Počítačové a komunikačné siete

**Analyzátor sieťovej komunikácie**

Emma Macháčová

**Meno cvičiaceho** : Ing. Lukáš Mastiľak

**Čas cvičení** : Štvrtok 16:00

**Dátum vytvorenia** : 17. Okt. 2021

Obsah

[Cieľ práce 1](#_Toc85664710)

[Implementácia a použité knižnice 1](#_Toc85664711)

[Načítanie externého súboru 1](#_Toc85664712)

[Používateľské rozhranie 2](#_Toc85664713)

[Globálne (statické) premenné - main 3](#_Toc85664714)

[Bod 1 – výpis všetkých rámcov 3](#_Toc85664715)

[Bod 2 – výpis vnorených protokolov 4](#_Toc85664716)

[Bod 3 – analýza protokolov rodiny TCP/IPv4 5](#_Toc85664717)

[Bod 4 – všeobecné informácie 5](#_Toc85664718)

[Bod 4.a až 4.f – analýza TCP 6](#_Toc85664719)

[Bod 4.g – analýza protokolov TFTP 7](#_Toc85664720)

[Bod 4.h – analýza ICMP správ 8](#_Toc85664721)

[Bod 4.i – analýza protokolov ARP 8](#_Toc85664722)

[Ukážky výpisu 9](#_Toc85664723)

[Bod 1 až 3 9](#_Toc85664724)

[Ethernet II – IPv4 9](#_Toc85664725)

[Ethernet II – IPv6 10](#_Toc85664726)

[Ethernet II – ARP 11](#_Toc85664727)

[Ethernet II – iné 12](#_Toc85664728)

[IEEE 802.3 13](#_Toc85664729)

[Bod 4 14](#_Toc85664730)

[TCP komunikacie 14](#_Toc85664731)

[TFTP komunikácie 15](#_Toc85664732)

[ICMP komunikácie 15](#_Toc85664733)

[ARP komunikácie 17](#_Toc85664734)

Cieľ práce

Cieľom je navrhnutie a implementácia programového analyzátora Ethernet siete, ktorý analyzuje komunikácie v sieti zaznamenané v .pcap súbore a poskytuje nasledujúce informácie o týchto komunikáciách:

* Výpis všetkých rámcov v hexadecimálnom tvare postupne tak, ako boli zaznamenané v súbore.
  + Poradové číslo rámca v analyzovanom súbore.
  + Dĺžku rámca v bajtoch poskytnutú pcap API, ako aj dĺžku tohto rámca prenášaného po médiu.
  + Typ rámca – Ethernet II, IEEE 802.3 (IEEE 802.3 s LLC, IEEE 802.3 s LLC a SNAP, IEEE 802.3 – Raw).
  + Zdrojovú a cieľovú fyzickú (MAC) adresu uzlov, medzi ktorými je rámec prenášaný.
* Výpis vnorených protokolov pre rámce typu Ethernet II a IEEE 802.3
* Analýzu IP adries a počtu odoslaných packetov
* Analýzu komunikácií zadaných protokolov

Implementácia a použité knižnice

Program je implementovaný v jazyku Python verzie 3.9 s využitím knižnice Scapy, z ktorej využíva funkciu na načítanie .pcap súboru *rdpcap(pathname)*. Na nasledovnú analýzu jednotlivých polí a hlavičiek rámcov nevyužíva iné funkcie poskytnuté knižnicou alebo programovacím jazykom.

Načítanie externého súboru

Program je organizovaný tak, že 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 sú programom načítané z jedného externého textového súboru.

Názvy a čísla portov sa pri spustení načítajú funkciou *nacitajSubor(...)* do slovníkov, kde sú uložené po celý beh programu. Ku každému názvu v súbore začínajúcemu znakom # patrí jeden slovník.

