VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

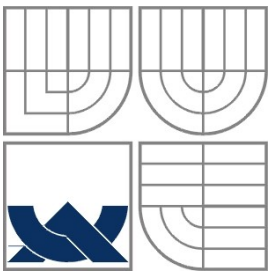
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

HTTP ANALYZÁTOR S WEBOVÝM ROZHRANÍM

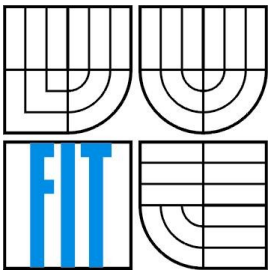
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE TOMÁŠ ROZSNYÓ
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

ANALYZÁTOR HTTP PROVOZU S WEBOVÝM ROZHRANÍM

HTTP ANALYZER WITH WEB USER INTERFACE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ ROZSNYÓ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. JIŘÍ TOBOLA

BRNO 2008

Zadání bakalářské práce/6952/2008/xrozsn01

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok

Zadání bakalářské práce

Řešitel: **Rozsnyó Tomáš**
Obor: Informační technologie
Téma: **Analyzátor HTTP provozu s webovým rozhraním**
Kategorie: Web

Pokyny:

1. Seznamte se s technologiemi pro tvorbu webových informačních systémů (HTML, CSS, PHP, Javascript, MySQL apod.).
2. Seznamte se s dostupnými nástroji pro logování navštívených URL na základě analýzy paketů.
3. Navrhněte aplikaci s webovým rozhraním, která v přehledné formě zobrazí výsledky logů Vámi vybraného nástroje. Navrženou aplikaci vhodně modelujte.
4. Navrženou aplikaci realizujte a otestujte, funkčnost aplikace demonstrujte na vhodně zvoleném vzorku dat.
5. Zhodnoťte dosažené výsledky a diskutujte možnosti dalšího rozšíření aplikace.

Literatura:

- Dle pokynů vedoucího.

Při obhajobě semestrální části projektu je požadováno:

- Splnění prvních tří bodů zadání.

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 2008
Datum odevzdání: 20. května 2009

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

Tato práce přináší popis návrhu teoretické a implementační části HTTP analyzátoru s webovým rozhraním. HTTP analyzátor zpracovává a ukládá odchycené HTTP pakety. HTTPry obsahuje klienta, jehož úlohou je ukládat analyzované logy do PostgreSQL databáze. Webové rozhraní poskytuje funkci správy PostgreSQL databáze a prostředek k přehlednému zobrazení statistik vytvořených z uložených logů. Správa databáze zahrnuje manipulaci s uloženými logy a administraci uživatelů přistupujících prostřednictvím webového rozhraní.

Abstract

This work provides a description of the theoretical and implementation part of HTTP analyzer with a web interface. HTTP Analyzer processes and saves captured HTTP packets. HTTP Analyzer includes a client, whose task is to save analyzed logs to PostgreSQL database. Web interface provides the management of the PostgreSQL database and offers capability to readily and clearly see the statistics generated from the stored logs. Database management involves the manipulation of stored logs and administration of users accessing the web interface.

Klíčová slova

analýza HTTP hlaviček, HTTP analyzátor, logovaní, odchycování HTTP toku

Keywords

analyzing HTTP headers, HTTP logger, HTTP analyzer, capture HTTP flow

Citace

Tomáš Rozsnyó: Analyzátor HTTP provozu s webovým rozhraním , bakalárska práca, Brno, FIT VUT v Brne, 2009

Analyzátor HTTP provozu s webovým rozhraním

Prohlášení

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

.....
Tomáš Rozsnyó
27. mája 2009

Poděkování

Rád by som paďakoval svojmu vedúcemu bakalárskej práce Ing. Jiřímu Tobolovi za usmernenie, akým sa má práca uberať. Rovnako za silnú motiváciu akú mi do práce dodal.

Obsah

1 Úvod

Rapídne rozširovanie internetu prináša so sebou aj stále narastajúce nebezpečie hroziace z neho. S narastajúcim nebezpečím narastájú nároky na zabezpečenie bezpečnosti sietí. Pre maximalizáciu bezpečnosti je nevyhnutné mať naištalovaný kvalitný a spoľahlivý harware na zabezpečenie ako napríklad firewall, UTM (Unified Threat Management) a iné. Ďalším krokom je ich správna konfigurácia. Tieto opatrenia, ale nepostačujú. Rovnako dôležitá je pokročilá správa siete. Tá v dnešnej dobe vyžaduje nielen kontrolu samotnej infraštruktúry siete ale aj jej užívateľov. Jedným z možností ako nadobudnúť vačší prehľad, väčšiu kontrolu nad sieťou je analýza HTTP toku v sieti. Potreba väčšej kontroly pramení z čoraz vačšieho ohrozenia prichádzajúceho z vnútra siete. Či už umyselného alebo neúmyselného. Z toho pramení požiadavka kotroly klientov siete. Znižovanie rizika by mohlo byť dosiahnuté preškolovaním zamestnancov. Je to vhodné riešenie ale nepostačujúce.

Alternatívnym alebo lepšie doplnkovým riešením je kontrola navštívených webových stránok, stiahnutých súborov a veľkosti stiahnutých súborov prenášaných pomocou protokolu HTTP. Splnenie tejto požiadavky je možné docieliť HTTP analyzátorom. HTTP analyzátor by zachytával, spracovával HTTP správy. Spracované HTTP správy by ukladal v štandardnej forme, čo by umožnilo automatizované spracúvanie.

Analýzu HTTP toku pod linuxom nám umožňujú programy tcpdump a Wireshark. Tie sú však zamerané všeobecne na analýzu širokej škály protokolov. Pre účely logovania HTTP toku je lepšie pouziť nástroj špeciálne na to určený. Tým nástrojom je sofistikovaný HTTP analyzátor , ktorý by pomocou prehľadného webového rozhrania umožnil rýchly a jednoduchý prehľad štatistík odchytených logov.

