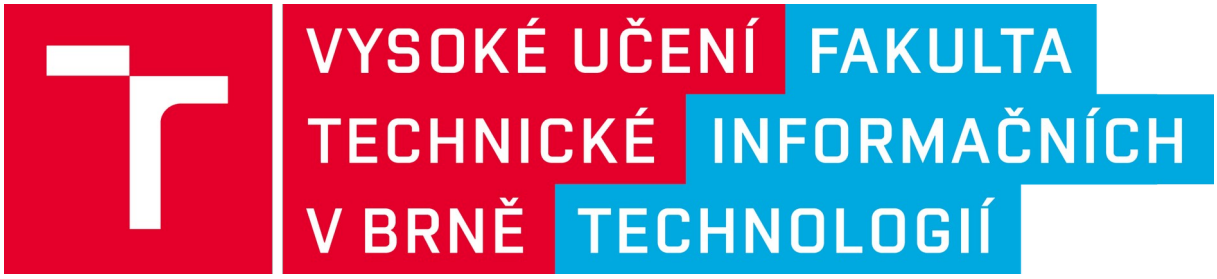
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií



POČÍTAČOVÉ KOMUNIKACE A SÍTĚ

2021/2022

Projekt 2 – Varianta ZETA

Sniffer paketů

Tomáš Bártů (xbartu11)

Třešť, 24. dubna 2022

1) Obsah

2) Úvod

Úkolem projektu bylo vytvořit v síťový analyzátor, který dle zadaných parametrů bude moci filtrovat {rámce, pakety, datagramy, segmenty} a vypsat zajímavé údaje, které obsahují. Například zdrojovou nebo cílovou Media Access Control adresu, zkráceně MAC adresu. Či vypsat zdrojový a cílový port protokolu TCP. Dalším úkolem bylo vypsat payload v bajtové podobě tak i ve znakové.

3) Implementace

V programu se nacházejí, mimo funkce main(), následující funkce: parse_args(), interface(), make_filter(), handler() a funkce začínající p_*název*, kde název je některé jmébo z množiny {time, mac, length, ip, ip6, tcp, udp, arp, payload}. Tyto funkce vypisují na standardní výstup data.

```
timestamp: 2022-04-24T21:08:13.654516+0200
src MAC: 00:00:00:00:00:00
dst MAC: 00:00:00:00:00:00
frame length: 98 bytes
src IP: 127.0.0.1
dest IP: 127.0.0.1

0x0010  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00     ........ ......E.
0x0020  00 54 23 b1 40 00 40 01  18 f6 7f 00 00 01 7f 00     .T#.@.@. ........
0x0030  00 01 08 00 1f 2c 00 01  00 01 1d a0 65 62 00 00     .....,.. ....eb..
0x0040  00 00 8d fc 09 00 00 00  00 00 10 11 12 13 14 15     ........ ........
0x0050  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25     ........ .. !"#$%
0x0060  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35     &'()*+,- ./012345
0x0070  36 37                                               67
```

Obr 1.: Příklad výstupu dat

Program se nachazí v souboru packetsniffer.c, kde pomocné proměnné, definice funkcí a podobně se nachazí v hlavičkovém souboru packetsniffer.h.

V tomto hlavičkovém souboru byly nedefinovány některá čísla protokolů, velikosti hlaviček a podobně.

```
#define IP_PROTOCOL_ICMP 0x01 /* ICMP */
#define IP_PROTOCOL_TCP  0x06 /* TCP */
#define IP_PROTOCOL_UDP  0x11 /* UDP */

#define ETHER_HEADER_SIZE 14  /* velikost ethernetové hlavičky */
#define IPV6_HEADER_SIZE  40  /* velikost IPv6 hlavičky */
```

