VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

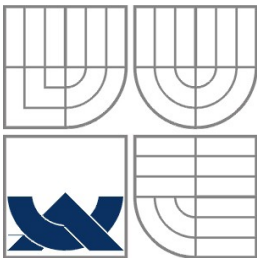
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

WEBOVÝ PORTÁL PRO ANOTACI SÍŤOVÉHO PROVOZU

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

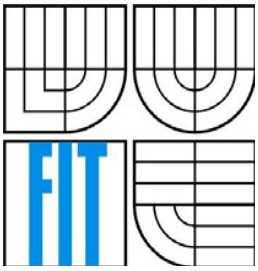
AUTOR PRÁCE PETER MÁLIK
AUTHOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

WEBOVÝ PORTÁL PRO ANOTACI SÍŤOVÉHO PROVOZU

Web Portal for Network Traffic Anotation

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — PETER MÁLIK
AUTHOR

VEDOUCÍ PRÁCE — Ing. MARTIN ŽÁDNÍK
SUPERVISOR

BRNO 2009

Abstrakt

Předmětem bakalářské práce je vytvorění systému pro anotaci síťového provozu. Za tímhle účelem je potřebné správne určení protokolů na všech vrstvách síťového modelu TCP/IP. Pro identifikaci protokolů aplikační vrstvy se využívá metoda založená na číslech portů a metoda založená na vyhledávání signatur v dátové části paketu pomocí regulárních výrazů. Webový portál je vytvořen v jazyce PHP s využitím rozšíření Phpcap a programu tcpdump, který zabezpečuje manuální anotaci síťového provozu.

Kľúčové slová

Model TCP/IP, síťový provoz, vyhledávání řetězců, phpcap, payload

Abstract

Bachelor´s thesis aim is to create a system for network traffic anotation. For this purpose it is necessary to correctly determine the protocols for all layers of the network model TCP/IP. To identify the application layer protocol is used a method based on port numbers and a method based on tracing the signatures in packet´s data field by using regular expressions. Web portal was written in PHP, using the extension phpcap and tcpdump program, which provides manual anotation of network traffic.

Keywords

TCP/IP model, network traffic, pattern maching, phpcap, payload

Citácia

Peter Málik: Webový portál pro anotaci síťového provozu, bakalářská práce, Brno, FIT VUT v Brně, 2009

Webový portál pro anotaci síťového provozu

Prehlásenie

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

….....
Peter Málik
19.mája 2009

Poďakovanie

Týmto by som sa chcel poďakovať Ing. Martinovi Žádníkovi za odbornú pomoc a konzultácie pri tvorbe tejto práce.

© Peter Málik, 2009.
Táto práca vznikla ako školské dielo na Vysokom učení technickom v Brne, Fakulte informačných technológií. Práca je chránená autorským zákonom a jeho použitie bez udelenia oprávnenia autorom je nezákonné, s výnimkou zákonom definovaných prípadov.

Obsah

1 Úvod

V dnešnej dobe čoraz viac vzrastá popularita a využívanie internetu. Vyvíjajú sa stále nové aplikácie, ktoré sú využívané či už vo firmách alebo v domácnostiach na komunikáciu, zdieľanie súborov, atď. Mohli by sme povedať, že svet sa stáva „IP-centrickým“ s veľkým množstvom pripojených zariadení každý deň. So zväčšovaním sa počtu zariadení a služieb na internete rastie aj množstvo nových protokolov, ktoré popisujú pravidlá komunikácie. K neustálemu rozširovaniu internetu sa viaže množstvo výhod ale aj problémov. Jedným z problémov môže byť napríklad zneužitie prístupu na internet.

Monitorovanie sieťovej komunikácie, inak nazvané sieťová analýza, je v dnešnej dobe potrebné z dôvodu detekovania aplikačných protokolov prípadne použitých aplikácií, pretože len dôkladná znalosť sieťovej komunikácie pomáha sieťovým administrátorom, bezpečnostným technikom prípadne programátorom úspešne diagnostikovať problémy či neoprávnené prístupy na sieti. Anotácia sieťových dát je nesmierne dôležitá práve kvôli tomu, aby sme boli schopní presne určiť zloženie sieťovej komunikácie, najmä z pohľadu aplikačných protokolov a aplikácií, ktoré ju využívajú. Z hľadiska hodnotenia systémov pre automatickú identifikáciu aplikácií je potrebné mať k dispozícií veľké množstvo dát na porovnanie a hlavne dát overených priamo užívateľom. Anotácia je využitá aj pri navrhovaní pomoci IDS (Intrusion detection systems) systémom, ktoré musia poznať aplikáciu, aby mohli vyhľadať príslušný škodlivý vzor. Presné dekódovanie sieťových, prenosových a aplikačných protokolov tiež umožňuje ľahké vyhľadanie problémov na sieti, ako je napríklad strata paketov, e-mailov a pod. Jedným z najznámejších sieťových analyzátorov je Wireshark [1], ktorý je schopný odhaliť protokoly na rôznych vrstvách sieťového modelu. Na analýzu aplikačných protokolov využíva len čísla portov, žiadnym spôsobom neanalyzuje payload paketu. Avšak pre správne a presné zistenie protokolu je často potrebná práve analýza payloadu. Medzi nástroje, ktoré využívajú analýzu dátovej časti paketu, patrí napríklad L7-filter [2], ktorý využíva regulárne výrazy na popis protokolov. Osobitnou kapitolou sú nástroje ako napríklad Snort [3] a Bro [4] tzv. IDS, ktoré majú za úlohu odhaľovať útoky alebo vírusy prenášané prostredníctvom počítačovej siete.

Druhá kapitola sa venuje popisu sieťového modelu TCP/IP a jeho jednotlivých vrstiev, na základe ktorých je možné pochopiť princípy sieťovej komunikácie. Podrobne je tu vysvetlený prenos dát, ich zapuzdrenie pomocou hlavičiek a využitie protokolov na jednotlivých vrstvách TCP/IP. Ďalej sú tu popísané rôzne metódy určené na identifikáciu aplikačných protokolov, prípadne samotných aplikácií.

Tretia kapitola pojednáva o implementácií systému pre analýzu sieťovej komunikácie. Systém analyzuje jednotlivé dátové toky spojenia a na základe čísel portov, prípadne vyhľadania určitých reťazcov v payloade paketu, určuje aplikačný protokol. Navrhnutý analyzátor využíva rôzne spôsoby určovania dátových tokov a protokolov nielen aplikačnej, ale aj nižších vrstiev.

Práca sa v štvrtej kapitole venuje testovaniu navrhnutého systému a jeho výsledkom. Bolo potrebné vykonať množstvo testov, podľa ktorých bolo možné určiť úspešnosť klasifikácie aplikačných protokolov. Rovnako bol testovaný aj čas potrebný pre samotnú analýzu.

Piata kapitola je záver, v ktorom sa nachádza krátke zhrnutie implementácie navrhnutého systému a jeho hodnotenie. Sú tu prítomné námety na ďalší vývoj a krátke zhrnutie práce s programom.

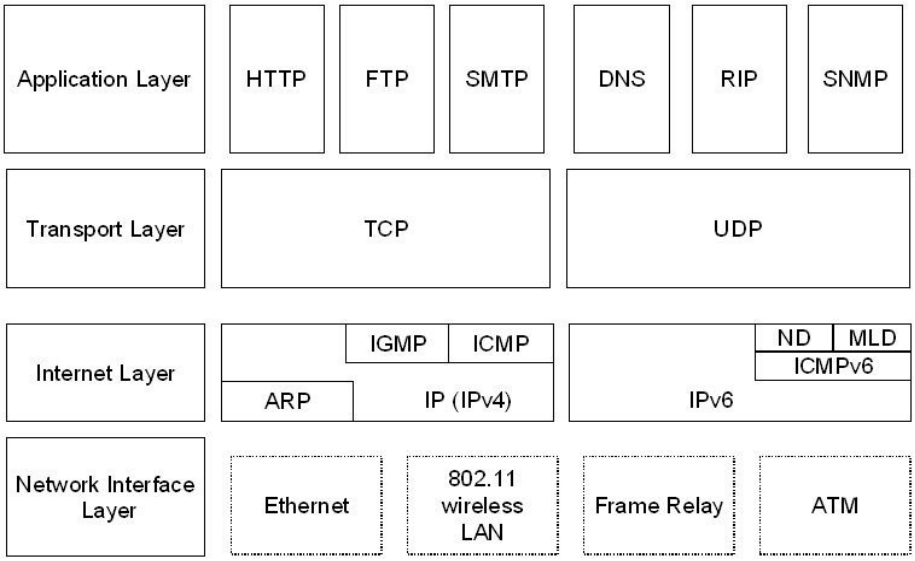
V šiestej kapitole je výpis všetkých prameňov, z ktorých som čerpal.

Siedma kapitola obsahuje zoznam príloh.

2 Teoretický rozbor

2.1 Model TCP/IP

Model TCP/IP (obrázok 2.1) má na rozdiel od ISO/OSI modelu jednoduchšiu a efektívnejšiu architektúru rozdelenú na štyri vrstvy. Hlavnou koncepciou tohto modelu je abstrakcia od fyzickej a linkovej vrstvy, definícia protokolu sieťovej vrstvy IP (internet protokol) a definícia transportných protokolov TCP a UDP. Model je veľmi robustný a odolný voči výpadkom siete.



