

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií



Počítačové komunikace a sítě
2019/2020

Paket sniffer

Varianta: ZETA

Obsah

Obsah	2
Paket Sniffer	3
Implementácia	3
Rozhranie	4
Zachytávanie paketov	4
Vypisovanie jednotlivých paketov	4
Výpis hlavičky paketu	5
Výpis dát paketu	5
Testovanie	5
Literatúra	5

Paket Sniffer

Našou úlohou bolo vytvoriť program na zachytávanie paketov na sieti. Zachytávanie komunikácie ktorá prebieha medzi zariadením na ktorom program spustíme a ostatnými zariadeniami s ktorými zariadenie spolupracuje. Náš projekt mal zameranie hlavne na TCP a UDP pakety, a preto program zobrazuje práve tieto.

Implementácia

Program je naprogramovaný v jazyku C. Tento jazyk som si vybral pretože som s ním mal z pomedzi jazykov na vybranie najviac skúseností.

Program využíva argumenty pri spustení na upresnenie funkcionality a určenie ake pakety chce užívateľ zachytávať.

Zoznam povolených parametrov je nasledujúci:

- -i eth0 rozhranie na ktorom sniffer očakáva pakety. Ak sa tento parameter neuvedie, vypíše sa aktívne rozhrania
- -p 23 Filtrovanie paketov na základe portu. Ak nebude parameter uvedený, očakávajú sa pakety na všetkých portoch
- -t alebo --tcp (bude zobrazovať iba tcp pakety)
- -u alebo --udp (bude zobrazovať iba udp pakety)
- Ak nebude -t ani -u uvedené, zobrazujú sa oba typy zároveň.
- -n 10 Počet paketov ktoré sa majú zobrazíť, ak sa parameter neudá tak sa zobrazí iba jeden parameter. Pri zadaní negatívneho počtu paketov beží program do zastavenia napríklad cez CTRL+C.

volanie programu potom vyzerá nasledovne:

```
./ipk-sniffer -i rozhraní [-p port] [--tcp|-t] [--udp|-u] [-n num]
```

Na parsing argumentov som vybral funkciu getopt_long() ktorá narozdiel od getopt povolí argumenty ako --tcp --udp ktoré sú zadané s dvomi pomlčkami namiesto jednej.

Argumenty slúžia na upresnenie toho čo bude program robiť. Po zparsovani argumentov pomocou funkcie getopt_long teda prebehne nastavenie premenných tak, aby program vedel ake argumenty boli zadané. Napríklad po zadaní argumentu -p sa nastaví port na ktorom zariadenie bude počúvať. Toto ovplyvní napríklad aj kompiláciu procesu na zachytávanie paketov, keďže sa pred kompiláciou nastaví filter, napríklad na zachytávanie iba paketov ktoré boli poslané na určený port.

Rozhranie

Pre zachytávanie paketov je treba určiť rozhranie pomocou argumentu -i.

Po zadaní rozhrania a spustení programu sa najskôr program uistí že zadané zariadenie existuje, a že má užívateľ dostatočné práva na spustenie a otvorenie tohoto zariadenia.

Pre zaistenie dostatočných oprávnení odporúčam spúšťať program ako sudo ./ipk-sniffer.

Po skontrolovaní existencie zariadenia a oprávnení, sa rozhranie nastaví na počúvanie paketov pomocou funkcie pcap_open_live.

Ak sa užívateľ rozhodol zadať argumenty na obmedzenie zachytávania paketov ako napríklad -p na zadanie portu alebo upresnenie protokolu na TCP alebo UDP, tak sa pred samotným spustením zachytávania nastaví filter na zachytávanie práve týchto paketov.

K tomuto mi poslúžila funkcia pcap_setfilter.

Zachytávanie paketov

Pomocou funkcie pcap_loop ktorá beží tak dlho kým sa nevypíše tolko paketov koľko bolo zadaných cez parameter -n alebo defaultne 1 paket. Táto funkcia volá definovanú funkciu processing ktorá na základe verzie IP a na základe protokolu ktorý je aktuálne na rozhraní rozhodne ako s paketom ďalej naložiť.

Funkcia **processing** po rozhodnutí o verzii a typu paketu volá najskôr funkciu ktorá vypíše zdrojové a cieľové IP adresy a porty. Z dôvodu problému s generovaním nadbytočných DNS dotazov pri resolvovaní IP adresy na doménové meno som sa rozhodol vypisovanie doménového mena vynechať. Po vypísaní týchto informácií sa prejde na samotné vypísanie jednotlivých paketov.

Vypisovanie jednotlivých paketov

Po vypísaní zdrojových a cieľových IP adries a portov sa prejde na výpis jednotlivých častí paketu, a to v dvoch častiach. A to z jednoduchého dôvodu a to toho že výpis tak vyzerá ucelenejší a na prvý pohľad je vidieť aká časť paketu sa týka hlavičky a akú časť predstavujú jednotlivé dáta. V prípade že dáta nevyplnia celý riadok (16 bytov) tak sa riadok pre krajší výpis vyplní medzerami. Z toho dôvodu aby výpis v ascii podobe zostal zachovaný.

