

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

POČÍTAČOVÉ A KOMUNIKAČNÉ SIETE

Zadanie 1: Analyzátor sieťovej komunikácie

Martin Hric

ak.rok 2021/2022

Znenie zadania:

Navrhните a implementujte programový 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:
 - a) Poradové číslo rámca v analyzovanom súbore.
 - b) Dĺžku rámca v bajtoch poskytnutú pcap API, ako aj dĺžku tohto rámca prenášaného po médiu.
 - c) Typ rámca – Ethernet II, IEEE 802.3 (IEEE 802.3 s LLC, IEEE 802.3 s LLC a SNAP, IEEE 802.3 – Raw).
 - 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) Pre rámce typu Ethernet II a IEEE 802.3 vypíšte vnorený protokol. 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 pre rámce Ethernet II a protokoly rodiny TCP/IPv4:
Na konci výpisu z bodu 1) uveďte pre IPv4 pakety:
 - a) Zoznam IP adries všetkých odosielaajúcich uzlov,
 - b) IP adresu uzla, ktorý sumárne odoslal (bez ohľadu na prijímateľa) najväčší počet paketov
a koľko paketov odoslal (berte do úvahy iba IPv4 pakety).

IP adresy a počet odoslaných / prijatých paketov sa musia zhodovať s IP adresami vo výpise
Wireshark -> Statistics -> IPv4 Statistics -> Source and Destination Addresses.

4) V danom súbore analyzujte komunikácie pre zadané protokoly:

- a) HTTP
- b) HTTPS
- c) TELNET
- d) SSH
- e) FTP riadiace

f) FTP dátové

g) TFTP, uveďte všetky rámce komunikácie, nielen prvý rámec na UDP port 69

h) ICMP, uveďte aj typ ICMP správy (pole Type v hlavičke ICMP), napr. Echo request, Echo

reply, Time exceeded, a pod.

i) Všetky ARP dvojice (request – reply), uveďte aj 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 rámcov ARP-Request na rovnakú IP adresu, vypíšte všetky.

Ak

sú v súbore rámce ARP-Request bez korešpondujúceho ARP-Reply (alebo naopak ARP-

Reply bez ARP-Request), vypíšte ich samostatne.

Implementácia:

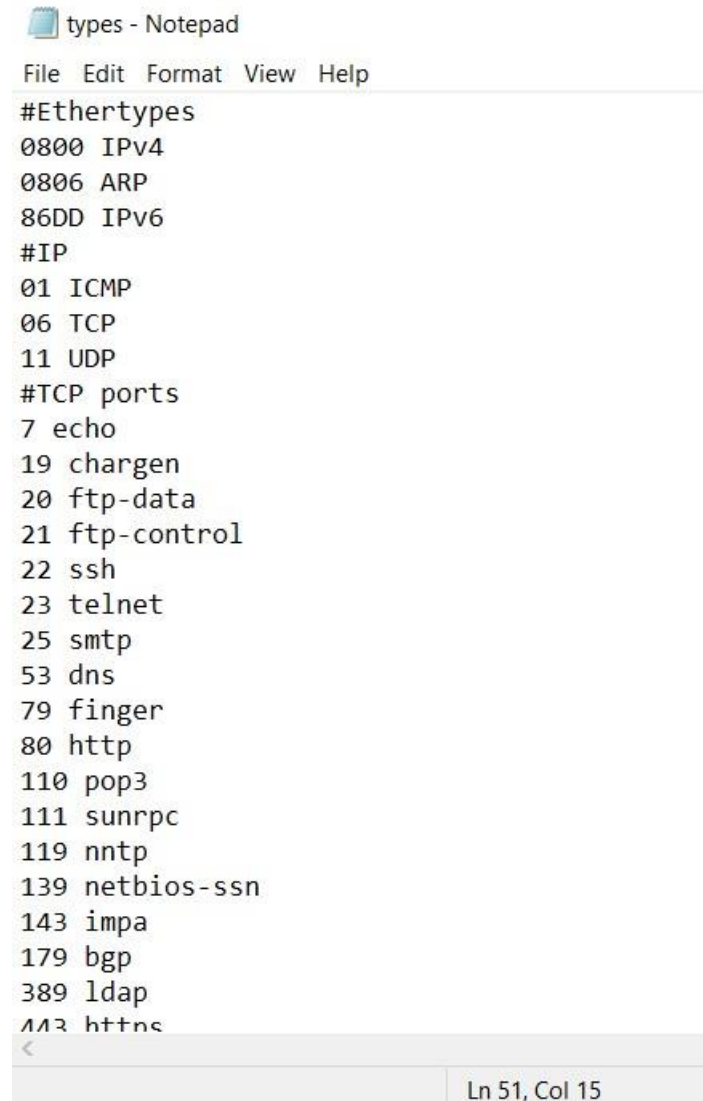
Začnem tým, že si daný .pcap súbor dekodujem a prepíšem všetky rámce do .txt súboru, ktorý budem potom analyzovať.

```
def uloz_pcap_do_txt():  
    paket = rdpcap(path)  
    textak = open('paket.txt', 'w')  
  
    for ramec in paket:  
        textak.write(hexlify(raw(ramec)).decode()+'\n')  
  
    textak.close()
```