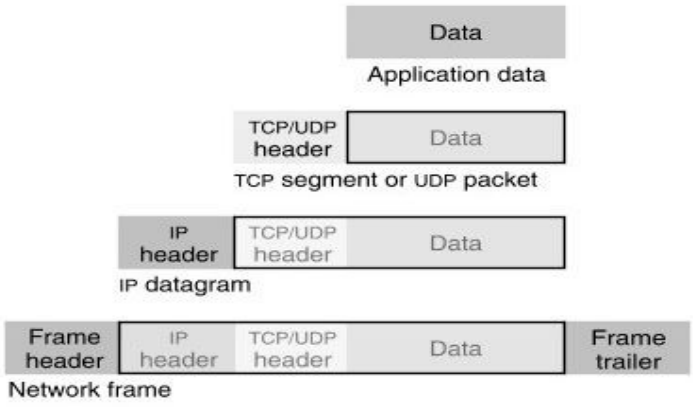
Obrázok 2.1 : Sieťový model TCP/IP

- **Vrstva sieťového rozhrania** špecifikuje ako sú pakety prenášané cez fyzickú vrstvu avšak nie je modelom TCP/IP bližšie špecifikovaná, pretože je závislá na použitej prenosovej technológií.
- **Sieťová vrstva** rieši problémy prenosu paketov medzi sieťami a zakrýva rozdiely medzi jednotlivými prenosovými technológiami. Na tejto vrstve sú definované protokoly ICMP, IGMP, ARP, IPv4 a nová verzia Ipv6. IP je schopný niesť dáta množstva rozličných vysokoúrovňových protokolov a stará sa o úlohy prenosu paketov zo zdroja do cieľa.
- **Transportná vrstva** ponúka spojovanú aj nespojovanú komunikáciu. Protokol TCP (Transmission Control Protocol) [5] je spoľahlivý transportný mechanizmus, ktorý zabezpečuje že dáta budú prenesené úplné, nepoškodené a v správnom poradí. UDP protokol (User Datagram Protocol) [6] je nespojový a nezaručuje spoľahlivý prenos dát.

- **Aplikačná vrstva** je vrstva bežne používaná väčšinou programov pracujúcich so sieťou na komunikáciu prostredníctvom siete. Procesy na tejto vrstve závisia na konkrétnej aplikácií a dáta sú kódované do štandardného protokolu. Táto vrstva obsahuje protokoly ako napr. HTTP, DHCP, DNS, FTP, SSH, IMAP, atď. [7]

2.2 Prenos dát

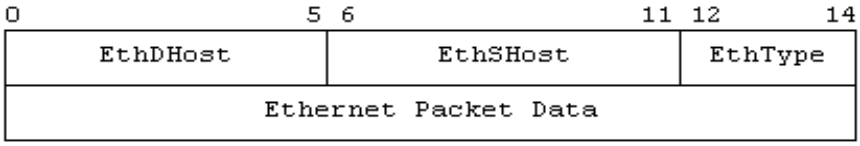
Dáta prenášané cez sieť sa rozdeľujú na časti nazývané PDU. Od fyzickej vrstvy sú to podľa modelu ISO/OSI postupne : bity (bits), rámce (frames), datagramy (datagrams), pakety (segments, packets) a dáta (data). Pri prenose dát dochádza k zapuzdrovaniu, ktoré poskytuje abstrakciu protokolov. Aplikácie medzi sebou komunikujú na najvyššej vrstve (aplikačnej) a využívajú pritom nižšie vrstvy, ktoré pridávajú k dátam potrebné informácie. Komunikácia je možná len medzi rovnakými vrstvami sieťového modelu, ktoré sú popísané rovnakými protokolmi.
Princíp a postup zapuzdrovania sa nachádza na obrázku 2.2.



Obrázok 2.2 : Postup zapuzdrovania

2.2.1 Vrstva sieťového rozhrania (Ethernet)

Ethernet je jeden z typov lokálnych sietí, ktorý realizuje vrstvu sieťového rozhrania. Z pohľadu analýzy protokolov je dôležité poznať formát Ethernetového rámca. Ten obsahuje : Preambulu, Príznak začiatku rámca, Cieľovú adresu (fyzická MAC adresa o dĺžke 48 bitov), Zdrojovú adresu (MAC), Typ protokolu alebo dĺžku, Dáta, Dátovú výplň a Kontrolný súčet (Checksum). Avšak nie všetky tieto údaje sú podstatné z hľadiska analýzy sieťových protokolov. Potrebná je iba hlavička, ktorá zahrňuje cieľovú a zdrojovú adresu a typ, a ktorá je spolu s dátami zobrazená na obrázku 2.3.

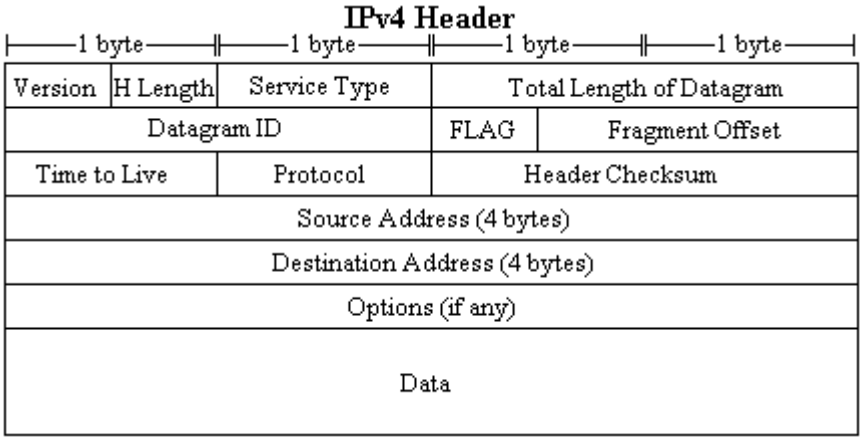


Obrázok 2.3 : Formát Ethernetového rámca

2.2.2 Sieťová vrstva

2.2.2.1 Protokol IP

Protokol IP je dátovo orientovaný komunikačný protokol sieťovej vrstvy, používaný zdrojovou a cieľovou stanicou na výmenu dát s prepínaním paketov. Dáta v IP sieti sa posielajú v blokoch nazývaných datagramy. Každý datagram má špecifický formát hlavičky pre určitý protokol. V modely TCP/IP je týmto protokolom IPv4 [8] alebo IPv6 [9]. IP spolu s transportnou vrstvou tvoria základ dnešného internetu. Na obrázku 2.4 je formát datagramu IPv4. Dôležitými prvkami pri analýze sú Total Length (Celková dĺžka datagramu), Protocol (určuje typ protokolu na vyššej vrstve), Source a Destination Address (Zdrojová a cieľová IP adresa).



Obázok 2.4 : Formát IPv4 datagramu

2.2.2.2 ARP a RARP

Protokol ARP (Address Resolution Protocol) [10] sa v počítačových sieťach používa k získavaniu Ethernetovej (MAC) adresy susedného stroja. Môže sa použiť pre preklad MAC adries rôznych protokolov na sieťovej vrstve.
Naopak RARP (Reverse Address Resolution Protocol) [11] sa používa k získaniu IP adresy počítača pri znalosti MAC adresy. Prakticky sa však RARP nepoužíva, pretože pre automatickú konfiguráciu staníc sa častejšie nasadzujú lepšie protokoly, ako napríklad DHCP alebo BOOTP [7].

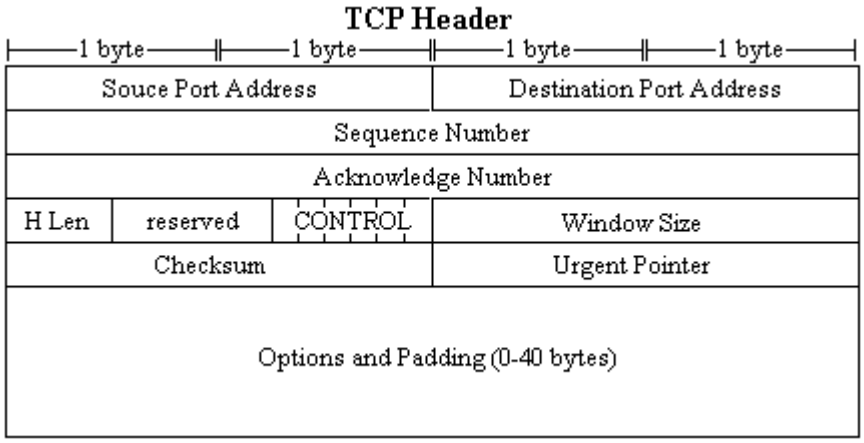
2.2.2.3 ICMP

Protokol ICMP (Internet Control Message Protocol) [12] sa používa pre odosielanie chybových správ a je súčasťou protokolu IP. Svojím účelom sa líši od TCP a UDP protokolov tým, že sa obvykle nepoužíva sieťovými aplikáciami priamo (jedinou výnimkou je nástroj *ping*).

2.2.3 Transtportná vrstva

2.2.3.1 TCP

Protokol TCP (Transmission Control Protocol) [5] je, rovnako ako UDP a IP, jedným zo základných protokolov internetu. Poskytuje spoľahlivý prenos dát medzi aplikáciami, pričom využíva služby IP protokolu opakovaným odosielaním stratených paketov a usporiadaním prijatých paketov do správneho poradia. Je to stavový protokol a preto zaisťuje riadenie spojenia a dátového toku. K rozlíšeniu komunikujúcich aplikácií používa TCP protokol čísla portov. Každá strana TCP spojenia má pridružené jedno 16 bitové bezznamienkové číslo portu. Tie sú rozdelené do troch skupín : Dobre známe, Registrované a Dynamické (Privátne). Na obrázku 2.5 sa nachádza formát hlavičky protokolu TCP.

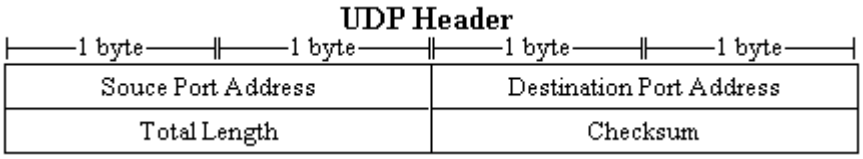


Obrázok 2.5 : Formát hlavičky protokolu TCP

Pre analýzu sú dôležité tieto hodnoty : Source a Destination Port Address (Zdrojové a Cieľové číslo portu), Sequence a Acknowledge Number (Sekvenčné a Potvrdzovacie číslo) a Control Flags (Kontrolné príznaky).
Sekvenčné a potvrdzovacie čísla slúžia na detekciu straty paketu alebo na usporiadanie prijatých paketov do správneho poradia. Kontrolné príznaky sú dôležité z hľadiska nadväzovania (three way hand shaking, obrázok 3.1) a ukončovania spojenia (obrázok 3.2).
Výhodou protokolu TCP je spoľahlivosť a jeho schopnosť rozlišovať dáta pre viacnásobné súčasne bežiace aplikácie na jednom počítači.