Cieľom bakalárskej práce je odchytávať informácie z HTTP hlavičky pomocou HTTP analyzátora a ukladať ich do databázy PostgreSQL. Dáta z databázy triediť a prehľadne zobrazovať pomocou webového rohzrania. Požiadavka bola aby HTTP analyzátor pracoval pod systémom linux(unix).

Druhá kapitola Teoretický rozbor je zameraný na analýzu jednotlivých častí systému. Analýza je bližšie zameraná na popis HTTP protokolu, libpcap knižnice, porovnanie HTTP analyzátorov, t.j. škály použitia a rýchlosti. V ďalšej časti sa nachádza popis PostgreSQL databázy a jeho knižnice libpq. Tretia kapitola Návrh systému špecifikuje návrh na vytvorenie celkovej štruktúry t.j. jednotlivých častí spolu fungujúcich ako celok. Štvrtá kapitola Implementácia obsahuje popisy techník riešenia jednotlivých častí a poradie v akom sa jednotlivé časti riešili. Piata kapitola Testovanie obsajuje výsledky testovania rýchlosti a funkčnosti HTTP analyzátora a PostgreSQL databázy. A výsledné testovanie práce ako celku. V závere by som chcel zhodnotiť dosiahnuté výsledky. Popísať príklady nasadenia v reálnej sieti. Priniesť návrhy na zlepšenie a definovať ďalšie budúce smerovanie vývoja práce.

2 Teoretický rozbor

2.1 HTTP protokol

2.1.1 Popis

HTTP alebo (Hypertext Transfer Protocol) je jedným z najčastejšie využívaných protokolov súčasnosti. Pôvodné bol navrhnutý pre výmenu hypertextových dokumentov HTML(Hypertext Markup Languege). Do dnešného dňa boli uverejnené dve verzie: HTTP/1.0 a najnovšia HTTP/1.1. Je implementovaný klient/ server architektúrou. Klientom je prehliadač, ktorý zasiela požiadavky na server. Následne čaká na odpoveď servera. Medzi prehliadače patrí Mozilla Firefox, Opera, Internet Explorer, Google Chrome. Tie patria do kategórie grafických prehliadačov. Menej známe sú textové prehliadače napríklad Lynx a w3m.

HTTP protokol pracuje na Aplikačnej vrstve, čo je siedma vrstva ISO/OSI modelu. Účelom vrstvy je poskytnúť aplikáciam prístup ku komunikačnému systému a umožniť tak ich spoluprácu.

Obrázok 2.1: Referenčný model ISO/OSI

HTTP sa na Transpornej vrstve prenáša pomocou TCP protokolu. Spolu s UDP protokolom tvoria transportnú vrstvu. TCP protokol využíva spojovanú službu. Nevýhodou TCP je skutočnosť, že má vyššie réžie prenosu ako aj spracovania v porovnaní s UDP protokolom. Výhody TCP protokolu sú spoľahlivý prenos, riadenie toku a riadenie zahltenia. Vlastnosti TCP sú nevyhnutné pre spoľahlivý prenos HTTP protokolu.

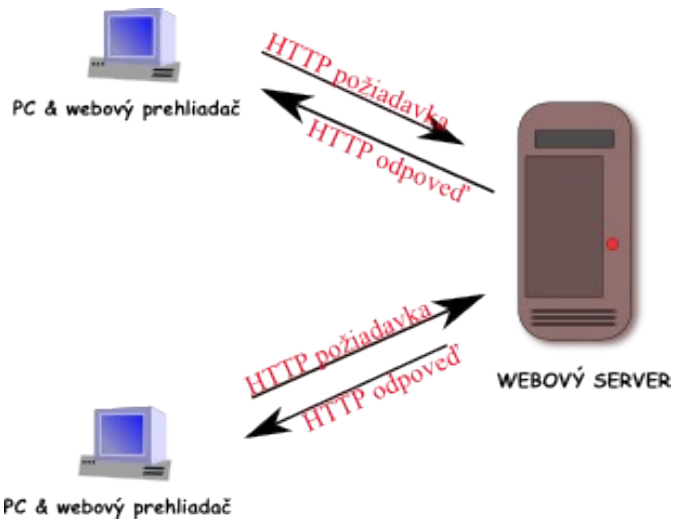
2.1.2 HTTP spojenie

a) Bez stáleho spojenia (nonpersistent) - maximálne jeden objekt je poslaný TCP spojením
- HTTP/1.0

b) Stále spojenie (persistent) - viac objektov môže byť zaslaných TCP spojením medzi klientov a serverom
- HTTP/1.1

2.1.3 Priebeh spojenia

1. Klient inicializuje TCP spojenie (vytvára socket) so serverom na cieľový port 80.
2. Server akceptuje spojenie.
3. Nastáva výmena HTTP správ.
4. Uzatvorí sa TCP spojenie.



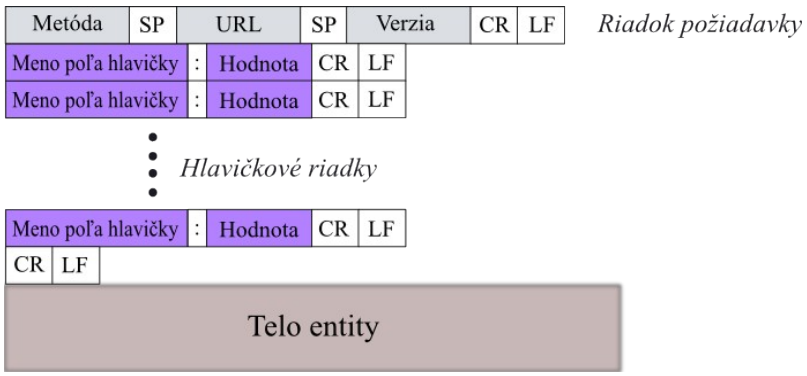
Obrázok 2.2: Ukážka spojenia klient/server

2.1.4 HTTP správy

HTTP správa môže byť typu požiadavka alebo odpoveď. Je to ASCII správa, ktorá je čitateľná človekom. Každá správa má svoj presne definovaný formát. To nám umožňuje jeho automatizované spracovanie.