Ukážka externého súboru:

#Ethertypes  
0800 IPv4  
0806 ARP  
0842 Wake-on-LAN  
22f0 AVTP  
86dd IPv6  
88cc Link Layer Discovery Protocol  
9000 Loopback  
#LSAPs  
42 STP  
aa SNAP  
e0 IPX  
ff IPX  
#IPProtocolNumbers  
01 ICMP  
02 IGMP  
06 TCP  
11 UDP  
67 PIM  
3a ICMPv6  
#TCPs  
0007 ECHO  
0013 CHARGEN  
0014 FTP-DATA  
0015 FTP-CONTROL  
0016 SSH  
0017 TELNET  
0050 HTTP  
01bb HTTPs  
#UDPs  
0035 DNS  
0045 TFTP  
0208 Route Info Protocol  
0089 NetBIOS Name Service  
008a NetBIOS Datagram Service  
076c Simple Service Discovery Protocol  
14eb Link-local Multicast Name Resolution  
#ICMPs  
00 Echo Reply  
03 Destination Unreachable  
04 Source Quench  
05 Redirect  
08 Echo Request  
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

Používateľské rozhranie

Používateľské rozhranie funguje v rámci konzoly, program sa ovláda vstupmi z klávesnice. Menu pozostáva z dvoch hlavných možností – prvej časti (body zadania 1-3) a druhej (bod 4). Okrem toho je možné vypísať obsah externého súboru (resp. už naplnených slovníkov) a tiež ukončiť program.

Obrázok, na ktorom je text

Automaticky generovaný popisObrázok, na ktorom je text

Automaticky generovaný popisPo zvolení funkcie si program buď vypýta ďalšie upresnenie, alebo názov .pcap súboru, ktorý sa má spracovať. Program tiež kontroluje správnosť vstupov (názov súboru aj písmeno označujúce pokyn), a pri nesprávnom vstupe si vyžiada opätovné zadanie informácie.

Globálne (statické) premenné - main

Program pre ľahkú prehľadnosť a zmeniteľnosť (vyhnutie sa priamemu umiestňovaniu čísiel do kódu) využíva globálne premenné v triede *glob*, v ktorých sú uložené začiatočné (a konečné) pozície analyzovaných polí jednotlivých typov rámcov.

Obrázok, na ktorom je text

Automaticky generovaný popisObrázok, na ktorom je text

Automaticky generovaný popisObrázok, na ktorom je text

Automaticky generovaný popisTaktiež pre lepšiu čitateľnosť pracuje program s triedou farby (farebné odlíšenie výpisov).

Bod 1 – výpis všetkých rámcov

Súbor vypíše všetky rámce v hexadecimálnom tvare postupne tak, ako boli zaznamenané v súbore. Pre každý rámec uvedie:

* poradové číslo rámca v analyzovanom súbore,
* dĺžku rámca v bajtoch poskytnutú pcap API, ako aj dĺžku tohto rámca prenášaného po médiu,
* typ rámca,
* zdrojovú a cieľovú fyzickú (MAC) adresu uzlov, medzi ktorými je rámec prenášaný.

Táto časť programu je sústredená v súbore *zadanie1.py*, a využíva najmä funkcie *start(...), get\_typ\_ramca(...), get\_src\_mac(...), get\_dest\_mac(...), print\_ramec(...) a pkt\_print(...)*, ktoré som implementovala.

Vo funkcii start sa program pokúsi otvoriť súbor – ak sa mu to podarí prečíta jeho dĺžku (počet rámcov v súbore), a pre každý z rámcov spustí funkciu *pkt\_print*. Po vykonaní všetkých iterácií spustí funkciu pre IP štatistiku (viď. Bod 3).

Funkcia *pkt\_print* postupne spracuje jednotlivé rámce súboru (vždy po jednom). Vypíše poradové číslo rámca (ktoré dostáva ako argument funkcie), vypočíta a vypíše dĺžku rámca (dĺžku API ako len(ramca), a dĺžku prenášanú po médiu buď ako minimálnu dĺžku 64 B, alebo dĺžku API spolu s hlavičkou 4 B).

Následne volanou funkciou *get\_typ\_ramca* program zistí, či je rámec typu Ethernet II alebo niektorý z typov IEEE 802.3. Funkciami *get\_src\_mac* a *get\_dest\_mac* zistí MAC adresy (podľa príslušných polí) a funkciou *getVnorenyProtokol* zistí vnorené protokoly a IP adresy (viď. Bod 2).