2.2.3.2 UDP

Protokol UDP (User Datagram Protocol) [6] je rovnako ako TCP jedným zo sady protokolov internetu, avšak je oveľa jednoduchší. Nie je spojovaný a preto nezaručuje správny prenos dát ani ich doručenie v správnom poradí. Preto nepotrebuje ani sekvenčné a potvrdzovacie číslo a jeho hlavička je jednoduchšia ako vidieť na obrázku 2.6.



Obrázok 2.6 : Formát hlavičky protokolu UDP

Pre analýzu sú potrebné len Source a Destination Port Address (Zdrojové a Cieľové číslo portu). UDP je nespoľahlivý a preto slúži na rýchly prenos dát (VoIP, audio-video-streaming, online hry) alebo pre aplikácie pracujúce systémom otázka-odpoveď (DNS [7]).

2.2.4 Aplikačná vrstva

Ako Payload označujeme segment (paket) bez TCP alebo UDP hlavičky. Jednotlivé aplikácie komunikujúce po sieti si medzi sebou vymieňajú dáta na základe určitého aplikačného protokolu (HTTP, FTP, DNS, atď. [7]) a payload sú vlastne tieto dáta. Mnohé nástroje analyzujúce aplikačné protokoly pracujú práve na základe informácií obsiahnutých v payloade. Ten môže mať formu ASCII textu, ktorý ako výstup používajú programy Wireshark [1] a tcpdump [13], alebo binárnych dát (pcap súbory).

2.3 Metódy identifikácie sieťovej komunikácie

Identifikácia typu protokolu na vrstve sieťového rozhrania, sieťovej vrstve a transportnej vrstve nie je zložitá. Táto informácia je obsiahnutá v hlavičkách protokolov nižšej vrstvy. V prípade sieťového rozhrania je dôležité poznať typ dátovej linky. Typ protokolu (množiny protokolov, aplikácie) na aplikačnej vrstve nie je možné spoľahlivo určiť na základe hlavičky segmentu z transportnej vrstvy. Tá poskytuje iba informáciu, na ktorom porte daná služba beží.
Existuje veľké množstvo spôsobov klasifikácie sieťovej komunikácie [14]. Jedným z najstarších je identifikácia podľa čísel portov transportnej vrstvy. Tieto porty sú registrované organizáciou IANA [15] a reprezentujú známe aplikácie. Vzrastajúca popularita aplikácií, ktoré používajú dynamické prideľovanie portov, alebo využívajú známe porty iných aplikácií (napríklad Peer-to-Peer, P2P), sa odzrkadľuje na spoľahlivosti tejto metódy a tá sa stáva čoraz viac nepostačujúcou [16].

Naproti tomu je určovanie aplikačného protokolu pomocou vyhľadávania špecifických reťazcov (tzv. Signatúr) v payloade paketu oveľa presnejšie a spoľahlivejšie. Táto metóda však vyžaduje veľký výpočtový výkon a nefunguje pri zašifrovaných dátach. Spôsobuje tiež závažné narušenie súkromia, pretože z odchytených dát je možno získať niektoré chúlostivé informácie, napr. prihlasovacie mená a heslá. Preto je vhodné používať pri analýze metódy, ktoré sa vyhýbajú kontrole obsahu payloadu. Prvá z nich je založená na správaní sa staníc a jednotlivých pripojení. Získavajú sa tým informácie o spojeniach a ich interakcie s ostatnými zložkami siete (internetu) [17].
Druhá metóda skúma vlastnosti dátových tokov, ako napr. celkový čas spojenia, počet a veľkosť paketov alebo čas medzi jednotlivými paketmi [18].
Je veľmi ťažké určiť, ktorá z týchto metód je najlepšia, alebo ktorá sa najviac hodí pre analýzu spojení odchytených na rôznych miestach a za rôznych podmienok.

2.3.1 Identifikácia aplikačných protokolov pomocou portov

Identifikácia pomocou portov je najjednoduchším spôsobom rozpoznávania aplikácie a závisí len od informácie o čísle portu v hlavičke protokolu transportnej vrstvy. Napriek tomu je to najrýchlejšia metóda. Jej úspešnosť, pri klasifikácií dátových tokov, je podľa niektorých štúdií len 70% [14]. Mnohé aplikácie v dnešnej dobe používajú štandardné čísla portov na svoje zamaskovanie. Dôveryhodné určenie aplikácie je v takomto prípade nemožné. Výsledky analýz, využívajúcich porty, preto kolíšu v závislosti od charakteru odchytených dát. Za hlavný nedostatok tejto metódy je považovaná nemožnosť overenia správnosti identifikácie dátových tokov. Najznámejším programom pracujúcim na tomto princípe je Wireshark [1].

	Metódy identifikácie	Príklady aplikácie
I	Klasifikácia na základe portov	-
II	Hlavička paketu (vrátane I)	jednosmerný tok
III	Signatúry v jednom pakete	červy, vírusy
IV	Sémantika protokolu v jednom pakete	IDENT
V	Signatúry v prvom KByte	P2P
VI	Sémantika protokolu v prvom KByte	SMTP
VII	Sémantika protokolu vybraného toku	FTP
VIII	Sémantika protokolu celého toku	VNC, CVS
IX	História pripojenia	port-scanning

Obrázok 2.7 : Jednotlivé kroky identifikovania aplikácie

2.3.2 Identifikácia aplikácie pomocou payloadu

Pri klasifikácií dátových tokov pomocou tejto metódy sa využíva kontrola obsahu, teda dát, paketu. Niektoré aplikácie sa dajú určiť na základe signatúr obsiahnutých v týchto dátach. Signatúry sú reťazce, ktoré sa dajú popísať regulárnymi výrazmi, a sú jednoznačné pre každú aplikáciu. Ak je dobre definovaná množina regulárnych výrazov, dochádza k veľmi presnému určeniu aplikačného protokolu. Z tohto dôvodu sa táto metóda používa, pri porovnávaní ostatných, pre stanovenie skutočnej aplikácie. Jeden zo systémov využívajúcich tento spôsob klasifikácie je založený na postupnej analýze a potvrdzovaní jej výsledkov [16]. Tento systém v prvom rade, a ak je to možné, určí aplikáciu na základe portu. V prípade neúspechu, či nedostatočnej dôveryhodnosti, pokračuje postupne ďalšími krokmi (obrázok 2.7). Vyhľadávanie signatúr a sémantiky protokolu sa aplikuje na dáta (payload). Deje sa tak postupne na stále väčšej vzorke dát. Vo väčšine prípadov je postačujúci prvý KByte spojenia. Výsledkom celého procesu sú jednotlivé dátové toky a k nim priradené aplikácie.

2.3.3 Metóda založená na správaní spojenia

Monitorovanie správania sieťového spojenia je potrebné z dôvodu výskytu čoraz väčšieho množstva škodlivého softwaru a zároveň čoraz hlasnejšieho volania o ochranu súkromných údajov. Bolo vyvinuté aby zachytilo interaktivitu medzi jednotlivými stanicami a to dokonca aj pri komunikácií v šifrovanej forme. Programy pracujúce na tomto princípe vytvárajú tzv. profil stanice, zachytávaním portov a cieľových destinácií, s ktorými komunikuje. Porovnaním týchto prvkov správania sa so vzormi správania aplikačných serverov dochádza k určeniu aplikácie.
Na tejto metóde je položený aj jeden z algoritmov identifikujúcich P2P aplikácie [17]. Ten vytvára páry zdrojových a cieľových IP adries využívajúcich oba protokoly, TCP aj UDP, pričom z nich odfiltrováva protokoly, ktoré komunikujú prostredníctvom známych čísiel portov. Preveruje tiež všetky páry IP adresa – port (srcIP-srcport, dstIP-dstport) a hľadá tie, pri ktorých sa počet odlišných IP adries približne rovná počtu odlišných portov. Tieto dve jednoduché heuristiky efektívne identifikujú väčšinu P2P komunikácie.

2.3.4 Metóda založená na vlastnostiach dátových tokov

Metóda skúma a využíva jednotlivé charakteristické vlastnosti dátových tokov. V spojitosti s ňou sa využívajú algoritmy strojového učenia. Pre tie sú potrené určité dáta (tréningové dáta), pomocou ktorých sú schopné sa učiť a popri tom vyhotovovať model, ktorý presne sedí na tieto dáta.

Táto metóda našla svoje uplatnenie pri zostavovaní algoritmu na detekciu VoIP (Voice over Internet Protocol) komunikácie [18]. Ako prvá bola vykonaná analýza dátových tokov, z ktorej boli určené charakteristické znaky jednotlivých typov komunikácia a samozrejme aj samotných dátových tokov, využívaných VoIP aplikáciami. Vzájomným porovnávaním a dôkladným pozorovaním bolo možné určiť, že tieto aplikácie posielajú (prijímajú) väčší počet paketov za sekundu ako ostatné, pričom veľkosť samotného paketu je niekoľko násobne nižšia. Napokon vznikol algoritmus, ktorý je schopný podľa týchto jedinečných vlastností identifikovať a klasifikovať akúkoľvek VoIP komunikáciu.

3 Systém pre analýzu sieťovej komunikácie

V tejto kapitole je popísaný návrh riešenia a samotná implementácia systému pre analýzu sieťovej komunikácie. Tento systém má za úlohu čo najpresnejšie určiť protokol aplikačnej vrstvy prípadne samotnú aplikáciu. Ako implementačný jazyk sa využíva jazyk PHP. Tento jazyk poskytuje vysokú mieru abstrakcie a preto je pomalší ako napríklad jazyk C, avšak pre potrebu vytvorenia webového portálu sa javí ako najlepšia možnosť.

