Slovenská technická univerzita

Fakulta infromatiky a infromačných technológií

Ilkovičova 3, 842 19 Bratislava 4

Zadanie č. 1 analyzátor sieťovej komunikácie

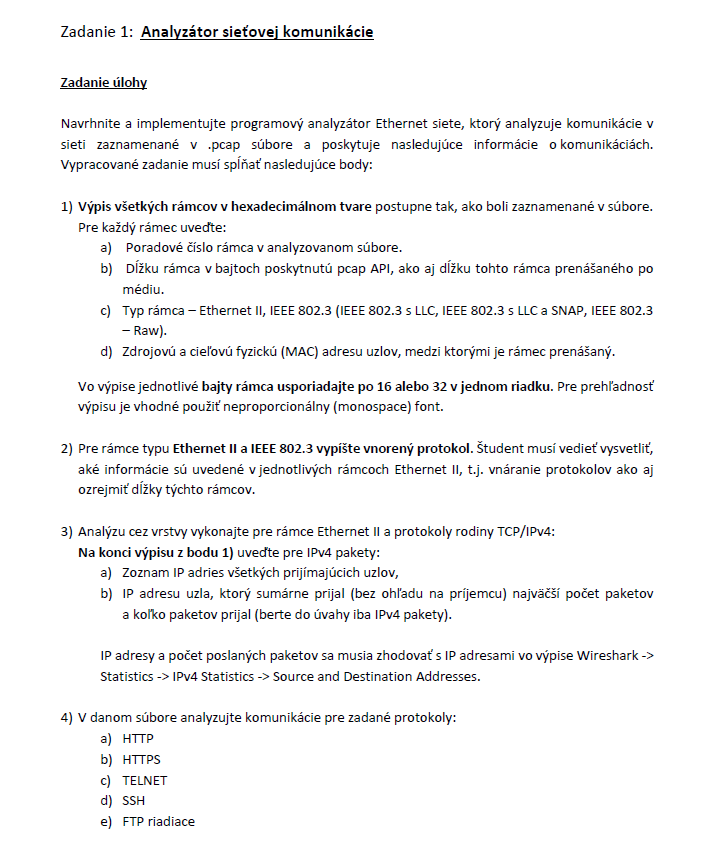
Autor: Michal Greguš

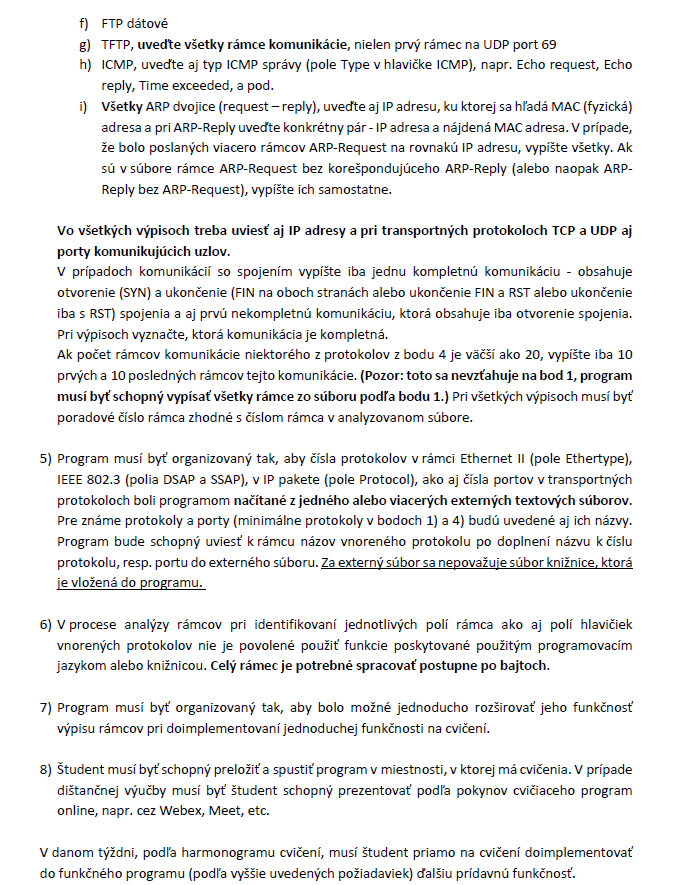
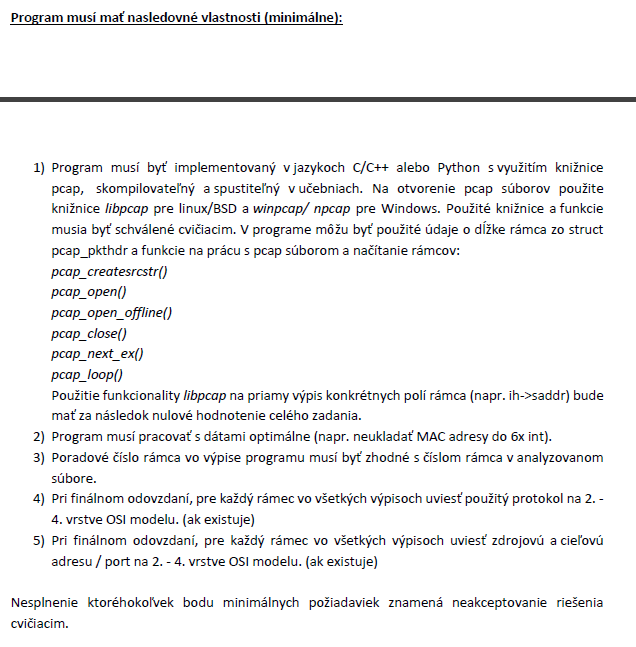
Predmet: Počítačové Komunikačné siete