Ako posledné vypíše celý prenášaný rámec funkciou *print\_ramec.*

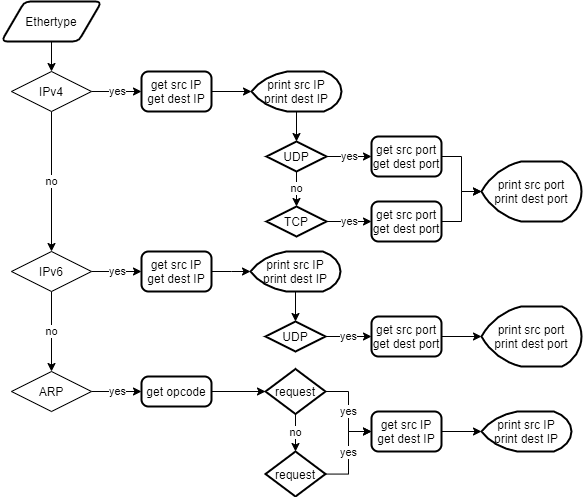
Bod 2 – výpis vnorených protokolov

Funkcie pre výpis vnorených protokolov sú v súbore zadanie2\_3.py, a to napríklad funkcie *ipv4\_getSrcIp(...),* *ipv6\_getSrcIp(...)* alebo *getVnorenyProtokol(...).*

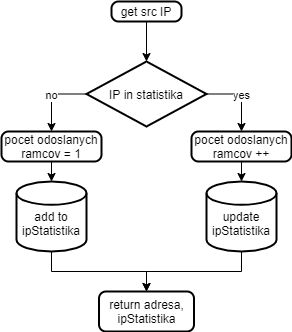
Tieto funkcie sú volané v predošlom bode funkciou *pkt\_print* pre individuálne rámce súboru.

Funkcia *getVnorenyProtokol* dostane ako argument typ rámca (zisťovaný už v Bode 1 – Ethernet II alebo IEEE 802.3), a na základe toho sa pozrie na príslušné pole v rámci (pole Ethertype pre Ethernet II, a polia DSAP a SSAP pre IEEE 802.3). Prečítané informácie porovná so slovníkom obsahujúcim čísla a názvy protokolov – v prípade zhody vráti názov protokolu, inak vráti číslo/označenie portu.

Podľa zistenej informácie o vnorenom protokole ďalej analyzuje rámec – pre IPv4, IPv6 a ARP. Pre rámce typu IPv4 a IPv6 skúma zdrojové a cieľové IP adresy funkciami *getSrcIp* a *getDestIp* (ktoré sú rozdielne podľa toho, či ide o IPv4 alebo IPv6) a skúma tiež protokol UDP (a TCP pre IPv4) funkciami *ipv4\_getVnorenyProtokol* a *ipv6\_getVnorenyProtokol*. Pri ARP zisťuje, či ide o typ Request alebo Reply, a taktiež skúma IP adresy.



Bod 3 – analýza protokolov rodiny TCP/IPv4

Pri využívaní funkcie *ipv4\_getSrcIp* v Bodoch 1 a 2 sa zaznamenané IPv4 adresy uložia do slovníka *ipStatistika* spolu s počtom rámcov, ktoré odoslali. Ak daná IPv4 adresa v slovníku nie je, uloží sa s počtom rámcov 1. Ak sa už v slovníku nachádza, počet rámcov sa inkrementuje.

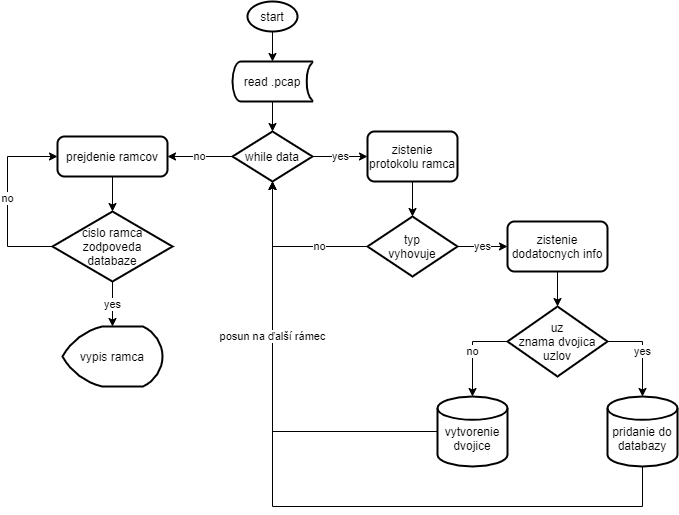
Funkcia *vypisIPstatistuku(...)* následne prejde slovník *ipStatistika* a vypíše všetky kľuče (IPv4 adresy odosielajúcich uzlov). Podľa hodnôt týchto kľúčov vie určiť IP adresu uzla, ktorý sumárne bez ohľadu na prijímateľa odoslal najväčší počet paketov.

