Vysoké učení technické v Brně
Fakulta informačních technologií

Počítačové komunikace a sítě
Packet Sniffer

Řešeno pomocí jazyka **C++**

2021 Martin Pech (xpechm00)

Obsah

1 Úvod

Náplní projektu byla tvorba síťového analyzátoru, který na předem určeném síťovém rozhraní filtruje příchozí a odchoží packety. Samotný analyzátor by měl být schopný filtrovat ICMP, TCP a UDP packety. Dále by měl umět vypsat ARP rámce.

2 Kompatibilita a spouštění

Program je kompatibilní s Linuxovými operačními systémy. Pro překlad využívá překladače `G++`. Pro usnadnění překladu lze využít přiloženého souboru `makefile`, který provede automatické přeložení. Pro jeho použití je potřeba mít `GNU Make`.

2.1 Překlad

Program je možné spustit pomocí `makefile`:

```
$ make
```

Nebo vykonáním následujícího překladu:

```
$ g++ ipk-sniffer.c -o ipk-sniffer -lpcap
```

2.2 Spouštění

Po přeložení dojde k vytvoření spustitelného souboru `ipk-sniffer`.
Pro následné smazání zdrojových soubrů lze použít příkaz `make` s cílem `clean` a pro smazání všech souborů lze použít `make` s cílem `cleanAll`. Nově vzniklý soubor `ipk-sniffer` lze spouštět následujícím způsobem:
`$ ./ipk-sniffer [-i` *rozhrani* `| --interface` *rozhrani*`] {-p` *port*`} {[-t | --tcp] [-u | --udp] [--arp] [--icmp]} {-n` *number*`}`.

- `--interface`
  Očekává jméno síťového rozhraní, na kterém bude odchytávat packety. Chybí-li tento parametr, vypíše se seznam aktivních síťových rozhraní.

- `-port` *(nepovinný)*
  Očekává číslo portu, který bude filtrovat. Port se může vyskytovat jako odesílatel i příjemce.

- `--tcp`, `--udp`, `--arp`, `--icmp` *(nepovinné)*
  Umožňují nastavit filtrování pouze na packety dané služby. Jsou-li vybrány všechny, chování je analogické vstupu: `./ipk-sniffer -i` *rozhrani*.

- `-n` *(nepovinný)*
  Umožňuje filtrovat určitý počet packetů. Defaultní nastavení je `n = 1`.