HTTP definuje osem metód identifikujúce požadovanú akciu, ktorá má byť vykonaná na identfikovanom zdroji. Čo predstavuje zdroj, či pre-existujúce údaje alebo údaje, ktoré sú generované dynamicky, záleží na implementácii servera. Často sa stáva, že zdroj zodpovedá súboru alebo výstup spustitelnému programu sídliaceho na serveri.

2.1.5 Obecný formát HTTP požiadavky



Obrázok 2.3: Obecný formát HTTP požiadavky

Vysvetlivky:

Metóda: - udáva použitú metódu HEAD, GET, POST, PUT, DELETE, TRACE, OPTIONS, CONNECT

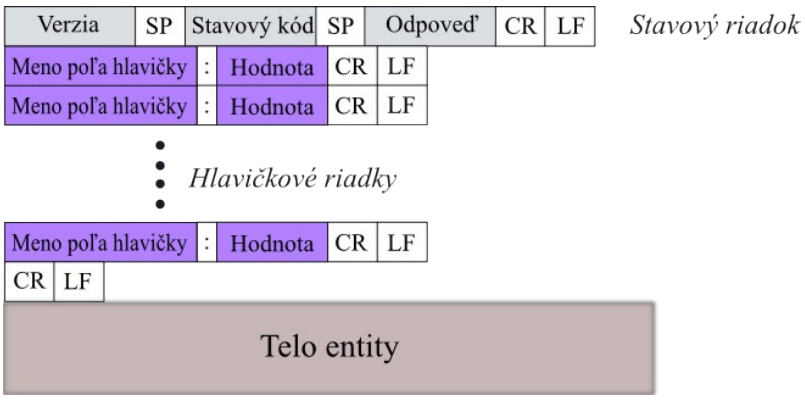
URL: - udáva cestu k zdroju

Verzia: - určuje akej verzie je HTTP protokol

Ukážka HTTP požiadavky:

```
GET /wiki/Wikipedie HTTP/1.1
Host: cs.wikipedia.org
User-Agent: Mozilla/5.0 Gecko/20040803 Firefox/0.9.3
Accept-Charset: UTF-8,*
```

2.1.6 Obecný formát HTTP odpovede



Obrázok 2.4: Obecný formát HTTP odpovede

Vysvetlivky:

Verzia: - určuje akej verzie je HTTP protokol(napríklad HTTP/1.0, HTTP/1.1)
Stavový kód: - trojmiestne číslo, udáva odpoveď na HTTP požiadavku (napríklad 200, 404)
Odpoveď: - špecifikuje význam Stavového kódu

Ukážka HTTP odpovede:

```
HTTP/1.0 200 OK
Date: Fri, 15 Oct 2004 08:20:25 GMT
Server: Apache/1.3.29 (Unix) PHP/4.3.8
X-Powered-By: PHP/4.3.8
Vary: Accept-Encoding,Cookie
Cache-Control: private, s-maxage=0, max-age=0, must-revalidate
Content-Language: cs
Content-Type: text/html; charset=utf-8
```

Základný oboznámenie sa s protokolom HTTP je nevyhnutné pre riešenie problematiky logovania HTTP správ. Pre vyčerpávanejšie vysvetlenie je možné nájsť v RFC. Každý protokol a verzia je popísaná zvlášť. Popis pre všeobecné použitie HTTP/1.1 sa nachádza v RFC 2616. Štandardný popis je dostupný na v RFC 2068. Podrobný popis HTTP/1.0 sa nachádza v RFC 1945.

2.2 Libpcap

Je knižnica, ktorá slúži na odchytávanie paketov. Názov knižnice je '*pcap.h*'. Knižnica je napísaná v programovacom jazyku C a je šírená pod BSD licenciou. Poskytuje široké možnosti rozhrania pre program na odchytávanie paketov systémov. Všetky pakety v sieti, a to aj tie, ktoré sú určené pre ďalšieho hostiteľa, sú prístupné prostredníctvom tohto mechanizmu.

Pakety odchytáva v momente ako opustia sieťovú kartu. Spraví si kópiu paketu, predtým než pakety spracuje operačný systém. Základné funkcie knižnice pcap.h sú:

pcap_lookupdev()
Vráti ukazovateľ na sieťové zariadenie vhodné pre použitie s pcap_open_live () a pcap_lookupnet ().

pcap_lookupnet()
Sa používa na určenie IP adresy a masky siete spojenú so sieťovým rozhraním zariadenia.

pcap_open_live()
Sa používa na získanie deskriptora paketu, ktorý umožňuje sledovanie paketov v sieti.

pcap_compile()
Sa používa na preklad reťazeca string, ktorý je parametrom pcap_compile(). Hodnoty preloženého raťazca vytvárajú filter programu.

pcap_setfilter()
Slúži na aplikovanie filtra vytvoreného funkciou pcap_compile().

pcap_freecode()
Slúži na odalokovanie pamäte spotrebovaných funkciou pcap_compile().

pcap_loop()
Číta predefinovaný počet paketov alebo kým nenastane chyba.

pcap_breakloop()
Nastaví príznak, ktorý spôsobí ukončenie pcap_loop(). Vráti počet doteraz spracovaných paketov.