3.1 Odchytené dáta

Navrhnutý systém je zameraný na analýzu dát vopred odchytenej komunikácie. K tomuto účelu slúži množstvo programov, ktoré sieťovú komunikáciu ukladajú do súborov pcap. Medzi tieto programy patria napríklad Wireshark [1] alebo tcpdump [13].
Unixové systémy, pre ktoré je tento webový portál určený implementujú prácu s pcap súbormi pomocou knižnice libpcap napísanej v jazyku C. Pre naše potreby sme však využili PHP modul nazývaný phpcap [19]. Toto rozšírenie PHP je implementáciou knižnice libpcap a zaručuje prístup ku všetkým pcap funkciám. Modul je rozdelený na 2 rozšírenia : phpcap a pkttool.
Kým phpcap implementuje, ako bolo vyššie spomínané, pcap funkcie, pkttool zaručuje prístup k funkciám, ktoré dokážu dekódovať odchytené pakety.
Hlavička každého paketu obsahuje informácie potrebné k analýze prislúchajúceho dátového toku. Táto hlavička je uložená v dynamickom poli PHP a obsahuje tieto hodnoty :

- čas odchytenia paketu (v sekundách aj mikrosekundách)
- počet bytov paketu, ktoré sú dostupné v zachytenom súbore
- celkový počet bytov paketu

Phpcap nám poskytuje ukazateľ na začiatok každého paketu v súbore a zároveň na hlavičku. Navrhnutý systém dovoľuje odchytávať určité časti komunikácie, napríklad v Ethernete len prvých 14 bytov a tak analyzovať danú časť (nap. Ethernetovú hlavičku).
Pomocou pkttool systém detekuje nielen časť komunikácie, ale všetky vrstvy sieťového modelu a preto odchytáva celé rámce vrátane payloadu.

3.2 Dekódovanie komunikácie

Dekódovanie, a teda samotná analýza odchytených dát, začína na prvej vrstve modelu TCP/IP (vrstve sieťového rozhrania). Analýzu je potrebné vykonávať postupne, pretože nie je možné ihneď analyzovať aplikačnú vrstvu. Takáto analýza je nevyhnutná, lebo na rôznych vyšších vrstvách sú použité rôzne protokoly a tie nemusia mať pevne stanovenú dĺžku hlavičky. Z toho vyplýva, že sa nedá napríklad z Ethernetového rámca určiť kde sa nachádza samotný payload paketu. Kedže payload paketu je vstupom pre ďalšie fázy analyzátoru, treba sa k nemu postupným dekódovaním hlavičiek protokolov na jednotlivých vrstvách prepracovať. Protokol vyššej vrstvy je závislý, ako už bolo spomenuté v teoretickom rozbore, od hlavičky protokolu nižšej vrstvy.
Jednotlivé hlavičky protokolov a ich formáty, ktoré analyzátor podporuje sú popísané v kapitole Teoretický rozbor. Nástroj pkttool obsahuje funkcie pre dekódovanie špecifických hlavičiek a následné vrátenie ich hodnôt v dynamickom poli. Analyzátor dokáže dekódovať a následne analyzovať iba štandard Ethernet, ktorý popisuje fyzické rozhranie a komunikáciu na LAN sieťach. Jeho hlavička obsahuje dôležitú hodnotu „ether_type“, pomocou ktorej dokážeme určiť typ protokolu sieťovej vrstvy. Ak je táto hodnota väčšia ako 600 hexadecimálne, potom táto hodnota predstavuje typ protokolu inak sa jedná o dĺžku odchytených dát. Navrhnutý systém podporuje na sieťovej vrstve protokoly IPv4[8], ARP[10] a RARP[11]. Pri protokoloch ARP a RARP dochádza k ukončeniu analýzy. Ďalšia funkcia z rozšírenia pkttool zabezpečuje dekódovanie hlavičky protokolu IPv4. Najdôležitejšími položkami v hlavičke sú : celková dĺžka datagramu, zdrojová a cieľová IP adresa a IP protokol. Práve podľa poslednej položky tj. IP protokolu je systém schopný rozpoznať protokoly použité na transportnej vrstve. Prípadne, čo je nezvyčajné, dokáže analyzátor detekovať aj protokol ICMP [12] práve pomocou hodnoty tejto položky. Pri detekcii protokolu ICMP dochádza rovnako ako v prípade ARP a RARP, k ukončeniu analýzy.
Následne sú detekované a dekódované hlavičky protokolov TCP a UDP, ktorých štruktúra a formát sú popísané na obrázkoch 2.5 a 2.6.
Pri každom jednom kroku dekódovania komunikácie dochádza k odstráneniu hlavičiek jednotlivých protokolov. Po odstránení hlavičky protokolu transportnej vrstvy získame posunutím ukazateľa na príslušnú pozíciu samotný payload paketu. Počas celého procesu dekódovania je rovnako dôležité zhromažďovať určité informácie, ako sú napríklad zdrojové a cieľové IP adresy a porty alebo flagy TCP paketu.

3.3 Monitorovanie dátových tokov

Pre navrhnutý systém je dôležité sieťovú komunikáciu, načítanú zo súboru, triediť do dátových tokov. Samotná analýza aplikačného protokolu sa nevykonáva na každom pakete zvlášť, ale práve na daných dátových tokoch. Dátový tok je možné jednoznačne určiť pomocou zdrojových a cieľových IP adries a čísel portov. Tieto informácie spoľahlivo identifikujú spojenie medzi dvoma koncovými bodmi v sieti.

Navrhnutý analyzátor prechádza postupne všetky pakety obsiahnuté v súbore odchytenej komunikácie. Pre každý paket sa vytvára jedinečný kód tvorený IP adresou zdroja, IP adresou cieľa, zdrojovým portom, cieľovým portom a protokolom transportnej vrstvy vo formáte :

IP_source::IP_destination::source_port::destination_port::TCP/UDP

Posledná položka je závislá na použitom protokole transportnej vrstvy. Tento kód určuje odchytený dátový tok.
Zároveň sa vytvára aj takzvaný reverzný kód vo formáte :

IP_destination::IP_source::destination_port::source_port::TCP/UDP

Pri každom ďalšom odchytenom pakete sa zisťuje či patrí do určitého spojenia (dátového toku), ktoré sa už v systéme nachádza. Ak sa daný kód, prípadne jeho reverzná forma, v systéme ešte nenachádza, vytvorí sa nová položka pola popisujúceho dátové toky. Jednotlivé položky tohto pola obsahujú v prípade TCP :

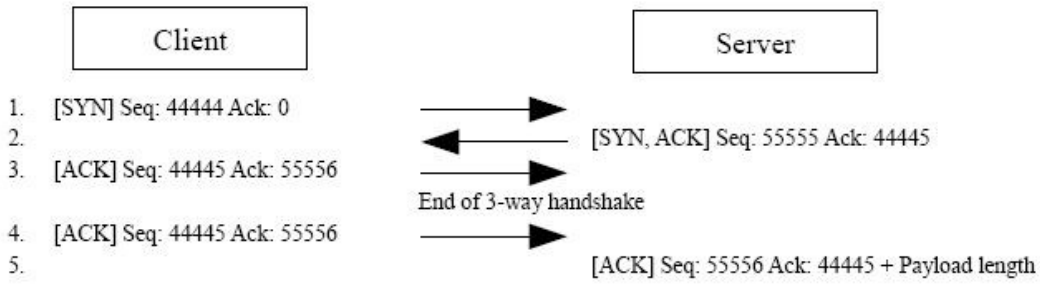
- kód daného dátového toku
- stav spojenia
- IP adresu zdroja a cieľa
- sekvenčné a potvrdzovacie číslo
- počet paketov
- celkovú veľkosť v KByte
- začiatok v sekundách a mikrosekundách
- koniec v sekundách a mikrosekundách
- protokol aplikačnej vrstvy určený pomocou portu
- protokol aplikačnej vrstvy určený pomocou signatúry v payloade

Položky pola obsahujúceho dátové toky protokolu UDP sú rovnaké, avšak nie je potrebné si pamätať IP adresy a sekvenčné a potvrdzovacie číslo. IP adresy sú obsiahnuté aj v samotnom kóde určujúcom dátový tok.

3.3.1 Rozpoznávanie stavu spojenia

Protokol transportnej vrstvy TCP je stavový tzn. že princíp jeho fungovania je založený na určitých stavoch. Pri každom spojení, a teda aj dátovom toku, je potrebné kontrolovať v akých stavoch sa momentálne nachádza. Rovnako treba tieto stavy v určitých prípadoch aj meniť. Zmeny stavov sa odohrávajú na transportnej vrstve, teda na protokoloch TCP a UDP. Podľa kódu daného dátového toku (dátového spojenia) sme schopní určiť, ktorý koncový bod spojenia je klient, a ktorý je server.

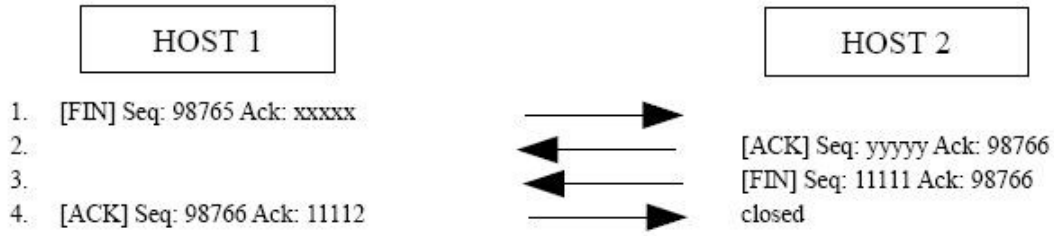
Detekcia IP adries z kódu je však výpočtovo náročná, a preto je v poli daného spojenia uložená aj zdrojová a cieľová IP adresa. Zdrojová IP adresa zodpovedá klientovi a cieľová IP adresa serveru. Nadväzovanie spojenia za pomoci protokolu TCP sa nazýva three-way handshake. Táto metóda je zobrazená na obrázku 3.1.



