VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

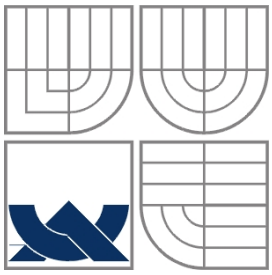
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

ANALYZÁTOR SIEŤOVÝCH PROTOKOLOV

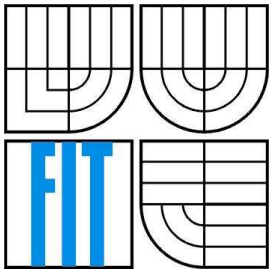
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE TAMÁS TAKÁCS
AUTHOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

ANALYZÁTOR SÍŤOVÝCH PROTOKOLŮ

NETWORK PROTOCOL ANALYZER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — TAMÁS TAKÁCS
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ TOBOLA
SUPERVISOR

BRNO 2010

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2009/2010

Zadání bakalářské práce

Řešitel: **Takács Tamás**
Obor: Informační technologie
Téma: **Analyzátor síťových protokolů**
Network Protocol Analyzer
Kategorie: Web

Pokyny:

1. Seznamte se s architekturou TCP/IP a se síťovými protokoly aplikačních vrstev.
2. Prostudujte možnosti detekce typu protokolu na základě obsahu paketů. Zaměřte se zejména na aplikační vrstvu, dynamické internetové protokoly a algoritmy jejich detekce pro vysokorychlostní sítě.
3. Navrhněte systém pro automatickou detekci vybraných protokolů včetně přehledného grafického webového rozhraní.
4. Implementujte prototyp navrženého systému.
5. Funkčnost prototypu demonstrujte na vhodně zvoleném vzorku dat.
6. Zhodnoťte dosažené výsledky a diskutujte možnosti dalšího rozšíření systému.

Literatura:

- Gerald Combs: Wireshark. Elektronické dokumenty dostupné na http://www.wireshark.org/.
- Sourcefire Inc. Snort. Elektronické dokumenty dostupné na http://www.snort.org/.

Při obhajobě semestrální části projektu je požadováno:

- Bez požadavků.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Tobola Jiří, Ing.**, UPSY FIT VUT
Datum zadání: 1. listopadu 2009
Datum odevzdání: 19. května 2010

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

doc. Ing. Zdeněk Kotásek, CSc.
vedoucí ústavu

Abstrakt

Bakalářská práce popisuje implementaci síťového analyzátoru spolu s přehledným grafickým webovým rozhraním. Síťový analyzátor zachycuje síťový provoz a umožňuje jeho podrobný rozbor. Jeho hlavním úkolem je sledování datových toků a identifikace protokolů na aplikační vrstvě modelu TCP/IP, které spojení využívalo. Práce popisuje dvě možnosti identifikace aplikačních protokolů a to na základě čísla portu a obsahu payloadu TCP paketu/UDP datagramu. Webové rozhraní poskytuje statistický přehled nad výstupnými hodnotami síťového analyzátoru.

Abstract

Bachelor's thesis describes an implementation of a network analyzer with an easy graphical web interface. Network Analyzer captures network communication and allows the detailed analysis. Its main task is to monitor data flows and identify protocols, which are situated on application-layer model TCP / IP. Thesis describes two types of possible identification of application protocols, which are based on the port number and the payload content TCP packet / UDP datagram.
Web interface provides an overview of the output values of the network analyzer.

Klíčová slova

síťový analyzátor, zachytávání paketov, libpcap, sniffer, web

Keywords

packet analyzer, packet capturing, libpcap, sniffer, web

Citace

Takács Tamás: Analyzátor sieťových protokolov, bakalářská práce, Brno, FIT VUT v Brně, 2010

Analyzátor sieťových protokolov

Prohlášení

Prehlasujem, že túto bakalársku prácu som vypracoval samostatne pod vedením Ing. Jiřího Tobolu. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

……
Tamás Takács
12. apríla 2010

Poděkování

Ďakujem vedúcemu práce Ing. Jiřímu Tobolovi za poskytnutú odbornú pomoc a konzultáciu pri priebehu vytvárania tejto práce.

Obsah

1 Úvod

V dnešnej dobe je Internet najrýchlejšie rozširujúcim sa fenoménom. Na počiatku vzniku Internetu ho využívali hlavne služby k prenosu elektronických správ a prenosu súborov. Postupom času Internet začali využívať nové aplikácie, čím vznikali nové protokoly, ktoré popisujú syntaktické a sémantické pravidlá komunikácie.

Pre monitorovanie sietí, k pochopeniu sieťovej komunikácie a k odhaľovaniu prípadných chýb vznikli nástroje volané sieťové analyzátory. Sieťový analyzátor je nástroj, ktorý môže byť použitý s dobrým, ale aj zlým úmyslom. Slúži na zachytávanie sieťovej komunikácie a na jeho detailný rozbor.

Práca popisovaná v tomto dokumente sa zaoberá návrhom a implementáciou sieťového analyzátora nazvaného Pacpt, spolu s webovým rozhraním k nemu vytvoreným.
Sieťový analyzátor je implementovaný v jazyku C, za použitia knižnice *libpcap.h* pre zachytávanie a filtrovanie sieťovej komunikácie. Hlavnou úlohou navrhnutého sieťového analyzátora je identifikovanie aplikačných protokolov najvyššej vrstvy modelu TCP/IP [4, 12]. Sieťový analyzátor popisovaný v práci detekuje aplikačné protokoly na základe detekovania čísla portu z hlavičky protokolu TCP/UDP transportnej vrstvy a na základe signatúr, ktoré obsahuje payload (dáta aplikačného protokolu).

Práca je v druhej kapitole zameraná na vysvetlenie výrazu "analýza siete", teoretickým popisom základného modelu ISO/OSI [4, 12] a podrobnejším popisom modelu TCP/IP. Podrobnejší popis modelu TCP/IP [4, 12] zahrňuje popis architektúry modelu spolu s hlavičkami jednotlivých vrstiev. Ďalej obsahuje popis spôsobu, ako je možné ochrániť sieť proti zachytávaniu sieťovej komunikácie a popis o možných detekciách aplikačného protokolu.

Tretia kapitola sa zaoberá návrhom a implementáciou sieťového analyzátora nazvaného Pacpt. Kapitola obsahuje popis princípu zachytávania dát a jej analýzu, princíp identifikovania TCP/UDP spojenia, princíp detekovania aplikačného protokolu ako aj popis komunikácie programu Pacpt s webovým rozhraním.

Štvrtá kapitola popisuje návrh a architektúru webového rozhrania vytvoreného k programu Pacpt ako aj popis štruktúry jednotlivých častí webu.

Piata a šiesta kapitola obsahuje vykonané testy programom Pacpt a záver obsahujúci zhrnutie práce, výhody programu Pacpt a webového rozhrania, ako aj možné vylepšenia pre budúci vývoj programu Pacpt/webového rozhrania.

2 Teoretický rozbor

2.1 Analýza siete

Analýza siete [4] je proces zachytávania sieťovej komunikácie a jeho detailný rozbor. Cieľom zachytávania sieťovej komunikácie je porozumenie diania v sieti a odhaľovanie prípadných chýb. Sieťový analyzátor dekóduje dátové pakety protokolov a zobrazuje ich v zrozumiteľnej podobe. Programom, ktoré sledujú a zachytávajú sieťovú komunikáciu sa v počítačovej oblasti hovorí "sniffer". V práci implementovaný program pre zachytávanie sieťovej komunikácie sa nazýva Pacpt (názov vznikol ako skratka Packet Capturing).

Sieťový analyzátor [4] môže mať formu programovú, to znamená že aplikácia je nainštalovaná na pracovnú stanicu, alebo tvorenú hardwarovým zariadením so špecializovaným programom.

Použitie sieťových analyzátorov má široké uplatnenie u systémových administrátorov, sieťových a bezpečnostných technikov, ale aj u programátorov sieťových aplikácií. Pre ľudí s dobrým úmyslom znamená "sniffer" neoceniteľný nástroj pri riešení problémov so sieťou, konfiguráciou systémov a sieťových aplikácií, avšak pre ľudí s opačným úmyslom znamená "sniffer" nástroj pre zachytávanie dôverných dát. Zachytávaním dôverných dát sa rozumie zachytávanie mien a hesiel v podobe textu, zachytávanie telefónnej konverzácie ktorá sa odohráva prostredníctvom protokolu VoIP, mapovanie sieťovej topológie a ďalšie.

Použitie programu "sniffer" neoprávnenou osobou vyjadruje značnú hrozbu pre bezpečnosť siete.

2.2 Ochrana siete proti zachytávaniu dát

Bezpečný spôsob zabezpečenia siete proti zachytávaniu dát je šifrovanie [4]. Šifrovanie je veľkou prekážkou do cesty útočníkov, ktorí sa snažia pasívne monitorovať sieťovú komunikáciu siete.

Najbežnejšie šifrovacie protokoly:

SSH (Secure Shell)

Protokol SSH [4] je kryptograficky bezpečnou náhradou štandardnej implementácie Telnet na unixových systémoch. Konkrétne príkazu rlogin, Remote Shell a Remote Copy Protocol. Komunikácia prostredníctvom SSH zahrňuje klientský systém a server, ktorý používa šifrovanie pomocou verejného kľúča k zaisteniu šifrovanej komunikácie.

SSL

Protokol SSL [4] poskytuje služby pre overenie, šifrovanie a môže byť použitý aj pre tvorbu virtuálne privátnych sietí (VPN). Protokol SSL poskytuje šifrovanie na aplikačnej úrovni.

Pretty Good Privacy a Secure/Multipurpose Internet Mail Extensions

Pretty Good Privacy a Secure/Multipurpose Internet Mail Extensions [4] sú protokoly pre šifrovanie správ elektronickej pošty. Ako odosielateľ tak aj príjemca musí používať špeciálnu aplikáciu, ktorá umožňuje šifrovanie a dešifrovanie správ.

Pri sieťach, kde nie je možné použiť šifrovanie, je zabezpečenie siete [4] veľmi obtiažné.
Jedna z možností je, zisťovanie sieťových kariet v sieti, ktoré fungujú v tkz. promiskuitnom režime. Práve tieto sieťové karty vytvárajú podozrenia pre možné zachytávanie sieťovej komunikácie. Mnohé operačné systémy poskytujú mechanizmus pre zisťovanie, či sieťová karta pracuje v promiskuitnom móde. Napr. na unixových systémoch je to príkaz *ifconfig -a*.

2.3 Spôsoby identifikácie aplikačného protokolu

Najjednoduchším, ale aj najčastejšie používaným typom detekcie aplikačného protokolu je detekcia založená na základe detekovania čísla portu z hlavičky protokolu TCP/UDP transportnej vrstvy. Za priraďovanie čísiel portov špecifickým aplikáciám zodpovedá organizácia IANA [10, 11]. Môže sa však stať, že daný port používa iná aplikácia ako je predpísaná. Môže sa to stať v prípade bežne používaného sieťového programu, ale aj škodlivého trójskeho koňa, ktorý

pomocou pripojenia k danému portu prijíma/odosiela nežiaduce informácie. To je hlavnou príčinou nedostatku identifikovania aplikačného protokolu iba na základe čísla portu.

