Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Počítačové a komunikačné siete
Analyzátor sieťovej komunikácie

Peter Plevko

2020/2021

a) zadanie úlohy,

POČÍTAČOVÉ A KOMUNIKAČNÉ SIETE

cvičenia

ak. rok 2020/21, zimný semester

Zadanie 1: **Analyzátor sieťovej komunikácie**

Zadanie úlohy

Navrhnite 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.

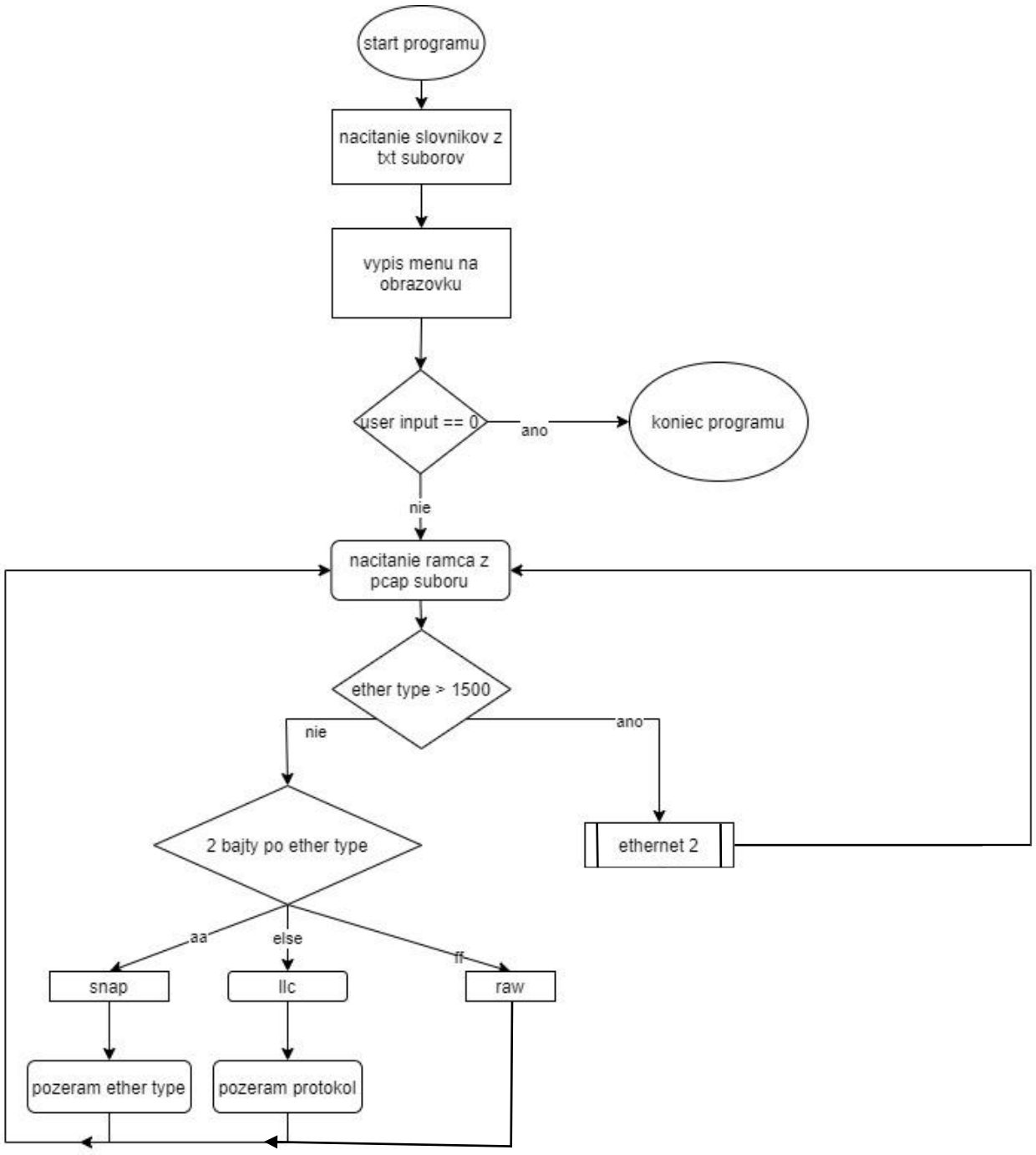
3) Analýzu cez vrstvy vykonajte pre rámce Ethernet II a protokoly rodiny TCP/IPv4:
Na konci výpisu z bodu 1) uveďte pre IPv4 pakety:
a) Zoznam IP adries všetkých prijímajúcich uzlov,
b) IP adresu uzla, ktorý sumárne prijal (bez ohľadu na odosielateľa) najväčší počet paketov a koľko paketov prijal (berte do úvahy iba IPv4 pakety).
IP adresy a počet poslaných paketov sa musia zhodovať s IP adresami vo výpise Wireshark -> Statistics -> IPv4 Statistics -> Source and Destination Addresses.

