Vysoké učení technické v Brně

Fakulta informačních technologií

Počítačové komunikace a sítě

2020/2021

Varianta: **Zeta**

Projekt: **Sniffer paketů**

Author: Kozhevnikov Dmitrii

Login: xkozhe00

Brno, 25. dubna 2021

1 Obsah

2 Úvod

Cílem projektu je návrh a implementace síťového analyzátoru, ktery bude schopný na určitém síťovém rozhraní zachytávat a filtrovat pakety. Také nutné vytvořit k projektu relevantní manual/dokumentaci.

Volání programu:

./ipk-sniffer [-i *rozhraní* | --interface *rozhraní*] {-p *port*} {[--tcp|-t] [--udp|-u] [--arp] [--icmp] } {-n *num*}

kde

- -i *eth0* (právě jedno rozhraní, na kterém se bude poslouchat. Nebude-li tento parametr uveden, či bude-li uvedené jen -i bez hodnoty, vypíše se seznam aktivních rozhraní)
- -p *23* (bude filtrování paketů na daném rozhraní podle portu; nebude-li tento parametr uveden, uvažují se všechny porty; pokud je parametr uveden, může se daný port vyskytnout jak v source, tak v destination části)
- -t nebo --tcp (bude zobrazovat pouze TCP pakety)
- -u nebo --udp (bude zobrazovat pouze UDP pakety)
- --icmp (bude zobrazovat pouze ICMPv4 a ICMPv6 pakety)
- --arp (bude zobrazovat pouze ARP rámce)
- Pokud nebudou konkrétní protokoly specifikovány, uvažují se k tisknutí všechny (tj. veškerý obsah, nehledě na protokol)
- -n *10* (určuje počet paketů, které se mají zobrazit; pokud není uvedeno, uvažujte zobrazení pouze jednoho paketu)
- argumenty mohou být v libovolném pořadí

3 Implementace

Program byl vytvořen v jazyce C++. Při implementace jsem použil knihovnu *libpcap.* Tato knihovna pro zachycení paketů poskytuje rozhraní na vysoké úrovni pro systémy pro zachycení paketů. Všechny pakety v síti, dokonce i ty, které jsou určeny pro jiné hostitele, jsou přístupné prostřednictvím tohoto mechanismu.

Project se skladá z:

- *ipk-sniffer.cpp* – samotný kod
- *Makefile*
- *manual.pdf* – manual k projektu
- *README.md*

4 Funkce program

Funkce, které program používá:

- *int main(int argc, char *argv[]* - hlavní funkce programu, kde probíhá kontrola vstupních argumentů a vytvoření filtru pro zpracování paketů
- *void clear(int signal)* - funkce kontroluje, zda je program správně odhlášen
- *void printPacketInfo(const u_char *data, int size)* - funkce pro vypisování informací z balíčků
- *void printPacket(u_char *args, const struct pcap_pkthdr *header, const u_char *packet)* - tato funkce zapisuje základní informace: vypočtený čas, IP adresy, čísla portů, počet bajtů přijatých v paketu. Prohlášení je provedeno s přihlédnutím k typu protokolu přijatého paketu.
- *int snifferFunction(char *myInterfaceName, int packetNumber, char *filter)* – vytvaření, filtering a zpracovává pakety
- *pcap_if_t *findInterface(string interface)* - kontrola existence zadaného rozhraní v seznamu rozhraní
- *printAllInterfaces()* - funkce vypíše seznam všech dostupných rozhraní