Ďalšou možnosťou identifikácie aplikačného protokolu je na základe signatúr, ktoré obsahuje payload (dáta aplikačného protokolu). Výhodou tohto druhu identifikácie aplikačného protokolu je takmer stopercentná istota správnej detekcie. Zaistí napríklad sieťovému správcovi zablokovanie použitia protokolu užívateľovi, pričom danému užívateľovi nepomôže ani zmena čísla portu pre uskutočnenie komunikácie pomocou daného protokolu. Nevýhodou tejto formy identifikácie aplikačného protokolu je veľká výpočtová-časová náročnosť a zložitý popis niektorých protokolov regulárnym výrazom.

Obidva vyššie uvedené metódy detekcie aplikačných protokolov sú neúčinné ak komunikácia je šifrovaná a aplikácia využíva dynamické prideľovanie čísiel portov. Príkladom šifrovanej komunikácie je protokol Skype [20]. Aplikácia Skype používa pre šifrovanie sieťovej komunikácie šifrovací algoritmus AES [21] a pre výmenu kľúčov využíva šifru RSA [22]. Na odhalenie sieťových aplikácií podobných aplikácii Skype (P2P protokoly) sa využívajú špecifické vlastnosti každého jedného protokolu. Napríklad komunikácia Skype sa dá odhaliť na skutočnosti, že po pripojení sa k serveru a po zahájení komunikácie s konkrétnym užívatelom Skype udržuje spojenie i s užívateľmi, ktorých má v zozname priateľov. Zisťuje stavy jednotlivých užívateľov pomocou protokolu UDP v rovnakých časových rozmedziach.

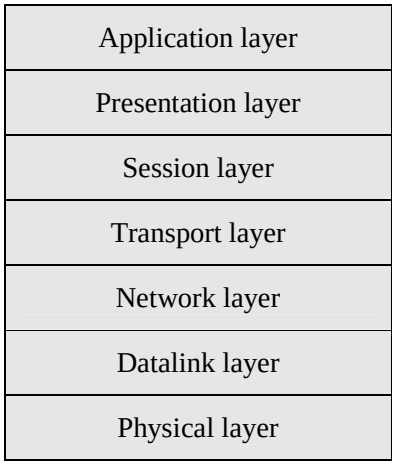
Jednou z vlastností komunikácií P2P aplikácií je, že súčasne využívajú pre komunikáciu protokol UDP a aj protokol TCP. Po zachytení sieťovej komunikácie a detekovaní dátových tokov využívajúcich protokol TCP a UDP súčasne s rovnakou zdrojovou a cieľovou IP adresou je možné s veľkou istotou určiť, že sa jedná o P2P aplikáciu komunikujúcu cez danú sieť.

Zachytávanie a sledovanie daného dátového toku a zistenie týchto špecifických vlastností (veľkosť paketov, pravidelnosť komunikácie, rýchlosť prichádzania paketov a ďalšie) pre daný protokol umožňuje aj detekovanie aplikácií, ktorých komunikácia je šifrovaná.

Analyzátor popisovaný v tejto práci umožňuje identifikovanie aplikačného protokolu na základe čísla portu a obsahu paylodu.

2.4 Model ISO/OSI

Model OSI(Open Systems Interconnection) [4, 14] bol vyvinutý začiatkom 80. rokov organizáciou ISO(Internet national Standards Organization) za účelom definovania spolupráce jednotlivých komponentov a sieťových protokolov. Rozdeľuje sieťové funkcie do siedmych vrstiev, kde každá z nich predstavuje určitú množinu súvisiacich špecifikácií, funkcií a činností.



Obrázok 2.1: Model ISO/OSI

2.4.1 Popis jednotlivých vrstiev Modelu ISO/OSI

Fyzická vrstva

Fyzická vrstva [14] popisuje elektrické či optické signály používané pre komunikáciu medzi počítačmi. Na fyzickej vrstve je vytvorený tzv. fyzický okruh. Na fyzický okruh medzi dvoma počítačmi bývajú často vkladané ďalšie zariadenia ako sú napríklad modemy.

Linková vrstva

Linková vrstva [14] zaisťuje v prípade sériových liniek výmenu dát medzi susednými počítačmi a v prípade lokálnych sietí výmenu dát v rámci lokálnej siete. Základnou jednotkou pre prenos dát je na linkovej vrstve dátový rámec. Dátový rámec obsahuje v hlavičke linkovú adresu príjemcu,

odosielateľa a ďalšie riadiace informácie. Základnými zariadeniami pracujúcimi na linkovej vrstve sú switch a bridge.

Sieťová vrstva

Sieťová vrstva [14] zabezpečuje prenos dát medzi vzdialenými počítačmi WAN. Základnou jednotkou prenosu je sieťový paket, ktorý sa balí do dátového rámca. Na sieťovej vrstve je jednoznačne v celej WAN adresované sieťové rozhranie . Sieťovým rozhraním rozumieme napríklad kartu pre Ethernet. Základným zariadením pracujúcim na sieťovej vrstve je router.

Transportná vrstva

Transportná vrstva [14] zabezpečuje spojenie medzi aplikáciami na vzdialených počítačoch. Poskytuje transparentný prenos dát medzi uzlami a zaisťuje riadenia toku dát od uzla k uzlu, detekciu chýb a zotavenia sa z chýb. Protokoly transportnej vrstvy vytvárajú spojenia na špecifických portoch počítačov a vytvárajú virtuálne okruhy. Dvoma najznámejšími protokolmi transportnej vrstvy sú protokoly TCP (Transmision Control Protocol) a UDP (User Datagram Protocol). TCP je spojovaným protokolom a UDP je nespojovaným protokolom.

Relačná vrstva

Relačná vrstva [4, 14] zabezpečuje nadväzovanie, sledovanie a ukončovanie relácie za použitia virtuálnych okruhov vytvorených transportnou vrstvou. Ďalšou dôležitou úlohou relačnej vrstvy je určovanie, či bude komunikácia odosielaná formou tzv. full-duplex alebo half-duplex správ.

Prezentačná vrstva

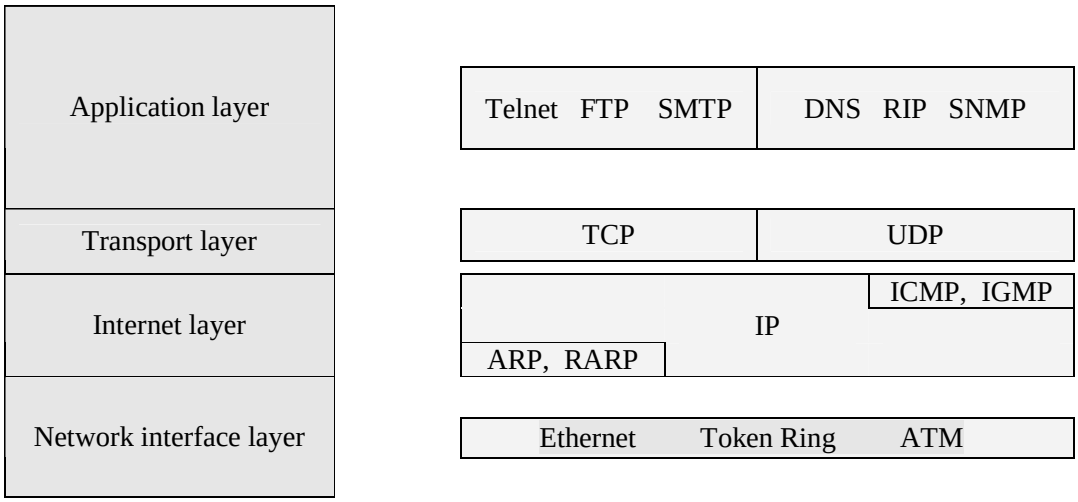
Prezentačná vrstva [4, 14] je zodpovedná za reprezentáciu a zabezpečenie dát. Reprezentácia dát môže byť na rôznych počítačoch iná. Zabezpečením sa rozumie šifrovanie, zabezpečenie integrity dát alebo aj digitálne podpisovanie.

Aplikačná vrstva

Aplikačná vrstva [4, 14] predpisuje v akom formáte a ako majú byť dáta prijímané/posielané od aplikačných programov. Na aplikačnej vrstve pracuje mnoho protokolov používaných napríklad na príjem či odosielanie elektronických správ elektronickou poštou (SMTP, POP3, IMAP), protokol pre správu sietí (SNMP) ako aj protokol pre prenos hypertextových dokumentov pre web (HTTP).

2.5 Model TCP/IP

Model TCP/IP [4, 14] bol vytvorený v sedemdesiatich rokoch 20. storočia Ministerstvom obrany USA. Stal sa modelom, ktorého sa držia hlavné internetové protokoly. Najznámejšími protokolmi modelu TCP/IP sú TCP (Transmission Control Protocol) a IP (Internet Protocol) [4, 14]. Model TCP/IP je zhustenou variantou modelu OSI a tvoria ho štyri vrstvy.



Obrázok 2.2: Model TCP/IP

2.5.1 Popis jednotlivých vrstiev modelu TCP/IP

Vrstva sieťového rozhrania

Vrstva sieťového rozhrania [4, 14] popisuje štandardy pre fyzické médium a elektrické signály ako sú napríklad Ethernet, Token Ring alebo Frame Relay. Vrstva sieťového rozhrania taktiež popisuje funkcie pre prístup k fyzickému médiu.

Sieťová vrstva

Sieťová vrstva [4, 14] slúži na vytváranie datagramov, adresuje a smeruje ich na dané miesta. Zaisťuje prenos s tzv. “doručenie s najväčšou snahou“ (best-effort delivery), čo znamená, že IP

vrstva sa snaží doručiť poslané dáta najvhodnejšou cestou. Pri nepodarenom prenose dát je užívateľ obratom informovaný. Sieťová vrstva poskytuje protokoly: IP (Internet Protocol) protokol pre prenos a smerovanie datagramov, ARP protokol(Adress Resolution Protocol) a RARP (Reverse ARP) pre mapovanie IP adries a MAC adries, ICMP (Internet Control Message Protocol) pre riadenie toku a detekciu nedosiahnuteľných uzlov a IGMP(Internet Group Management Protocol) pre prihlasovanie sa do multicastových skupín. Protokol IP je nespoľahlivým protokolom, pretože nevykonáva žiadnu kontrolu správnosti prenosu a opravu dát.

Transportná vrstva

Transportná vrstva [4, 14] zabezpečuje prenos dát medzi aplikáciami. Transportná vrstva zabezpečuje spojovanú službu TCP a aj nespojovanú službu UDP (viď. Architektúra TCP/IP - Protokol TCP a UDP).

Aplikačná vrstva

Aplikačná vrstva [4, 14] slúži na podporu sieťových aplikácií. Aplikačná vrstva sa stará o spracovanie dát na najvyššej úrovni vrátane reprezentácie dát, kódovania a riadenia dialógu. Aplikačnými protokolmi sú napríklad: FTP, SMTP, Telnet, HTTP, DNS, NNTP.

2.6 Architektúra TCP/IP

2.6.1 Ethernet

V modely TCP/IP Ethernet [4, 12, 14] realizuje vrstvu sieťových rozhraní. Ethernet je najpopulárnejším protokolom v LAN sietiach, kvôli jeho jednoduchosti a ľahkej implementácii. Jednotlivé PC sú v LAN sieti identifikované so svojimi hardwarovými adresami - MAC adresa. V závislosti od použitého prenosového média(je možná kombinácia) sa rozlišujú druhy Ethernetu ako sú napr. Ethernet 10base5 alebo 10baseT. Pre prístup k zdieľanému prenosovému médiu sa používa metóda CSMA/CD.