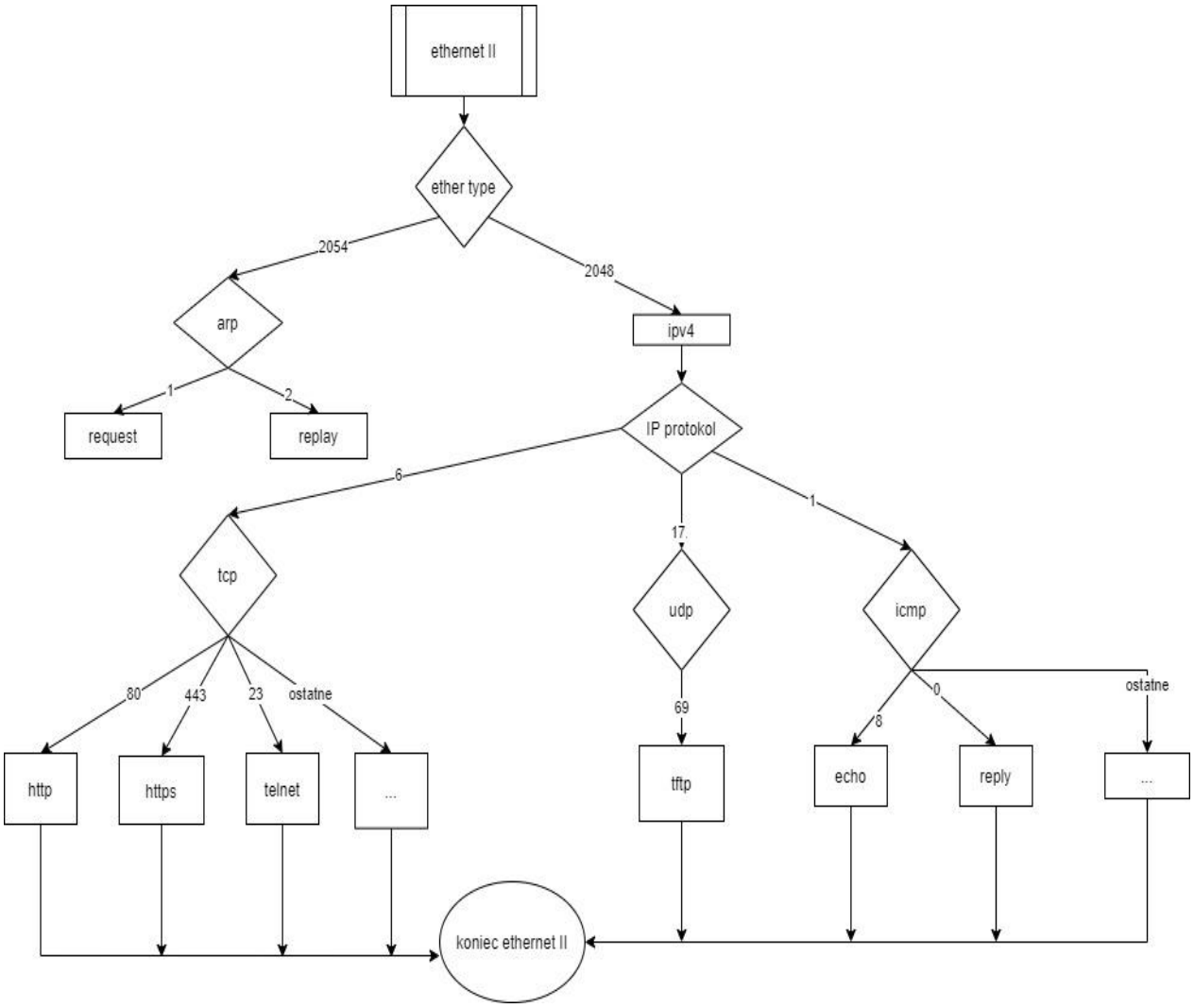
4) V danom súbore analyzujte komunikácie pre zadané protokoly:
a) HTTP
b) HTTPS
c) TELNET
d) SSH
e) FTP riadiace
f) FTP dátové
g) TFTP, **uveďte všetky rámce komunikácie**, nielen prvý rámec na UDP port 69
h) ICMP, uveďte aj typ ICMP správy (pole Type v hlavičke ICMP), napr. Echo request, Echo reply, 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 ARPReply bez ARP-Request), vypíšte ich samostatne.
Vo všetkých výpisoch treba uviesť aj IP adresy a pri transportných protokoloch TCP a UDP aj porty komunikujúcich uzlov.
V prípadoch komunikácií so spojením vypíšte iba jednu kompletnú komunikáciu - obsahuje otvorenie (SYN) a ukončenie (FIN na oboch stranách alebo ukončenie FIN a RST alebo ukončenie iba s RST) spojenia a aj prvú nekompletnú komunikáciu, ktorá obsahuje iba otvorenie spojenia. Pri výpisoch vyznačte, ktorá komunikácia je kompletná.
Ak počet rámcov komunikácie niektorého z protokolov z bodu 4 je väčší ako 20, vypíšte iba 10 prvých a 10 posledných rámcov tejto komunikácie. **(Pozor: toto sa nevzťahuje na bod 1, program**
musí byť schopný vypísať všetky rámce zo súboru podľa bodu 1.) Pri všetkých výpisoch musí byť
poradové číslo rámca zhodné s číslom rámca v analyzovanom súbore.
5) Program musí byť organizovaný tak, aby čísla protokolov v rámci Ethernet II (pole Ethertype), IEEE 802.3 (polia DSAP a SSAP), v IP pakete (pole Protocol), ako aj čísla portov v transportných protokoloch boli programom **načítané z jedného alebo viacerých externých textových súborov.** Pre známe protokoly a porty (minimálne protokoly v bodoch 1) a 4) budú uvedené aj ich názvy. Program bude schopný uviesť k rámcu názov vnoreného protokolu po doplnení názvu k číslu protokolu, resp. portu do externého súboru. Za externý súbor sa nepovažuje súbor knižnice, ktorá