Bod 4 – všeobecné informácie

Funkcie k tomuto bodu sú sústredené v súbore zadanie4.py. Obsahuje pomocné funkcie na prevod medzi sústavami (resp. na formátovanie adries), funkcie na finálne výpisy pre už analyzovaný súbor, a dve hlavné funkcie – *start* a *vyberProtokol*.

Funkcia *start* slúži obdobne ako pri bodoch 1, 2 a 3. Program načíta súbor a postupne pre každý rámec volá funkciu *vyberProtokol*, ktorá daný rámec analyzuje. Výstupom tejto analýzy je slovník, ktorý obsahuje informácie o komunikáciach – komunikujúce dvojice (adresy) ako key, a zoznam poradových čísiel rámcov so správnym typom komunikácie ako value.

Na koniec sa volá funkcia pre výpis (podľa zadaného vstupu), ktorá vypíše rámce zodpovedajúce poradovému číslu v slovníku.



Bod 4.a až 4.f – analýza TCP

Analýza všetkých protokolov rodiny TCP prebieha rovnako.

Vo funkcii *vyberProtokol* sa postupne prejdú všetky rámce, a tie, ktorých protokol sa zhoduje s tým hľadaným sa ďalej analyzujú.

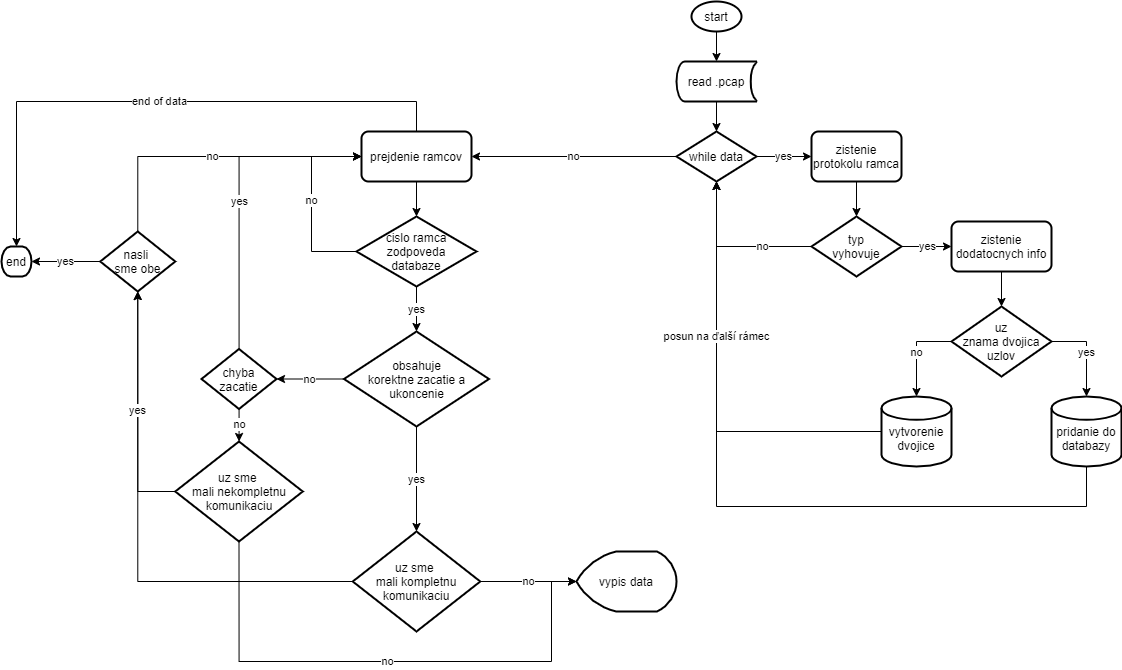
Podľa IP adries a portov komunikujúcich uzlov sa vytvorí ID. To potom program hľadá medzi už známymi uzlami. Ak ho nenájde, vytvorí nový záznam. Ak ho nájde, pridá číslo rámca do databázy.