0-5	6-11	12-13
Destination MAC Address	Source MAC Adress	Ethernet type

Obrázok 2.3: Formát Ethernetovej hlavičky

2.6.2 Protokol IP

Protokol IP (Internet Protocol) [4, 12, 14] slúži pre komunikáciu medzi vzájomne prepojenými sieťovými rozhraniami v sieti. Dáta v IP sieti sú posielané v blokoch a nazývajú sa segmenty (TCP) alebo datagramy (UDP). Protokol IP je nespoľahlivou službou to znamená, že neposkytuje spoľahlivé doručenie dát ani kontrolu nad posielaním/ prijímaním. Na jednoznačnú identifikáciu a smerovanie medzi sieťovými rozhraniami slúži IP adresa. V súčasnosti najpoužívanejším protokolom sieťovej vrstvy je IPv4 a aktuálne rozširujúca sa verzia IPv6.

0-3	4-7	8-15	16-18	19-31
Version	Header Lenght	Differentiated Services	Total Lenght	
Identification			Flags	Fragment Offset
Time to Live		Protocol	Header Checksum	
Source Address				
Destination Address				
Options				
Data				

Obrázok 2.4: Formát hlavičky IPv4

2.6.3 Protokol TCP

Protokol TCP (Transmission Control Protocol) [4, 12, 14] slúži pre prenos dát WAN sieťami medzi dvoma konkrétnymi aplikáciami. Protokol TCP je spojovanou službou, ktorý je plne duplexný. Spojenie je typu point-to-point čo znamená, že komunikácia prebieha medzi dvoma PC (aplikáciami) vo WAN sieti, kde jeden PC vystupuje ako odosielateľ a druhý PC ako príjemca. Protokol TCP zaisťuje spoľahlivý prenos dát , zreťazený prenos (pipelined). Protokol TCP je spojovo orientovanou službou – vytvorenie spojenia sa uskutočňuje výmenou správ (handshaking). Na jednoznačnú identifikáciu, že ktorej aplikácii sa majú dáta poslať slúži číslo portu.
Základnou jednotkou v protokole TCP je segment, niekedy nazývaný aj ako TCP paket.

0-3	4-7	8-15	16-31
Source Port			Destination Port
Sequence Number			
Acknowledgement Number			
Data Offset	Reserved	Flags	Window
Checksum			Urgent Pointer
Options			
Data			

Obrázok 2.5: Formát TCP hlavičky

2.6.4 Protokol UDP

Protokol UDP (User Datagram Protocol) [4, 12, 14] je jednoduchý protokol transportnej vrstvy pre prenos dát WAN sieťami medzi dvoma konkrétnymi aplikáciami. Protokol UDP je datagramová služba bez záruky doručenia a kontroly poriadia paketov. Protokol UDP je nespojovanou službou. Na jednoznačnú identifikáciu, ktorej aplikácii sa majú dáta poslať slúži číslo portu. Základnou jednotkou v protokole UDP je UDP datagram.

0-15	16-31
Source Port	Destination Port
Lenght	Checksum
Data	

Obrázok 2.6: Formát UDP hlavičky

2.6.5 Payload

Payloadom sa nazývajú dáta bez TCP/UDP hlavičky, ktoré medzi sebou posielajú aplikácie. Payload môže mať textový alebo binárny obsah.

3 Program Pacpt

Program Pacpt je nástroj slúžiaci na zachytávanie sieťovej komunikácie a na jeho podrobný rozbor. Hlavnou úlohou programu Pacpt je sledovanie dátových tokov a identifikácia protokolov na aplikačnej vrstve modelu TCP/IP, ktoré spojenia využívali. Program Pacpt je konzolovou aplikáciou implementovanou v jazyku C na platforme Unix. Pre zachytávanie sieťovej komunikácie využíva knižnicu *libpcap.h* .

Program Pacpt je schopný identifikovať aplikačné protokoly na aplikačnej vrstve modelu TCP/IP dvoma spôsobmi . Identifikácia aplikačných protokolov na základe čísla portu a obsahu signatúr v payloade TCP paketu/UDP datagramu.

Programom Pacpt sú podporované aplikačné protokoly, ktoré sú uvedené v špeciálnom .txt súbore. Meno tohto .txt súboru je uvedené v konfiguračnom súbore *config.txt*.

Základné nastavenia programu Pacpt sa nastavujú pred samotným spusteným programu v konfiguračnom súbore *config.txt*. Nastavuje sa hlavne meno sieťového rozhrania, na ktorom bude prebiehať samotná analýza, filter pre filtrovanie zachytených paketov, meno výstupného súboru, kde sa budú zapisovať výstupné údaje a mnoho ďalších dôležitých nastavení.

Program Pacpt funguje iba v prípade, ak je možné pripojiť sa k databáze MySQL [2], ktorá slúži ako ukladacie miesto niektorých výstupných údajov (detekované spojenia a detekované aplikačné protokoly), ktoré zobrazuje v zrozumiteľnej forme webové rozhranie k nemu vytvorené.

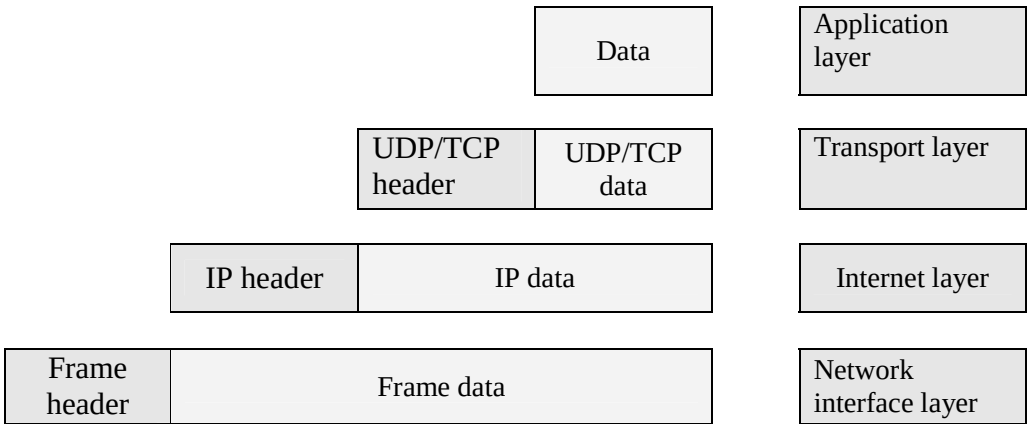
3.1 Zachytávanie dát

Zachytávanie surových (raw) dát zo siete umožňujú programátorovi na to špeciálne vytvorené knižnice. Najznámejšou knižnicou pre platformu Unix je *libpcap*. Knižnicu *libpcap* využíva mnoho známych nástrojov na zachytávanie a analýzu dátových paketov. Príkladom je napríklad unixový nástroj tcpdump [7]. Výhodou použitia knižnice libpcap je možnosť zachytávať všetky dáta z daného segmentu siete a nielen tie, ktoré sú určené danému sieťovému rozhraniu (sieťová karta musí pracovať v promiskuitnom režime).

Knižnica libpcap poskytuje aj súbor filtrovacích pravidiel pre filtrovanie zachytených paketov. Filtrovanie paketov je realizované pomocou Berkeley Packet Filter (BPF) [18].

3.1.1 Analýza zachytených dát

Ako bolo spomínané o kapitolu vyššie, dáta posielané aplikácii sú v nespracovanej forme nazývané aj ako raw. Každé zachytené dáta sa skladajú z jednej alebo viacerých vrstiev, kde každá vrstva predstavuje jednotlivý typ protokolu. V sieťových modeloch ako je aj model TCP/IP platí, že dáta vyšších vrstiev sú zabalené do dát nižších vrstiev. To umožňuje vyšším vrstvám používať vlastnosti vrstiev nachádzajúcich sa pod ňou. Túto skutočnosť znázorňuje aj nasledujúci obrázok.



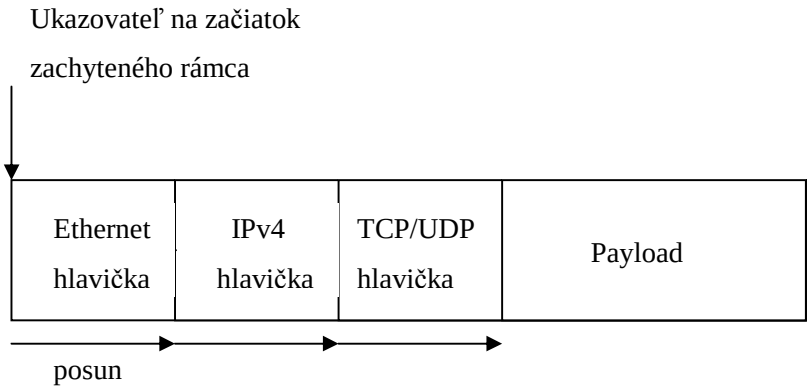
Obrázok 3.1: Zapúzdrovanie dát vyšších vrstiev do PDU nižších vrstiev v modely TCP/IP

Z vyššie uvedených poznatkov vyplýva, že k dátam ktoré sú na najvyššej aplikačnej vrstve sa nedá priamo pristupovať, ale až po odbalení jednotlivých vrstiev a získaní ukazovateľa na nami požadované miesto. Pri odbaľovaní jednotlivých vrstiev je nutné poznať štruktúry hlavičiek a ich veľkosti, ktoré môžu byť navyše premenlivé.

Analyzátor popisovaný v tejto práci k tomu používa štruktúry deklarované v hlavičkových súboroch OS Linux v adresári */usr/include/netinet* a */usr/include/net*. Formát štruktúr je zhodný s formátom hlavičiek jednotlivých protokolov modelu TCP/IP popisovaných vo vyššie uvedenej kapitole "Architektúra TCP/IP".

K dátam aplikačnej vrstvy sa analyzátor dostáva po jednotlivom posúvaní ukazovateľa. Ukazovateľ sa posúva od začiatku rámca po veľkostiach hlavičiek v poradí posun o veľkosť Ethernetovej hlavičky, IP hlavičky a nasledovnej TCP/UDP hlavičky. Tento postup získavania

ukazovateľa k jednotlivým hlavičkám protokolov na daných vrstvách a nakoniec k dátam aplikačnej vrstvy zobrazuje nasledovný obrázok.



Obrázok 3.2: Posun ukazovateľa od začiatku rámca k jednotlivým hlavičkám protokolov až k dátam aplikačnej vrstvy

3.2 Detekovanie a vytvorenie záznamu o novo detekovanom TCP/UDP spojení

TCP/UDP spojenie je jednoznačne identifikované zdrojovým, cieľovým číslom portu a zdrojovou, cieľovou IP adresou. Analyzátor využíva na triedenie TCP segmentov a UDP datagromov do jednotlivých dátových tokov funkciu, ktorá porovná či daná kombinácia čísiel portov a IP adries už existuje v záznamoch analyzátora.

Na ukladanie záznamov - jednotlivých detekovaných TCP/UDP spojení analyzátor využíva hashovaciu tabuľku, ktorá je zvlášť pre komunikáciu smerujúcu smerom von zo sieťového rozhrania a smerujúcu smerom dnu do sieťového rozhrania. Po zavolaní hashovacej funkcie s parametrami, ktoré jednoznačne identifikujú spojenie, funkcia vráti návratovú hodnotu, ktorá slúži ako index do hashovacej tabuľky vytvorenej pre daný smer.

