Richard Šléher, ID:120890, ak.r.:2023/2024

Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Počítačové a komunikačné siete

Zadanie 1: Analyzátor sieťovej komunikácie

Obsah

Richard Šléher, ID:120890, ak.r.:2023/2024

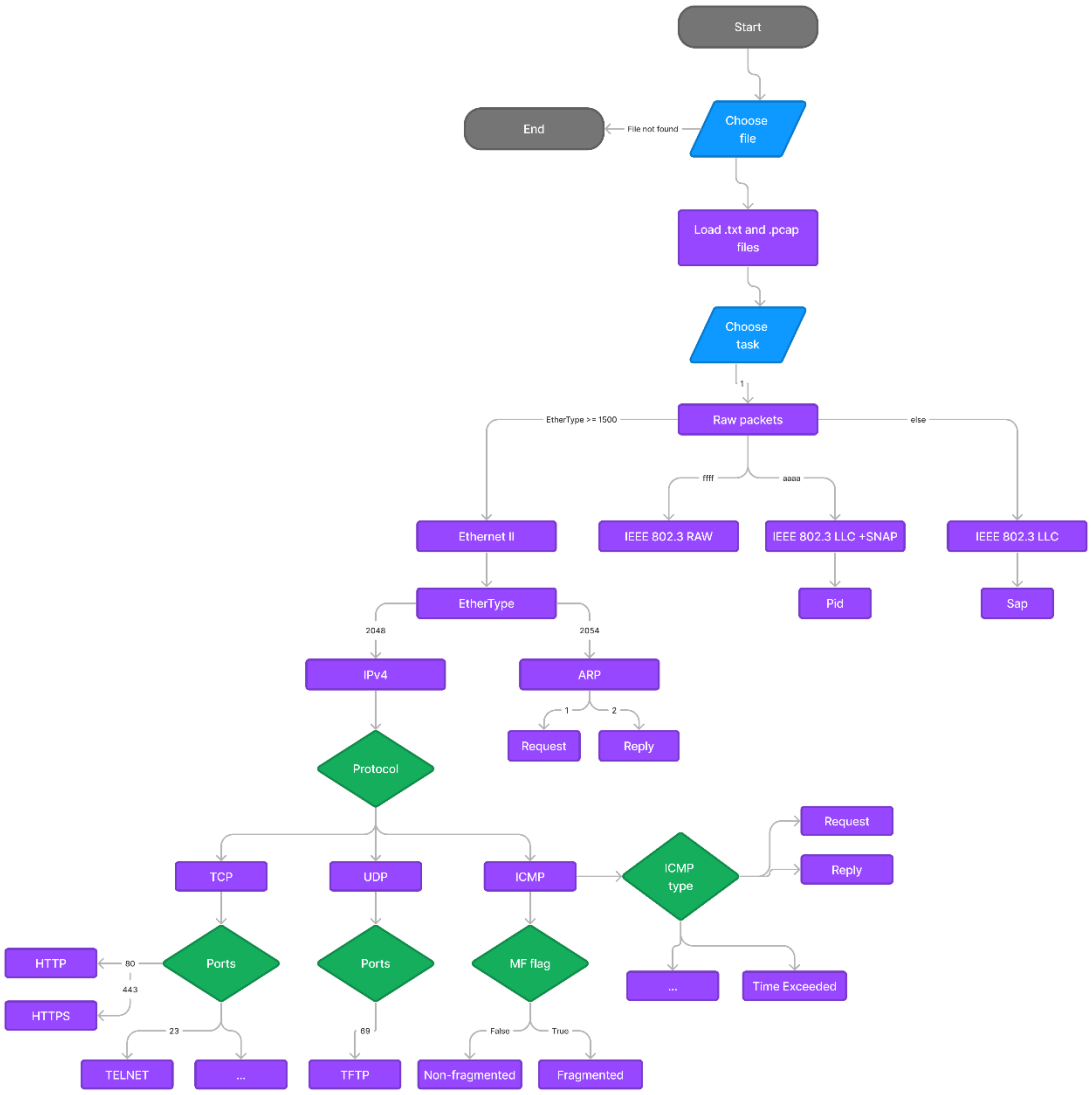
Zadanie

Navrhnite analyzátor sieťovej komunikácie, kde program dostane na vstupe .pcap súbor a druh filtru podľa ktorého bude program analyzovať pakety a komunikácie. Komunikácie následne rozdelí do kompletných a čiastočných podľa zadaného filtra.