Po vytvorení databázy dochádza ešte k filtrovaniu kompletných a nekompletných komunikácií vo funckií *vypisTCP*. Táto funkcia má zo všetkých záznamov vypísať iba prvú kompletnú a prvú nekompletnú komunikáciu.

Zo zadania, kompletná komunikácia je začatá otvorením SYN na oboch stranách, a ukončená FIN na oboch stranách, FIN a RST, alebo iba RST. Nekompletná neobsahuje ukončenie. Kompletná aj nekompletná komunikácia musí byť pritom začatá korektne.

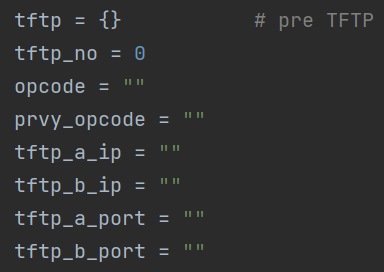
Funkcia v rámcoch komunikácie skontroluje začiatok a hľadá ukončenie. Podľa toho, čo nájde (resp. nenájde) nastaví flagy, ktoré kontrolujú výpis.

Ak sa v súbore nenájde kompletná alebo nekompletná komunikácia, program o tom informuje.

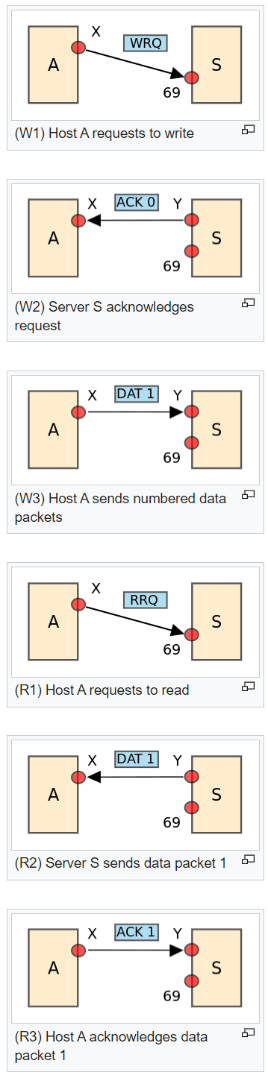
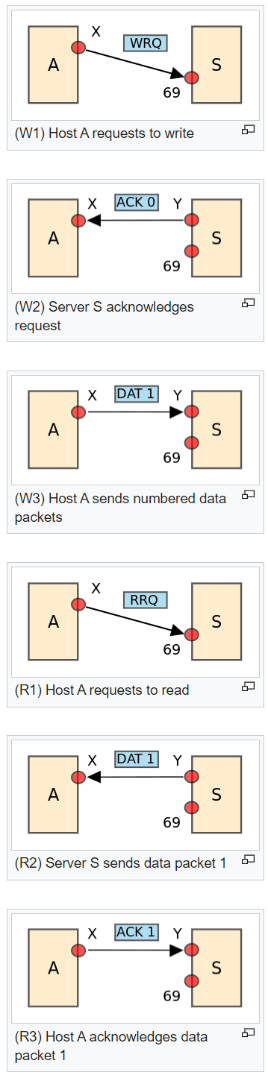


Bod 4.g – analýza protokolov TFTP

Pre analýzu protokolov TFTP využívam tieto globálne premenné v triede *kom* v súbore *zadanie4.py*:

* Slovník *tftp*, v ktorom ukladám komunikujúce uzly (ako key) a zoznam obsahujúci poradové čísla rámcov, v ktorých som našla TFTP komunikáciu (ako value).
* Premenná *tftp\_no* označuje priebežne poradové číslo TFTP komunikácie.
* Premenná *opcode* slúži ako história pre predchádzajúci opcode pred tým, ktorý sa aktuálne spracúva (opcode predchádzajúceho rámca TFTP komunikácie).
* Premenná *prvy\_opcode* značí prvý opcode aktuálnej komunikácie.
* Premenné *tftp\_a\_ip* a *tftp\_a\_port* nesú informácie o uzly ktorý inicializoval komunikáciu, premenné *tftp\_b\_ip* a *tftp\_b\_port* zasa informácie o adresátovi.