Detekcia nového TCP spojenia je založená na vlastnosti, že protokol TCP je spojovo orientovanou službou. Vytvorenie nového TCP spojenia je založené na prijatí/odoslaní paketu s príznakom SYN. Po prijatí/odoslaní paketu s príznakom SYN sa alokuje nové pamäťové miesto v hashovacej tabuľke pre uloženie spojenia.

UDP protokol je bezstavovým protokolom čo znamená, že detekovanie nového UDP spojenia je založené čisto na kontrole či daná kombinácia čísiel portov a IP adries už existuje v záznamoch hashovacej tabuľke pre daný smer. Po prijatí datagramu s kombináciou IP adries a čísiel portov, ktorá neexistuje v hashovacej tabuľke pre daný smer sa alokuje nové pamäťové miesto v hashovacej tabuľke pre uloženie spojenia.

Zistenie smeru komunikácie funguje na princípe porovnania IP adresy sieťového rozhrania so zdrojovou IP adresou zachyteného UDP datagramu alebo TCP paketu. Ak sa IP adresa sieťového rozhrania rovná zdrojovej IP adrese odosielaného UDP datagramu alebo TCP paketu, jedná sa o komunikáciu smerom von a ak sa nerovná, jedná sa o komunikáciu smerom dnu.

3.3 Ukladanie payloadu a identifikácia spojenia

Po analýze zachyteného TCP paketu/UDP datagramu a zistení prítomnosti aplikačných dát (payload) sa volá funkcia na kontrolu či TCP paket alebo UDP datagram patrí do analyzátorom zaznamenaného dátového toku (spojenia). V prípade ak záznam o spojení do ktorého rámec patrý neexistuje, rámec sa ďalej nespracováva. V opačnom prípade ak záznam o spojení existuje, skontroluje sa či pre daný dátový tok už bol identifikovaný aplikačný protokol. Ak aplikačný protokol identifikovaný zatiaľ nebol, skontroluje sa veľkosť odchytených dát v danom dátovom toku a smere komunikácie. Ak veľkosť odchytených dát v danom dátovom toku je väčší, alebo rovný maximálnej veľkosti danej parametrami programu, TCP paket/UDP datagram sa ďalej nespracováva. V prípade, keď aplikačný protokol zatiaľ nie je identifikovaný pre daný dátový tok a ani zachytená veľkosť dát v danom dátovom toku nie je väčší, ako maximálna veľkosť odchytených dát. dáta aplikačného protokolu (payload) sa pridajú do hashovacej tabuľky na index vrátený hashovacou funkciou na koniec jednosmerného zoznamu.

3.4 Životnosť detekovaného TCP/UDP spojenia z pohľadu analyzátora

Životnosťou sa myslí doba, po ktorú drží analyzátor záznam o existencii TCP/UDP spojenia počas behu programu.

V prípade záznamu o existujúcom TCP spojení sa záznam z obidvoch hash tabuliek (smer von, dnu) maže po detekovaní TCP paketu s príznakom FIN alebo s príznakom RST.

V prípade záznamu o existujúcom UDP spojení sa záznam z obidvoch hash tabuliek (smer von, dnu) maže po uplynutí určitého času, t.j. od príchodu posledného UDP datagramu v danom spojení.

3.5 Detekovanie aplikačného protokolu

3.5.1 Detekovanie aplikačného protokolu na základe obsahu payloadu

Pre vyhľadávanie signatúr v payloade paketu/datagramu analyzátor využíva funkcie z hlavičkového súboru *regex.h* . Po skompilovaní regulárneho výrazu funkciou *regcomp()* sa vytvorí štruktúra, ktorou je deterministický konečný automat. Po skompilovaní a získaní potrebnej štruktúry nastáva samostatné vyhľadávanie hľadaných signatúr v textovom reťazci (v payloade) predanom ako parameter funkcie *regexec()*.

Protokoly, ktoré analyzátor je schopný identifikovať sú uvedené nižšie v Tabuľke 3.1. Pre každý jeden analyzátorom podporovaný protokol existuje jeden príslušný k nemu vytvorený regulárny výraz – deterministický konečný automat. Regulárne výrazy pre jednotlivé aplikačné protokoly boli prevzaté z open-source nástroja L7-filter [8].

Po spustení analyzátora sa načítajú regulárne výrazy, čísla portov spolu s názvami aplikačných protokolov a typu transportného protokolu z špeciálne preto vytvoreného textového súboru. Názov tohto súboru je daný v konfiguračnom súbore *config.txt*. Po načítaní z tohto súboru sa načítané údaje ukladajú do dvoch tabuliek rozdelených podľa typu transportného protokolu

TCP/UDP. Do týchto dvoch tabuliek sa ukladajú regulárne výrazy už v prekompilovanej podobe, kvôli zvýšeniu efektívnosti analyzátora.

Samotné vyhľadávanie signatúr v paylode paketu/datagramu prebieha hneď po prvom odchytenom payloade, kvôli skutočnosti, že práve analyzátorom vyhľadávané signatúry sa nachádzajú v prvých desiatkach bajtov dát v dátovom toku (v detekovanom spojení). Po neúspešnom vyhľadaní hľadaných signatúr sa vyhľadávanie uskutoční pri ďalšom odchytenom payloade paketu/datagramu patriaceho do dátového toku, alebo pokiaľ sa neidentifikuje typ aplikačného protokolu, alebo sa nedosiahne maximálna veľkosť odchytených dát v dátovom toku pre daný smer. Aplikovanie jednotlivých regulárnych výrazov na vyhľadávanie signatúr v danom textovom reťazci prebieha sekvenčne - v poradí od nultého indexu tabuľky až po koniec. Výber tabuľky, z ktorej sa budú regulárne výrazy brať je daný typom transportného protokolu TCP/UDP.

3.5.2 Detekovanie aplikačného protokolu na základe čísla portu

Načítanie čísiel portov, názvov aplikačných protokolov, ukladanie záznamov ako aj porovnávanie prebieha rovnakým princípom ako v prípade detekcie aplikačného protokolu na základe obsahu payloadu s tým rozdielom, že po príchode TCP paketu alebo UDP datagramu sa neprehľadáva obsah payloadu, ale porovnáva sa zdrojové číslo portu protokolu TCP/UDP transportnej vrstvy s číslom portu uloženým v tabuľke. Identifikovanie aplikačného protokolu na základe čísla portu nastáva hneď po prvom odchytenom paketu/datagramu v danom dátovom toku (v danom spojení).

Zoznam aplikačných protokolov podporovaných analyzátorom popisovaným v tejto práci.

Číslo portu	Služba	Protokol
21	FTP	TCP
22	SSH	TCP
23	Telnet	TCP
25	SMTP	TCP
53	DNS	UDP
68	DHCP	UDP
69	TFTP	TCP
80	HTTP	TCP
110	POP3	TCP
161	SNMP	UDP
443	HTTPS	TCP

Tabuľka 3.1: Zoznam aplikačných protokolov podporovaných analyzátorom

3.6 Komunikácia s MySQL databázou

Ku komunikácii programu Pacpt s MySQL [2] databázou sa využívajú funkcie z hlavičkového súboru *mysql.h*.

Po úspešnom pripojení sa k MySQL [2] databáze, analyzátor komunikuje s databázou v troch prípadoch:

1. Zápis základných informácií o sieťovom rozhraní a času spustenia analyzátora po spustení programu Pacpt.
2. Zápis o detekovanom TCP/ UDP spojení a o type aplikačného protokolu, ktoré spojenie využívalo v určitom časovom intervale.
3. Zápis štatistických údajov o počte zachytených TCP paketov a UDP datagramov ako aj o počte detekovaných TCP/ UDP spojení v určitom časovom intervale.

4 Webové rozhranie k programu Pacpt

4.1 Požiadavky kladené na web rozhranie k programu Pacpt

Webové rozhranie bude umožňovať štatistický prehľad nad dekódovanou sieťovou komunikáciu zachytenou programom Pacpt.

Rozhranie bude jednoduché, užívateľsky veľmi prívetivé. Jednotlivé výstupné údaje programu Pacpt budú zhrnuté v tabuľkách. Webové rozhranie bude obsahovať "koláčové" grafy, ktoré budú zobrazovať názvy detekovaných protokolov na aplikačnej vrstve modelu TCP/IP v rôznych časových rozmedziach. Všetky premenlivé údaje sa budú automaticky aktualizovať na webovom rozhraní bez zásahu užívateľa každú jednu minútu. Web budú môcť prehliadať iba registrovaný užívatelia. Registrovaný (prihlásený) užívateľ bude jedinou vystupujúcou rolou v systéme.

Aplikácia bude komunikovať s MySQL databázou, kde program Pacpt ukladá výstupné hodnoty. Ďalej bude umožňovať stiahnutie .txt súboru z webového serveru, kde je uložený program Pacpt spolu s webovým rozhraním.

Webové rozhranie bude optimalizované pre najbežnejšie webové prehliadače, najmä pre prehliadač Mozilla Firefox. Programová časť webového portálu bude oddelená od grafickej časti, čo bude umožňovať jednoduchú zmenu grafického štýlu webu.

4.2 Návrh a architektúra webového rozhrania

Pre implementáciu webového rozhrania bol použitý značkovací jazyk HTML [1] skombinovaný s populárnym skriptovacím jazykom PHP [2]. Výber na použitie jazyka PHP padlo kvôli rozsiahlych knižniciach už v základnej inštalácii, natívnej podpore databázových systémov, širokej podpore hostingov na uloženie webu a veľa ďalších. Jazyk PHP umožňuje objektový

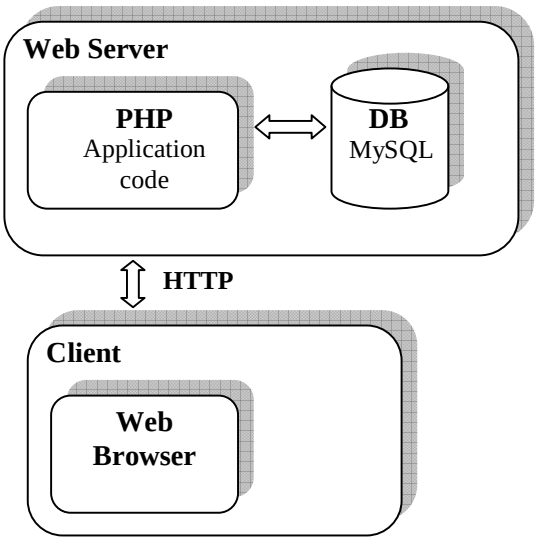
princíp programovania, ktorý však pri implementácii webového rozhrania k programu Pacpt využitý nebol.

Na zobrazovanie grafov bol použitý online nástroj Google Chart API [17], ktorý na základe parametrov predaných v URL vygeneruje graf v obrázkovom formáte PNG. Online nástroj Google Chart API umožňuje generovanie grafov rôznych typov v rôznych farbách a nastavenie ďalších parametrov pre nastavenie výzoru grafu. Práve preto bol nástroj Google Chart API uprednostnený pred výberom iného nástroja slúžiaceho na generovanie grafov.