Obrázok 3.1 : Nadväzovanie spojenia pomocou protokolu TCP

Analyzátor dokáže detekovať flagy TCP paketu a na ich základe vykonávať rôzne operácie v rôznych prípadoch. Pri ustanovovaní spojenia klient posiela serveru SYN paket, podľa ktorého systém určí na ktorej strane spojenia je klient a na ktorej server, a zároveň sa vytvorí nový dátový tok, v prípade že ešte neexistuje, a stav daného spojenia sa nastaví na SYN_SENT. Server odpovedá zaslaním paketu s flagmy SYN a ACK, pričom sa spojenie dostáva do stavu SYN_RCVD. Nakoniec, aby mohlo byť spojenie úspešne nadviazané, klient zasiela serveru paket s flagom ACK a spojenie je v stave ESTABLISHED. Server nemusí prijať požiadavku klienta na nadviazanie spojenia. V tomto prípade mu posiela paket s flagom RST. Týmto sa spojenie ukončí, pretože ide o reset spojenia zo strany servera. Po týchto troch paketoch nasleduje samotný prenos dát, ktoré sa ukladajú do špecifických polí, ktoré budú popísané v ďalšej podkapitole.
V prípade, že bolo spojenie nadviazané ešte pred začiatkom odchytávania dát do súboru, toto spojenie pravdepodobne nebude možné určiť pomocou three-way handshake, a preto sa pri každom zaslaní paketu s flagom ACK kontroluje či daný paket patrí k nejakému dátovému toku. Pokiaľ nie, vytvorí nový dátový tok so stavom nastaveným na ESTABLISHED. Po úspešnom prenose dát sa spojenie vo väčšine prípadov uzatvára spôsobom nazývaným four-way handshake, zobrazenom na obrázku 3.2. Strana, ktorá chce spojenie uzatvoriť, posiela paket s flagom FIN, pričom sa stav mení na FIN_WAIT. Druhá strana odpovedá paketom ACK a stav je CLOSE_WAIT. Toto sa nazýva pasívne uzatvorenie spojenia, v ktorom môže ešte druhá strana posielať ďalšie pakety obsahujúce dáta. Po ich prenose zasiela aj ona paket s flagom FIN a stav spojenia sa mení na TIME_WAIT, na čo následne odpovedá stanica, ktorá chcela spojenie ukončiť ako prvá, paketom ACK a spojenie je úspešne ukončené a v stave CLOSED. Pri tomto stave sa výsledné informácie o dátovom toku zapíšu do výsledkového súboru a dochádza k uvoľneniu prostriedkov potrebných pre ich uloženie. Rovnako sa uvoľnia polia pre zber dát zo strany klienta a servera.

Ak od odchytenia posledného paketu v danom spojení uplynul určitý čas, toto spojenie sa stáva z pohľadu systému ukončené. Je zavedený takzvaný časovač, ktorý vždy po tomto čase kontroluje čas odchytenia posledného paketu spojenia a porovnáva ho s aktuálnym časom. Pri neaktivite spojenia dlhšej ako je nastavený čas pre daný protokol, sa toto spojenie stáva ukončeným a uvoľnia sa polia obsahujúce payload. Časy, po ktorých uplynutí sa detekuje neaktívne spojenie, je možné upraviť v nastaveniach programu.
Systém ošetruje aj iné spôsoby uzatvárania spojenia [5], avšak pre názornosť bol uvedený len ten základný.



Obrázok 3.2 : Ukončovanie spojenia pomocou protokolu TCP

Protokol transportnej vrstvy UDP je bezstavový, ale analyzátor poskytuje stavy aj pre neho. Prvý paket v spojení určí, ktorá strana je klient a ktorá server a zároveň sa vytvorí nová položka pola dátových tokov UDP so špecifickým kódom a stavom UDP_REQ. Odpoveď zo strany servera zmení stav spojenia na UDP_ACTIVE. Pri spojení UDP je nemožné určiť koniec spojenia na základe analýzy UDP paketu, preto sa využíva opäť časovač, ktorý podľa času posledného odchyteného paketu v dátovom spojení zisťuje či je daný dátový tok aktívny. Rovnako ako pri protokole TCP, ak uplynie určitý čas, spojenie je ukončené, pričom toto je jediný spôsob ako ho ukončiť. Po uzavretí sa informácie o danom dátovom toku zapíšu do výsledkového súboru a dochádza k uvoľneniu prostriedkov potrebných pre uloženie dát daného dátového toku.
Pri oboch protokoloch, TCP aj UDP, analyzátor považuje časovačom ukončené spojenia za neaktívne. Môže sa stať, že to nie je pravda. V takomto prípade je pokračujúce spojenie detekované ako nové a kedže nie je k dispozícii počiatočná komunikácia, jeho identifikácia pravdepodobne zlyhá.

3.3.2 Spracovanie payloadu

Pre potreby analyzátora je potrebné odchytávať 1 Kbyte prenášaný zo strany klienta a 1 KByte prenášaný zo strany serveru. Táto počiatočná komunikácia obsahuje signatúry, podľa ktorých systém detekuje aplikačný protokol prípadne aplikáciu.

V prípade TCP spojenia sa samotné dáta ukladajú do polí, ktoré obsahujú špecifický kód, sekvenčné číslo a tieto dáta. Pri každom novom pakete sa kontroluje veľkosť dát uložených v poli. Ak táto hodnota presiahne 1024 (1 Kbyte), dáta v poli sa usporiadajú podľa sekvenčných čísiel, ktoré k nim prislúchajú. Následne sa vykoná samotná analýza payloadu a tieto dáta sú označené ako skontrolované (CHECKED). Tieto polia sú usporiadané do dvoch polí, jedno pre dáta od klienta a druhé pre dáta od serveru. Obe sa uvoľnia pri ukončovaní spojenia.

UDP spojenie nepoužíva sekvenčné čísla, preto nie je možné kontrolovať či dáta prichádzajú v správnom poradí. Do polí určených pre klientskú a serverovú časť komunikácie sa preto ukladajú v takom poradí, v akom prichádzajú. Z toho istého dôvodu sa ukladajú až prvé 2 KByte z každej strany a následne ich nie je potrebné usporiadavať. Rovnako ako pri TCP dátach sa vykonaná analýza dát UDP tokov a tieto dáta sú označené ako skontrolované (CHECKED). Polia určené pre zachytenie payloadu zo strany klienta a servera sa uvoľnia pri ukončení spojenia.

3.4 Identifikácia aplikačného protokolu

Systém je navrhnutý tak, aby využíval rôzne druhy filtrov. Tieto sa nachádzajú v externých súboroch.

3.4.1 Analýza pomocou portu

Súbor, v ktorom sa nachádzajú názvy jednotlivých protokolov a im prislúchajúcich portov, má formát :

protocol_name/port_number/transport_protocol

Protocol name je názov aplikačného protokolu.
Port number definuje číslo portu prislúchajúce danému protokolu.
Transport protocol určuje či daný protokol využíva služby TCP alebo UDP.

Čísla portov spolu s protokolmi sú prevzaté od organizácie IANA [15].
Mená protokolov, spolu s číslami portov, sú uložené do 2 polí, jedno pre TCP a druhé pre UDP, čo urýchľuje určenie aplikačného protokolu počas analýzy.
Číslo portu spojenia sa určuje z hlavičky protokolu transportnej vrstvy. Pri každom novom dátovom toku sa určí port, na ktorom naslúcha server a porovná sa s číslami portov uloženými v poliach. Po určení protokolu sa výsledok zapíše do pola tohto dátového toku.

3.4.2 Analýza payloadu

Pri analýze payloadu sa pre popis signatúr aplikačných protokolov používajú regulárne výrazy prevzaté z linuxového L7-filtra [2]. Regulárny výraz je textový reťazec, ktorý opisuje alebo pasuje na množinu reťazcov na základe určitých syntaktických pravidiel. Regulárne výrazy sa najčastejšie používajú pre vyhľadávanie a úpravu textu. V navrhnutom systéme majú úlohu vyhľadávania tzv. pattern matching. Regulárny výraz sa skladá z literálov textu, ktoré sa majú zhodovať a špeciálnych znakov, ktoré nie sú súčasťou hľadaného textu slúžiacich pre popis alternatív, množín, počtu výskytov a prepínačov.
Formát súboru, v ktorom sú zapísané jednotlivé protokoly a im prislúchajúce regulárne výrazy :

protocol_name;;regular_expression;;transport_protocol;;credibility

Protocol name je názov aplikačného protokolu prípadne aplikácie.
Regular expression je samotný regulárny výraz popisujúci aplikačný protokol (aplikáciu).
Transport protocol určuje či daný protokol využíva služby TCP alebo UDP.
Credibility určuje dôveryhodnosť s akou je možné po uskutočnení analýzy payloadu označiť výsledok ako pravdivý.

Podobne ako pri portoch sú aj regulárne výrazy uložené do 2 polí, jedno pre TCP a druhé pre UDP. Výrazne to znižuje čas trvania analýzy.
PHP využíva na určenie zhody regulárneho výrazu s dátami spojenia funkciu PREG_MATCH [20]. Keďže L7-filter využíva regulárne výrazy kompatibilné s programovacím jazykom PERL [21], tieto výrazy treba upraviť. Na to slúži funkcia ADDCSLASHES [20], ktorá do regulárneho výrazu pridá spätné lomítka pred znaky, ktoré sú v PHP označené, ale v PERL-y nie, ako metaznaky : /:<>=! .
Samotná analýza payloadu prebieha pri odchytení prvého paketu obsahujúceho dáta. Tie sa uložia do pola určeného pre dáta zaslané klientom a vykoná sa ich analýza pomocou už spomínanej funkcie PREG_MATCH. V prípade, že aplikačný protokol nebol určený systém pokračuje v ukladaní dát do pola. Pri odpovedi serveru sa dáta uložia do pola pre dáta serveru a vykoná sa analýza aj týchto dát. Ak sa ani teraz nepodarí určiť aplikačný protokol, čaká sa kým nie je veľkosť dát v oboch poliach väčšia ako 1 KByte v prípade TCP a 2 KByte v prípade UDP. Po dosiahnutí týchto hodnôt sa analýza oboch polí vykonáva znova. Ak sa podarí určiť aplikačný protokol, tak je priradený do prislúchajúceho dátového toku a neskôr zapísaný do výsledkového súboru spolu s hodnotou dôveryhodnosti.

3.5 Prezentácia výsledkov analýzy

Výsledky analýzy sú prezentované 2 spôsobmi. Tým prvým sú súbory nazvané rovnako ako vstupný pcap súbor s príponami *.nl* a *.tl* .
Súbor s príponou .tl obsahuje výsledky analýzy pre sieťový protokol IPv4. Tento súbor má nasledujúci formát :

start_s::start_ms::time::IP_src::IP_dst::port_src::port_dst::proto::count::length::port_proto::
L7_proto::credibility::user_proto

Start_s je čas začiatku spojenia vo formáte timestamp.
Start_ms je čas začiatku spojenia v mikrosekundách.
Time je celkový čas trvania spojenia.
IP_src, IP_dst sú zdrojová a cieľová IP adresa.
Port_src, port_dst sú zdrojový a cieľový port.
Proto je protokol transportnej vrstvy (TCP alebo UDP).
Count je počet paketov prenesených v danom dátovom toku.
Length určuje celkovú veľkosť dát prenesených v dátovom toku v bytoch.
Port_proto určuje výsledok identifikácie aplikačného protokolu pomocou známych portov.
L7_proto je výsledok určenia aplikačného protokolu prípadne aplikácie pomocou vyhľadania signatúr L7-filtru v payloade spojenia.
Credibility určuje dôveryhodnosť regulárneho výrazu L7-filtra v prípade úspešnej analýzy.
User_proto je aplikačný protokol určený užívateľom pri manuálnej identifikácií.
Ukončený znakom „\n“, ktorý predstavuje ďalší riadok.