Všetky potrebné hodnoty uchovávam v triede:

```
class a_ramec():  
    number: int  
    length_pcap_API: str  
    length_medium: str  
    payload: str  
    d_length: str  
    src_mac: str  
    dst_mac: str  
    type: str  
    protocol_2: str  
    protocol_3: str  
    protocol_4: str  
    src_port: str  
    dst_port: str  
    l_port: str  
    src_ipv4: str  
    dst_ipv4: str  
    druh: str
```

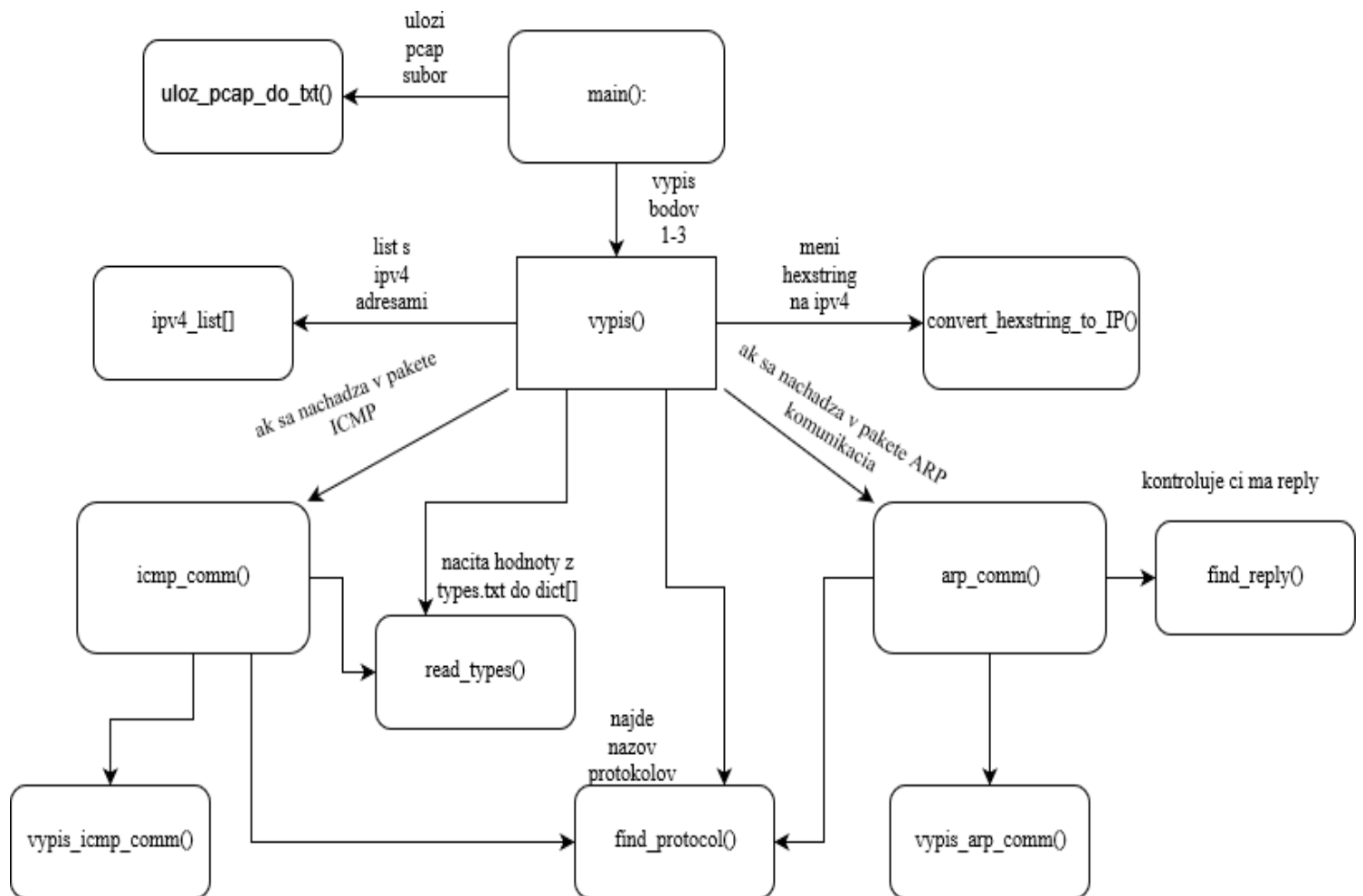
V súbore types.txt držím uchované hodnoty, ktoré potrebujem na identifikovanie daného protokolu/portu.



```
types - Notepad
File Edit Format View Help
#Ethertypes
0800 IPv4
0806 ARP
86DD IPv6
#IP
01 ICMP
06 TCP
11 UDP
#TCP ports
7 echo
19 chargen
20 ftp-data
21 ftp-control
22 ssh
23 telnet
25 smtp
53 dns
79 finger
80 http
110 pop3
111 sunrpc
119 nntp
139 netbios-ssn
143 imap
179 bgp
389 ldap
443 https
<
Ln 51, Col 15
```