Obr 2.: Některé nadefinované hodnoty

Při implementaci analyzátoru byla použita knihovna libpcap, která umožňuje zachytávat dat prochazející určitým zařízením. Pro zbracovaní přijatých dat bylo využito knihoven z rodiny netinet, případně arpa

a Funkce parse_args

Zájímavou částí této sekce je makro BETWEEN. Pomocí něho probíha kontrola zda při zadaném parametru -p *číslo*, číslo se nachazí ve správném rozsahu portů a ten je <0, 65535> .

b Funkce p_time

Důležitou částí byl výpis timestampu ze zachycených dat, kvůli zpříjemění práce při případném debbugovaní programu. Pro správný výpis času uloženého ve struktuře pcap_pkthdr, bylo nejdříve položku ts.tv_sec přetypovat do time_t, kterému už rozumí funkce localtime, který takto přetypovaný čas umí převést do strukury, která umožní formátovaně vypsat například den v týdny, měsíc, časové pásmo a to pomocí funkce strftime ke které, byly následně přidány milivteřiny uložené v původní struktuře.

c Funkce main

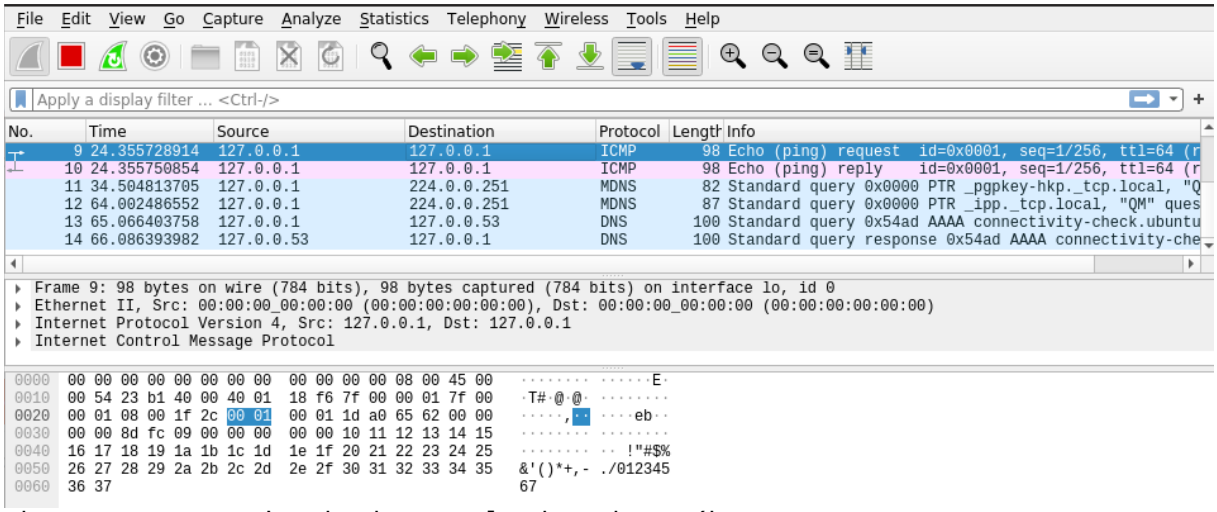
Nejpodstatnější část každého programu napsaného v jazyce C je funkce main. Zde tomu není výjimkou. Dochází zde postupně k ověření argmunetů programu (unformace o nich se uloží do strukury Options). Následně k vytvoření řetězcové reprezentace filtru, pokud již dříve nebyl program ukončen kvůli zadanému argumentu -i bez parametru (výpis všech dostupných zařízeních) nebo pokud nedošlo k chybě při zadávaní argumentů programu.

Následně se v vyhledá požadované zařízení, které je posléze otevřeno. Jelikož chceme pracovat s ethernetovými rámci. Musíme zkontrolovat, zda nám to zařízení umožňuje a to provedeme pomocí funkce pcap_dataling(*zařízení*), kde výstupem musí být DLT_EN10MB. Podstatnou částí aplikace je možnost filtrování dle typu protokolu. Aby bylo možné toto provádět, tak musíme na zařízení aplikovat filter a to provedeme pomocí funkce pcap_setfilter.