je vložená do programu.
6) V procese analýzy rámcov pri identifikovaní jednotlivých polí rámca ako aj polí hlavičiek vnorených protokolov nie je povolené použiť funkcie poskytované použitým programovacím jazykom alebo knižnicou. **Celý rámec je potrebné spracovať postupne po bajtoch.**
7) Program musí byť organizovaný tak, aby bolo možné jednoducho rozširovať jeho funkčnosť výpisu rámcov pri doimplementovaní jednoduchej funkčnosti na cvičení.
8) Študent musí byť schopný preložiť a spustiť program v miestnosti, v ktorej má cvičenia. V prípade
dištančnej výučby musí byť študent schopný prezentovať podľa pokynov cvičiaceho program online, napr. cez Webex, Meet, etc.
V danom týždni, podľa harmonogramu cvičení, musí študent priamo na cvičení doimplementovať
do funkčného programu (podľa vyššie uvedených požiadaviek) ďalšiu prídavnú funkčnosť.
Program musí mať nasledovné vlastnosti (minimálne):
1) Program musí byť implementovaný v jazykoch C/C++ alebo Python s využitím knižnice pcap, skompilovateľný a spustiteľný v učebniach. Na otvorenie pcap súborov použite knižnice *libpcap* pre linux/BSD a *winpcap/ npcap* pre Windows. Použité knižnice a funkcie musia byť schválené cvičiacim. V programe môžu byť použité údaje o dĺžke rámca zo struct pcap_pkthdr a funkcie na prácu s pcap súborom a načítanie rámcov:
pcap_createsrcstr()
pcap_open()