Funkcia, ktorú volám nájde #, a podľa toho vráti hodnoty do dvojrozmerného poľa, odkiaľ ďalšia funkcia nájde názov podľa kódu tomu prislúchajúcemu.

Diagram:



Navrhnutý mechanizmus:

Uložiť hodnoty z rámcov do premenných, s ktorými potom neskôr pracujem (porovnávam ich, mením na str/int ..)

Dĺžku pcap API zistím jednoducho z funkcie len() , dĺžka media je dĺžka pcap API + FCS(4), minimálne 64 stále.

Najprv zistiť, či je to Ethernet II / IEEE 802.3 (ak je ethertype väčší ako 1500, je to Eth II, ak menší tak na základe DSAP/SSAP rozlišujem IEEE 802.30.

Na základe ethertypu zisťujem, či je to IPv4 (0x0800), IPv6(0x08dd), ARP(0x0806)

Pre IPv4, zisťujem aj IP adresy, protokol and tým : ICMP(0x01) TCP(0x06) UDP(0x11), a pre TCP a UDP aj porty.

Pre porty rozlišujem http/https/ssh/telnet/dns....

Ak sa v rámci nachádza ARP, tak sa pustí funkcia na výpis ARP komunikácii, kde jednoducho porovnávam IP adresy a hľadám či request má reply.

Ak sa nachádza ICMP , tak sa načíta typ ICMP zo súbora a vypíšu sa všetky takéto komunikácie.

Zvyšné body 4. Úlohy som nerobil.

Používateľské rozhranie:

Veľmi jednoduché, stačí len do path=' ' napísať cestu k vybranému .pcap súboru.

Zvyšok už prebieha samostatne z funkcie main(), kde prv sa zavolá funkcia na dekodovanie .pcap súboru, a potom funkcia vypis() spraví zvyšok.

Implementačné prostredie:

Zadanie som robil v python, konkrétne PyCharm, keďže mám zatiaľ veľmi dobré skúsenosti s JetBrains produktmi a nakoľko programujem v pythone prvýkrát, nechcel som robiť v niečom inom.

V Pythone to robím najmä kvôli spätnej väzbe od spolužiakov, že je to takto jednoduchšie keďže nemusím nijak ošetrovať pamäť/ pointre.

Záver:

Veľmi zaujímavé zadanie, vďaka ktorému som sa mnoho naučil o protokoloch, najmä teda samoštúdiom keďže na prednáškach takmer nič k tomu podstatné povedané nebolo, dokonca ani celé zadanie nebolo ešte prebrané na prednáškach a už ho odovzdávame.

Aj vďaka tomu som and tým strávil mnoho, ale že mnoho času a nestihol som dorobiť všetko, resp. nemal som ani chuť tráviť niekoľko x hodín , kvôli tým pár bodom.

Slabo hodnotené zadanie, pretože 15 bodov maximum je výsmech vzhľadom k času ktorý som tomuto venoval.