Richard Šléher, ID:120890, ak.r.:2023/2024

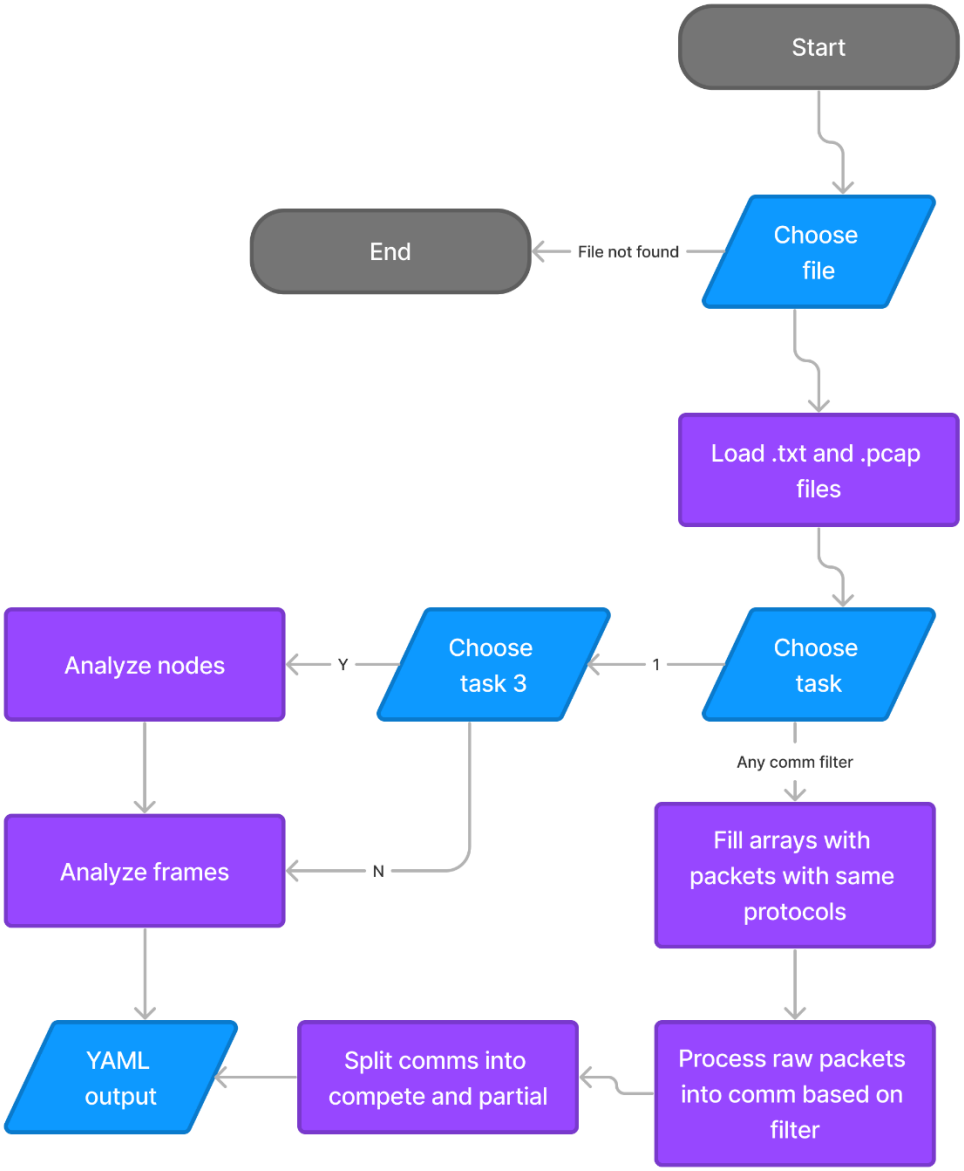
Diagramy spracovávania

Analyzovanie rámcov



Richard Šléher, ID:120890, ak.r.:2023/2024

Zjednodušený priebeh programu



Fungovanie riešenia

Fungovanie úloh 1-3

Do funkcie sa posielajú rámce, ktoré sa následne spracujú a ich všetky hodnoty sa zapíšu do slovníka a potom do súboru yaml. Do funkcie sa ešte posiela číslo paketu a pri výpise ICMP komunikácie aj flag či je rámec fragmentovaný alebo nie. Úloha 3 sa vypisuje iba pri úlohe vypisovania rámcov.