2.3 Logovanie sieťového toku

Dôležité je na aký aspekt analýzy sa zameriame, aké informácie chceme z HTTP správ získať aby nám poslúžili k čo najlepším štatistikám, najlepšej správe siete. Je preto veľmi dôležité, podrobne sa zoznámiť s jednotlivými analyzátormi, dôkladne prehodnotiť aké možnosti ponúkajú. Či už sú to

možnosti logovania, rýchlosť akým pracujú alebo iné vlastnosti. Ako výsledok mojej analýzy uvádzam skrátený popis jednotlivých HTTP analyzátorov:

Dsniff - Urlsnarf

- Licencia : GNU GENERAL PUBLIC LICENSE
- Programovací jazyk: C
- Možnosti logovania: zdrojová IP adresa, dátum, čas, HTTP metóda a verzia, host, user-agent
- Výhody: balíček v gentoo repozitároch, používa knižnicu libpcap
- Nevýhody: minimálne možnosti odchytávania, nefunguje ako démon, neodchytáva HTTP odpovede
- Web adresa: http://monkey.org/~dugsong/dsniff/

HTTPry

- Licencia : GNU GENERAL PUBLIC LICENSE
- Programovací jazyk: C
- Možnosti logovania: každá položka HTTP hlavičky
- Výhody: používa knižnicu libpcap, vlastné požiadavky na odychytávanie, spracováva aj HTTP odpovede, rýchly, dokáže pracovať ako démon
- Nevýhody: HTTP požiadavka a k nemu prislúchajúca odpoveď nie sú logované spolu ako pár, ale zvlášť
- Web adresa: http://www.dumpsterventures.com/jason/httpry/

HTTPTea

- Licencia : GNU GENERAL PUBLIC LICENSE
- Programovaci jazyk: Java
- Možnosti logovania: HTTP hlavička
- Výhody: vyuziteľný na testovanie webovych aplikacií, možnost meniť obsah HTTP hlavičiek
- Nevýhody: nefunguje ako démon, nie ja vhodný čisto na logovanie HTTP paketov
- Web adresa: http://httptea.sourceforge.net/

tcpdump

- Licencia : BSD license
- Programovaci jazyk: C
- Možnosti logovania: veľmi rozsiahle
- Výhody: zachytáva rôzne pakety, rýchly, široké možnosti odchytávania
- Nevýhody: nie je určený špecialne na logovanie HTTP paketov
- Web adresa: http://www.tcpdump.org/

2.4 Databáza PostgreSQL

2.4.1 Podpora OS

Podporuje všetky rozšírené OS, napr: Linux, UNIX (AIX, BSD, HP-UX, SGI-IRIX, Mac OS X, Solaris, Tru64) a Windows. Je dostupný v repozitároch distribucii alebo na stiahnutie z FTP serverov.

2.4.2 Licencia

- Je distribuovaná pod BSD licenciou.
- Súhlas používať, kopírovať, upravovať a šíriť tento softvér a jeho dokumentáciu pre akykoľvek účel, a to bez poplatku a bez písomnej dohody udelenou za predpokladu, že bude obsahovať autorské práva vo všetkých kópiach.
- Pre komerčné aj nekomerčné použitie.

2.4.3 Vlastnosti a SQL štandard

- Umožňuje beh uložených procedúr napísaných v niekoľkých programovacích jazykoch: Perl, Python, C alebo v špeciálnom PL/pgSQL jazyku.

- Obsahuje väčšinu SQL92 a SQL99 dátových typov, napr. INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL a TIMESTAMP.

- Osmičková verzia priniesla: stabilný výkon, MVCC, TOAST, WAL, Save Points, Point In Time Recovery, dvojfázové potrvrdzovanie, Partitioning, podpora SMP, informačné schémy, čiastočne a funkcionálne indexy, podpora SQL/XML, integrovany fulltext, integrovaná autentifikácia v prostredí MS Windows.

- Stopercentne splňuje podmienky ACID, plne podporuje cudzie kľúče, operácie JOIN, pohľady, spúšte a uložené procedúry.

ACID(*Atomic, Consistent, Isolated, Durable*) je obecne uznávaný zoznam požiadaviek na bezpečný tranzakčný systém:

- Atomičnosť
- Konzistencia
- Izolácia
- Trvanlivosť

2.4.4 Porovnanie s MySQL databázou

Výkonnostne nezaostáva za porovnatelnými komerčnými systémami, dokonca ich mnohokrát predčí. Podáva výborne výkony na multiprocesorových staniciach. Svojho najväčšieho open source konkurenta MySQL predci občas aj o viac ako 100%. Tohoto výsledku je možné dosiahnuť vyladením výkonu PostgreSQL databázy.