Pre oddelenie obsahu HTML od jeho vzhľadu bolo použité CSS [1] (Kaskádové štýly). CSS je rozšírenie jazyka HTML a slúži na vizuálne formátovanie webového rozhrania.

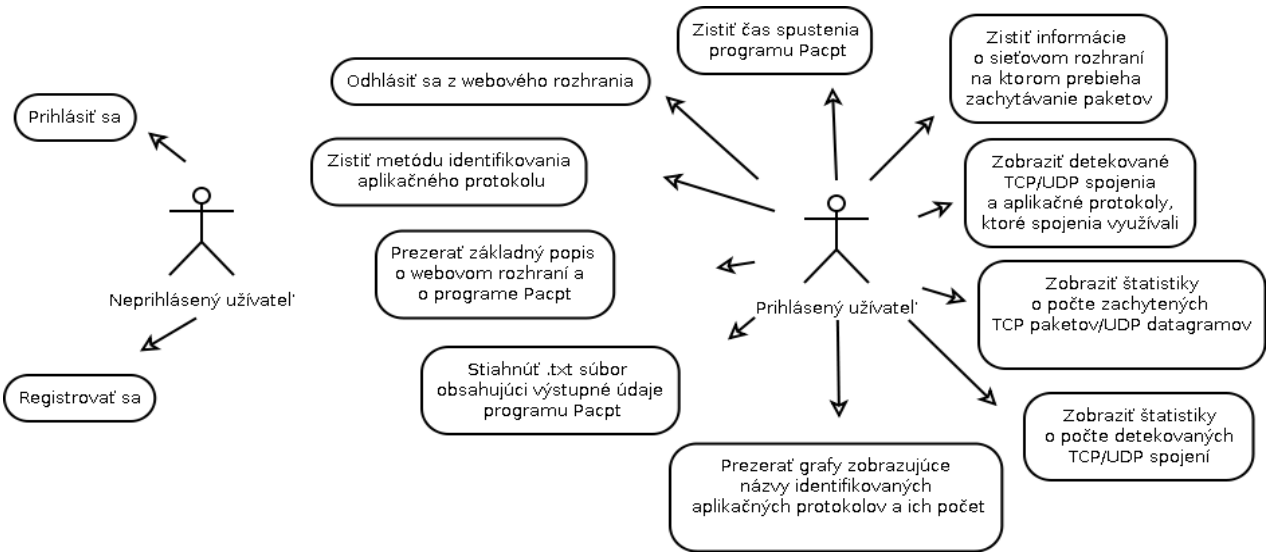
Aktuálnosť zobrazovaných hodnôt webovým rozhraním je zaistené v rozdelení zobrazenej web stránky do niekoľkých blokov (použitie HTML blokového elementu DIV), kde obsah bloku obsahujúci hodnoty, ktoré sa v závislosti na čase a výstupných hodnôt programu Pacpt menia, sa aktualizuje každú jednu minútu. Aktualizácia daných blokov je implementované pomocou použitia jQuery [19] skriptu. JQuery je JavaScript -ová knižnica obsahujúca skripty slúžiace na uľahčenie práci programátora.

Prístup ku databáze MySQL [2], ktorá je súčasťou webového serveru umožňujú funkcie jazyka PHP.



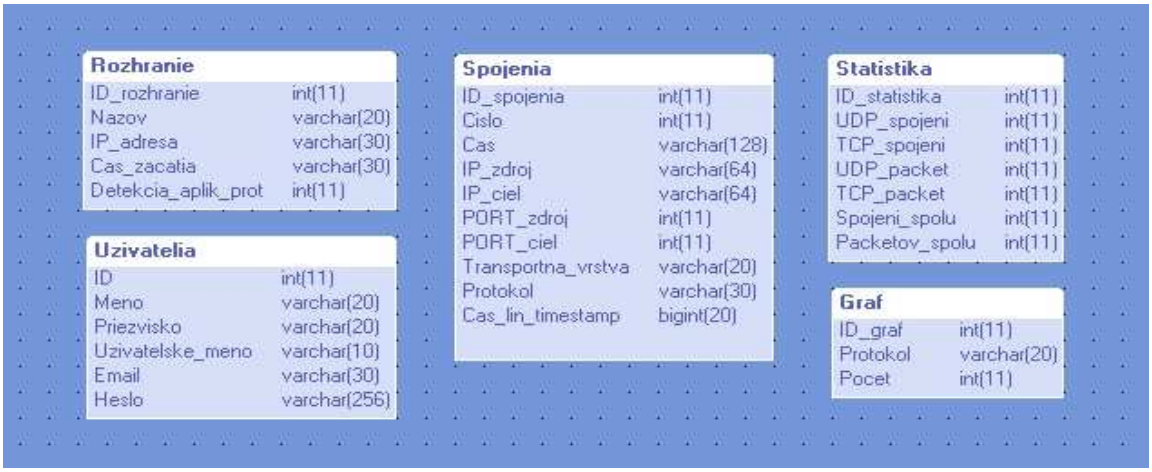
Obrázok 4.1: Architektúra Client-Server

4.2.1 UseCase diagram



Obrázok 4.2: UseCase diagram

4.2.2 ER diagram



Obrázok 4.3: ER diagram

4.2.3 MySQL databáza

Návrh databáze vychádza z ER diagramu znázorneného na obrázku 4.3 .
Popis jednotlivých tabuliek vytvorených v MySQL databáze:

Tabuľka **Graf**	- uchováva informácie o názvoch aplikačných protokolov a ich počet.
Tabuľka **Rozhranie**	- uchováva informáciu o názve sieťového rozhrania, jeho IP adrese a o čase začiatku behu programu.
Tabuľka **Spojenia**	- uchováva informácie o detekovaných spojeniach a o aplikačných protokoloch, ktoré spojenia využívali.
Tabuľka **Statistika**	-uchováva informácie o počte vytvorených TCP/UDP spojení a počte zachytených TCP paketov/UDP datagramov.
Tabuľka **Uzivatelia**	-uchováva osobné údaje o užívateľovi a jeho prihlasovacie údaje.

4.2.4 Návrh štruktúry webu

Webové rozhranie má jednostĺpcový návrh so spoločnou hlavičkou a pätou.

Štruktúra webu je rozdelená do šiestich častí pre dosiahnutie čo najväčšej prehľadnosti o údajoch, ktoré webové rozhranie obsahuje. Pre prepínanie sa medzi jednotlivými časťami webu slúži horizontálne menu, ktoré je uložené hneď pod hlavičkou webu. Názvy jednotlivých častí webu sú nazvané tak aby užívateľ intuitívne hneď po prvej návšteve webu vedel kam má kliknúť bez hocijakého hlbšieho skúmania webového rozhrania k programu Pacpt.

4.3 Popis jednotlivých častí webového rozhrania

4.3.1 Hlavička a horizontálne menu webového rozhrania

Obrázok 4.4: Hlavička a horizontálne menu webového rozhrania

4.3.2 Päta webového rozhrania

Obrázok 4.5: Päta webového rozhrania

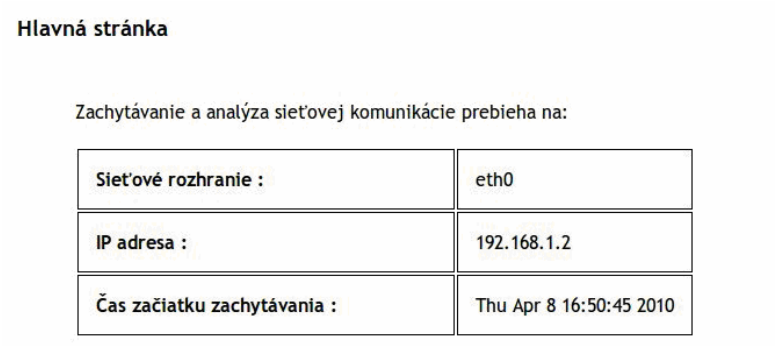
4.3.3 Úvodná stránka Webového rozhrania (Prihlasovacia)

Stránka Prihlasovacia obsahuje prihlasovací formulár pre už zaregistrovaných užívateľov webového rozhrania, alebo umožňuje zaregistrovanie nového užívateľa.

Obrázok 4.6: Úvodná stránka Webového rozhrania (Prihlasovacia)

4.3.4 Hlavná stránka

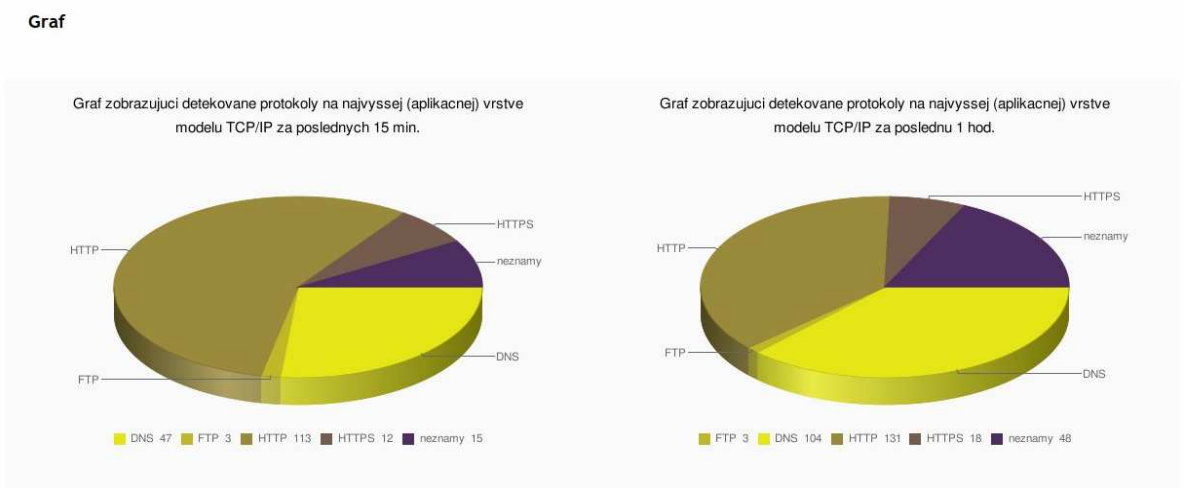
Hlavná stránka obsahuje tabuľku zobrazujúcu základné informácie o sieťovom rozhraní, na ktorom prebieha zachytávanie a analýza paketov. Základnými informáciami sú názov sieťového rozhrania, IP adresa sieťového rozhrania a čas spustenia programu Pacpt tj. čas začiatku zachytávania paketov.



