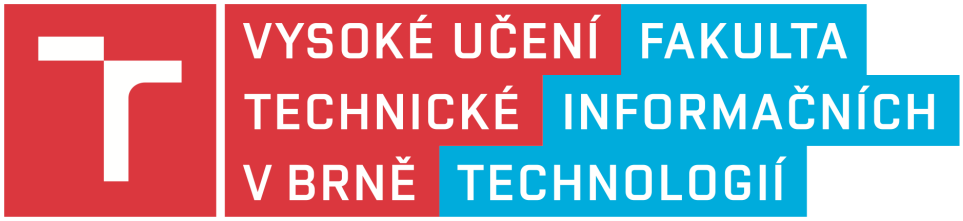


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

IPK - Počítačové komunikáce a siete

Projekt 2 -Varianta ZETA: Sniffer paketov

Lucia Makaiová (xmakai00)

20. dubna 2022

Obsah

1 Varianta ZETA: Sniffer paketov

1.1 Popis projektu

Ipk-sniffer je aplikácia jazyka C++, ktorá slúži ako sieťový analyzátor. Aplikácia zachytáva pakety na vybranom, užívateľom zadanom rozhraní a ďalej ich spracováva. Taktiež je možné pakety pomocou aplikácie filtrovať.

1.2 Formát volania programu

./ipk-sniffer [-i rozhranie] -p port [–tcp—-t] [–udp—-u] [–arp] [–icmp] -n num

Činnosť programu je možné dodatočne špecifikovať použitím nasledujúcich argumentov pri volaní programu(na poradí argumentov nezáleží):

- -i / –interface *rozhranie*, kde rozhranie je názov rozhrania na analýzu paketov
- -i bez uvedeného rozhrania vypíše zoznam rozhraní, ktoré sú k dispozícii
- -p *port*, kde port určuje číslo portu, na ktorom program bude zachytávať pakety (implicitne na všetkých) Filtrovanie paketov:
- -n *num*, kde num je číslo určujúce počet paketov ktoré má program spracovať
- -u / –udp -program bude spracovávať pakety používajúce protokol UDP
- -t / –tcp -program bude spracovávať pakety používajúce protokol TCP
- –icmp -program bude spracovávať pakety používajúce protokol ICMP
- –arp -program bude spracovávať pakety používajúce protokol ARP

Poznámka: pri uvedení všetkých argumentov pre filtráciu paketov sa program správa rovnako, ako keď užívateľ neuvedie žiadny z nich.

2 Použité knižnice a funkcie

- arpa/inet.h -funkcie htons a ntohs na prevod medzi hostiteľským a sieťovým poradím bajtu
- pcap.h -odposlúchavanie packetov a práca s nimi
- getopt.h -parsovanie argumentov príkazového riadka
  Knižnice pre prácu s hlavičkami pomocou typecastovania paketu:
- netinet/ether.h
- netinet/ip6.h
- netinet/tcp.h

- netinet/udp.h
- netinet/ip_icmp.h

3 Časti programu

3.1 Spracovanie argumentov

Argumenty príkazového riadka sú spracované pomocou knižnice getopt. Keďže takmer všetky dlhé verzie argumentov sú bez parametrov, použitá bola krátka verzia getopt príkazu, modifikovaná pre účely dlhých príkazov. Argument "–"je vnímaný ako argument v krátkom formáte podobne ako "-i"a ako jeho parameter je braný pôvodný dlhý príkaz. (Napríklad "–udp"je brané ako argument "–"s parametrom "udp")

3.2 Inicializácia odposluchávania (sniffing)

Inicializácia odposluchávania paketov je vyriešená podľa zdroja[1]. Pomocou funkcií knižnice pcap.h sa vytvorí nová relácia odposúchania v promiskuitnom móde a následne sa overí že sa ju na zadanom rozhraní podarilo vytvoriť a že toto rozhranie podporuje ethernetové pakety.