Teď je možné přejít k zachytávání paketů a to díky funkci pcap_loop, jejímž jedním parametrem je číslo, jež udává kolik paketů se zachytí. V programu je toto číslo defaultně 1, ale při spouštění je možné toto změnit pomocí přepínače -n *číslo*. Následně se pomocí tzv. Callbacku zpracují jednotlivé pakety a to se provádí ve funkci handler. Tato funkce dokáže zpracovávat přijatá data. Zjišťovat z nich hlavičky jednotlivých vrstev a z těchto hlaviček pak následně vypsta zajímavá data na standardní výstup

4) Testování

Testování probíhalo především na virtuálním stroji, ale i na lokálním pro testování IP verze 6, pomocí různých příkazů jako například arp či curl (curl -6 pro IPv6 testování).
Pro testování výstupnách dat jsem používal program Wireshark, ve kterém jsem si prohlížel zachycená data a porovnával s výstupem programu. Níže je možne vidět zachycený ICMP rámec a výstupní payload programu.



Obr .: Program wireshark a payload zachyceného ICMP

```
0x0010   00 00 00 00 00 00 00 00   00 00 00 00 08 00 45 00      ........ ......E.
0x0020   00 54 23 b1 40 00 40 01   18 f6 7f 00 00 01 7f 00      .T#.@.@. ........
0x0030   00 01 08 00 1f 2c 00 01   00 01 1d a0 65 62 00 00      .....,.. ....eb..
0x0040   00 00 8d fc 09 00 00 00   00 00 10 11 12 13 14 15      ........ ........
0x0050   16 17 18 19 1a 1b 1c 1d   1e 1f 20 21 22 23 24 25      ........ .. !"#$%
0x0060   26 27 28 29 2a 2b 2c 2d   2e 2f 30 31 32 33 34 35      &'()*+,- ./012345
0x0070   36 37                                                 67
```

Obr .: Payload, který byl vytištěn na výstup

Je zda vidět menší nepřesnot v paddingu posledním řádku znakového payloadu.

5) Zdroje

1. How to Perform Packet Sniffing Using Libpcap with C Example Code [online]. Dostupné z: https://www.thegeekstuff.com/2012/10/packet-sniffing-using-libpcap/
2. TCPDUMP&LIBPCAP public repository. TCPDUMP/&LIBPCAP public repository [online]. Dostupné z: https://www.tcpdump.org/
3. getopt_long() option with optional argument - Stack Overflow. Stack Overflow - Where Developers Learn, Share, & Build Careers [online]. Dostupné z: https://stackoverflow.com/a/40595790
4. inet_ntop(3) - Linux manual page. Michael Kerrisk – man7.org [online] Dostupné z: https://man7.org/linux/man-pages/man3/inet_ntop.3.html
5. inet_ntop printing incorrect IPv6 address - Stack Overflow. Stack Overflow - Where Developers Learn, Share, & Build Careers [online]. Dostupné z: https://stackoverflow.com/a/38849126
6. ntohs(3) - Linux man page. Linux Documentation [online] Dostupné z: https://linux.die.net/man/3/ntohs
7. read from a PCap file and print out IP addresses and port numbers in c, but my result seem wrong - Stack Overflow. Stack Overflow - Where Developers Learn, Share, & Build Careers [online]. Dostupné z: https://stackoverflow.com/questions/12999538/read-from-a-pcap-file-and-print-out-ip-addresses-and-port-numbers-in-c-but-my-r
8. getopt_long(3) - Linux man page. Linux Documentation [online] Dostupné z: https://linux.die.net/man/3/getopt_long
9. How to print time in format: 2009-08-10 18:17:54.811 Stack Overflow. Stack Overflow - Where Developers Learn, Share, & Build Careers [online]. Dostupné z: https://stackoverflow.com/questions/3673226/how-to-print-time-in-format-2009-08-10-181754-811