Kada se koristi funkcija *pcap_loop()* paketi se prosleđuju aplikaciji pomoću callback funkcije. Funkcija *pcap_loop()* ima kao svoj parametar callback funkciju *pcap_handler* koja služi za prijem svakog paketa. Ova funkcija se poziva od strane WinPcap/libpcap biblioteke za svaki novi paket pristigao sa mreže.

U nastavku je data deklaracije i objašnjenje callback funkcije koja služi za prihvatanje paketa.

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

- *param* odgovara istoimenom argumentu *pcap_loop()* funkcije, a sadrži stanje sesije za hvatanje paketa.
- *packet_header* je generičko zaglavlje koje drajver hvatanja paketa doda na svaki uhvaćeni paket. To nije zaglavlje koje protokoli stavljaju prilikom enkapsulacije paketa. Ovo zaglavlje sadrži podatke o vremenu prijema paketa, njegovoj dužini, kao i stvarnoj dužini podataka u paketu.

Struktura *pcap_pkthdr* ima 3 polja:

```
struct pcap_pkthdr {
    struct timeval ts;
    unsigned int caplen;
    unsigned int len;
};
```

Polje:	Značenje polja:
struct timeval ts	Vreme prijema paketa. Ovo je tzv. "time stamp".
unsigned int caplen	Ukupna dužina uhvaćenog paketa sa generičkim zaglavljem.
unsigned int len	Dužina samog paketa.

- *packet_data* pokazuje na početak podataka u paketu, uključujući i zaglavlja protokola. Treba naglasiti da u okviru uhvaćenog okvira (rama) nije prisutno CRC 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.

Primer:

```
/*-------------------------------------------------------------------------------*/
int main()
{
	…
	/* start the capture */
	pcap_loop(adhandle, 0, packet_handler, NULL);
	…
}

/* 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)
{
	time_t timestamp;      // Raw time (bits) when packet is received
	struct tm* local_time; // Local time when packet is received
	char time_string[16];  // Local time converted to string

	// Convert the timestamp to readable format
	timestamp = packet_header->ts.tv_sec;
	local_time = localtime(&timestamp);
	strftime( time_string, sizeof time_string, "%H:%M:%S", local_time);

	// Print timestamp and length of the packet
	printf("Packet: %s, %d byte\n", time_string, packet_header->len);

	return;
}
/*-------------------------------------------------------------------------------*/
```

3. Hvatanje paketa bez callback funkcije

Korišćenje funkcije *pcap_loop()* se oslanja na callback funkciju koja se poziva od strane drajvera za hvatanje paketa. Iz tog razloga korisnička aplikacija nema direktnu kontrolu nad ovim procesom.

Drugi pristup hvatanju paketa je pomoću funkcije *pcap_next_ex*, koja ne koristi callback funkciju. Funkcija *pcap_next_ex()* vraća paket sa direktnim pozivom, odnosno paketi se hvataju samo onda kada programer to želi.

int pcap_next_ex (pcap_t* device_handle, struct pcap_pkthdr** packet_header, const unsigned char** packet_data);

Funkcija:	Opis:
pcap_next_ex	Funkcija za hvatanje sledećeg dostupnog paketa.
Parametri:	**Opis:**
pcap_t *device_handle	*pcap_t* je struktura koja se dobija kao povratna vrednost funkcija *pcap_open_live()* i *pcap_open_offline()*. To je deskriptor adaptera koji je otvoren za hvatanje paketa na mreži.
struct pcap_pkthdr** packet_header	Funkcija popunjava ovaj parametar sa pokazivačem na zaglavlje sledećeg uhvaćenog paketa. Ovo zaglavlje na paket dodaje drajver za hvatanje paketa. (To nije zaglavlje koje protokoli stavljaju prilikom enkapsulacije paketa).
const unsigned char** packet_data	Funkcija popunjava ovaj parametar sa pokazivačem na podatke sledećeg uhvaćenog paketa. Ovi podaci uključuju i sva protokol zaglavlja.
Povratna vrednost:	**Opis:**
int	Povratne vrednosti: • 1 - ako je paket uspešno primljen (pročitan). • 0 - ako je isteklo vreme postavljeno pomoću funkcije *pcap_open_live()*. U tom slučaju *packet_header* i *packet_data* ne pokazuju na validne podatke. • -1 - ako se desi greška. • -2 - ako se dostigne EOF prilikom offline čitanja iz fajla.