Funkcia *vyberProtokol* zistí pomocou pomocných funkcií vnorený protokol rámca (IPv4 -> UDP -> TFTP), a ak nájde prvý rámec na port 69:

* inkrementuje poradové číslo TFTP komunikácie,
* zistí IP adresy komunikujúcich uzlov,
* vytvorí z týchto údajov ID pre slovník,
* nastaví hodnoty pre globálne premenné prvy\_opcode, opcode a lokálnu premennú opcode,
* uloží hodnoty portov do globálnych premenných,
* a na záver pridá ID komunikácie a poradové číslo rámca do slovníka.

Ak nájde iný rámec UDP:

* zistí, či tento rámec patrí do TFTP komunikácie podľa IP adries a portov (až v druhej správe zistí port uzlu b, a ten uloží pre ďalšie kontrolovanie),
* vytvorí ID tak, aby sa komunikácie správne spárovali (správa z uzla a -> b patrí do tej istej komunikácie ako z b -> a),
* skontroluje, či je správna následnosť opkódov (ako na obrázku, prípadne error code),
* aktualizuje slovník – pre dané ID pridá do zoznamu poradové číslo rámca.

Na záver (po tom ako sa zanalyzujú všetky rámce a priradia do komunikácie) sa správne rámce vypíšu pomocou funkcie *vypisTFTP*.

Bod 4.h – analýza ICMP správ

Pre analýzu protokolov ICMP využívam globálnu premennú - slovník *icmp* - v triede *kom* v súbore *zadanie4.py*. Slovník *icmp* ukladá komunikujúce uzly (ako key) a zoznam obsahujúci poradové čísla rámcov, v ktorých som našla ICMP komunikáciu (ako value).

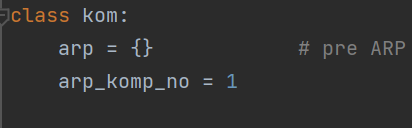
Funkcia *vyberProtokol* zistí pomocou pomocných funkcií vnorený protokol rámca (IP -> ICMP), a ak nájde takýto protokol:

* zistí IP adresy,
* vytvorí z nich ID,
* vytvorí nový záznam v slovníku alebo aktualizuje slovník o poradové číslo rámca pre danú dvojicu uzlov.

Vo výpise (funkcia *vypisICMP*) sa okrem iného zisťuje typ ICMP správy (Echo Request, Echo Reply...). Tieto hodnoty má program z externého súboru.

Bod 4.i – analýza protokolov ARP

Pre analýzu protokolov ARP využívam tieto globálne premenné v triede *kom* v súbore *zadanie4.py*:

* Slovník *arp*, v ktorom ukladám komunikujúce uzly (ako key) a zoznam obsahujúci poradové čísla rámcov, v ktorých som našla TFTP komunikáciu (ako value).
* Premennú *arp\_komp\_no*, ktorá uchováva počet kompletných komunikácií (poradové číslo kompletnej komunikácie sa využíva pre párovanie komunikácií)

Funkcia *vyberProtokol* zistí pomocou pomocných funkcií vnorený protokol rámca (Ethernet -> ARP), a ak nájde ARP protokol:

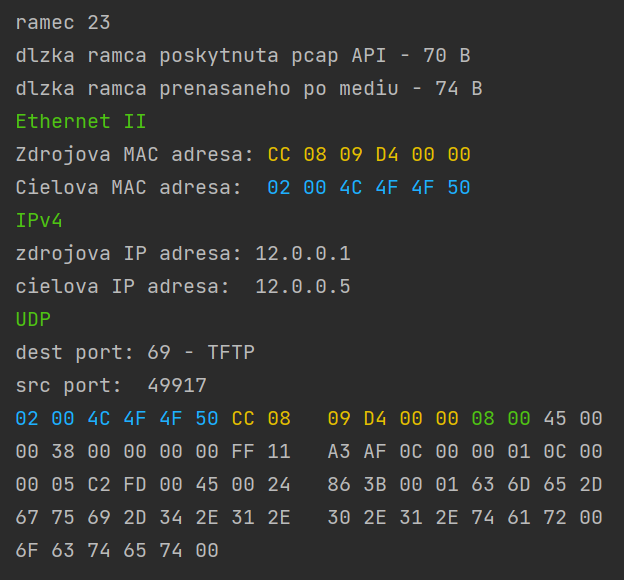
* zistí opcode a IP adresy,
* vytvorí z nich ID,
* ak dvojicu nepozná:
  + vytvorí nový záznam,
* ak dvojicu pozná:
  + ak ide o Reply a našli sme kompletnú komunikáciu
    - pridá rámec ako posledný pre danú komunikáciu uzlov
    - zmení ID uzavretej komunikácie, aby sa do nej nepridávali ďalšie prípadné správy medzi uzlami
  + ak je to ďalší Request pridá ho k predchádzajúcim