pcap_open_offline()
pcap_close()
pcap_next_ex()
pcap_loop()
Použitie funkcionality *libpcap* na priamy výpis konkrétnych polí rámca (napr. ih->saddr) bude mať za následok nulové hodnotenie celého zadania.
2) Program musí pracovať s dátami optimálne (napr. neukladať MAC adresy do 6x int).
3) Poradové číslo rámca vo výpise programu musí byť zhodné s číslom rámca v analyzovanom súbore.
4) Pri finálnom odovzdaní, pre každý rámec vo všetkých výpisoch uviesť použitý protokol na 2. - 4. vrstve OSI modelu. (ak existuje)
5) Pri finálnom odovzdaní, pre každý rámec vo všetkých výpisoch uviesť zdrojovú a cieľovú adresu / port na 2. - 4. vrstve OSI modelu. (ak existuje)
Nesplnenie ktoréhokoľvek bodu minimálnych požiadaviek znamená neakceptovanie riešenia cvičiacim.

b) blokový návrh (koncepcia) fungovania riešenia,




ethernet II
ether type
2054
2048
arp
ipv4
1
2
request
replay
IP protokol
6
17
1
tcp
udp
icmp
80
443
23
ostatne
69
8
0
ostatne
http
https
telnet
...
tftp
echo
reply
...
koniec ethernet II

c) navrhnutý mechanizmus analyzovania protokolov na jednotlivých vrstvách,

Úloha 1-3

Po spustení programu inicializujem potrebne premenne načítam si všetky slovníky ktoré budem potrebovať z textových súborov, vypíšem užívateľské menu užívateľ si vyberie čo chce analyzovať otvorím súbor na výstup otvorím pcap súbor a začnem prechádzať rámce. V rámci skontrolujem ether type ak je väčší ako 1500 viem že sa jedna o ethernet II a vypíšem z tabuľky jeho ether type ak je menší viem že sa jedna o 802.3 už mi ostáva iba zistiť o aký 802.3 sa jedna toto Zistím podľa nasledujúcich dvoch bajtoch ak sa tam nachádza ff jedna sa o Novell 802.3 RAW (Novell proprietary) ak sa tam nachádza aa jedna sa o IEEE 802.3 LLC + SNAP pričom u tohto snapu určujem ešte jeho vnorený protokol podľa ethertypu ak sa tam nachádza niečo iné jedna sa o IEEE 802 . LLC a pozriem do tabuľky aký ma vnorený protokol. Moja analýza rámca spočíva vo výpise poradového čísla rámca jeho dĺžku pcap APi a jeho dĺžku prenášanú po médiu následne typ ci sa jedna o ethernet II alebo 802.3 a podrobnejšie o akú 802.3 jeho zdrojovú mac adresu jeho cieľovú mac adresu. Na konci vypíšem zoznam všetkých ip adries prijímacích uzlov a adresu uzla s najväčším počtom odoslaných packetov s jeho ip adresou a počtom packetov ktoré poslal.

Úloha 4 a, b, c, d, e, f

Analýza rámcov v týchto bodoch spočíva v tom že sa pozriem ci sa jedna o ethernet II ak áno pozriem sa na IPv4 ak sa jedna o protokol TCP pozriem sa na jeho cieľový port ak sa tato hodnota rovná s niektorým z týchto hodnôt analyzujem to ďalej podľa toho čo zadal užívateľ.

20:ftp-data

21:ftp-control

22:ssh

23:telnet

80:http

443:https (ss1)

Užívateľ si na začiatku vybral ktorý well known port chce analyzovať, rámce tohto portu si roztriedim do jednotlivých komunikácii podľa zdrojových a cieľových. Ďalej riešim len komunikácie ktoré sa správne začali ostatne neberiem do úvahu . Správne začaté komunikácie majú flag v prvom rámci nastavený na :

1 rámec syn v desiatkovej je to hodnota flagu 2

2 rámec syn ack v desiatkovej je to hodnota flagu 18

3 rámec ack v desiatkovej je to hodnota flagu 2

Už viem čo je to dobre začatá komunikácia už potrebujem pozrieť iba ci sa jedna o kompletnú komunikáciu a o takúto komunikáciu ide ak sa dobre začala ale aj dobre skončila komunikácia sa môže skončiť rôznymi spôsobmi a tieto spôsobí sú :

1 spôsob:

Posledný rámec ma flag nastavený na rst čo je v desiatkovej hodnota 4

2: spôsob:

Posledný rámec ma flag nastavený na rst ack čo je v desiatkovej hodnota 20

3 spôsob:

Posledný rámec ma flag nastavený na ack v desiatkovej hodnota 16

Predposledný rámec ma flag nastavený na fin ack v desiatkovej je to hodnota 17

Pred predposledný ma hodnotu fin ack v desiatkovej je to hodnota 17

4 spôsob:

Posledný rámec ma flag nastavený na ack v desiatkovej je to hodnota 16

Predposledný rámec ma flag nastavený na fin ack v desiatkovej je to hodnota 17

Pred predposledný rámec ma flag nastavený na ack v desiatkovej je to hodnota 16

Pred pred pred posledný rámec ma flag nastavený na fin ack v desiatkovej je to hodnota 17

V tomto momente viem ci je moja komunikácia kompletná alebo nie a mojou úlohou je vypísať jednu Kompletnú a jednu nekompletnú komunikáciu ak ma tato komunikácia však viac ako 20 rámcov tak vypíšem prvých 10 a posledných desať.