Fungovanie jednotlivých filtrov

Všetky filtre majú podobnú logiku, kde sa najprv roztriedia rámce s rovnakým protokolom do polí. Potom sa rámce rozdelia do jednotlivých komunikácií podľa zadania a následne sa zistí, či daná komunikácia patrí do kompletnej alebo nie a vypíšu sa.

TCP komunikácia

Pri TCP sa delia rámce podľa IP adries a portov a následne sa najprv kontroluje, či je komunikácia začatá pomocou flagov v rámci a potom koniec. Začiatok môže nastať 2 spôsobmi. Prvý je „three-way handshake“ a druhý, že obe strany pošlú naraz príznak SYN. Ukončenie môže nastať 3 spôsobmi a to príznakom RST, naraz si pošlú príznak FIN alebo jedna strana pošle FIN a druhá mu odpovie, dostane naspäť ACK a komunikáciu ukončí. Potom už iba vypíšeme jednotlivé komunikácie do kompletných a prvú nekompletnú

UDP komunikácia

Pri UDP protokole sme pracovali s TFTP komunikáciou na porte 69. Keďže ide o connectionless protokol tak sme brali kompletnú komunikáciu, takú kde je posledný prijatý datagram s príznakom ACK. Ostatné berieme ako nekompletné.

ARP komunikácia

Rámce rozdeľujeme po pároch podľa IP adries a „opcode“. Ako kompletné považujeme pár REQUEST – REPLY. Do 2 nekompletných vypíšeme všetky REQUESTY a REPLY zvlášť bez páru.

ICMP komunikácia + fragmentácia

Pri tomto filtri sme najprv rozdelili rámce do dvojíc REQUEST – REPLY podľa IP adries, ID a SEQ čísel. Potom sme kontrolovali, či sú komunikácie kompletné alebo nie. V prípade, kde sa odpovie na REQUEST TIME EXCEED je komunikácia považovaná za nekompletnú. Ďalej sa kontroluje, či je rámec fragmentovaný. Po vytvorení komunikácií ešte pozrieme komunikácie s rovnakými ID a IP adresami a spojíme ich. Následne ich vypisujeme a rámce bez odpovede alebo s neplatnou odpoveďou vypisujeme ako nekompletné.

Štruktúra externého súboru

V externom súbore mam uložené protokoly a flagy s ich decimálnymi hodnotami, ktoré program načíta do slovníkov. Súbor je vo formáte .txt a má nasledovnú formu, kde „LLC:“, “IPPROTOCOL:“,... riadky slúžia ako rozdeľovač (na obrázku je iba časť súboru):

```
LLC:
66:STP
224:IPX
240:NETBIOS
ETHERTYPE:
2048:IPv4
2054:ARP
34525:IPv6
35020:LLDP
36864:ECTP
IPPROTOCOL:
1:ICMP
2:IGMP
6:TCP
9:IGRP
17:UDP
47:GRE
50:ESP
51:AH
57:SKIP
88:EIGRP
89:OSPF
103:PIM
115:L2TP
ICMP:
0:Echo Reply
3:Destination Unreachable
4:Source Quench
5:Redirect
8:Echo
9:Router Advertisiment
10:Router Selection
11:Time Exceeded
12:Parameter Problem
13:Timestamp
14:Timestamp Reply
```

Používateľské rozhranie

Používateľ spúšťa program cez menu, ktoré najprv musí dostať cestu k súboru a potom typ úlohy akú má vykonať. Pri zadaní „1“ sa program ešte opýta, či má vypísať aj úlohu 3 na výpis IP adries.

```
Enter the path to the pcap file: "C:\Users\riso\Desktop\vzorky_pcap_na_analyzu\trace-15.pcap"
Choose task:
Task 1-2 enter "1"
Task 4:
For HTTP -> "HTTP"
For HTTPS -> "HTTPS"
For TELNET -> "TELNET"
For SSH -> "SSH"
For FTP-CONTROL -> "FTPC"
For FTP-DATA -> "FTPD"
For TFTP -> "TFTP"
For ICMP -> "ICMP"
For ARP -> "ARP"
Enter the task:
```

Voľba implementačného prostredia

Zvolil som si IDE Visual Studio Code, kvôli rozsiahlej možnosti upravovania prostredia s rozšíreniami. Zadanie je písane v jazyku Python s použitím knižníc ako binascii, os, scapy a yaml.