**Analyzátor sieťovej komunikácie**

Autor: Peter Šípoš

**Zadanie úlohy**

Navrhnite a implementujte programový “post” analyzátor Ethernet siete, ktorý analyzuje komunikácie v sieti zaznamenané v .pcap súbore a poskytuje nasledujúce informácie o komunikáciách. Vypracované zadanie musí spĺňať nasledujúce body:

1) Výpis všetkých rámcov v hexadecimálnom tvare postupne tak, ako boli zaznamenané v súbore.

Pre každý rámec uveďte:

1. a) Poradové číslo rámca v analyzovanom súbore.
2. b) Dĺžku rámca v bajtoch poskytnutú pcap API, ako aj dĺžku tohto rámca prenášaného po médiu.
3. c) Typ rámca – Ethernet II, IEEE 802.3 (IEEE 802.3 - LLC, IEEE 802.3- LLC - SNAP, IEEE 802.3 – Raw).
4. d) Zdrojovú a cieľovú fyzickú (MAC) adresu uzlov, medzi ktorými je rámec prenášaný.

Vo výpise jednotlivé bajty rámca usporiadajte po 16 alebo 32 v jednom riadku. Pre prehľadnosť výpisu je vhodné použiť neproporcionálny (monospace) font.

2) Študent musí vedieť vysvetliť, aké informácie sú uvedené v jednotlivých rámcoch Ethernet II, t.j. vnáranie protokolov ako aj ozrejmiť dĺžky týchto rámcov.

3) Analýzu cez vrstvy vykonajte len pre rámce Ethernet II a protokoly rodiny TCP/IPv4:

Na konci výpisu z bodu 1) uveďte pre IPv4 pakety:

1. a) Zoznam IP adries všetkých vysielajúcich uzlov,
2. b) IP adresu uzla, ktorý sumárne odvysielal (bez ohľadu na príjemcu) najväčší počet bajtov a koľko bajtov odoslal.

V danom súbore analyzujte komunikácie pre zadané protokoly:

1. a) HTTP
2. b) HTTPS
3. c) TELNET
4. d) SSH
5. e) FTP riadiace
6. f) FTP dátové
7. g) Všetky TFTP
8. h) Všetky ICMP
9. i) Všetky ARP dvojice (request – reply).

Vo všetkých výpisoch treba uviesť aj IP adresy a pri transportných protokoloch aj porty komunikujúcich uzlov.

V prípade výpisu h) uveďte aj typ ICMP správy (pole Type v hlavičke ICMP), napr. Echo request, Echo reply, Time exceeded, a pod.

V prípade výpisu i) uveďte pri ARP-Request IP adresu, ku ktorej sa hľadá MAC (fyzická) adresa a pri ARP-Reply uveďte konkrétny pár - IP adresa a nájdená MAC adresa. V prípade, že bolo poslaných viacero rovnakých rámcov ARP-Request, vypíšte všetky.

Ak počet rámcov danej komunikácie je väčší ako 20, vypíšte iba 10 prvých a 10 posledných rámcov. Pri všetkých výpisoch musí byť poradové číslo rámca zhodné s číslom rámca v analyzovanom súbore.

4) Program musí byť organizovaný tak, aby čísla protokolov v rámci Ethernet II a v IP pakete ako aj čísla portov v transportných protokoloch boli programom určené z externého súboru a pre známe protokoly a porty boli uvedené aj ich názvy.

5) V procese analýzy rámcov pri identifikovaní jednotlivých polí rámca ako aj polí hlavičiek vnorených protokolov nie je povolené použiť funkcie poskytované použitým programovacím jazykom. Celý rámec je potrebné spracovať postupne po bajtoch.

6) Program musí byť organizovaný tak, aby bolo možné jednoducho rozširovať jeho funkčnosť o výpis rámcov podľa ďalších požiadaviek na protokoly v bode 3) - pri doimplementovaní jednoduchej funkčnosti na cvičení.

7) Študent musí byť schopný preložiť a spustiť program v miestnosti, v ktorej má cvičenia!