Súbor s príponou .nl obsahuje spojenia využívajúce protokoly sieťovej vrstvy ARP, RARP a ICMP. Pri týchto typoch protokolu neprebieha analýza aplikačného protokolu, preto aj formát súboru je jednoduchší a obsahuje len prvých 5 položiek, okrem hodnoty time a hodnotu proto obsiahnutých vo formáte súboru .tl.

Druhým spôsobom prezentácie výsledkov je vytvorenie tabuľky prístupnej užívateľovi na portáli hneď po ukončení analýzy. Táto tabuľka obsahuje rovnaké informácie ako súbor s príponou .tl, avšak s drobnými rozdielmi. Dátové toky sú očíslované a ich začiatok je prevedený do formátu : hodiny:minúty:sekundy.mikrosekundy. Užívateľ má možnosť prezrieť si prvých 10 paketov každého dátového toku vo formáte v akom ich prezentuje program tcpdump [13]. Podľa toho môže určiť aplikačný protokol a ten následne zapísať do položky PROTO-výsledné. Jeho rozhodovania uľahčujú prednastavené hodnoty tejto položky na základe dôveryhodnosti. Tú je možné nastaviť pre : porty do 1024, porty od 1024, porty celkovo, L7-filter a celkovú dôveryhodnosť.

Ak celková dôveryhodnosť presiahne hodnotu nastavenú užívateľom, je protokol určený analýzou vpísaný do položky PROTO-výsledné a program tým dáva užívateľovi najavo, že tento protokol je z jeho pohľadu určený správne. Po identifikácií všetkých dátových tokov sa hodnoty, užívateľom alebo programom určených aplikačných protokolov, zapíšu do súboru s príponou .tl na pozíciu user_proto.

4 Testovanie navrhnutého systému a jeho výsledky

Navrhnutý systém bol podrobený rôznym druhom testov. Najdôležitejším kritériom pri jeho hodnotení je percentuálna úspešnosť jednotlivých metód určovania aplikačných protokolov. Ďalším nemenej dôležitým kritériom je časová náročnosť. Tú však nie je možné objektívne zhodnotiť, pretože závisí od použitého hardwaru. Pamäťové nároky na systém nie sú vysoké. Pri bežnom užívaní boli najvyššie namerané hodnoty na úrovni 30 Mbytov. Pri kopírovaní súborov na server je však požadovaná minimálna veľkosť 1024 MiB RAM, kvôli dočasnému uloženiu súboru do tejto pamäte. Testy boli uskutočnené na počítači s procesorom Intel Core 2 Duo, 2,26 GHz, 3,0 GiB RAM.

4.1 Percentuálna úspešnosť

Testy prebiehali nad viacerými rôznymi sadami dát. Prvá sada bola poskytnutá vedúcim bakalárskej práce a odchytená dňa 12.12.2008 a 11.01.2009. Obsahuje dáta z akademickej siete. Druhá sada použitá pri testovaní pochádza z internátnej siete pripojenej pomocou statickej IP adresy. A tretia sada je odchytená na domácej sieti schovanej za NAT. Dáta boli zachytené pomocou programu tcpdump [13] v dňoch 14.05.2009 a 17.05.2009.

Dôveryhodnosti, pri ktorých boli dátové toky označené ako správne identifikované :

- porty do 1024 – 90
- porty od 1024 – 50
- porty celkovo – 100
- l7-filter – 90
- celková – 60

Ak celková dôveryhodnosť, vypočítaná z ostatných hodnôt dôveryhodností, presiahla hranicu 60% bol užívateľovi poskytnutý tento výsledok ako prednastavený text pomáhajúci pri manuálnej anotácií. Tú bolo potrebné vykonať pre správne identifikovanie tokov, ktoré systém označil za neurčené. Avšak vznikala tu aj potreba racionálne vysvetliť výsledky testov. Tie v rozpore s očakávaniami vykazovali nízke percento úspešne klasifikovaných dátových tokov.
Regulárne výrazy využívané pri analýze, na vyhľadávanie signatúr, museli byť v niektorých prípadoch pozmenené kvôli ich nižšej kvalite a nefunkčnosti. Príkladom je regulárny výraz popisujúci DNS, v ktorom bol vynechaný jeden z úvodných znakov, charakteristický pre tento protokol.

Prvé tri stĺpce všetkých tabuliek uvedených v tejto kapitole reprezentujú výsledky testov, vykonávaných na dátach odchytených z akademickej siete. Ďalšie dva sú z internátnej siete a napokon posledný odráža sieťovú komunikáciu na domácej sieti.

4.1.1 Percento úspešne rozpoznaných tokov

Tento test má za úlohu percentuálne určiť množstvo úspešne identifikovaných tokov odhalených analyzátorom. Najdôležitejší prvok pri výpočte úspešnosti je celkový počet tokov určených z daného súboru. Nasleduje počet tokov odhalených jednotlivými metódami (pomocou portov a pomocou regulárnych výrazov l7-filtra). Celkový úspech klasifikácie je určený pri zhode týchto dvoch postupov. Do výsledkov a konečného hodnotenia úspešnosti systému boli pridané aj hodnoty, ktoré boli užívateľovi navrhnuté ako pravdepodobne správne. To sa udialo na základe hore spomínaných hodnôt dôveryhodnosti, pričom podmienkou nebola detekcia pomocou oboch metód. Zvyšné dátové toky sú v tabuľke 4.1 označené ako neidentifikované. Dôležitým poznatkom je aj počet tokov komunikujúcich pomocou TCP [5] a UDP [6].

	11-1-09_1	**11-1-09_2**	**12-12-08_2**	**14-5-09_0**	**14-5-09_3**	**17-5-09_0**	**Priemer**
Celkový počet tokov	**5573**	**5087**	**3530**	**3098**	**2446**	**2064**	**3633**
Celkovo podľa portu	5495	5043	3458	2659	2227	1861	3457
	98,60%	99,14%	97,96%	85,83%	91,05%	90,16%	95,16%
Celkovo podľa l7-filtra	4855	3952	2869	1556	1360	1115	2618
	87,12%	77,69%	81,27%	50,23%	55,60%	54,02%	72,06%
Zhoda protokolov	3085	751	598	566	426	617	1007
	55,36%	14,76%	16,94%	18,26%	17,42%	29,89%	27,72%
Neidentifikované toky	1477	2620	1734	1548	1026	923	1555
	26,50%	51,50%	49,12%	49,97%	41,95%	44,72%	42,80%
Dôvera podľa portu	996	1696	1188	942	982	518	1054
	17,87%	33,34%	33,65%	30,41%	40,15%	25,10%	29,01%
Dôvera podľa l7-filtra	15	20	10	42	12	6	18
	0,27%	0,39%	0,28%	1,36%	0,49%	0,29%	0,50%
Počet TCP tokov	4952	3978	2936	240	272	678	2176
	88,86%	78,20%	83,17%	7,75%	11,12%	32,85%	59,90%
Počet UDP tokov	621	1109	594	2858	2174	1386	1457
	11,14%	21,80%	16,83%	92,25%	88,88%	67,15%	40,10%

Tabuľka 4.1 : Percentuálna úspešnosť klasifikácie dátových tokov

Už na prvý pohľad je vidieť veľký rozdiel medzi dátami odchytávanými na školskej a domácej sieti. V prvých menovaných prevláda TCP , kým v druhých UDP komunikácia, avšak len čo do počtu dátových tokov.

Avšak ešte zaujímavejšie sú výsledky samotnej analýzy. Vo všetkých sadách prevládala, podľa predpokladov, metóda založená na využití čísiel portov. Druhý riadok, teda rozpoznanie aplikácie podľa regulárneho reťazca však vykazuje určité nepresnosti. Sieťová komunikácia v domácej sieti a pod systémom Ubuntu, na ktorom boli testy uskutočnené, obsahuje veľké množstvo požiadaviek a odpovedí zasielaných len v rámci lokálnej siete. Bližším preskúmaním sa zistilo, že tieto nepresnosti sú spôsobené použitím protokolov ako LLMNR (komunikuje ako DNS), či ws-discovery (multicast na lokalizáciu služieb, napr. „People near Me“), ktoré nie sú popísané v l7-filtri. Prvý test sa svojou presnosťou vymyká ostatným. Je to najmä vďaka komunikácii pomocou HTTP protokolu, ktorá je ľahko rozpoznateľná. Vo zvyšných prípadoch je zhoda aplikácií percentuálne veľmi nízka. Je to dané viacerými faktormi.

- Už spomínané obmedzenie analýzy pomocou l7-filtra len na najznámejšie (http, dns, ssh atď.) alebo naopak v Európe exotické protokoly (100bao, kugoo, goboogy atď.). Pričom testované dáta obsahovali nepodporované protokoly.
- Aj keď je analýza úspešná pomocou oboch metód, nie je dosiahnutá zhoda názvov. Konkrétnym príkladom je napríklad aplikácia Zabbix (systém pre sieťový manažment), ktorá tvorí prevažnú časť zachytenej komunikácie z akademického prostredia. Tá nemá svoju alternatívu v l7-filtri a ten ju v drvivej väčšine prípadov identifikuje ako finger (regulárny výraz popisujúci tento protokol je totiž ohodnotený ako nespoľahlivý).
- Niektoré protokoly sú identifikované zhodne, avšak majú pre rôzne metódy, rôzne názvy alebo jedna z metód určí protokol pričom druhá aplikáciu, šifrovanie a pod.. Preto ich nie je možné zaradiť medzi rozpoznané dátové toky. Medzi tieto podobné, ale nie zhodné protokoly (aplikácie) patria napr. : netbios-dgm ~ smb, https ~ ssl, aol ~ aim, a iné.
- Ďalším problémom sú už spomínané chybné regulárne výrazy. Jedným z rozpoznaných a opravených je DNS. Pri DHCP bolo potvrdené rozpoznanie nielen samotného tohto protokolu, ale aj BOOTPS. Nepodarilo sa však upraviť výraz popisujúci protokol FTP a ten ostáva naďalej neidentifikovaný.
- Svojou troškou prispievajú k nesprávnemu určeniu aplikácie, pri HTTP [7], aj nepotvrdené požiadavky klienta, pretože jedinečná signatúra tohto protokolu je obsiahnutá práve v odpovedi zo servera.
- Pri násilnom ukončení spojenia pomocou časovača, je možné že v skutočnosti takéto spojenie ešte stále prebieha. Tým pádom sa akoby duplikuje, pričom jeho payload neobsahuje dáta zo začiatku komunikácie. Systém nie je schopný určiť aplikáciu takéhoto spojenia.