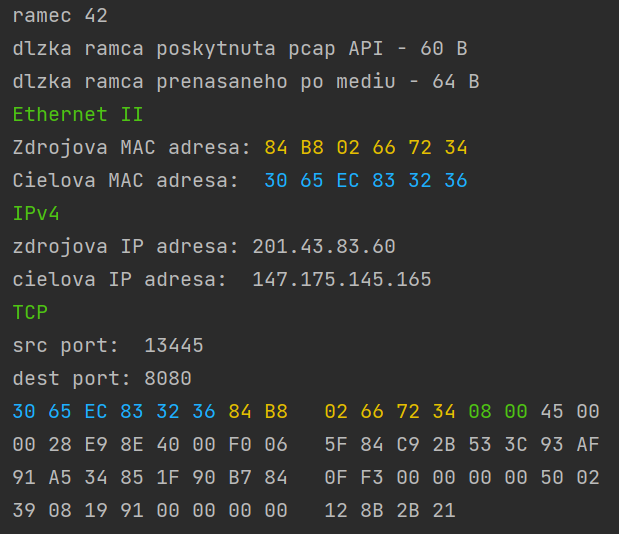
Po vykonaní tejto funkcie sa na záver volá výpis – *vypisARP*.

Ukážky výpisu

Bod 1 až 3

Ethernet II – IPv4





Obrázok, na ktorom je text, elektronika

Automaticky generovaný popis

Ethernet II – IPv6

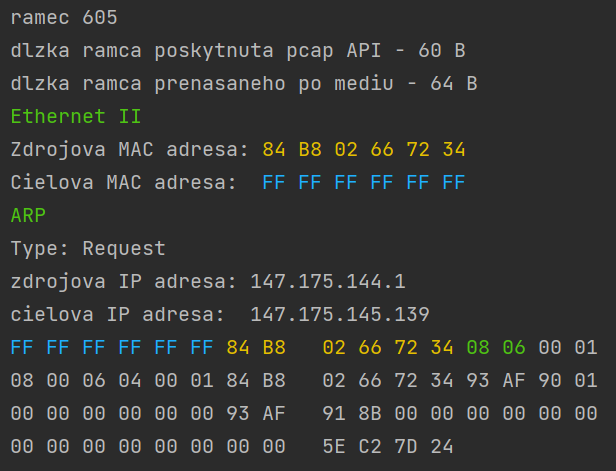
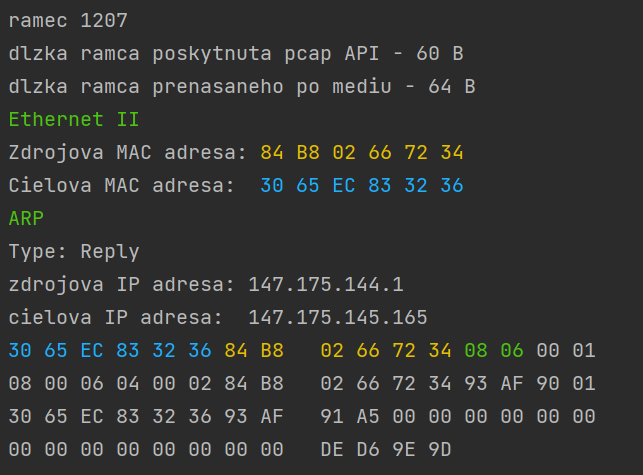
Obrázok, na ktorom je text