Cvičiaci: Ing. Rastislav Bencel, PhD.

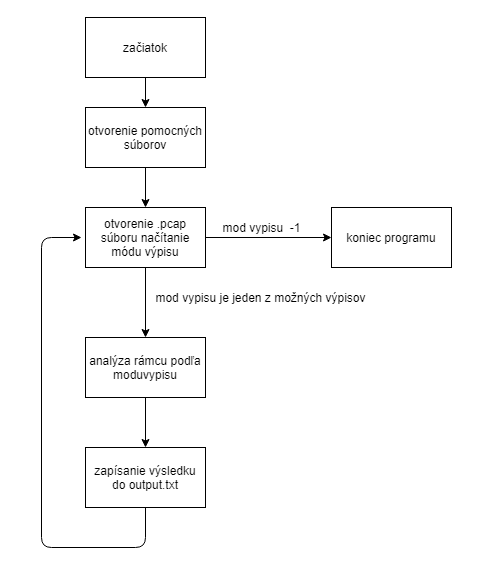
Cvičenie: Štvrtok 8:00

# Zadanie úlohy

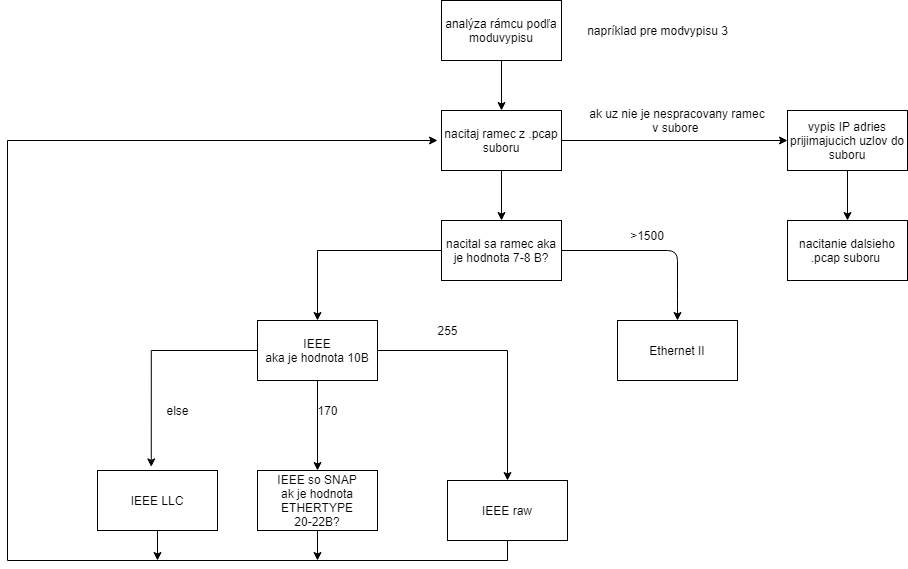


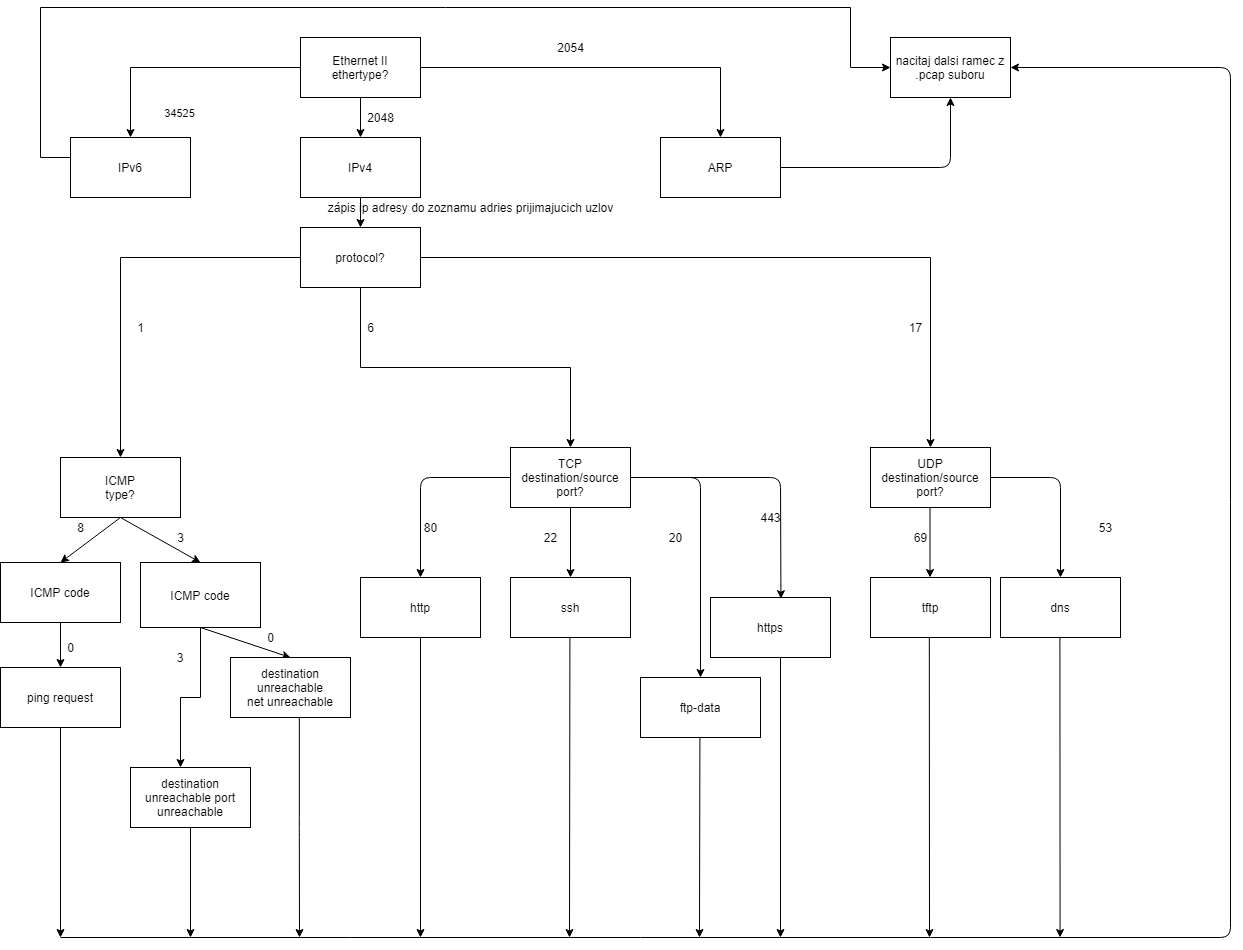
# Blokový návrh programu a koncepcia fungovanie riešenia