Primer:

```
/*-------------------------------------------------------------------------*/
int result; // result of pcap_next_ex function
int packet_counter = 0; // counts packets in order to have numerated packets
struct pcap_pkthdr* packet_header; // header of packet generated by WinPcap
const unsigned char* packet_data; // packet content
```

```
// Retrieve the packets
while((result = pcap_next_ex(adhandle, &packet_header, &packet_data)) >= 0)
{
	// Check if timeout has elapsed
	if(result == 0)
		continue;

	// Print length of received packet
	printf("New packet arrived. Size: %d byte\n", packet_header->len);
}

if(result == -1)
{
	printf("Error reading the packets: %s\n", pcap_geterr(adhandle));
	return -1;
}
/*-------------------------------------------------------------------------*/
```

4. Filtriranje primljenih paketa

Jedna od najznačajnijih odlika WinPcap/libpcap biblioteke jeste mogućnost filtriranja mrežnog saobraćaja. Filtriranje omogućava da se primi samo deo paketa sa mreže i ono je integrisano u mehanizam hvatanja paketa koji se nalazi u kernelu. Funkcije koje se koriste za filtriranje paketa su *pcap_compile()* i *pcap_setfilter()*.

Funkcija *pcap_compile()* prihvata izraz za filtriranje napisan na višem nivou i vraća kompajliran filter koji može biti primenjen tj. interpretiran na filtru u paketskom drajveru (na nivou kernela).

int pcap_compile (pcap_t *device_handle, struct bpf_program *fcode, char * filter_exp, int optimize, unsigned int netmask);

Funkcija:	Opis:
pcap_compile	Funkcija za kompajliranje paketskog filtra. Kompajlira string *filter_exp* u filterski program (*bpf_program*).
Parametri:	**Opis:**
pcap_t *device_handle	To je deskriptor adaptera koji je otvoren za hvatanje paketa na mreži.
struct bpf_program *fcode	Funkcija vraća filterski program putem pokazivača na strukturu *bpf_program*.
char * filter_exp	String koji sadrži izraz za filtriranje.
int optimize	Određuje da li se optimizuje rezultujući kod.
unsigned int netmask	Specificira IPv4 mrežnu masku mreže na kojoj se paketi hvataju. Ako se ne zna mrežna maska postavlja se na vrednost 0.
Povratna vrednost:	**Opis:**
int	Vraća -1 ako se desi greška. U tom slučaju može se pozvati *pcap_geterr()* radi ispisa greške.

Nakon što smo preveli izraz za filtriranje i dobili kompajlirani paketski filter, pozivamo funkciju *pcap_setfilter()* koja će povezati dati filter sa sesijom za hvatanje paketa.

int pcap_setfilter (pcap_t* device_handle, struct bpf_program* fcode);

Funkcija:	Opis:
pcap_setfilter	Funkcija za povezivanje paketskog filtra sa sesijom za hvatanje paketa.
Parametri:	**Opis:**
pcap_t *device_handle	*pcap_t* je deskriptor adaptera koji je otvoren za hvatanje paketa na mreži i na koji će se primeniti paketsko filtriranje.
struct bpf_program *fcode	Pokazivač na strukturu *bpf_program* koja sadrži kompajliran filterski program. Najčešće je dobijen po izvršenju funkcije *pcap_compile()*.
Povratna vrednost:	**Opis:**
int	Ako se uspešno izvrši, vraća 0. Vraća -1 ako se desi greška. U tom slučaju može se pozvati *pcap_geterr()* radi ispisa greške.