4.1.2 Percento úspešne rozpoznaných paketov

Percentuálna úspešnosť rozpoznávania paketov sa určuje podobne ako pri dátových tokoch s malými rozdielmi. Počet prenesených paketov je pri komunikácii v domácich sieťach menší ako pri dátach z akademickej siete. To sa prejavuje aj na priemernej úspešnosti, ktorá môže byť trochu skreslená a neodzrkadľuje úplne presne priemer pre všetky dáta. Dá sa však predpokladať, že ak by všetky dátové sady obsahovali rovnaký počet paketov, došlo by k posunu len o rádovo pár percent. Avšak stále je to len domnienka. Tabuľka 4.2 obsahuje výstupné údaje testov.

	11-1-09_1	**11-1-09_2**	**12-12-08_2**	**14-5-09_0**	**14-5-09_3**	**17-5-09_0**	**Priemer**
Celkový počet paketov	**923565**	**960787**	**941977**	**612453**	**103344**	**214021**	**626025**
Celkovo podľa portu	919555	954224	940962	434896	100109	213370	593853
	99,57%	99,32%	99,89%	71,01%	96,87%	99,70%	94,86%
Celkovo podľa l7-filtra	113481	155564	115142	221741	14968	105068	120994
	12,29%	16,19%	12,22%	36,21%	14,48%	49,10%	19,33%
Zhoda protokolov	69999	78541	38538	10965	7797	41655	41249
	7,58%	8,17%	4,09%	1,79%	7,54%	19,46%	6,59%
Neidentifikované pakety	825482	835333	838155	388420	86975	1613	495996
	89,38%	86,94%	88,98%	63,42%	84,16%	0,75%	79,23%
Dôvera podľa portu	27759	46516	57413	7243	8548	170654	53022
	3,01%	4,84%	6,09%	1,18%	8,27%	79,74%	8,47%
Dôvera podľa l7-filtra	325	397	7871	205825	24	99	35757
	0,04%	0,04%	0,84%	33,61%	0,02%	0,05%	5,71%
Počet TCP paketov	123055	171146	122488	183505	10368	210013	136763
	13,32%	17,81%	13,00%	29,96%	10,03%	98,13%	21,85%
Počet UDP paketov	800510	789641	819489	428948	92976	4008	489262
	86,68%	82,19%	87,00%	70,04%	89,97%	1,87%	78,15%

Tabuľka 4.2 : Percentuálna úspešnosť klasifikácie paketov

Už na prvý pohľad je zrejmé že dáta označené ako 14-5-09_0 a 17-5-09_0 sú odlišné od ostatných. Percentuálny podiel TCP paketov je v nich výrazne vyšší, čo má za následok ľahšie určenie prvého Kbytu spojenia. Na základe toho je systém schopný rozpoznať viacero regulárnych výrazov. Tento vysoký počet TCP paketov má odlišné príčiny v oboch prípadoch. Dňa 14.5.2009 bola pri odchytávaní komunikácie pravdepodobne spustená P2P aplikácia, ktorá pre prenos dát využíva TCP spojenie. V prípade dátovej sady zo dňa 17.5.2009 sa jedná o časté využívanie protokolov https, ssh, či http-video. Rovnako ako pri klasifikácii dátových tokov, je nízke percento zhody,

ovplyvnené vyššie vymenovanými faktormi. K nim sa však pridáva ďalší veľmi významný prvok a tým je využitie portu 1234. Ten používa mnoho aplikácií (hry Command and Conquer), trojanov a čo je najdôležitejšie program VLC pre UDP/RTP stream. Tento program je veľmi rozšírený a je viac ako pravdepodobné, že práve on využíval protokol identifikovaný na základe čísla portu 1234 ako search-agent. Opäť, ako v prípade LLMNR, systém nedisponuje regulárnym výrazom schopným určiť tento druh komunikácie. Samozrejme najväčší problém spôsobujú dátové toky, ktoré začali ešte pred samotným odchytením daného súboru. Pomerne často sa stáva, že neskončia až do konca súboru a to má za následok neschopnosť ich identifikácie. Aj v testoch uvedených v tejto kapitole sa takéto dátové toky vyskytujú a predstavujú podstatné časti jednotlivých dátových sád. Preto ostáva veľké množstvo paketov ako aj bytov (tabuľka 4.3) nerozpoznaných.

4.1.3 Percento úspešne rozpoznaných dát

Testy určené na percentuálne vyjadrenie úspešnosti systému z pohľadu veľkosti v bytoch sa nijak výrazne nelíšia od tých predchádzajúcich, rozpoznávajúcich pakety. Jedinou odlišnosťou je ešte výraznejšie prehĺbenie rozdielov medzi jednotlivými dátovými sadami a technikami určovania aplikácií. Výsledky testov sú zobrazené v tabuľke 4.3.

	11-1-09_1	**11-1-09_2**	**12-12-08_2**	**14-5-09_0**	**14-5-09_3**	**17-5-09_0**	**Priemer**
Celková veľkosť dát	**770336648**	**766182973**	**769906714**	**775269254**	**114290300**	**149376697**	**557560431**
Celkovo podľa portu	769449162	764830772	769259027	561340578	113984434	149297078	521360175
	99,88%	99,82%	99,92%	72,41%	99,73%	99,95%	93,51%
Celkovo podľa l7-filtra	53751688	99768561	67932663	259089755	5191662	82694437	94738128
	6,97%	13,02%	8,82%	33,42%	4,54%	55,36%	16,99%
Zhoda protokolov	36404623	69604232	31548686	4445433	2563983	28000378	29211223
	4,73%	9,08%	4,10%	0,57%	2,24%	18,74%	5,24%
Neidentifikované toky	716835800	668738074	697252593	516025774	106295746	237985	450897662
	93,05%	87,28%	90,56%	66,56%	93,01%	0,16%	80,87%
Dôvera podľa portu	17038853	27776700	39500162	1737316	2728402	121127256	34984782
	2,21%	3,63%	5,13%	0,22%	2,38%	81,09%	6,28%
Dôvera podľa l7-filtra	57372	63967	1605273	253060731	2169	11078	42466765
	0,01%	0,01%	0,21%	32,64%	0,00%	0,01%	7,62%
Veľkosť dát TCP	57744620	105239131	75827156	217856837	3850922	148592211	101518480
	7,50%	13,74%	9,85%	28,10%	3,37%	99,47%	18,21%
Veľkosť dát UDP	712592028	660943842	694079558	557412417	110439378	784486	456041951
	92,50%	86,26%	90,15%	71,90%	96,63%	0,53%	81,79%

Tabuľka 4.3 : Percentuálna úspešnosť vyjadrená vo veľkosti dát v bytoch

Je zaujímavé si všimnúť, ako postupne klesajú percentá úspechu s každým jeho vyjadrením. Nezaujatý pozorovateľ by mohol dôjsť k záveru že sa ľahšie identifikujú dátové toky s menším obsahom paketov a veľkosťou prenášaných dát. Avšak to nie je pravda. Najlepším príkladom je úspešnosť klasifikácie dátových tokov pri poslednom odchytenom súbore z dňa 17.5.2009. Ten obsahoval komunikáciu bežného užívateľa internetu a sietí celkovo. Tá je veľmi dobre známa a preto sú aj regulárne výrazy spoľahlivejšie ako v prípade iných typov komunikácie. Za neúspechom našich testov, najmä z pohľadu paketov a dát, stojí neschopnosť identifikácie tokov, ktoré začali ešte pred samotným zachytením sieťovej prevádzky.

4.2 Časová náročnosť

Časová náročnosť systému je závislá od viacerých faktorov. Najdôležitejším z nich je konfigurácia serveru, na ktorom portál beží. Pre úspešnú analýzu musí byť nastavený čas vykonávania skriptov jazyka PHP na neurčito príkazom set_time_limit(0). Samotné určovanie časov pri testovaní bolo zabezpečené, ďalšími funkciami time a mikrotime (pre presnejšie určenie). Odchytávali sa časy začiatku behu programu, samotnej analýzy a ich konce. Týmto spôsobom bolo možné určiť čas potrebný pre vykonanie samotnej analýzy načítaného súboru ako aj celkový čas od kliknutia na odkaz, až po vypísanie tabuľky, zobrazujúcej práve klasifikované dátové toky. V tabuľke 4.4 sa nachádzajú tieto hodnoty, spolu s časmi potrebnými pre identifikáciu prvých 10,000 prípadne 100,000 paketov uložených v danom súbore.

V sekundách a mikrosekundách	**11-1-09_1**	**11-1-09_2**	**12-12-08_2**	**14-5-09_0**	**14-5-09_3**	**17-5-09_0**	**Priemer**
Čas samotnej analýzy	259,114583	352,844533	275,788050	1011,367479	289,885423	334,299952	nedostupné
Čas celého procesu analýzy	278,371965	367,465556	282,570183	1016,946307	293,160933	336,241203	nedostupné
Čas identifikácie prvých 10.000 paketov	1,311216	1,703544	2,649728	10,517426	13,831976	16,786540	7,800072
Čas identifikácie prvých 100.000 paketov	27,756764	13,189761	20,534724	186,998748	133,639409	150,090958	88,701727

Tabuľka 4.4 : Časová náročnosť jednotlivých testov

Porovnaním počtu dátových tokov s dĺžkou trvania jednotlivých analýz je možné dôjsť k záveru, že analýza dát, odchytených na internátnej a domácej sieti trvá niekoľkonásobne dlhšie ako tých zachytených v akademickej sieti. Príčinou je väčší počet dátových tokov využívajúcich protokol UDP v pomere k celkovému počtu. UDP spojenie sa ukončuje len na základe časovača a preto je v pamäti uložený väčší počet týchto tokov.