Rozhodovanie o type na 2 vrstve



Analýza ethernet rámca cez 3-4 vrstvu



# Mechanizmus analyzovanie rámcov

Program na začiatku otvorí zvolený .pcap súbor a následne ho analyzuje tak, aby výsledný výpis zodpovedal požadovanému výpisu. V zásade pre jednotlivé výpisi 4-14 si najskôr prejdem celý súbor aby som si poznačil potrebné údaje pre výpis. A v druhom prechode vypisujem potrebné rámce.

Jednotlivé rámce v súbore analyzujem postupne po bajtoch, pričom potrebné resp. pre analýzu významné bajty si ukladám do príslušných štruktúr a ich polí určených pre tieto údaje. Tieto štruktúry boli vytvorené na mieru pre jednotlivé protokoly podľa ich hlavičiek.

**Opis procesu spracovania jedného rámcu:**

Na začiatku zo štruktúry typu pcap\_pkthdr zistím dľžku rámca, a následne vypočítam reálnu dľžku rámca prenášaného po médiu pomocou mojej funckie dlzka\_paketu\_po\_mediu(). Ďalej analyzujem rámec. Následne analyzujem rámec po bajtoch z údajov uložených v poli data\_packetu. Významné údaje si kopírujem počas analýzy do príslušných štruktúr pre potreby ďaľšej analýzy alebo výpisu.

1. načítam rámec z prvých 12B zistím mac adresy

2. zo 13,14B zistím ethertype

ak je ethertype > 1500 ide o ethernet II hlavičku

inak je to IEEE hlavička

IEEE hlavička

Zisťujem podľa 15B pole SAP o aký konkrétny typ IEEE hlavičky sa jedná

SAP = 255 IEEE raw

SAP = 170 IEEE s LLC a SNAP header

Inak IEEE s LLC

Následne pre všetky nižšie spomenuté protokoly sledujem už len časti hlavičky, ktoré sú definované v rámci štruktúr pre jednotlivé protokoly

3. ETHERNET hlavička na základe pola ethertype zistím o aký vnorený protokol ide

Ethertype == 2054 ARP

Ethertype == 2048 Ipv4

Ethertype == 34525 Ipv6

4. Protokol 3. vrstvy

Ak IPv6 – zistím zdrojovú a cielovú IP adresu a podľa poľa next header aj protokol vyššej vrstvy

Ak ARP – podľa poľa operation určím či ide o request alebo reply. Ďalej z rámcu zisťujem source/target hardaware a protocol adresu.

Ak IPv4 – zistím zdrojovú a cieľovú Ip adresu, podľa poľa protokol protokol vyššej vrstvy, a taktiež IHL aby som sa vedel správne posunúť za IP hlavičku.

5. Protokol 4. vrstvy

Podľa poľa protokol v IPv4 hlavičke zistíme o aký protokol ide.

Ak protocol == 6 TCP

Ak protocol == 17 UDP

Ak protocol == 1 ICMP

Pre ICMP zisťujem následne pole type a code, ktoré potrebujem pre výpis ICMP správy.

Pri TCP/UDP sa zameriavam na zdrojový a cieľový port.

Pri TCP navyše zisťujem aj hodnoty v poli FLAG pre potreby určenia kompletnosti komunikácie.

6. Port identifikujúci aplikáciu na 5./7. vrstve

Pri TCP a UDP na základe čísiel portov určím aká aplikácia používa daný protokol.

7. Výpis rámcu v hexa formáte – tento výpis vypíše len samotný rámec v hexa formáte, jednotlivé potrebné údaje ako napr. názov protokolu alebo adresy sú vypisované priebežne počas procesu analýzy rámcu.