U sledećem primeru zadaje se filter pomoću stringa "*ip and udp*" što označava da želimo da zadržimo samo pakete koji su istovremeno generisani od strane IPv4 i UDP protokola, i da samo te pakete prosledimo aplikaciji.

Primer:

```
/*-------------------------------------------------------------------------*/
unsigned int netmask;
char filter_exp[] = "ip and udp";
struct bpf_program fcode;

if (device->addresses != NULL)
	// Retrieve the mask of the first address of the interface
	netmask=((struct sockaddr_in *)(device->addresses->netmask))->sin_addr.s_addr;
else
	// If the interface is without an address we suppose to be in a C class network
	netmask=0xffffff;

// Compile the filter
if (pcap_compile(adhandle, &fcode, filter_exp, 1, netmask) < 0)
{
	printf("\n Unable to compile the packet filter. Check the syntax.\n");
	return -1;
}

// Set the filter
if (pcap_setfilter(adhandle, &fcode) < 0)
{
	printf("\n Error setting the filter.\n");
	return -1;
}
/*-------------------------------------------------------------------------*/
```

5. Sintaksa Berkli paketskih filtera (Berkeley Packet Filter - BPF)

WinPcap/libpcap filtri se na visokom nivou zapisuju kao ASCII string koji sadrži izraz za filtriranje. Funkcija *pcap_compile()* uzima ovaj izraz i prevodi ga u program koji će se izvršavati u paketskom filtru na nivou kernela.

Filterski izraz određuje koje ćemo pakete uhvatiti. Ako ne postoji paketski filter, onda će svi paketi sa mreže biti uhvaćeni. U suprotnom, samo paketi koji zadovoljavaju uslove filterskog izraza će biti prihvaćeni.

Filterski izraz se sastoji od jedne ili više primitiva. Svaka primitiva se obično sastoji od identifikatora (ime ili broj) kome prethodi jedan ili više kvalifikatora.

Postoje tri vrste kvalifikatora:

1. **Tip** – ovaj kvalifikator bliže opisuje identifikator. Može imati vrednosti: *host, net, port* i *portrange*.
    - host 192.168.55.54
    - net 192.168.0.0/24
    - port 20
    - *portrange 9000-9003*
2. **Smer** – ovaj kvalifikator opisuje smer prenosa (od/ka identifikatoru). Moguće vrednosti su : *src* i *dst*
    - src net 192.168.0.0/24
    - *dst host 192.168.55.54*
3. **Protokol** – ovaj kvalifikator oganičava hvatanje paketa generisanog određenim protokolom. Neke od mogućih vrednosti su: *ether, ip, ip6, arp, tcp and udp*.

Primeri:

- ip
- ether host 11:22:33:44:55:66
- tcp port 21
- *udp portrange 8000-8009*

Složeniji izrazi za filtriranje se dobijaju kombinovanjem primitiva korišćenjem zagrada i upotrebom reči *and, or* i *not*.

Primeri:

- port not 27015 and not arp
- arp or dns
- tcp dst port 27016 or udp dst port 27015

Sintaksa Berkley filtera data je na sledećem linku: http://biot.com/capstats/bpf.html

6. Dobijanje informacije o grešci

Ukoliko se prilikom rada sa bilo kojom funkcijom u libpcap biblioteci desi greška, informacija o njoj se dobija pomoću funkcije *pcap_geterr()*.

char* pcap_geterr(pcap_t* device_handle);

Funkcija:	Opis:
pcap_geterr	Vraća tekst greške koja odgovara zadnjoj grešci prilikom rada sa libpcap bibliotekom.
Parametri:	**Opis:**
pcap_t *device_handle	Deskriptor adaptera otvorenog za hvatanje paketa na mreži. Greška koja se desi vezana je za rad pcap bibliotečkih funkcija sa ovim adapterom.
Povratna vrednost:	**Opis:**
char *	Pokazivač na string koji sadrži tekst greške.