5 Příklady výstupu program

```
[redacted] $ sudo ./ipk-sniffer -i wlp2s0
2021-04-25T21:56:40.078+02:00 51.144.22.67 : 443 > 192.168.1.124 : 38594, length: 54 bytes
0x0000:      34 41 5d 93 bf 3e c8 3a 35 19 90 90 08 00 45 00        4A]..>.:5.....E.
0x0010:      00 28 f9 84 40 00 6d 06 08 54 33 90 16 43 c0 a8        .(..@.m..T3..C..
0x0020:      01 7c 01 bb 96 c2 81 53 be 3d 37 cd 9d c9 50 10        .|.....S.=7...P.
0x0030:      08 02 ee 35 00 00                                      ...5..
```

```
[redacted]$ sudo ./ipk-sniffer -i wlp2s0 --tcp -n 2
2021-04-25T21:57:34.004+02:00 192.168.1.124 : 51398 > 95.142.206.90 : 443, length: 294 bytes
0x0000:      c8 3a 35 19 90 90 34 41 5d 93 bf 3e 08 00 45 00         .:5...4A]..>..E.
0x0010:      01 18 5d b6 40 00 40 06 ec 1c c0 a8 01 7c 5f 8e         ..].@.@......|_.
0x0020:      ce 5a c8 c6 01 bb e7 0e fe 76 4f 39 0f 55 80 18         .Z.......vO9.U..
0x0030:      01 f5 2c 68 00 00 01 01 08 0a a8 5e fc 62 f7 0b         ..,h.......^.b..
0x0040:      f2 7f 17 03 03 00 df c2 8c cb 4c 4b 0a 7a 69 30         ..........LK.zi0
0x0050:      73 53 09 5d bf 01 51 95 b0 5c e8 68 97 f7 62 d1         sS.]..Q..\.h..b.
0x0060:      3c 0e 49 b3 13 a8 22 88 69 ce 44 f1 e3 20 7c 00         <.I..."."i.D.. |.
0x0070:      17 3e b9 a6 6d 95 29 3b 70 15 34 17 c7 95 3c 80         .>..m.);p.4...<.
0x0080:      e4 b4 5f f5 43 a0 81 df e0 79 f0 69 c7 59 8f 88         .._.C....y.i.Y..
0x0090:      2a 60 22 0f 9e 4c 57 e4 48 7b 4d f1 49 4d 82 8d         *`"..LW.H{M.IM..
0x0100:      d4 05 3e 12 7b a2 a2 24 96 21 41 71 48 e8 47 7b         ..>.{..$.!AqH.G{
0x0110:      f0 a2 3f ab 5d a2 87 9b 65 8e b3 d2 fb 10 6f f3         ..?.]...e.....o.
0x0120:      7a cc f2 15 e8 1b 58 f8 34 71 7d 68 44 de cf 6d         z.....X.4q}hD..m
0x0130:      35 68 0c f4 b1 d1 44 32 ca 08 ed a0 c5 b5 a7 e6         5h....D2........
0x0140:      35 ad 2b 37 95 39 86 9c 34 c6 67 7f 84 dc 27 3d         5.+7.9..4.g...'=
0x0150:      f3 4d d5 8e 1d 01 92 7d 45 2f a5 b6 ac 6d b2 cc         .M.....}E/...m..
0x0160:      f6 85 b2 d6 a9 be 97 0b 94 d9 c8 55 40 30 8a 6f         ...........U@0.o
0x0170:      e5 66 ef 67 2e 30 19 ee e1 8b f1 ab f4 4e 5f 6c         .f.g.0.......N_l
0x0180:      a4 31 a8 97 74 c4                                       .1..t.

2021-04-25T21:57:34.018+02:00 192.168.1.124 : 38594 > 51.144.22.67 : 443, length: 105 bytes
0x0000:      c8 3a 35 19 90 90 34 41 5d 93 bf 3e 08 00 45 00         .:5...4A]..>..E.
0x0010:      00 5b 79 86 40 00 40 06 b5 1f c0 a8 01 7c 33 90         .[y.@.@......|3.
0x0020:      16 43 96 c2 01 bb 37 ce c9 8e 81 53 f0 49 50 18         .C....7....S.IP.
0x0030:      1d f7 3d 17 00 00 17 03 03 00 2e b4 71 90 6b 9a         ..=.........q.k.
0x0040:      e6 00 db 8d a8 0c af 73 b1 9e b1 00 61 b1 36 71         .......s....a.6q
0x0050:      6b c8 22 5a d8 d4 db 2c 93 3b 89 7c fa 23 82 83         k."Z...,.;.|.#..
0x0060:      e5 e9 b6 c9 90 67 22 c9 d8                              .....g"..
```