## Použité funkcie

V programe som použil

**Funkcie z pcap.h knižnice:**

pcap\_next\_ex() na načítanie ďalšieho rámcu

pcap\_open\_offline() na otvorenie pcap súborov

a tiež štruktúru struct pcap\_pkthdr

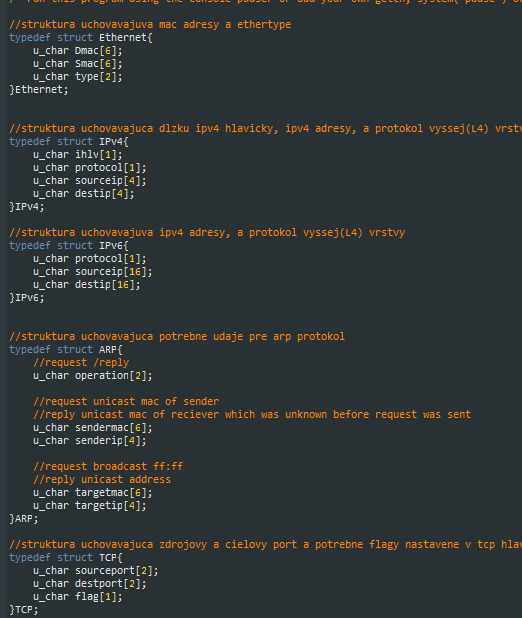
**Funkcie z knižnice string.h:**

ako napríklad strcmp, alebo strstr;

**Vlastné funckie:**

Mnou vytvorené funkcie najmä na výpis a čítanie zo súborov a na prácu zo spájaným zoznamom ip adries uzlov, ktoré sú popísané v zdrojovom kóde programu.

**Vlastné štruktúry:** V programe používam aj vlastné štruktúry na uchovávanie potrebných dát pre analýzu resp. výpis infromácií z rámcov

****

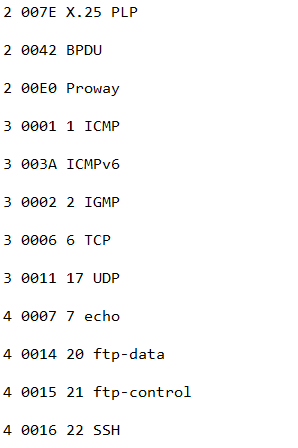
# Príklad štruktúry externých súborov, pre určenie protokolov a portov

Štruktúra súboru messages.txt

# 

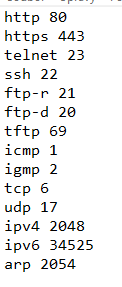
Príklad štruktúry súbory obsahujúceho icmp správy prvé čislo v riadku definuje ICMP type a druhé ICMP code nasledovaný samotným obsahom správy. Obe čísla sú v decimálnom tvare.

štruktúra súboru protokoly1.txt



Príklad štruktúry súboru pre čísla protokolov a portov kde prvé číslo(decimálne) určuje vrstvu v hlavičky protokolu v ktorej sa nachádza nasledujúce hexadecimálne číslo a to kód samotného protokolu nasledovaný názvom daného protokolu/portu. Priradí k číslu portu/protokolu názov.

štruktúra súboru cisla.txt



Príklad štruktúry súboru na priradenie čísla protokolu k názvu. Použité najmé pri modvypisi(4 - 9)

Prvý údaj je názov protokolu/portu nasledovaný decimálnym číslom daného portu.

# Opis používateľského rozhrania

Pred samotným spustením programu je potrebné vytvoriť 3 pomocné textové súbory. Prípadne skontrolovať správnosť ich názvu v programe.



Ďalej treba nastaviť do premennej filepath cestu k .pcap súborom, ktoré chceme analyzovať. Momentálne je nastavená tak, aby program hľadal názov .pcap súboru zo vstupu v priečinku pcap nachádzajúcom sa v aktuálnom adresári.