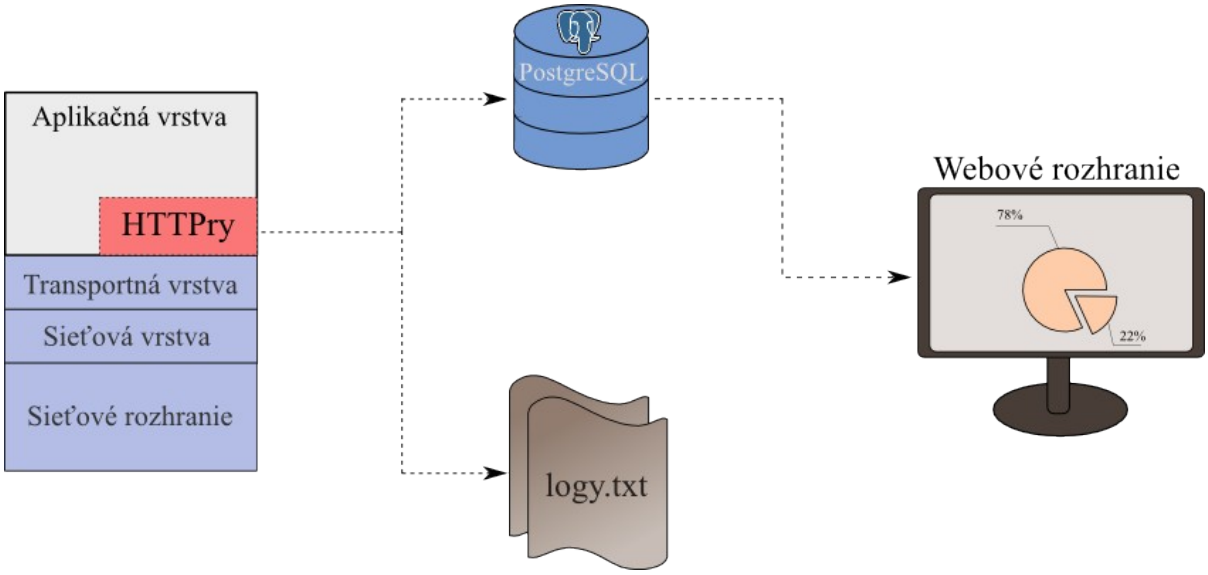
2.5 Libpq

Libpq je aplikačné rozhranie naprogramované v jazyku C pre PostgreSQL. Libpq je súbor knižníc funkcií, ktoré umožňujú klientskym programom vykonávať dotazy na PostgreSQL servery a prijímať výsledky týchto dotazov. Libpq je tiež základným motorom pre niekoľko ďalších PostgreSQL aplikačných rozhraní, vrátane tých, napísaných pre C + +, Perl, Python, Tcl a ECPG. Klientske programy, ktoré používajú libpq musia obsahovať hlavičku súboru '*libpq-fe.h*' a odkaz na libpq knižnice.

Prvý krok je pripojiť sa pomocou funkcie PQconnectdb(), kde je nutné udať minimálne jeden parameter a to názov databázy. Počet a hodnota parametrov závisí na nastavení konfiguračného súboru pg_hba.conf. Súbor pg_hba.conf obsahuje nastavenie pripojenia do databáze. Umožňuje nastaviť autentizáciu, autentizačnú metódu, užívateľov ktorí majú prístup do databázy a spôsob ako sa pripájajú do databáze, či už je to lokálne alebo vzdialene IP adresou. Následne je potrebné overiť spojenie s databázou, k tomu poslúži funkcia PQstatus(). Ak vráti hodnotu CONNECTION_OK, spojenie je pripravené. Teraz je možné vytvárať dotazy pomocou hore uvedených funkcií.
Po vykonaní všetkých spojení je nutné spojenie ukončiť funkciou PQfinish().

3 Návrh systému

Pred definovaním návrhu je potrebné si určiť výsledné ciele, ktoré by mala aplikácia spĺňať. Ako prvé by mala aplikácia odchytávavať HTTP správy. Odchytávané správy následne bezstratovo ukladať do databáze a pokiaľ je požiadavka aj do súboru alebo posieľať na štandardný výstup. Administrátor alebo užívateľ by mal mať prístup k logom uložených v databáze cez webové rozhranie. Kde by mal možnosť ich prehľadne zobrazovať.



Obrázok 3.1: Návrh systému

3.1 Odchýtávanie logov pomocou HTTPry

3.1.1 HTTPry-0.1.4

Po zhodnotení výhod a nevýhod každého analyzátora som ako najvhodnejšiu alternatívu vybral program HTTPry. HTTPry ponúka široké možnosti odchytávania a to vďaka možnosti si nadefinovať aké údaje má odchýtávať z hlavičky. Pre odchytávanie paketov používa knižnicu libpcap. Ako ďalšiu výhodu považujem, že je naimplementovaný v programovacom jazyku C. Čo by malo prispieť jeho rýchlosti, ktorú nakoniec potvrdil v zaťažových testoch. HTTPry podľa zadaných parametrov posiela logy na štandardný výstup stdout alebo ich ukladá do súboru.

Základné dôležité vlastnosti:

- Program dokáže sparsovať hocijakú hlavičku paketu, nemusí to byť HTTP hlavička.
- Pomocou parametra **'-s'** umožňuje meniť východzí(defaultný) formatovaný string, t.j. údaje ktoré sa majú odchytávať z HTTP hlavičky(viď RFC2616 hlavičky).
- Pomocou parametra **'-m'** umožňuje meniť metódy, ktoré sa budú odchytávať.
- Môže fungovať ako démon.

Základné parametre:

-n	Určuje po koľkých spracovaných hlavičkách skončí. Defaultne je 0 čo je nekonečno.
-i	Rozhranie, na ktorom bude prebiehať odchytávanie.
-o	Cesta a názov súboru do ktorého sa ukladaju logy. *#httpry -i eth0 -o /home/tomapirin/Desktop/httpry_skuska2.txt* *#httpry -i eth0 -o httpry_skuska2.txt* (uloží do aktuálneho adresára)
-u user	Umožnuje nadefinovať nového užívateľa, ktorý dostane práva na proces a výstupné súbory. Na nadefinovanie je potreba mať práva roota. Na nového užívateľa sa prepne po inicializácii.
'expression'	Je to filter, ktorý funguje podobne ako ACL(Access-List). *'tcp port 80 or 8080'* *'tcp dst port 80'* *'tcp dst port 80 and src host 192.168.1.1'*
-q	Potlačí nekritický výstup (štartovací výpis, štatistiky o počte spracových paketov, atď.).
-d	Pracuje v démon režime.
-m	Parameter umožňuje zadať, ktoré metody sa budú odytávať.
-s	Parameter umožňuje zadať formátovaný reťazec t.j. akú infromáciu chceme z HTTP hlavičky parsovať.

Východzí formátovaný výstup (bez parametra -s)**:**
Timestamp - Source-IP - Dest-IP - Direction - Method - Request-URI - HTTP-Version - Status-Code - Reason-Phrase

Nadefinovaný formátovaný výstup (s parametrom -s)**:**
Timestamp - Source-IP - Dest-IP - Direction - Method - Request-URI - HTTP- Version - Status-Code - Reason-Phrase - Content-Length

Východzie odchytávané metódy (bez parametra -m)**:**
options - get - head - post - put - delete - trace - connect

HTTP analyzátor musí podporavať parsovanie zdrojovej, cieľovej IP adresy, hostu t.j. doménového mena, cesty (URI) odkiaľ sú súbory získavané použitím metódy GET. HTTP správy dostatočne rýchlo parsovať, aby sa žiadne HTTP logy nestrácali, tým sa dosiahnu 100% spoľahlivé štatistiky. Všetky tieto požiadavky HTTPry-0.1.4 splňuje.
Parsovanie hore zmienených údajov som overil testami. Rýchlosť som overil zaťažovacími testami, bližšie informácie sa nachádzajú v kapitole 5 Testovanie.