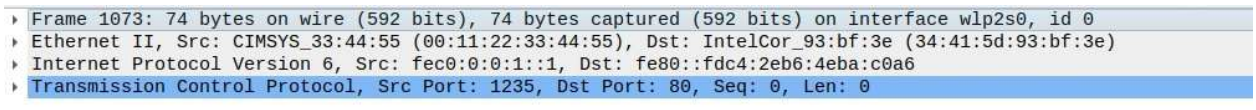
6 Testování programu

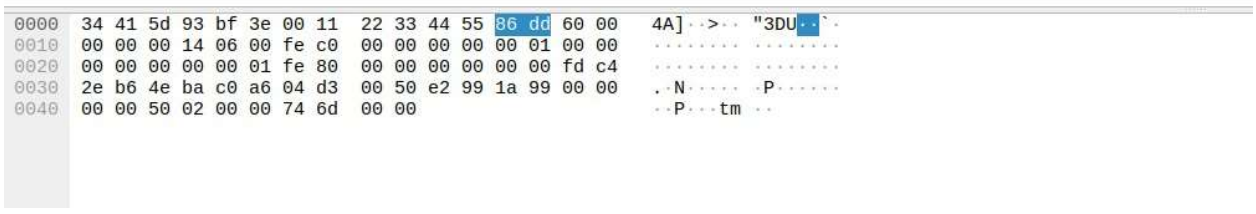
Program jsem testoval pomoci porovnání přijatých balíčků programem a Wiresharkem.

Můj terminal

```
[redacted]$ sudo ./ipk-sniffer -i wlp2s0 -p 80
2021-04-25T22:12:35.044+02:00 fec0:0:0:1::1 : 1235 > fe80::fdc4:2eb6:4eba:c0a6 : 80, length: 74 bytes
0x0000:      34 41 5d 93 bf 3e 00 11 22 33 44 55 86 dd 60 00         4A]..>.."3DU..`.
0x0010:      00 00 00 14 06 00 fe c0 00 00 00 00 00 01 00 00         ................
0x0020:      00 00 00 00 00 01 fe 80 00 00 00 00 00 00 fd c4         ................
0x0030:      2e b6 4e ba c0 a6 04 d3 00 50 e2 99 1a 99 00 00         ..N......P......
0x0040:      00 00 50 02 00 00 74 6d 00 00                           ..P...tm..
```

Wireshark





7 Zdroje

Seznam zdroju:

1. http://www.tcpdump.org - oficiální webové stránky tcpdump
2. https://www.winpcap.org/docs/docs_412/html/group__language.html – filtrování syntaxe výrazu; Uživatelská příručka WinPcap
3. https://android.googlesource.com/platform/bionic/+/master/libc/include/netinet/ip.h - informace o knihovně ip.h
4. https://support.huawei.com/enterprise/en/doc/EDOC1000178017/dd76ea1f/ipv4-packet-format – informace o IPv4
5. https://pro-prof.com/forums/topic/libpcap - informace o prace s libPcap v C++
6. https://man7.org/linux/man-pages/man7/ipv6.7.html - informace o IPv6
7. https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol - ICPV informace
8. https://www.juniper.net/documentation/us/en/software/junos/interfaces-security-devices/topics/topic-map/security-interface-ipv4-ipv6-protocol.html – prace s IPv4 a IPv6 protokoly
9. http://embeddedguruji.blogspot.com/2014/01/pcapfindalldevs-example.html - prace s pcap_findalldevs
10. inet_ntop(3) - Linux manual page (man7.org) – prace s inet_ntop
11. https://stackoverflow.com – hledání chyb v kodu
12. https://www.tcpdump.org/pcap.html
13. https://www.binarytides.com/packet-sniffer-code-c-libpcap-linux-sockets/

8 Závěr

Pro mě tento projekt byl složitý, ale zajímavý. Kvůli tomuto projektu jem se naučil odchytávat pakety z internetové komunikace. Naučil jsem se pracovat s protokoly a balíčky, a rozšířil jsem své znalosti o provozu sítí. Rozšiřil znalost o protokolech a naučil jsem se pracovat s knihovnou *libpcap*.

Tento program jsem zkontroloval a otestoval pomocí Wiresharku.