Automaticky generovaný popis

Obrázok, na ktorom je text

Automaticky generovaný popis

Ethernet II – ARP

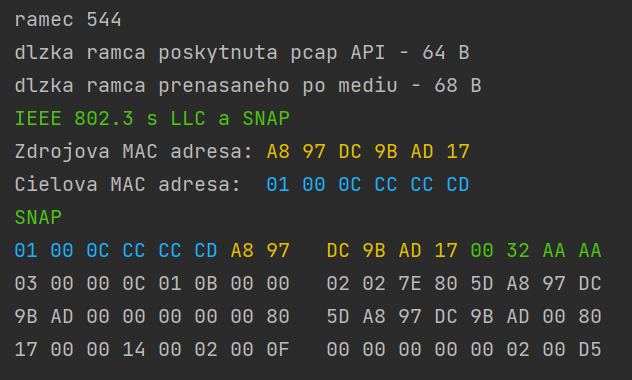
 

Ethernet II – iné

Obrázok, na ktorom je text, elektronika, klávesnica

Automaticky generovaný popis

IEEE 802.3



Obrázok, na ktorom je text, elektronika

Automaticky generovaný popis

Bod 4

TCP komunikacie

Obrázok, na ktorom je text

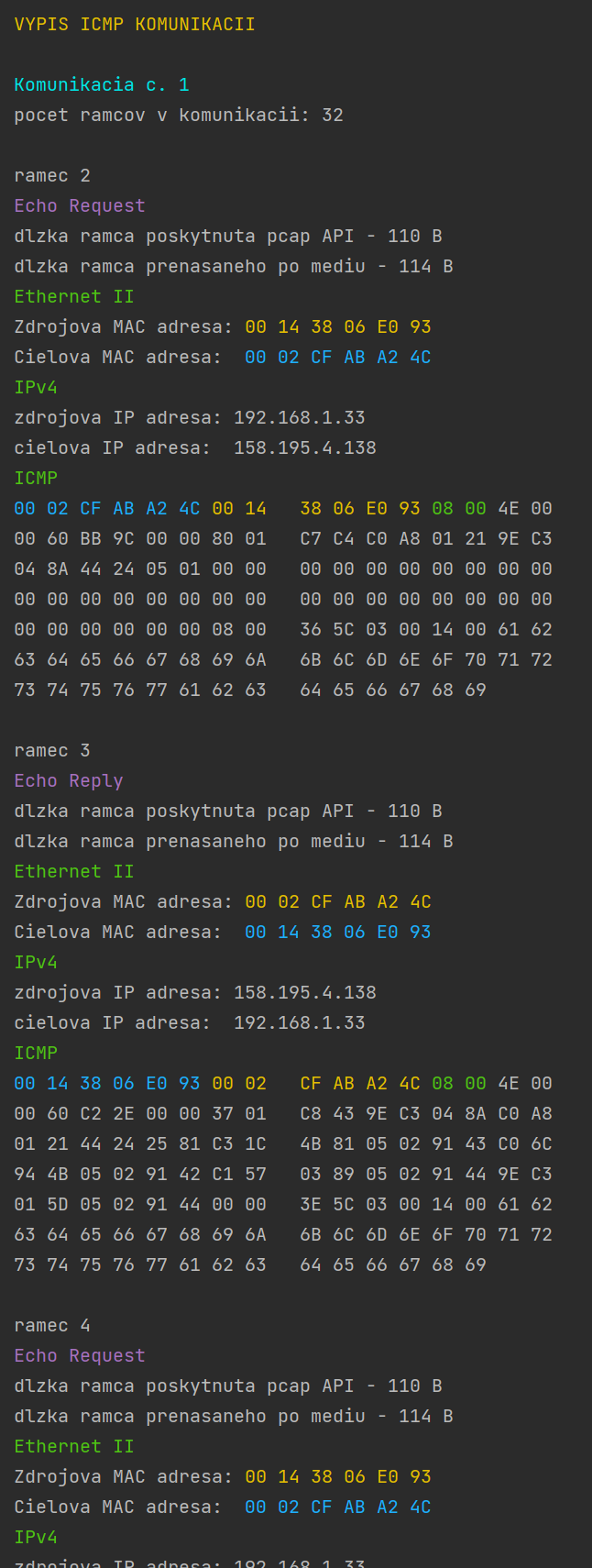
Automaticky generovaný popis

TFTP komunikácie

Obrázok, na ktorom je text

Automaticky generovaný popis

ICMP komunikácie



ARP komunikácie

Obrázok, na ktorom je text

Automaticky generovaný popis