Obrázok 4.7: Hlavná stránka

4.3.5 Graf

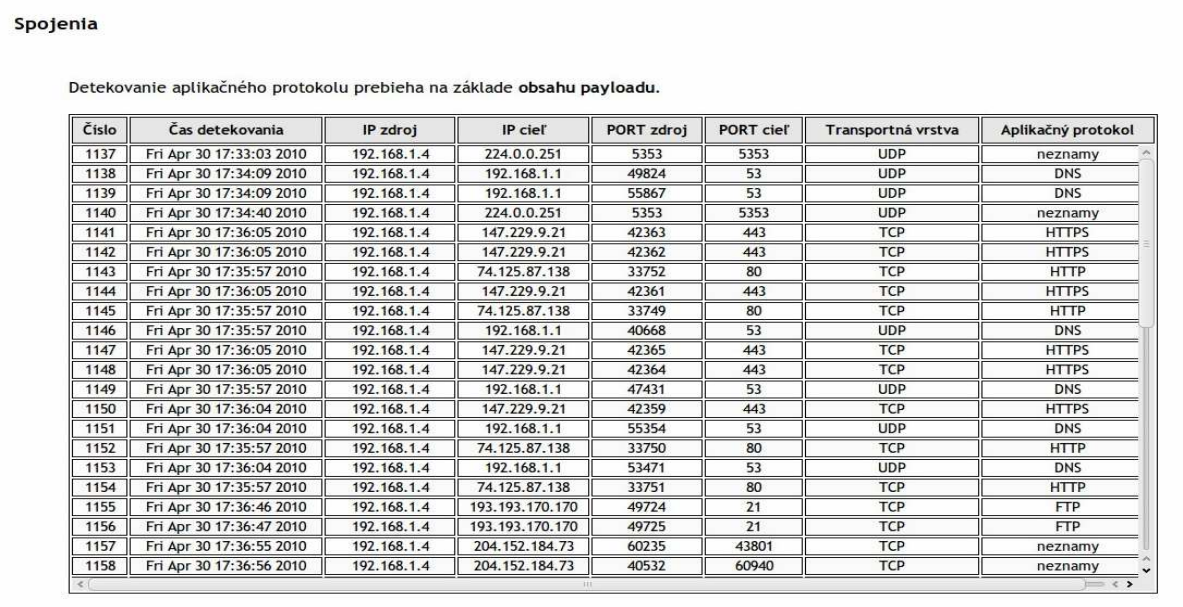
Stránka “Graf“ obsahuje 4 grafy zobrazujúce detekované protokoly najvyššej aplikačnej vrstvy modelu TCP/IP. Graf za posledných 15 minút, za poslednú 1 hodinu, za posledných 12 a posledných 24 hodín. Legenda ku grafu obsahuje názov aplikačného protokolu a k nemu počet výskytov.



Obrázok 4.8: Graf

4.3.6 Spojenia

Stránka “Spojenia“ obsahuje tabuľku zobrazujúcu posledných X (configurovateľné) vytvorených spojení. Jeden riadok tabuľky obsahuje čas vytvorenia spojenia, jednoznačnú identifikáciu spojenia tj. IP adresy, čísla portov a názov detekovaného aplikačného protokolu. Ďalej stránka informuje o metóde detekcie aplikačného protokolu.



Spojenia

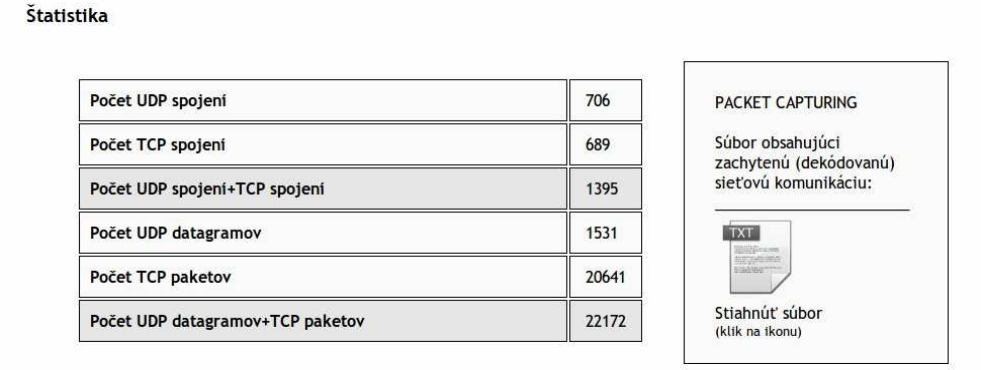
Detekovanie aplikačného protokolu prebieha na základe **obsahu payloadu.**

Číslo	Čas detekovania	IP zdroj	IP cieľ	PORT zdroj	PORT cieľ	Transportná vrstva	Aplikačný protokol
1137	Fri Apr 30 17:33:03 2010	192.168.1.4	224.0.0.251	5353	5353	UDP	neznamy
1138	Fri Apr 30 17:34:09 2010	192.168.1.4	192.168.1.1	49824	53	UDP	DNS
1139	Fri Apr 30 17:34:09 2010	192.168.1.4	192.168.1.1	55867	53	UDP	DNS
1140	Fri Apr 30 17:34:40 2010	192.168.1.4	224.0.0.251	5353	5353	UDP	neznamy
1141	Fri Apr 30 17:36:05 2010	192.168.1.4	147.229.9.21	42363	443	TCP	HTTPS
1142	Fri Apr 30 17:36:05 2010	192.168.1.4	147.229.9.21	42362	443	TCP	HTTPS
1143	Fri Apr 30 17:35:57 2010	192.168.1.4	74.125.87.138	33752	80	TCP	HTTP
1144	Fri Apr 30 17:36:05 2010	192.168.1.4	147.229.9.21	42361	443	TCP	HTTPS
1145	Fri Apr 30 17:35:57 2010	192.168.1.4	74.125.87.138	33749	80	TCP	HTTP
1146	Fri Apr 30 17:35:57 2010	192.168.1.4	192.168.1.1	40668	53	UDP	DNS
1147	Fri Apr 30 17:36:05 2010	192.168.1.4	147.229.9.21	42365	443	TCP	HTTPS
1148	Fri Apr 30 17:36:05 2010	192.168.1.4	147.229.9.21	42364	443	TCP	HTTPS
1149	Fri Apr 30 17:35:57 2010	192.168.1.4	192.168.1.1	47431	53	UDP	DNS
1150	Fri Apr 30 17:36:04 2010	192.168.1.4	147.229.9.21	42359	443	TCP	HTTPS
1151	Fri Apr 30 17:36:04 2010	192.168.1.4	192.168.1.1	55354	53	UDP	DNS
1152	Fri Apr 30 17:35:57 2010	192.168.1.4	74.125.87.138	33750	80	TCP	HTTP
1153	Fri Apr 30 17:36:04 2010	192.168.1.4	192.168.1.1	53471	53	UDP	DNS
1154	Fri Apr 30 17:35:57 2010	192.168.1.4	74.125.87.138	33751	80	TCP	HTTP
1155	Fri Apr 30 17:36:46 2010	192.168.1.4	193.193.170.170	49724	21	TCP	FTP
1156	Fri Apr 30 17:36:47 2010	192.168.1.4	193.193.170.170	49725	21	TCP	FTP
1157	Fri Apr 30 17:36:55 2010	192.168.1.4	204.152.184.73	60235	43801	TCP	neznamy
1158	Fri Apr 30 17:36:56 2010	192.168.1.4	204.152.184.73	40532	60940	TCP	neznamy

Obrázok 4.9: Spojenia

4.3.7 Štatistika

Stránka „Štatistika“ obsahuje prehľadnú tabuľku zobrazujúcu skutočný počet zachytených TCP paketov/UDP datagramov a skutočný počet detekovaných TCP/UDP spojení medzi zdrojovými a koncovými sieťovými rozhraniami (PC) od času zachytávania sieťovej komunikácie. Stránka ďalej umožňuje stiahnutie .txt súboru obsahujúceho výstupné hodnoty programu Pacpt (číslo paketu, hlavičky jednotlivých vrstiev paketu, veľkosť payloadu, payload). Obsah súboru záleží od spustiteľných parametrov programu Pacpt.



Štatistika

Počet UDP spojení	706
Počet TCP spojení	689
Počet UDP spojení+TCP spojení	1395
Počet UDP datagramov	1531
Počet TCP paketov	20641
Počet UDP datagramov+TCP paketov	22172

PACKET CAPTURING

Súbor obsahujúci zachytenú (dekódovanú) sieťovú komunikáciu:

TXT

Stiahnúť súbor
(klik na ikonu)

Obrázok 4.10: Štatistika

4.3.8 Popis webu

Stránka Popis webu obsahuje stručné vysvetlenie programu Pacpt ako aj popis webového rozhrania. Popisuje v nej použité implementované programovacie jazyky, výstupné hodnoty programu Pacpt, opis možných detekcií aplikačného protokolu a základných častí webu.

5 Testy vykonané na programe Pacpt

Pre overenie funkčnosti implementovaného analyzátora Pacpt boli uskutočnené rôzne testy.
Testy vykonané na programe Pacpt boli robené na sieti Ethernet s prenosovou rýchlosťou 100 Mb/s. Výsledky testov v ktorých sa meral čas záležia najviac od hardwarového vybavenia počítača a to hlavne na výkone procesoru.

Hardwarové vybavenie počítača na ktorom boli uskutočnené testy: procesor Intel Core 2 Duo P8400 (3M Cache, 2.26 GHz, 1066 MHz FSB), pamäť 3GB DDR3 RAM.

Testy uskutočnené na analyzátore boli robené pomocou unixových nástrojov Tcpdump [7, 14], Tcpreplay [14, 15] a programu Wireshark [4, 6].

Tcpdump

Tcpdump [7, 14] je nástroj k pasívnemu zachytávaniu sieťovej komunikácie prebiehajúcej sieťou. Umožňuje užívateľovi zobraziť informácie o paketoch prenášaných cez TCP protokol. Výhodou použitia tohto nástroja na odchytávanie sieťovej komunikácie je v možnosti nastavenia maximálnej veľkosti obsahu zachyteného TCP paketu v bytoch.
Príklad použitia nástroja Tcpdump:
- zachytávanie sieťovej komunikácie do súboru: *tcpdump -w capture.pcap -i eth0 -s 1024*

Tcpreplay

Tcpreplay [14, 15] je nástroj pre spätné prehrávanie zachytenej sieťovej komunikácie z uloženého súboru vytvoreného pomocou nástroja Tcpdump [7].
Príklad použitia nástroja Tcpreplay:
- spätné prehrávanie sieťovej komunikácie: *tcpreplay --mbps=100.0 --intf1=eth0 capture.pcap*

Wireshark

Program Wireshark [4, 6] je jeden z najznámejších sieťových analyzátorov distribuovaných zdarma pod open-source licenciou. Program Wireshark umožňuje čítať pakety zo siete, dekódovať a zobrazovať ich v zrozumiteľnej podobe.

Pre meranie času potrebného k vykonaniu určitej časti programu boli použité funkcie a štruktúry z hlavičkového súboru *time.h*.

Príklad merania času:

```
struct timespec start, stop;                          //štruktúry na uloženie začiatočného a
                                                      konečného času
clock_gettime( CLOCK_REALTIME, &start);               //zaznamenanie začiatočného času
.
.                                                     //vykonanie príkazov (časť programu)
.
clock_gettime( CLOCK_REALTIME, &stop);                //zaznamenanie konečného času
time_prg=(double)(stop.tv_nsec-start.tv_nsec);        //výpočet potrebného času na vykonanie
                                                      časti programu
```

5.1 Test základnej funkčnosti analyzátora Pacpt

Cieľom testu je overenie správnej činnosti analyzátora Pacpt. Myslí sa tým overenie činností: zachytávanie sieťovej komunikácie, dekódovanie hlavičiek jednotlivých protokolov, identifikácia jednotlivých dátových tokov a identifikácia aplikačných protokolov.

Správna činnosť programu Pacpt sa overí porovnaním zachytených údajov programom Wireshark [6]. Pri testovaní bola použitá vzorka dát (sieťová komunikácia) zachytená programom Tcpdump [7].