3.2 Ukladanie logov

HTTPry vo verzii 0.1.4 umožňuje ukladať logy do súboru alebo posielať na štandardný výstup. Logy je potrebné ukladať do databázy pre jednoduchšiu správu a prácu nad získanými údajmi. Preto je potreba vytvoriť klienta, ktorý bude dáta ukladať do databázy.

3.2.1 Ukladanie logov do PostgreSQL databázy

PostgreSQL klient musí umožňovať ukladanie do PostgreSQL databázy bez straty dát. Základom je HTTPry, do ktorého sa naimplementuje POstgreSQL klient na pripojovanie a ukladanie do databázy. Funkcia ukladania do súboru alebo posielania na štandardný výstup zostane zachovalá.

3.3 Webové rozhranie na zobrazovanie logov

Webové rozhranie je prostriedkom umožňujúcim vytvárať si štatistiky v tabuľkách a grafoch. Spravovať užívateľov webového rozhrania na základe práv nad databázou. Nevyhnutnou súčasťou je jednoduchá správa databázy.

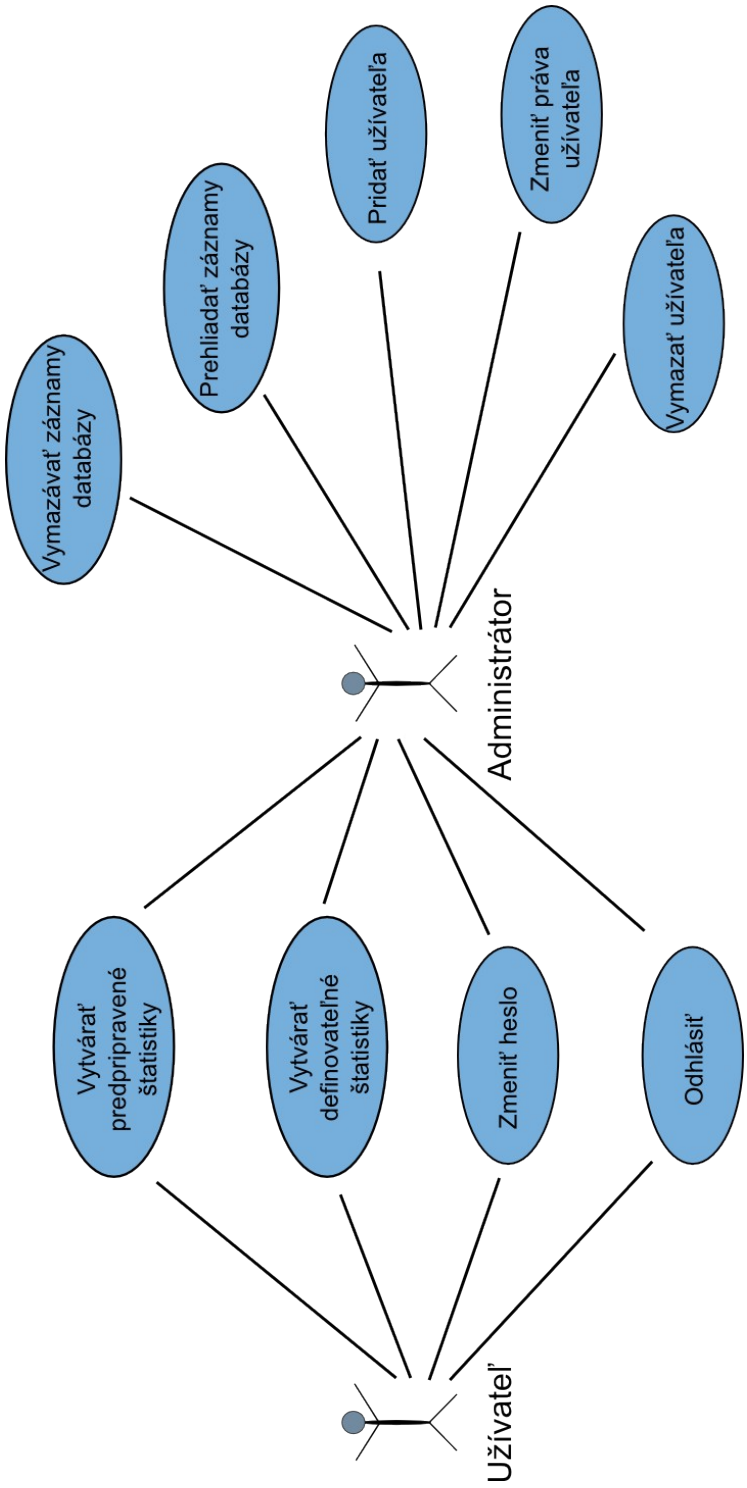
3.3.1 Administrácia

Užívatelia webového rozhrania môžu mať právo administrátora alebo užívateľa. Podľa toho majú prístup k možnostiam správy systému.