3.3 Vytvorenie filtra paketov

Filtrovanie paketov prebieha v troch fázach. Ako prvé je nutné **vytvoriť** podľa zadaných argumentov reťazec s výrazom popisujúcim spôsob filtrovania paketov. V ďalšom kroku je nutné tento reťazec **skompilovať** a až po úspešnej kompilácii je možné ho **aplikovať**.Táto časť je taktiež inšpirovaná rovnakým zdrojom ako samotné odposlúchavanie.

3.4 Hlavná smyčka odposluchávania paketov.

Zachytenie samotného paketu prebieha pomocou funkcie $pcap_next$, $ktorávraciaukazateľtypuconstu\_char.Tátofunkciaanásle$
n"$spracujeodpovedajúcipočetpaketov(implicitne1).Informáciezhľaviečiekprotokolovpaketusazískavajupoužitímknižnícpr$

- ***Cieľ:*** *Výpis aktívnych rozhraní:*
  Volanie:
  ./ipk-sniffer -i
  Výstup:
  List of all interfaces:
  ens33
  lo
  any
  bluetooth-monitor
  nflog
  nfqueue

- ***Cieľ:*** *Výpis prvých 10 zachytených paketov používajúcich protokol ICMP:*

  Volanie:

  ./ipk-sniffer -n 3 –icmp

  Výstup:

  timestamp: 2022-04-20T18:43:46.866
  src MAC: 00:0c:29:64:af:ce
  dst MAC: 00:50:56:e6:ae:40
  frame length: 98 bytes
  src IP: 192.168.6.128
  dst IP: 142.251.37.110

  0x0000: 00 50 56 e6 ae 40 00 0c 29 64 af ce 08 00 45 00 .PV..@..)d....E.
  0x0010: 00 54 1b d4 40 00 40 01 a3 43 c0 a8 06 80 8e fb .T..@.@..C.....
  0x0020: 25 6e 08 00 0e 50 00 08 00 01 42 38 60 62 00 00 %n...P....B8‘b..
  0x0030: 00 00 7b 39 0d 00 00 00 00 00 10 11 12 13 14 15 ..{9.....
  0x0040: 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25 ..... !”#$%
  0x0050: 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 &’()+,-./012345*
  0x0060: 36 37 67

  timestamp: 2022-04-20T18:43:46.885
  src MAC: 00:50:56:e6:ae:40
  dst MAC: 00:0c:29:64:af:ce
  frame length: 98 bytes
  src IP: 142.251.37.110
  dst IP: 192.168.6.128

  0x0000: 00 0c 29 64 af ce 00 50 56 e6 ae 40 08 00 45 00 ..)d...PV..@..E.
  0x0010: 00 54 30 91 00 00 80 01 8e 86 8e fb 25 6e c0 a8 .T0.....%n..
  0x0020: 06 80 00 00 16 50 00 08 00 01 42 38 60 62 00 00 .....P....B8‘b..
  0x0030: 00 00 7b 39 0d 00 00 00 00 00 10 11 12 13 14 15 ..{9.....
  0x0040: 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25 ..... !”#$%
  0x0050: 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 &’()+,-./012345*
  0x0060: 36 37 67

  timestamp: 2022-04-20T18:43:47.867
  src MAC: 00:0c:29:64:af:ce
  dst MAC: 00:50:56:e6:ae:40
  frame length: 98 bytes

src IP: 192.168.6.128

dst IP: 142.251.37.110

```
0x0000:  00 50 56 e6 ae 40 00 0c 29 64 af ce 08 00 45 00  .PV..@..)d....E.
0x0010:  00 54 1c 91 40 00 40 01 a2 86 c0 a8 06 80 8e fb  .T..@.@.........
0x0020:  25 6e 08 00 05 4b 00 08 00 02 43 38 60 62 00 00  %n...K....C8`b..
0x0030:  00 00 83 3d 0d 00 00 00 00 00 10 11 12 13 14 15  ...=............
0x0040:  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  .......... !"#$%
0x0050:  26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35  &'()*+,-./012345
0x0060:  36 37                                            67
```

- ***Cieľ:*** *Výpis udp paketu na porte 443 (ako zdrojový alebo cieľový port):*

  Volanie:

  ./ipk-sniffer -i ens33 -p 443 -u

  Výstup:*:*

  timestamp:2022-04-20T18:47:08.967

  src MAC: 00:50:56:c0:00:08

  dst MAC: ff:ff:ff:ff:ff:ff

  frame length: 86 bytes

  src IP: 192.168.6.1

  dst IP: 192.168.6.255

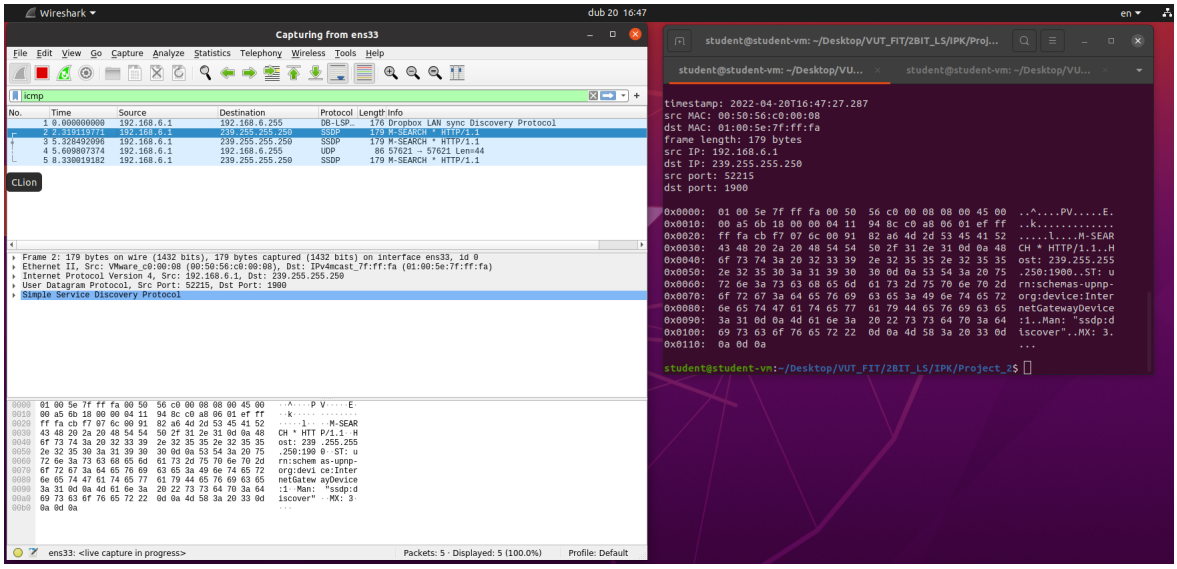
  src port: 57621

  dst port: 57621

```
0x0000:  ff ff ff ff ff ff 00 50 56 c0 00 08 08 00 45 00  .......PV.....E.
0x0010:  00 48 41 2c 00 00 80 11 6b 28 c0 a8 06 01 c0 a8  .HA,....k(......
0x0020:  06 ff e1 15 e1 15 00 34 3c 65 53 70 6f 74 55 64  .......4<eSpotUd
0x0030:  70 30 d4 5f 40 51 54 1b 15 ef 00 01 00 04 48 95  p0._@QT.......H.
0x0040:  c2 03 99 e1 e7 1d 77 71 70 f1 55 4e 33 36 c5 4b  ......wqp.UN36.K
0x0050:  ce 1c 4a 49 91 38                                ..JI.8
```

5 Testovanie

Testovanie funkčnosti aplikácie prebehlo manuálne za použitia referenčného open-source programu na zachytávanie a filtrovanie packetov Wireshark. Testovanie bolo úspešné, keďže údaje o paketoch zachytené pomocou programu Wireshark a pomocou ipk-snifferu boli zhodné.



Autor

Lucia Makaiová [xmakai00]

Literatúra

[1] CARSTENS, T.: PROGRAMMING WITH PCAP. [online], rev. 20. leden 2022.
URL `https://www.tcpdump.org/pcap.html`