Popis základných krokov vykonaných pri testovaní:

1. Zachytenie sieťovej komunikácie slúžiacej ako vzorka dát pre testovanie.
Pomocou Tcpdump.
2. Spustenie programu Pacpt. Zachytávanie sieťovej komunikácie.
3. Spustenie nástroja Tcpreplay pre spätné prehranie zachytenej sieťovej komunikácie zachytenej nástrojom Tcpdump.
4. Vypnutie programu Pacpt.
5. Spustenie programu Wireshark. Zachytávanie sieťovej komunikácie.
6. Spustenie nástroja Tcpreplay pre spätné prehranie zachytenej sieťovej komunikácie zachytenej nástrojom Tcpdump.
7. Vypnutie programu Wireshark.
8. Porovnanie zachytenej sieťovej komunikácie programom Wireshark a programom Pacpt.

Vyhodnotenie testu

Test bol vykonaný rôznymi nastaveniami programu Pacpt v konfiguračnom súbore *config.txt*. Testovalo sa funkčnosť filtrácie zachytávania sieťovej komunikácie, detekcia aplikačných protokolov obidvoma metódami identifikácie popisovanými v tejto práci, dekódovanie hlavičiek protokolov a ďalšie.

Testom bolo zistené, že analyzátor Pacpt pracuje správne. Zachytená sieťová komunikácia programom Pacpt a programom Wireshark [6] bola zhodná. Zhodovali sa aj údaje o dekódovaných paketoch, údaje o detekovaných dátových tokoch a identifikovaných aplikačných protokoloch.

5.2 Meranie času

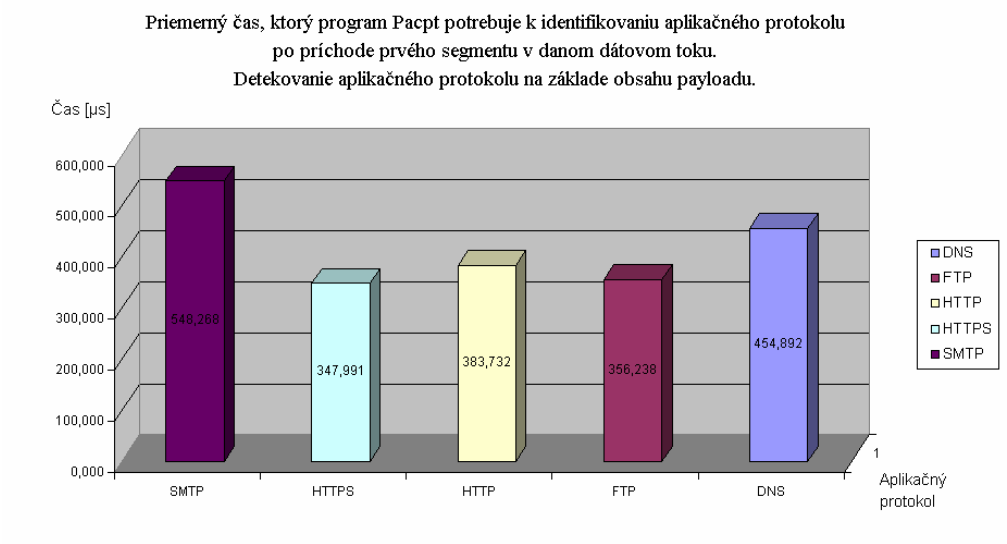
Meraním času sa rozumie, zachytenie času ktorý program Pacpt potrebuje k identifikovaniu aplikačného protokolu po príchode prvého segmentu v danom dátovom toku.

Meranie bolo vykonané nastavením obidvoch metód identifikácie aplikačného protokolu. Nastavenia konfiguračného súboru v dobe vykonávania merania: hash_t_size 7919, print_headers 3, payload yes, udp_timer 30, detect 1/2. Pre vysvetlenie jednotlivých konfiguračných nastavení viď. súbor *config.txt.* Pri vykonávaní merania nebol procesor zaťažený inými procesmi.

Meranie doby potrebnej k identifikovaniu aplikačného protokolu bolo robené na aplikačných protokoloch SMTP, HTTP, HTTPS, FTP, DNS. Výsledné časy pre dané protokoly sú priemerom päťdesiatich časových údajov.

5.2.1 Identifikovanie aplikačného protokolu z obsahu payloadu

Z nameraných hodnôt (časov) bol vytvorený graf. Graf slúži pre rýchly prehľad porovnania časov potrebných pre identifikáciu jednotlivých aplikačných protokolov, metódou identifikovania aplikačného protokolu z obsahu payloadu.



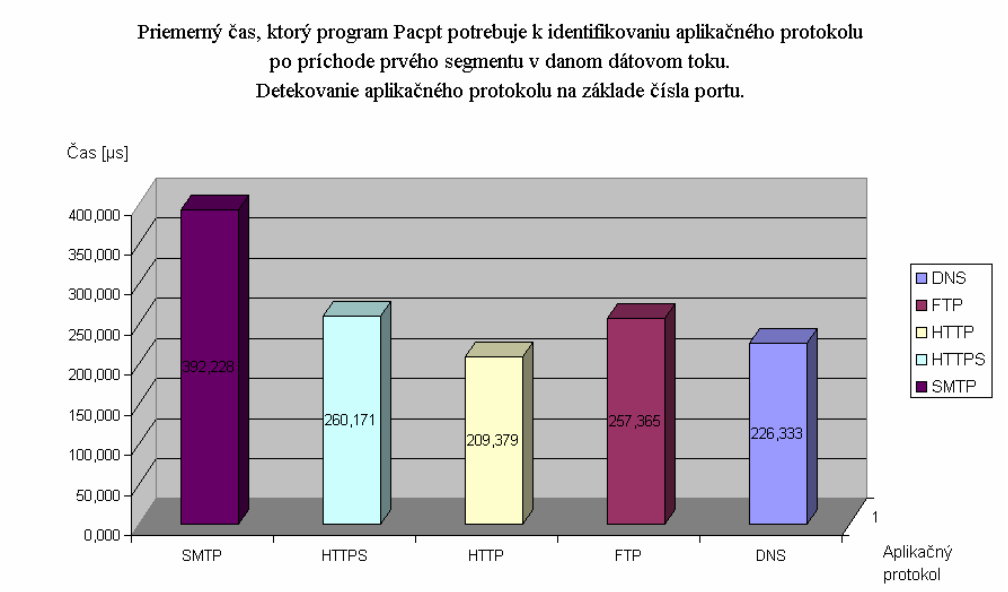
Graf 5.1: Graf zobrazuje namerané časové údaje potrebné pre identifikáciu daných aplikačných protokolov.

Na rýchlosť identifikovania aplikačného protokolu má veľký vplyv hustota obsahu hľadaných signatúr v paylode a zložitosť regulárnych výrazov, ktorými boli signatúry vyhľadávané. Keďže aplikovanie regulárnych výrazov na vyhľadávanie signatúr v paylode je sekvenčné t.j. v poradí v akom sú uložené za sebou, má poradie uloženia regulárnych výrazov taktiež veľkú váhu čo sa týka časovej náročnosti. Z čoho vyplýva, že zvýšením počtu podporovaných protokolov sa zvyšuje aj čas potrebný na identifikovanie aplikačného protokolu. Zvýšením počtu podporovaných protokolov aplikáciou Pacpt sa časová náročnosť k identifikovaniu aplikačného protokolu lineárne zväčšuje. Súvisí to s počtom prechodov potrebných k identifikovaniu aplikačného protokolu. Prechodom sa rozumie prechod deterministickým konečným automatom, ktorý popisuje jednotlivý aplikačný protokol.

Meraním bolo zistené, že priemerný čas, ktorý program Pacpt potrebuje pre identifikáciu aplikačného protokolu od príchodu prvého segmentu metódou vyhľadávania signatúr v paylode je 418 µs.

5.2.2 Identifikovanie aplikačného protokolu na základe čísla portu

Ako pri vyššie uvedenom meraní bolo uvedené, z nameraných hodnôt(časov) bol vytvorený graf. Graf slúži pre rýchly prehľad porovnania časov potrebných pre identifikáciu jednotlivých aplikačných protokolov metódou identifikovania aplikačného protokolu na základe čísla portu z transportného protokolu.



Graf 5.2: Graf zobrazuje namerané časové údaje potrebné pre identifikáciu daných aplikačných protokolov

Identifikovanie aplikačného protokolu na základe čísla portu z transportného protokolu je najrýchlejšou a najjednoduchšou metódou identifikovania aplikačného protokolu. V porovnaní s metódou identifikovania aplikačného protokolu na základe skúmania obsahu payloadu nie je potrebné skúmať obsah dát aplikačného protokolu, ale postačuje dekódovanie hlavičiek jednotlivých protokolov na daných vrstvách modelu TCP/IP. Tento fakt má veľký význam v časovej náročnosti identifikácie aplikačného protokolu.

Výsledkom merania bolo overenie teoretických znalostí. Identifikácia aplikačného protokolu na základe čísla portu z transportného protokolu je rýchlejšia ako identifikácia aplikačného protokolu na základe skúmania obsahu paylodu. Rýchlosť identifikovania

aplikačného protokolu na základe čísla portu oproti metóde identifikovania, kde sa preskúmava obsah payloadu je rádovo rýchlejšie (dva a viackrát).

Priemerný čas potrebný pre identifikáciu aplikačného protokolu od príchodu prvého segmentu metódou identifikovania aplikačného protokolu na základe čísla portu z transportného protokolu je 268.8 µs.

5.3 Presnosť identifikácie aplikačného protokolu

Test skúmania presnosti identifikácie aplikačného protokolu je založený na porovnaní presnosti detekcie aplikačného protokolu pre dané spojenie aplikáciou Pacpt a open-source nástrojom Wireshark [4, 6].

Ako bolo už v práci veľakrát spomínané, nástroj Pacpt umožňuje detekovanie aplikačného protokolu na základe čísla portu z transportnej vrstvy TCP/IP modelu a metódou založenou na skúmaní obsahu payloadu. Program Wireshark identifikuje aplikačný protokol iba na základe čísla portu z transportnej vrstvy TCP/IP modelu.

Pri teste boli použité unixové nástroje uvedené na začiatku 5. kapitoly. Pri tomto teste bol použitý rozširujúci nástroj T*cpwrite* [14, 15], ktorý obsahuje unixový nástroj *Tcpreplay* [14, 15]. Nástroj Tcpwrite slúžil na mapovanie preddefinovaného výstupného portu pre danú aplikáciu na iný ľubovoľný voľný port.

Príklad použitia nástroja Tcpwrite:

- mapovanie výstupného portu *tcprewrite --infile=capture.pcap --outfile=capture-portmap.pcap -- portmap=80:8080*

Po vytvorení súborov zachytávajúcich sieťovú komunikáciu jednotlivých protokolov pomocou nástroja Tcpdump boli sieťové komunikácie jednotlivo spätne prehrávané pomocou nástroja Tcpreplay. Pri spätnom prehrávaní zachytenej sieťovej komunikácii bola sieťová komunikácia zachytávaná nástrojom Pacpt ako aj programom Wireshark. Spätné prehrávanie zachytenej sieťovej komunikácie jednotlivých protokolov bolo vykonané viackrát, pričom sa prehrávala komunikácia na štandardný výstupný port, ale aj na výstupný port mapovaný užívateľom pomocou nástroja *Tcpwrite*.