Tak isto je pri každom novom pakete potrebné kontrolovať, či nepatrí do už existujúceho toku. To je výpočtovo najnáročnejšia časť a preto vyšší počet dátových tokov uložených v pamäti spôsobí enormný nárast časovej náročnosti danej analýzy. TCP spojenie sa ukončuje pomocou stavov a je možné určiť, kedy je potrebné uvoľniť pamäťové prostriedky.
Z tabuľky 4.4 sa dá vyčítať, že čas potrebný na anotáciu určitého počtu paketov rastie približne priamo úmerne s počtom týchto paketov. Je však badať nepatrné zvyšovanie pri väčšom množstve paketov. Jediné výrazne odchýlky od priamej úmery sú pri prvej a štvrtej sade dát. Opäť sa to dá vysvetliť nárastom počtu dátových tokov uložených v dočasnej pamäti.
Celkový čas potrebný pre identifikáciu aplikácií by sa dal obmedziť skrátením doby, po ktorej sa spojenia stávajú neaktívnymi. V testoch boli využívané hodnoty 5 minút pre TCP aj UDP.

4.3 Protokoly sieťovej vrstvy

Okrem klasifikácie aplikácií a transportných protokolov TCP a UDP je systém schopný určiť aj protokoly sieťovej. Konkrétne ide o ARP [11], RARP [12] a ICMP [13]. Tieto protokoly sú bližšie popísané v teoretickom rozbore. Ako je možno si všimnúť, v tabuľke 4.5 chýba protokol RARP. Dôvodom je jeho zastaranosť a absencia vo všetkých testovaných vzorkách dát.
V školskej, teda akademickej, sieťovej komunikácii je pomerne rozšírený protokol ICMP. Možno len predpokladať, že sa jedná o jeho využitie programom ping.
Na druhej strane v internátnej sieti, s veľkým množstvom ťažko kontrolovateľných staníc a pripojení, bol podľa očakávania zaznamenaný zvýšený výskyt použitia protokolu ARP.

	11-1-09_1	**11-1-09_2**	**12-12-08_2**	**14-5-09_0**	**14-5-09_3**	**17-5-09_0**	**Priemer**
Počet ARP	306	385	249	2135	2812	41	988
Počet ICMP	60	69	34	6	3	1	29

Tabuľka 4.5 : Počet využití protokolov sieťovej vrstvy

5 Záver

V tejto práci bol vytvorený systém pre analýzu sieťovej komunikácie. Analyzátor je schopný rozpoznávať a detekovať hlavičky odchytených dát na jednotlivých vrstvách sieťového modelu TCP/IP. Jedná sa o Ethernet na vrstve sieťového rozhrania, protokoly IPv4, ARP, RARP a ICMP na sieťovej vrstve a protokoly TCP, UDP na transportnej vrstve. Okrem toho bol implementovaný aj analyzátor, ktorý podľa čísiel portov a podľa signatúr, nachádzajúcich sa v payloade paketov, určuje aplikácie.. Systém bol implementovaný v jazyku PHP a je určený pre webové servery bežiace na unixových systémoch.
Pre identifikáciu aplikačného protokolu, prípadne aplikácie, sa používajú čísla portov definované organizáciou IANA [15], prípadne regulárne výrazy čerpané z L7-filtra [2].
Navrhnutý systém má za úlohu pomôcť užívateľovi s anotáciou sieťovej komunikácie. A to tým, že ponúkne užívateľovi výsledky analýzy pomocou portu a pomocou signatúr. Napokon je ponúknutá aj dôveryhodne určená aplikácia, ktorá uľahčuje užívateľom správnu klasifikáciu dátového toku. Súbory s výsledkami sú voľne prístupné užívateľovi a môžu byť použité inými programami.
V rozšírení phpcap použitom v tejto práci museli byť spravené určité úpravy umožňujúce jeho kompiláciu do jadra PHP.
Výsledky testov ukázali, že doba potrebná pre identifikáciu dátových tokov je neprimerane dlhá. Pre jej urýchlenie bol do systému pridaný tzv. časovač, ktorý označí spojenie ako uzavreté po stanovenej dobe nečinnosti. Ďalším elementom spomaľujúcim beh programu je nutnosť pamätať si aj dátové toky, ktoré začali ešte pred zachytením analyzovaného súboru. Tieto toky nie je možné spoľahlivo určiť pomocou analýzy payloadu a preto je zbytočné si pamätať ich dĺžku trvania, prípadne ich veľkosť. Z pohľadu ďalšieho vývoja je teda potrebné skrátiť dobu behu analýzy napr. odstránením neidentifikovateľných dátových tokov z jej priebehu.
Pri testovaní bola zistená istá nepresnosť v regulárnych výrazoch popísaných L7-filtrom. Niektoré z nich nedokázali správne určiť aplikáciu a museli byť opravené. Nepodarilo sa však upraviť všetky a preto je potrebné ďalej tieto výrazy skúmať a vylepšovať ich. Rovnako je potrebné rozšírením modulu pkttool pridať podporu pre ďalšie protokoly (Wi-Fi, IPv6, PPP, SCTP, atď.).
Ako hlavné rozšírenie by som však navrhoval pridanie ďalších techník určených pre identifikáciu aplikácie. Jedná sa hlavne o IDS systémy Snort, Bro, Hippie a o rôzne iné heuristiky založené na analýze payloadu.

6 Literatúra

[1] WWW stránky projektu Wireshark.. http://www.wireshark.org, (máj 2009)

[2] WWW stránky projektu L7-filter. http://l7-filter.sourceforge.net, (máj 2009)

[3] WWW stránky projektu Snort. http://www.snort.org, (máj 2009)

[4] WWW stránky projektu Bro. http://www.bro-ids.org, (máj 2009)

[5] Postel, J.: Transmission control protocol, RFC 793, 1981. Dokument dostupný na http://tools.ietf.org/html/rfc793, (máj 2009)

[6] Postel, J.: User datagram protocol, RFC 768, 1980. Dokument dostupný na http://tools.ietf.org/html/rfc768, (máj 2009)

[7] Yan, K.: Network Protocols Handbook. 2nd Edition. In: Javvin Technologies Inc.. ISBN 0-9740945-2-8, 2005

[8] Postel, J.: Internet protocol, RFC 791, 1981. Dokument dostupný na http://tools.ietf.org/html/rfc791, (máj 2009)

[9] Deering, S., Hinden, R.: Internet protocol version 6 (IPv6) specifikation, RFC 2460, 1998. Dokument dostupný na http://tools.ietf.org/html/rfc2460, (máj 2009)

[10] Plummer, D.C.: An ethernet address resolution protocol, RFC 826, 1982. Dokument dostupný na http://tools.ietf.org/html/rfc826, (máj 2009)

[11] Finlayson, R., Mann, T., Mogul, J., Theimer, M.: A reverse address resolution protocol, RFC 903, 1984. Dokument dostupný na http://tools.ietf.org/html/rfc903, (máj 2009)

[12] Postel, J.: Internet control message protocol, RFC 792, 1981. Dokument dostupný na http://tools.ietf.org/html/rfc792, (máj 2009)

[13] WWW stránky projektu tcpdump. http://www.tcpdump.org, (máj 2009)

[14] Kim, H., Claffy, K., Fomenkov, M., Barman, D., Faloutsos, M., Lee, K.: Internet traffic classification demystified: Myths, caveats, and the best practices. In: ACM CoNEXT 2008. Madrid, Spain, 12 November 2008. Dokument dostupný na http://www.caida.org/publications/papers/2008/classification_demystified/classification_demystified.pdf, (máj 2009)

[15] IANA. Port numbers. Dokument dostupný na http://www.iana.org/assignments/port-numbers, (máj 2009)

[16] Moore, A.W., Papagiannaki, K.: Toward the Accurate Identification of Network Applications. In: PAM 2005: Passive and Active Network Measurement, 6th International Workshop. Boston, MA, USA, 31 March – 1 April 2005. Dokument dostupný na http://128232.0.20/research/srg/netos/papers/2005-moore2005toward.pdf, (máj 2009)

[17] Karagiannis, T., Broido, A., Faloutsos, M., Claffy, K.: Transport layer identification of p2p traffic. In: IMC 2004: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement. New York, NY, USA, October 2004. Dokument dostupný na http://www.caida.org/publications/papers/2004/p2p-layerid/p2p-layerid.pdf, (máj 2009)

[18] Idrees, F., Khan, U. A.: A generic technique for Voice over Internet Protocol (VoIP) traffic detection. In: IJCSNS International Journal of Computer Science and Network Security, Vol. 8, No. 2, pp. 52-59, February 2008. Dokument dostupný na http://paper.ijcsns.org/07_book/200802/20080207.pdf, (máj 2009)

[19] WWW stránky. Stránka pre stiahnutie rozšírenia phpcap. http://sourceforge.net/projects/phpcap, (máj 2009)

[20] WWW stránky. Manuálová stránka pre php funkcie. http://php.net/manual/en, (máj 2009)

[21] WWW stránky. Perl regular expresions. http://perldoc.perl.org/perlre.html, (máj 2009)

7 Zoznam príloh

Príloha 1. CD – obsahuje adresár *www*, v ktorom sú umiestnené všetky zdrojové kódy. Priložená je aj táto technická správa vo formáte pdf a v zdrojovom tvare. V adresári *install* sú umiestnené inštalačné súbory webového serveru Apache (2.2.11), jazyka php (verzia 4.4.9), rozšírenia phpcap (0.2e) a mnohých ďalších. Tieto programy sú dostupné zdarma a poskytujú prostredie pre beh tohto webového portálu. V koreňovom adresári je umiestnený súbor *INSTALL*, ktorý obsahuje návod na inštaláciu a konfiguráciu tohto portálu. Ďalej je priložený *doc.html* popisujúci jednotlivé súbory uložené v adresári www. CD tiež obsahuje všetky položky potrebné pre úspešné rozbehnutie webového portálu.