Úloha 4g (TFTP)

Sledujem ci sa vnorený protokol ipv4 je UDP ak áno pozerám ci destination port == 69 ak áno začína sa mi komunikácia, všetky komunikácie si roztriedim podľa zdrojových a cieľových ip

adries. Tieto komunikácie následne prechádzam a pozerám opcode ak v ňom nájdem hodnotu 5 viem že ide o error tento rámec ešte pridám do komunikácie avšak uzatváram ju a ďalšie rámce prechádza a ignorujem dokiaľ nepríde ďalšia 69. Výpis sa nelíši od úlohy 4a iba v tom že vypisujem dvojice a číslo komunikácie.

Úloha 4h (ICMP)

Sledujem ci sa jedno o ip protokol ICMP ak áno zase si to roztriedim podľa ip adries a v tejto úlohe je mojou úlohou naparovať icmp rámce pozriem sa na type a hľadám dvojice echo (ping) request a echo (ping) reply.

Výpis sa nelíši od úlohy 4a okrem toho že robím dvojice.

Úloha 4i (ARP)

Ak sa jedna o ethernet II a jeho vnorený je protokol arp musím nájsť dvojice request reply. Komunikácie triedim podľa ip zdrojovej a cieľovej a mac adresy. AK mam jeden request a nájdem jeden reply považujem komunikáciu za uzavretý. Môžu mi však nastať aj krajne prípady to znamená napríklad 2 requesty a jeden reply takúto považujem tiež za uzatvorenú a musím vypísať všetky requesty. Keď sa mi stane že budem mat v komunikácii napríklad len jeden request alebo reply tak nemá par. Výpis sa líši od 4a v tom že vypisujem pary a mam pridaný výpis ip adresy a mac adresy.

d) príklad štruktúry externých súborov pre určenie protokolov a portov,

takto vyzerá môj textový súbor pre tcp

7:echo

19:chargen

20:ftp-data

21:ftp-control

22:ssh

23:telnet

25:smtp

53:domain

79:finger

80:http

110:pop3

111:sunrpc

119:nntp

139:netbios-ssn

143:imap

179:bgp

389:ldap

443:https (ss1)

445:microsoft-ds

1080:socks

e) opísané používateľské rozhranie,

```
Analyzator packetov
Autor: Peter Plevko
priklad cesty: vzorky_pcap_na_analyzu/trace-27.pcap
zadaj cestu k suboru :
```

Program si následne vypýta cestu ku súboru ktorý chcem otvoriť

```
Stlac 1 pre vypis do suboru a hocico ine pre vypis na monitor:
```

Následne sa používateľa opýtam ci chce mat výpis na obrazovku alebo do súboru

```
Stlac 1 pre vypis do suboru a hocico ine pre vypis na monitor: 2
Zadaj 0 pre ukoncenie programu
Zadaj 1 pre vypis uloh 1-3
Zadaj 4a pre analyzu HTTP
Zadaj 4b pre analyzu HTTPS
Zadaj 4c pre analyzu TELNET
Zadaj 4d pre analyzu SSH
Zadaj 4e pre analyzu FTP riadiace
Zadaj 4f pre analyzu FTP datove
Zadaj 4g pre analyzu TFTP
Zadaj 4h pre analyzu ICMP
Zadaj 4i pre analyzu ARP
Zadaj vyber ulohy:
```

Po zadaní ci chcem výstup do súboru alebo na obrazovku sa nás program opýta čo chceme konkrétne chceme analyzovať.

f) voľbu implementačného prostredia.

Môj program som programoval v pythone v ide éčku od jetbrains pyCharm. Toto prostredie som si zvolil kvôli tomu aby som si ušetril prácu s bajtami využil som vstavane funkcie pythonu ako napríklad binascii hexlify následne som použil scapy na načítanie rámcov packetov.