Primer:

```
/*-------------------------------------------------------------------------*/
char filter_exp[] = "port 80"; /* The filter expression */
if (pcap_compile(adhandle, &fp, filter_exp, 0, net) == -1)
{
	printf("Couldn't parse filter %s: %s\n", filter_exp, pcap_geterr(adhandle));
	return(2);
}
/*-------------------------------------------------------------------------*/
```

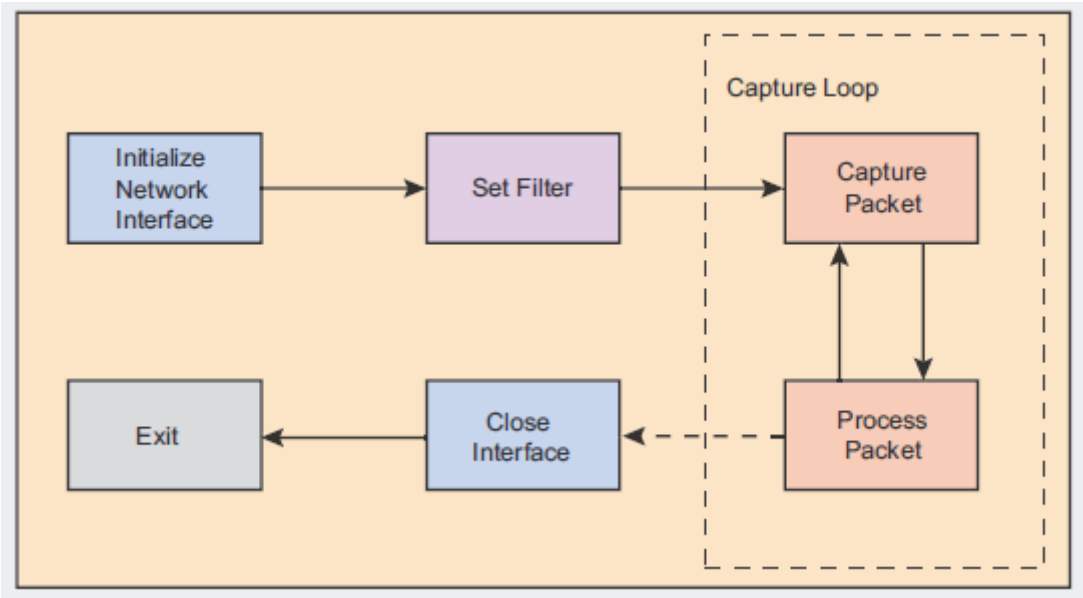
ZADATAK

U prilogu vežbe dat je primer *vezba5.c* u kome je demonstrirano hvatanje paketa. Nakon dobavljanja i ispisa liste mrežnih adaptera *pcap_findalldevs()*, korisnik bira mrežni adapter na kome želi vršiti analizu paketa *select_device()*. Mrežni adapter se stavlja u slanje prisluškivanja *pcap_open_live()* i pomoću callback funkcije *pcap_loop()* se zadaje metoda *packet_handler()* koja će vršiti obradu presretenih paketa. Na ekranu se prikazuje pseudo zaglavlje *packet_header* svakog paketa generisano od libpcap/WinPcap biblioteke sa lokalnim vremenom kada je paket uhvaćen *packet_header→ts* i dužinom paketa *packet_header→len*.

Postojeću implementaciju potrebno je unaprediti sledećim mogućnostima:

1. Omogućiti da se obrađuju samo dolazni paketi koji su adresirani na logičku adresu mrežnog adaptera na kome je pokrenuta aplikacija i koji za transport koriste TCP protokol. Predfiltriranje paketa vršiti na kernelu korišćenjem *pcap_compile()* i *pcap_setfilter()*. (**1.5 bod**)

2. Ispisati sadržaj svakog primljenog paketa *packet_data* koristeći heksadecimalan zapis. U jednom redu prikazati 16 bajta sadržaja istog po ugledu na Wireshark. (**1.5 bod**)



Slika 1. Tok programa tipične pcap aplikacije