- Užívatelia - zobrazujú preddefinované štatistiky, ktoré zvolia pomocou filtra
  - zobrazujú definovateľné štatistiky
  - majú možnosť si zmeniť heslo

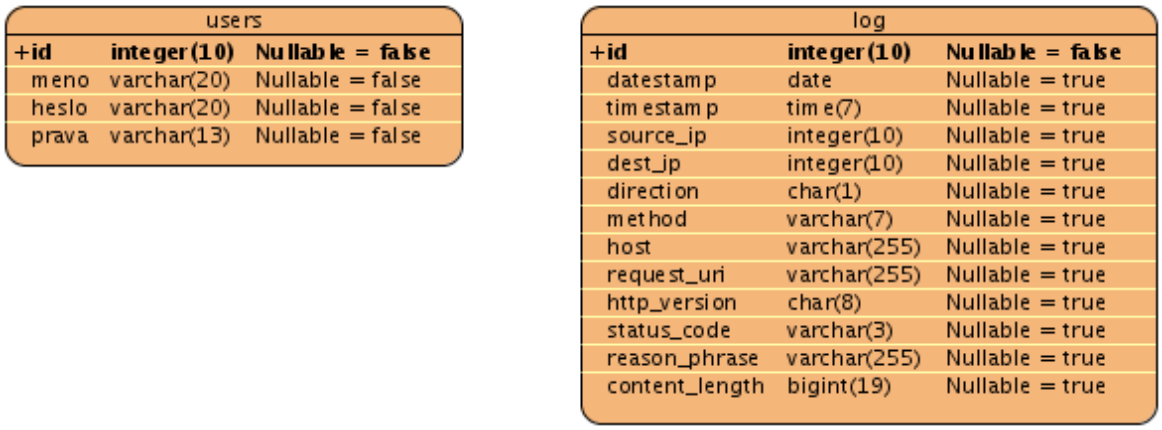
- Administrátori - majú rovnaké mmožnosti ako užívatelia plus rožšírené o administráciu
  - možu vytvárať užívateľov, meniť ich práva a vymazať užívateľov
  - vymazávať riadky databázovej tabuľky, v ktorej sú uchovávané logy

3.3.2 Use Case Diagram



Obrázok 3.2: Use Case Diagram

3.3.3 ER Diagram



Obrázok 3.3: ER diagram

3.4 Bezpečnosť

Požiadavka bezpečnosti by mala byť samozrejmá. Bral som na to ohľad počas návrhu aplikácie. Základné opatrenia sú:

- prenášať heslá v šifrovanej podobe
- heslá užívateľov web rozhrania musia byť uložené v databáze v šifrovanej podobe
- administrátor databázy musí prideliť práva užívateľovi, ktorí sa pomocou PostgreSQL klienta môže pripojiť na ukladanie logov do databázy

Splnením týchto podmienok výrazne zvýšime ochranu hesiel užívateľov. V prípade pripájania sa na databázu diaľkovo, t.j. iné ako localhost spojenie sa odporúča použit SSL šifrované spojenie. Toto opatrenie výrazne prispenie k zabezpečeniu spojenia a ochrany prenášaných dát.

4 Implementácia

Táto kapitola opisuje implementáciu systému podľa návrhu z predošlej kapitoly. Je rozdelená do troch hlavných častí.

- HTTP analyzátor HTTPry
- PostgreSQL databáza
- Webové rozhranie

4.1 Použité technológie

Využil som programovací jazyk C. Nutnosť využiť ho vyplívala z dôsledku, že HTTP analyzátor je v jazyku C naprogramovaný. To nám však vyhovuje pretože prináša výhody ako rýchlosť a prenositeľnosť. Klient naimplementovaný do analyzátora využíva knižnicu *libpq*. Ako databázove riešenie je použitá PostgreSQL databáza. Vysoká spoľahlivosť a dobrý výkon pri súbežnom spracovaní koplikovanejších SQL dotazov bez ohľadu na objem dát [22]. Kvalitná dokumentácia vytvárajú dobrý predpoklad pre širšie nasadenie do budúcnosti.

Na implementáciu webového rozhrania som použil jazyka PHP. Doplnení o kaskádové štýly, pre prehľadnejšie a prívetivešie webové rozhranie.

4.2 HTTP analzyzátor HTTPry

Úlohov je ukladať logy do súboru a zároveň do databázy. To dosiahnem naimplementovaním databázového klienta priamo do programu HTTPry. Klient potrebuje na svoju správnu funkciu modul **libpq**. Pre správnu kompiláciu som do makefilu vložiť parameret **-lpq**. Do súborov httpry.c a format.c som pripojil hlavičku **libpq-fe.h**. Pro preštudovaní zdrojových kódov som zisti, že výpis logov do súboru alebo na štandardný výstup obsluhuje funkcia *print_format_values()* v súbore *format.c*, hlavičkou je súbor *format.h*. Hlavný súbor je *httpry.c*.

Najprv som vytvril funkciu *connect_to_database()* na vytvorenie spojenia s databázou, ktorú som umiestnil do hlavného súboru. Volanie funkcie som umiestnil pred volanie funkcie *pcap_loop()*, ktorá ma na starosti čítanie paketov. Pre výpis hodnôt sa volá funkcia *print_format_values()*. V nej rozkladám typ timestamp zvlášť na dátum a čas a tak ich ukladám do databázy. Toto riešenie som zvolil aby som si zjednodušil dotazy a zároveň umožnil hľadať dátum a čas nezávisle na sebe. Ako prvé uložím všetky hodnoty riadka tabuľky do pola const char *insertValues[12]. Následne volám funkciu *PQexecParams()*. Tá vykoná INSERT dotaz na databázu. Tento proces sa opakuje, pre každú HTTP správu. Informácie z jednej správy sú uložené do jedného riadka tabuľky **log**. V závislosti od nastavenia postgreSQL databázy, sa po každom dotaze automaticky vykoná dotaz COMMIT. To spomaľovalo ukladanie do databáze. Vyriešil som to manuálnym volaním COMMIT. Začnem

dotazom BEGIN, potom nasleduje dvadsať dotazov INSERT a dotaz COMMIT. Tým to som zvýšil rýchlosť ukladania.