Výpis paketu je teda rozdelený na:

- Výpis hlavičky
- Výpis dát

Výpis hlavičky paketu

Na rozdelenie paketu na header a data si najskôr zistím dĺžku hlavičky a podľa toho pošlem dĺžku na vypísanie funkcie `process_data`, ktorá vypíše z paketu určitý počet dát.

Po vypísaní hlavičky paketu si uloží counter jednotlivých bytov ktoré boli vypísané aby som tento potom pričítal pri vypisovaní jednotlivých dát paketu.

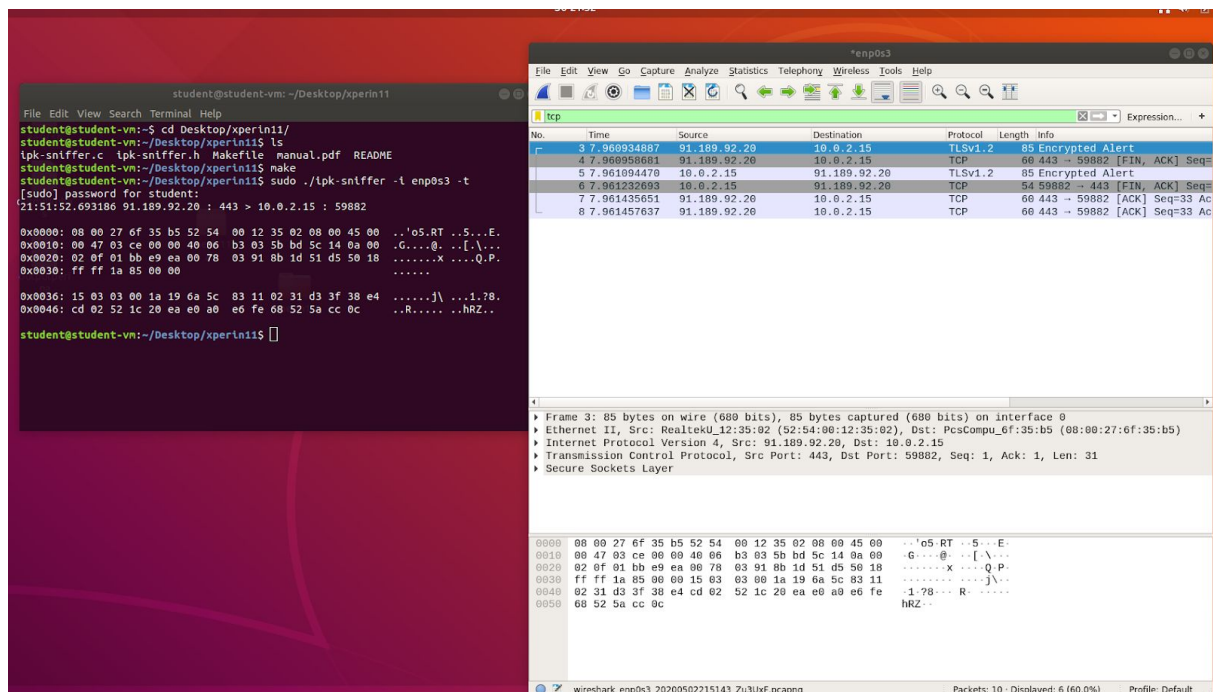
Výpis dát paketu

Po vypísaní hlavičky paketu sa následne vypíšu dáta paketu, ak nejaké existujú- niektoré pakety obsahujú iba hlavičku. Dáta sa vypisujú pomocou rovnakej funkcie ako výpis hlavičky paketu, pretože sa vlastne jedná o ten istý typ dát, ide iba o vypísanie. Dáta sa v ľavej časti terminálu vypíšu v hexadecimálnej podobe a v pravej časti terminálu v podobe ascii znakov. Ak daný hexadecimálny znak nemá v ascii tabulke význam, vypíše sa iba bodka. Toto je veľmi častý úkaz.

Testovanie

Program bol testovaný v dvoch krokoch, v prvom kroku boli testované parametre programu a ich správnosť a v druhom kroku bol kontrolovaný hlavne výpis jednotlivých paketov.

Výpis paketov bol kontrolovaný a overený na základe výstupov z programu Wireshark, ktorý bol pri tomto projekte veľkou oporou hlavne pri pochopení správania sa paketov a výpis paketov rozdelený na hlavičku a dáta paketu.



The screenshot displays two windows side-by-side. The left window is a terminal running a program named `tpk-sniffer`. The terminal output shows the program's execution, including the command `./tpk-sniffer -l enp0s3 -t` and the resulting packet capture data in hex and ASCII. The right window is Wireshark, showing a list of captured packets. The selected packet is a TLSv1.2 Encrypted Alert, which is a TCP Reset (RST) with sequence number 59882. The packet details pane shows the structure of the alert, including the alert type and the alert data.

Literatúra

Na vypracovanie projektu som používal hlavne knihu "Computer networking : a top-down approach featuring the internet" vypožičanú z knihovny FIT.

Pri programovaní mi asistovala taktiež online literatúra a to v podobe článku "Programming with pcap" a článok "Using libpcap library" taktiež prístupný online.