# 

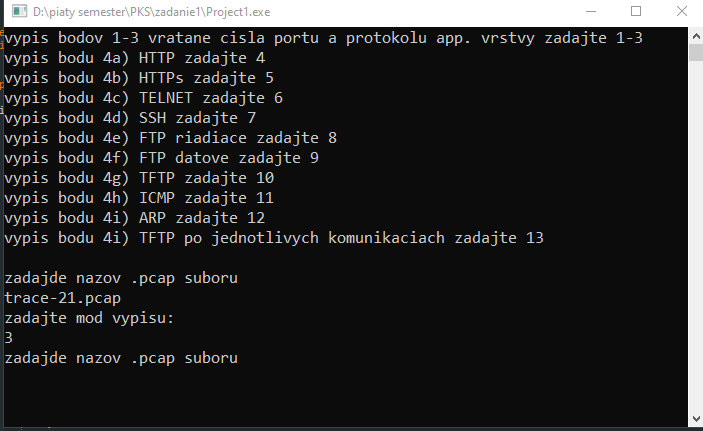
Po spustení programu možno zadať názov súboru, ktorí chceme analyzovať a následne aj jeho mód výpisu pre príslušný výpis ako vidieť na obrázku. Pre zvolenie módu 1,2,3 program vypíše vždy všetky požiadavky zo zadanie v bodoch 1 až 3. Teda vypíše všetky rámce analyzované cez 2-4 vrstvu a na konci sú uvedené ip adresy prijímajúcich uzlov. Následne pre jednotlivé konkrétne módy vypíše len konkrétnu komunikáciu.

Pri výpisoch 4-9 je najskôr vypísaná kompletná a následne nekompletná komunikácia.(Ak sa také nachádzajú v analyzovanom súbore)

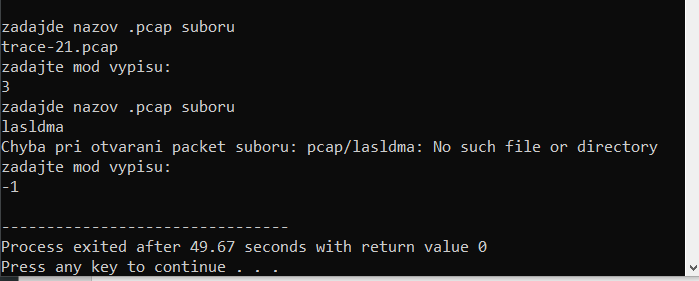
TFTP 10 vypíše len jednu TFTP komunikáciu

TFTP 13 vypíše všetky TFTP komunikácie po jednotlivých komunikáciach.

Pri ARP výpise je pre každý request reply uvedené do ktorej komunikácie patria.



Po zadaní módu výpisu program spracuje súbor a výpis zapíše do súboru output x(kde x je číslo súboru). Z každým ďalším načítaním a výpisom sa číslo x o jedna zväčší teda je zápis vykonaný vždy do ďaľšieho súboru.



Ak chceme program ukončiť možno zadať ľubovoľný názov súboru a následne ako mód výpisu zvoliť

-1. V prípade ak sme omylom zadali nesprávny názov súboru treba zadať nejaký mód výpisu(okrem -1). Program sa následne automaticky opäť dopytuje na názov súboru na spracovanie.

# Zvolené implementačné prostredie

Program som vytvoril v prostredí Dev-Cpp 5.11 s použitím 32-bit verzie TGM-GCC compileru. Predtým som si na windowse musel nainštalovať knižnicu pcap(automaticky spolu s wiresharkom). Program wireshark som používal pri kontrole správností výpisu môjho programu.

Následne som podľa návodu vytvoril projekt, s príslušnými parametrami, ktorí umožňuje využitie funkcionalít z knižnice pcap.h potrebných na otvorenie .pcap súborov. Program možno následne štandardne spustiť na (windows) zariadení ak ho otvoríme v Dev-Cpp prostrí ako Dev-C++ project.

návod: <https://www.csie.nuk.edu.tw/~wuch/course/csc521/lab/ex1-winpcap/>