Dokumentace k 2. projektu z IPK

Varianta ZETA: sniffer paketů

Hung Do
xdohun00@stud.fit.vutbr.cz

24. dubna 2022

Obsah

1 Úvod

1.1 Co je to paket?

V dnešní době zařízení spolu komunikují tak, že si mezi sebou přeposílají zprávy. Aby se zprávy mohly odesílat efektivně, je potřeba tyto zprávy rozříznout na menší úseky. A těmto úsekům se říkají *pakety*.

Paket je tedy malý segment velké zprávy. Každá zpráva se rozdělí před odesláním na pakety, které se pak posílají po síti. Každý paket pak může k příjemci doputovat jinou cestou a v jiný okamžík. Koncové zařízení ale dokáže po přijetí všech paketů zprávu znovu sestavit [2].

1.2 Co je to sniffer packet?

Packet sniffer, česky analyzátor paketů nebo paketový sniffer, je prgram, který monitoruje provoz sítě. Program funguje tak, že sleduje a zachytává příchozí a odchozí datové pakety v síti. Uživatel pak může analyzovat obsah těchno paketů. Jsou dva režimy zachytávání paketů: **nefiltrovaný režim**, který zachytává všechny pakety, nebo **filtrovaný režim**, který zachytává jenom pakety splňující parametry filtru (např. použitý protokol, nebo IP adresa destinace) [3].

2 Použité prostředky

V projektu byly použité prostředky *libpcap* a aplikace *Wireshark*. **Libpcap** je open-source knihovna, která obsahuje rozhraní pro zachytávání paketů v síti [1]. **Wireshark** je volně dostupná aplikace pro analýzu provozu v počítačových sítích[1].

3 Implementace

Celá aplikace je napsaná v jazyce C a je rozdělen do tří částí: načítání argumentů, zachytávání paketů a jejich následovné zpracování.

3.1 Načítání argumentů

Argumenty se zpracovávají v souborech `arguments.c` a `arguemnts.h`. Tyto soubory používají API `getopts.h`. Jádrem tohoto modulu je funkce `getopt_long`, která podporuje načítání slovních argumentů (argumentů začínající prefixem `--`). Volitelné parametry argumentů muselo být ručně řešeno pomocí knihovní proměnné `optind`, která si uchovává pozice parametru v seznamu argumentů[2].

Po zpracování argumentů se modul chová jako *read-only* globální struktura. Jedná se o simulaci zapozdření ve strukturovaném programovacím jazyce C.

3.2 Zachytávání paketů

V druhé části aplikace inicializuje zachytavač paketů. Poté, co uživatel vybere rozhraní, ze kterého chce pakety zachytávat, se vytvoří zachytávač pomocí funkce `pcap_create`, nastaví se potřebná data, jako například časovač, a následně se aktivuje pomocí funkce `pcap_activate`.

Struktura zachytávače je poté předána do funkce `pcap_loop`. Ten až do přerušení čte pakety. Jakmile se zacbytí paket, tak se spustí uživatelem definovaná fce `packet_handler`, který se už stará o zpracování paketu[3].

[1] Oficiální stránka Wireshark: `https://www.wireshark.org/`

[2] Více informací na manuálové stránce: `https://linux.die.net/man/3/getopt_long`

[3] Více informací na manuálové stránce: `https://www.tcpdump.org/manpages/pcap.3pcap.html`

3.3 Zpracování a výpis obsahu paketu

Jak už víme ze sekce 1.1, paket je malý segment velké zprávy. Jedná se o pole bajtů, který se řídí jasně daným tvarem/rozhraním. V projektu má aplikace být schopna zpracovat 4 protokoly:

- TCP
- UDP
- TCMP
- ARP

Paket se může skládat ze 3 až 4 částí (headerů). Každý paket má **Ethernet Frame**, podle kterého se zjistí, jakého protokolu přijatý paket je. Pod ním se může schovávat **Internet Protocol** (IP) nebo **Address Resolution Protocol** (ARP). A IP může obsahovat **Tangosol Cluster Management Protocol** (TCMP), **Transmission Control Protocol** (TCP), nebo **User Diagram Protocol** (UDP).

Součástí UNIX operačních systémů jsou hlavičkové soubory pod `netinet` API, který na všechny tyto sekce má vytvořenou strukturu (např. `struct iphdr` pro IPv4 protokol nebo `struct ethhdr` pro Ethernet rámec). Stačilo tedy jenom součítat offset v poli bajtů pro danou sekci a přetypovat pole na danou strukturu. Získané informace se pak zpracovávaly v `header_display.c` a `header_display.h` souborech. Soubory `packet_handler.c` a `packet_handel.h` obsahují definici funkci `packet_handler` pro zpracování zachyceného packetu.

Příklad získání IP sekce v poli bajtů v jazyce C:

```
void packet_handler(u_char *user, const struct pcap_pkthdr *h, const u_char *bytes) {
    // ...
    struct iphhdr *ip4 = (struct iphhdr *)(bytes + sizeof(struct ethhdr));
    // ...
}
```

4 Příkladné výstupy programu

Program byl testován na operačním systému `Manjaro Linux x86_64`, s jádrem `5.15.32-1-MANJARO`. Výstupy aplikace souhlasí s výstupy programu **Wireshark**.

```
[thinkpad-e14 Project_2]# ./ipk-sniffer
enp2s0
ap0
any
lo
wlp3s0
virbr223
virbr0
bluetooth0
bluetooth-monitor
nflog
nfqueue
dbus-system
dbus-session
[thinkpad-e14 Project_2]# ./ipk-sniffer -i enp2s0 -n 1
--------------- Frame info ---------------
Arrival time:         2022-04-24T01:51:44.7028+02:00
Frame length:         60 bytes
--------------- Ethernet II --------------
Src MAC:              94:3f:c2:07:ca:10
Dst MAC:              ff:ff:ff:ff:ff:ff
Type:                 0x0806
------ Address Resolution Protocol -------
Hardware type:        1
Protocol type:        0x0800
Hardware size:        256
Protocol size:        4
Opcode:               1
Sender MAC address:   94:3f:c2:07:ca:10
Sender IP address:    147.229.208.1
Target MAC address:   00:00:00:00:00:00
Target IP address:    147.229.208.15
-------------- Data Dump -----------------
0x0000   ff ff ff ff ff ff 94 3f  c2 07 ca 10 08 06 00 01   .......? ........
0x0010   08 00 06 04 00 01 94 3f  c2 07 ca 10 93 e5 d0 01   .......? ........
0x0020   00 00 00 00 00 00 93 e5  d0 0f 00 00 00 00 00 00   ........ ........
0x0030   00 00 00 00 00 00 00 00  00 00 00 00               ........ ....
```

Literatura

[1] Jacobson, V.; Leres, C.; McCanne, S.: Home: TCPDUMP & LIBPCAP. [online], 2010 [cit. 2022-04-23]. Dostupné z: `https://www.tcpdump.org/index.html`

[2] Kurose, J. F.; Ross, K. W.: *Computer networking: A Top-Down Approach.* Pearson, 2017, ISBN 978-0-13-359414-0.

[3] www.kaspersky.com: What is a Packet Sniffer? [online], 2022 [cit. 2022-04-23]. Dostupné z: `https://www.kaspersky.com/resource-center/definitions/what-is-a-packet-sniffer`