Cieľom testu bolo overenie skutočnosti, že sieťové analyzátory, ktoré vykonávajú identifikáciu aplikačného protokolu iba na základe čísla portu nie sú schopné správne identifikovať aplikačný protokol po zmene štandardného výstupného portu. Dôkazom toho sú aj informácie zapísané v nižšie uvedenej tabuľke 5.3, ktorá zobrazuje údaje, ktoré boli zistené pri teste. Tabuľka 5.3 zobrazuje pre lepšiu názornosť iba časť získaných údajov pri teste.

		nástroj **Pacpt**		program **Wireshark**
		metóda identifikácie aplikačného protokolu		
aplikačný protokol/ preddef. cieľ. číslo portu	cieľové číslo portu (portmap)	na základe obsahu payloadu	na základe čísla portu	na základe čísla portu
		identifikovaný aplikačný protokol		
HTTP /80	80	HTTP	HTTP	HTTP
HTTP /80	21	HTTP	FTP	FTP
FTP /21	21	FTP	FTP	FTP
FTP /21	25	FTP	SMTP	SMTP
Telnet /23	23	Telnet	Telnet	Telnet
Telnet /23	80	Telnet	HTTP	HTTP
Skype/80	80	neznámy	HTTP	HTTP
DNS/53	53	DNS	DNS	DNS
DNS/53	80	DNS	HTTP	HTTP

Tabuľka 5.3: Zoznam identifikovaných aplikačných protokolov

5.4 Priepustnosť systému

Priepustnosť systému znamená, aký počet údajov vie analyzátor maximálne spracovať za jednu sekundu.

Priepustnosť systému patrí medzi najdôležitejšie vlastnosti sieťových aplikácií, ktoré slúžia na sledovanie a zachytávanie sieťovej prevádzky na danom segmente siete. Na priepustnosť systému navrhnutého analyzátora má veľký vplyv hardwarové vybavenie PC (sieťová karta, rýchlosť procesoru), momentálna vyťaženosť procesora a veľa iných. Medzi nastaviteľné vlastnosti, ktoré majú vplyv na priepustnosť navrhnutého analyzátora patrí nastavenie BPF [18] filtra, možnosť nastavenia metódy detekcie aplikačného protokolu ako aj možnosť nastavenia sprievodného výpisu na konzolu a výpisu do súboru. Po nastavení filtrovania sieťovej komunikácie obmedzenej napríklad iba na "udp", sa celková priepustnosť sieťového analyzátora zvýši. Je to dané architektúrou navrhnutého systému a to tým, že sieťová komunikácia sa filtruje pred tým, ako sa posielajú dáta samotnému programu Pacpt. Priepustnosť systému sa získava buď analytickým výpočtom alebo praktickým meraním a testovaním systému. Priepustnosť aplikácie *Pacpt* popisovanej v tejto práci bola odmeraná.

Štandardná sieťová komunikácia (HTTP, FTP, SHTTP, SMTP, DNS) bola zachytená do súboru pomocou nástroja *Tcpdump* [7].

Spätné prehrávanie sieťovej komunikácie bolo robené pomocou nástroja *Tcpreplay* [14, 15]. Pri spätnom prehrávaní sieťovej komunikácie bola využitá možnosť nastavenia rýchlosti odosielania údajov nástrojom Tcpreplay. Nástroj Tcpreplay umožňuje nastavenie väčšej rýchlosti odosielania údajov na sieťové rozhranie než akou boli údaje v skutočnosti zachytené.

Unixová knižnica *libpcap*, na ktorej je sieťový analyzátor postavený umožňuje volanie funkcie, ktorá vypíše výslednú štatistiku zobrazujúcu skutočný počet zachytených paketov analyzátorom a počet zahodených paketov (nespracovaných).

Meranie prebiehalo na princípe zaznamenávania strát údajov pri spätnom prehraní zachytenej sieťovej komunikácie rôznymi rýchlosťami. Po vykonaní testov tj. viacnásobným prehraním spätnej komunikácie a sledovaním počtu zahodených paketov analyzátorom Pacpt pri rôznych nastaveniach filtrovania sa dá odhadnúť, že výsledná priepustnosť navrhnutého sieťového analyzátora Pacpt sa pohybuje okolo 100 Mb/s.

6 Záver

Cieľom tejto bakalárskej práce bola implementácia sieťového analyzátora Pacpt spolu s prehľadným grafickým webovým rozhraním. Sieťový analyzátor bol implementovaný v programovacom jazyku C.

Medzi základné funkcie navrhnutého sieťového analyzátora patrí: zachytávanie a filtrovanie sieťovej komunikácie, dekódovanie hlavičiek jednotlivých protokolov, detekovanie vzniknutých TCP/UDP spojení ako aj identifikácia aplikačného protokolu a to nie len na základe čísla portu z transportného protokolu, ale aj metódou založenou na princípe vyhľadávania signatúr v obsahu dát aplikačného protokolu. Analyzátor využívajúc túto možnosť identifikovania aplikačného protokolu umožňuje aj identifikovanie aplikačných protokolov, ktoré využívajú dynamické prideľovanie čísiel portov.

Medzi veľké výhody navrhnutého sieťového analyzátora patrí aj komunikácia s webovým rozhraním. Webové rozhranie umožňuje vzdialený prehľad o sieťovej komunikácii daného segmentu siete, kde beží pasívne monitorovanie sieťovej komunikácie programom Pacpt.

V práci sú popísané základné teoretické znalosti k pochopeniu princípu fungovania sieťového analyzátora Pacpt, implementácia ako aj výsledky testov vykonaných na programe Pacpt.

Výsledky testov ukázali, že sieťový analyzátor Pacpt je plne fungujúcim nástrojom. Testované boli obidve metódy identifikácie aplikačného protokolu. Priemerný čas potrebný k identifikovaniu aplikačného protokolu od príchodu prvého segmentu v danom dátovom toku metódou na základe čísla portu je 268.8 µs a metódou identifikovania aplikačného protokolu z obsahu payloadu je 418 µs. Ďalej bola testovaná presnosť identifikácie aplikačného protokolu sieťovým analyzátorom Pacpt a open-source nástrojom *Wireshark* [4, 6]. Testom bolo zistené, že sieťové analyzátory, ktoré identifikujú aplikačné protokoly iba na základe čísla portu (*Wireshark*), nedokážu správne identifikovať aplikačný protokol po zmene štandardného výstupného portu na iný, čo znamená, že sa stávajú nepresnými. Posledným meraním na navrhnutom sieťovom analyzátore Pacpt bolo zistené, že priepustnosť navrhnutého systému sa pohybuje okolo 100 Mb/s.

Pokračovanie vývoja navrhnutého sieťového analyzátora má veľký význam kvôli zvyšujúcemu sa dopytu na trhu po nástrojoch umožňujúcich prehľad sieťovej komunikácie sietí v domácnostiach i na pracoviskách. Ďalší vývoj by spočíval v zrýchlení zachytávania sieťovej komunikácie, v zrýchlení identifikácii dátových tokov, v pridaní podpory identifikácie ďalších

protokolov a vo vývoji webového rozhrania smerujúceho k možnému ovládaniu sieťového analyzátora cez Internet.

Literatúra

[1] E. Holzschlag, M.: HTML a CSS jdi do toho. Praha, Grada Publishing, 2006. ISBN 80-247-1454-X

[2] E. Hugh,W., Lane,D.: PHP a MySQL: Vytváříme webové databázové aplikace. Praha, Computer Press, 2002. ISBN 80-7226-760-4

[3] Václavek,P.: JavaScript Hotová řešení. Brno, Computer Press, 2003. ISBN 80-7226-854-6

[4] Orebaugh,A., Ramirez,G., Burke,J., Morris,G., Pesce,L., Wright, J.: Wireshark a Ethereal Kompletní průvodce analýzou a diagnostikou sítí. Brno, Computer Press, 2008. ISBN 978-80-251-2048-4

[5] Packet capturing s libpcap. [online],[rev. 2009-07-05],[cit. 2010-04-24].
URL http://www.linuxos.sk/clanok/345/index.html

[6] Wireshark. [online],[rev. 2010-03-09],[cit. 2010-04-24].
URL http://www.wireshark.org/

[7] Tcpdump. [online],[rev. 2010-04-06],[cit. 2010-04-24].
URL http://www.tcpdump.org/

[8] L7-filter. [online],[rev. 2009-01-07],[cit. 2010-04-24].
URL http://l7-filter.sourceforge.net/

[9] Snort. [online],[rev. 2010-02-24],[cit. 2010-04-24].
URL http://www.snort.org/

[10] IANA. [online],[rev. 2010-04-20],[cit. 2010-04-24].
URL http://www.iana.org/assignments/port-numbers

[11] Zoznam známych portov. [online],[rev. 2010-04-20],[cit. 2010-04-24].
URL http://sk.wikipedia.org/wiki/Zoznam_známych_portov

[12] Dostálek,L., Kabelová,A.: Velký průvodce protokoly TCP/IP a systémem DNS. Praha, Computer Press, 2000. ISBN 80-7226-323-4

[13] Cracking a obrana – pasivní odposlech. [online],[rev. 2009-03-28],[cit. 2010-04-25].
URL http://fei.abba.cz/~st15886/sos/

[14] Studijní opora k predmetu ISA. [online],[rev. 2009-09-22],[cit. 2010-04-25].
URL http://www.fit.vutbr.cz/study/courses/index.php?id=6299

[15] Tcpreplay. [online],[rev. 2010-03-28],[cit. 2010-04-25].
URL http://tcpreplay.synfin.net/

[16] Use case diagram. [online],[rev. 2010-04-2],[cit. 2010-04-25].
URL http://en.wikipedia.org/wiki/Use_case_diagram/

[17] Google Chart Tools. [online],[rev. 2010-04-10],[cit. 2010-04-25].
URL http://code.google.com/intl/sk-SK/apis/charttools/

[18] Berkeley Packet Filter. [online],[rev. 2010-02-18],[cit. 2010-04-25].
URL http://en.wikipedia.org/wiki/Berkeley_Packet_Filter/

[19] jQuery. [online],[rev. 2010-04-20],[cit. 2010-05-04].
URL http://jquery.com/

[20] Skype. [online],[rev. 2010-05-1],[cit. 2010-05-05].
URL http://skype.com/

[21] AES. [online],[rev. 2010-04-20],[cit. 2010-04-05].
URL http://cs.wikipedia.org/wiki/Advanced_Encryption_Standard

[22] RSA. [online],[rev. 2010-04-20],[cit. 2010-04-05].
URL http://cs.wikipedia.org/wiki/RSA

Zoznam príloh

Obsah CD

Priložené CD obsahuje:

- adresár **webove_rozhranie** - obsahuje zdrojové súbory k webovému rozhraniu k programu Pacpt
- adresár **Pacpt –** obsahuje zdrojové a hlavičkové súbory programu Pacpt
- adresár **dokumentacia** – obsahuje textovú časť(dokumentáciu) k bakalárskej práce
- adresár **test –** obsahuje súbor *test.sql* a *test.pcap*. Súbor *test.sql* obsahuje testovacie údaje na naplnenie MySQL databáze. Súbor *test.pcap* obsahuje zachytenú sieťovú komunikáciu na možnosť otestovania programu Pacpt.
- **README.pdf –** popisuje obsah CD(jeho štruktúru), inštrukcie k inštalácii a príklad spustenia projektu.
- **db.sql** – SQL skript na vytvorenie databáze