**Obsah externého súboru pre určenie protokolov a portov**

Ethernet\_II 05DC

IPv4 0800

IEEE\_802.3\_Raw FFFF

IEEE\_802.3\_SNAP AAAA

IEEE\_802.3\_LLC 05DC

ARP 0806

Request 01

ICMP 01

TCP 06

UDP 11

FTP\_DATA 14

FTP\_CONTROL 15

SSH 16

TELNET 17

HTTP 50

HTTPS 1BB

TFTP 45 END\_NORMAL

00 Echo\_Reply

03 Destination\_uncreachable

04 Source\_Quench

05 Redirect

08 Echo

09 Router\_Advertisement

0A Router\_Selection

0B Time\_Exceeded

0C Parameter\_Problem

0D Timestamp

0E Timestamp\_Reply

0F Information\_Request

10 Information\_Reply

11 Address\_Mask\_Request

12 Address\_Mask\_Reply

1E Traceroute

999 Frag\_protocol\_continue

**Blokový návrh fungovania riešenia**

V programe som sa snažil maximalizovať využitie vlastných funkcií. Výpis všetkých rámcov z .pcap súboru ukladám do súboru „vypis\_pcap\_suboru.txt“, ktorý sa automaticky vytvorí. Výpis do súboru je istejší, pretože ak je v .pcap súbore priveľa rámcov, tak sa môže stať, že nie všetky sa zmestia na výpis v konzole. Na tento výpis používam povolenú funkciu *pcap\_loop*, v ktorej ako callback funkciu volám *packet\_to\_do*. Tá následne vždy zavolá jednotlivé funkcie pre výpis konkrétnych údajov (od *print\_name\_number* na výpis čísla rámca až po *print\_hexagulash* na celkový výpis rámca v hexadecimálnom tvare).

Následne sa zavolá funkcia *print\_ips*. Tá zapíše všetky rôzne IP adresy uzlov, ktoré vysielali dáta (program počíta s dĺžkou rámca prenášaného po médiu), taktiež do súboru „vypis\_pcap\_suboru.txt“. Nakoniec zapíše IP adresu uzla, ktorý odvysielal najviac, spolu s počtom bajtov, ktoré odvysielal. Funkcia si najprv spočíta množstvo IPv4 komunikácii v súbore. Potom si vytvorí 2D pole, do ktorého bude ukladať jednotlivé IP adresy ako aj sumárny počet jej odvysielaných dát. Vždy sa najprv prechádzajú už existujúce IP adresy. Ak práve čítaná IP adresa už bola predtým načítaná, tak iba pripočíta množstvo odvysielaných dát. Ak nie tak, tak túto novú adresu uloží do poľa. Na záver už len vyberie maximum a vypíše ho.

Nakoniec má užívateľ možnosť zadať, či chce vypísať niektorú z možných komunikácií alebo ukončiť program. Ak sa rozhodne pre výpis niektorej komunikácie, tak ten obstará funkcia *print\_exac\_com*.

**Opis mechanizmu analyzovania protokolov na jednotlivých vrstvách**

Analýzu cez vrstvy zabezpečuje výpis z funkcie *print\_exac\_com*. Funkcia si najprv nasčíta celkový počet výskytu daného typu komunikácie v .pcap súbure a následná z nej vypíše prvých a posledných 10 rámcov. Tento výpis je realizovaný do konzoly, keďže tá je v tomto prípade dostačujúca. Funkcia vie o aký druh komunikácie sa jedná vďaka pomocným funkciám *has....* (napr. *has\_ipv4, has\_http, has\_ssh...*). Každá z týchto pomocných funkcií vráti 1, ak sa daný typ komunikácie v danom rámci nachádza, inak vráti 0. Samotné overenie prebieha tak, že funkcia porovná číselnú hodnotu daného protokolu či portu (načítanú z externého súboru) s hodnotou z pozície, kde by sa tento protokol či port mal v rámci nachádzať. Ak došlo k zhode, tak logicky sa daný protokol či port v rámci nachádza. V prípade portov je takto kontrolovaný zdrojový aj cieľový port.