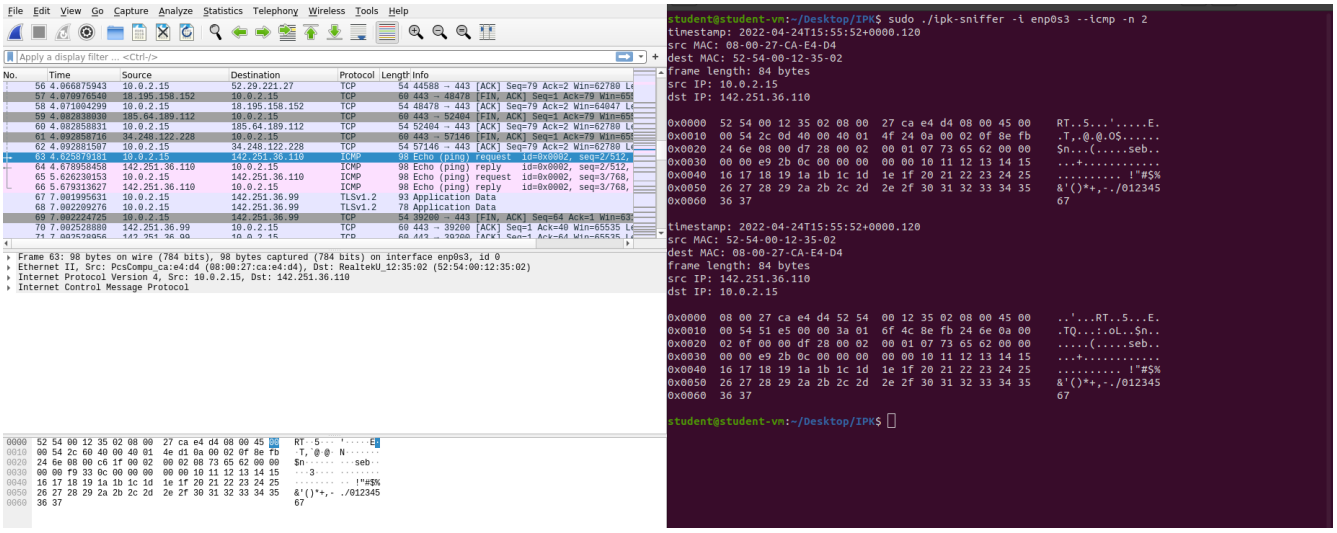
3 Výpis výstupu a testování

3.1 Výpis výstupu

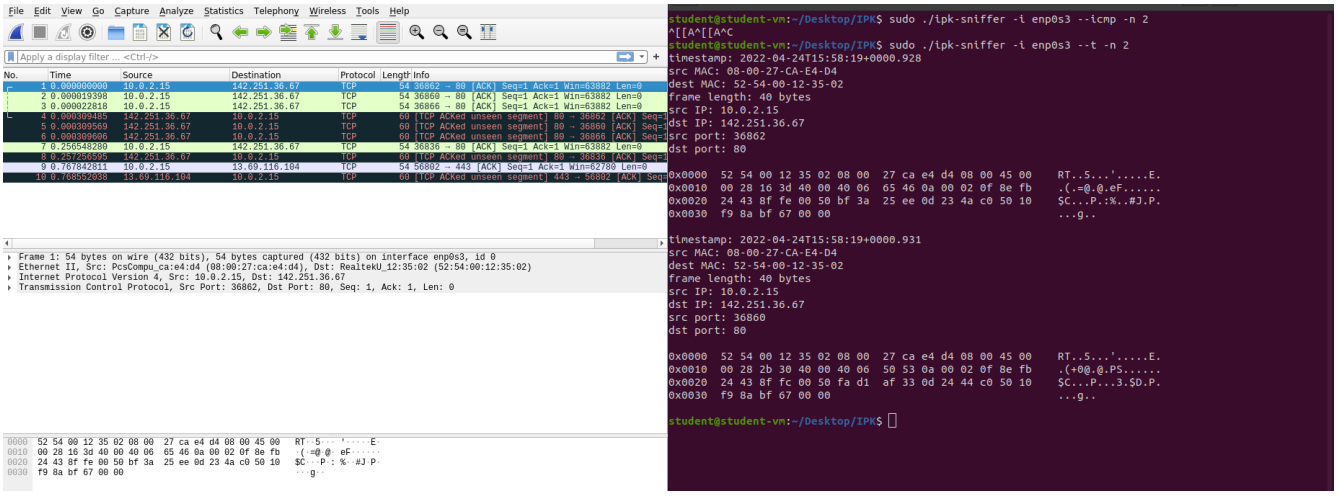
O výpis výstupu se stará řada funkcí. Za zmínku stojí funkce `printData`, která kontroluje délku řádku dle požadavků na výpis. Dále volá funkci `hexIt`, která převede obsah packetu (data) na příslušné znaky. Není-li to možné, nahradí znak tečkou. Pro výpis časové značky (timestamp) se stará funkce `timeIs`, která s pomocí knihovny `time.h` a `std::chrono` zformátuje čas do podle formátu `RFC3339`.

3.2 Testování

Kontrola správného výpisu byla ověřována programem `Wireshark`. Na následujícím příkladu je možné vidět dva ICMP packety. Tyto dva packety byly odchyceny i v programu `Wireshark`. Obsah těchto packetů byl shodný.



Na následujícím příkladu je obdobný příklad, avšak narozdíl od předchozího byl zachycen TCP packet.



Podobně se dají filtrovat i další služby (viz sekce 2.2).

4 Implementační detaily

Pro implementaci byla vybrána knihovna `libpcap` (pcap.h). Tato knihovna nese název open source aplikačního rozhraní pro odchytávání síťové komunikace. V případě, že dojde k chybě při práci s funkcemi této knihovny, je program ukončen s návratovou hodnotou `1`.

4.1 Funkce main

Ve funkci `main` dochází ke zpracování argumentů příkazové řádky. Je-li argument vyhodnocen jako neplatný, nastává chyba `99`. V opačném případě je nastaven příznak platnosti (například `tcpFLAG == true`) a případně uložena vsutpní hodnota. Hodnoty jsou dále validovány a jsou nastaveny příslušné příznaky. Dále dochází k volání funkcí z knihovny `pcap.h`.

4.2 Funkce pcap_lookupnet

Zjišťuje číslo a masku sítě. V případě chyb vrací hodnotu `1`.

4.3 Funkce pcap_open_live

Zajišťuje přístup ke sledovaným packetům na síti. Pro provedení je potřeba spouštět program s administrátorskými právy.

4.4 Funkce pcap_loop

Vytváří smyčku, ve které kontroluje packety na síti. Má atribut `cnt`, který ukončuje smyčku, najde-li se packet s příslušným pořadovým číslem. V mém projektu se této funkce nevyužívá, protože jsou implementovány příznaky a počítadla, která nahrazují tuto funkci.

4.5 Funkce pcap_close

Ukončuje veškeré spojení, uzavírá soubory otevřené v souladu s odchytáváním packetů a dealokuje veškeré zdroje.

4.6 Funkce process_packet

Zodpovídá za veškerou řídící logiku analyzování a filtrování. Vyskytují se v ní veškeré potřebné zdroje pro vypsání výstupních údajů.

5 Použtá literatura

BinaryTides: *How to code a Packet Sniffer in C with Libpcap on Linux* [online]. 31.7.2020, [cit. 2022-04-23]. Dostupné z: https://www.binarytides.com/packet-sniffer-code-c-libpcap-linux-sockets/

TCPDUMP: *Man page of PCAP* [online]. 9.9.2020 [cit. 2022-04-23]. Dostupné z: https://www.tcpdump.org/

Wikipedia: *pcap* [online]. 22.4.2022 [cit. 2022-04-23]. Dostupné z: https://en.wikipedia.org/wiki/Pcap

Stackoverflow: *I'm trying to build an RFC3339 timestamp in C. How do I get the timezone offset?* [online]. 22.4.2022 [cit. 2022-04-23]. Dostupné z: https://stackoverflow.com/questions/48771851/im-trying-to-build-an-rfc3339-timestamp-in-c-how-do-i-get-the-timezone-offset/48772690#48772690