V pri ukončení programu HTTPry sa volá funkcia *cleanup()*, ktorá má na starosti čisté ukončenie aplikácie, to je uvolnenie obsadenej pamäte, zatvorenie súboru a iné. V nej si volám funkciu *end_connection_to_database()*, tá má na starosti ukončenie spojenia s databázou. Pri volaní funkcií klienta, vždy kontrolujem návratovú hodnotu. Ak by sa vyskytla chyba, vypíše sa chybová správa na chybový výstup a ukončí sa spojenie s databázou volaním funkcie *exit_nicely()*.

4.3 PostgreSQL databáza

Vytvoril som databázu pod menom *tom. Tom* má všetky práva superužívateľa.

$ createdb -U tom -W testdb

Vytvoril som užívateľa, pod ktorým sa bude HTTPry prihlasovať do databáze.

#CREATE ROLE uzivatel_1 ENCRYPTED PASSWORD 'uzivatel' ;

4.3.1 Tabuľky

Pri návrhu tabuliek som vychádzal z potrieb aplikácie. Prvú tabuľku som nazval **log**. Je vytvorená na uchovávanie logov vytvorených analyzátorom. Analyzátor ukladá spracované HTTP požiadavky a odpovede do riadka s formátovými hodnotami nadefinovanými pri spustený analyzátora(hodnotami, ktoré sa majú odchytávať z HTTP správ). Pričom pri HTTP požiadavke a odpovedi odchytávam rozdielne údaje. Toto ukladanie nie je najefektívnejšie, nakoľko do niektorých stĺpcov sa neukladajú dáta nikdy. Ak nemám hodnotu na uloženie do poľa tabuľky, ukladám '-'.
Tabuľka log:

```
TABLE log (
          datestamp       date,
          timestamp       time,
          source_ip       inet,
          dest_ip         inet,
          direction       char(1),
          method          varchar(7),
          host            text,
          request_uri     text,
          http_version    char(8),
          status_code     varchar(3),
          reason_phrase   text,
          content_length  bigint,
          id              SERIAL,
          primary key(id)
)
```

Pri každej HTTP správe sa ukladajú hodnoty do *datestamp, timestamp, source_ip, dest_ip,* direction a *http_version*. HTTP požiadavka obsahuje hodnoty *method, host, request_uri*. HTTP odpeveď obsahuje hodnoty *status_code*, *reason_phrase, content_lenght*.

Tabuľka **users** obahuje uživateľov, ktorí majú prístup k webovému rozhraniu. Obsahuje meno, heslo, práva užívateľov. Užívateľ môže mať právo administrátora alebo užívateľa.

Tabuľak users:

```
TABLE users (
          meno       varchar(20) NOT NULL,
          heslo      varchar(100) NOT NULL,
          prava      varchar(13) NOT NULL,
          id         SERIAL,
          primary key(id)
)
```

4.3.2 Inštalácia

Súbory na vytvorenie tabuliek a vloženie hodôt do tabuliek je v adresáry *postgresq*.

Vytvorenie tabuľky log, vytvorenie tabuľky users, naplnenie tabuľky users užívateľmi:

#\i create_table_log.sql

#\i create_table_users.sql

#\i insert_table_users.sql

Pridelie práv užívateľovi *uzivatel_1* na vykonávanie dotazov nad tabuľkou:

#GRANT ALL ON log TO uzivatel_1;

#GRANT ALL ON log_id_seq TO uzivatel_1;

4.3.3 Bezpečnosť

Zvýšenie bezpečnosti som dosiahol pomocou nasledujúcich opatrení:

- Heslo medzi PostgreSQL klientom a PostgreSQL databázou sa prenáša v md5 šifrovanej podobe.
- Heslá užívateľov sú v databáze uchovávané md5 hashom.
- Kontrola hesla vo webovom rozhraní prebieha porovnávaním hesiel v md5 šifrovanej podobe.

4.4 Webové rozhranie

Implementácia vychádza z navrhnutého prípadu užitia na Obrázku 3.2. Pri návrhu dizajnu som sa zameral na prehľadnosť a vecnosť. Úžívateľ sa rýchlo zorientuje. Jazyk rozhrania som zvolil slovenský jazyk. Do budúcnosti plánujem rozširiť o angličtinu. Pri programovaní som využil jazyky PHP, CSS a JavaScript. Architektúra rozhrania je založená na súbore *index.php*. V tomto súbore sa získavajú parametre pomocou GET.
Využité toolkity:

- *Google Chart API* - slúži na dynamické generovanie grafov[23], široká prisôsobiteľnosť ho predurčovali ako správnu voľbu na generovanie grafov

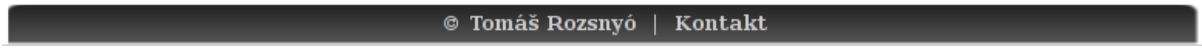
- *The php calendar component, written by TJ @triconsole* - slúži na výber dátumu pomocou prehľadného výberu z minikalendára alebo klasického dropdown menu

- *jQuery* - je rýchla a presná JavaScript knižnica, ktorá zjednodušuje kríženie HTML dokumentu, spracovanie manipulačných udalostí a oživenie využitím animácie[25].

4.4.1 Rozloženie prvkov

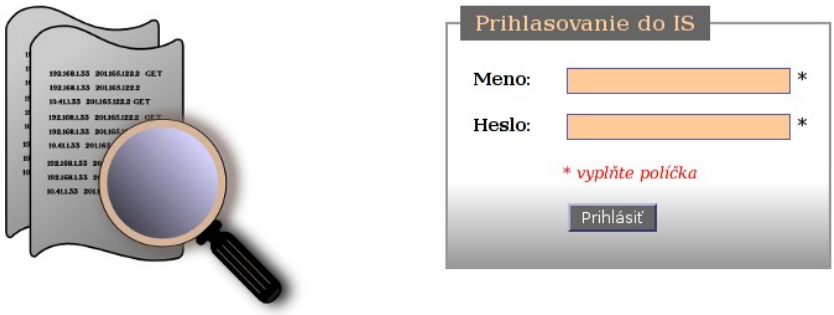
Zvolil som pevnú širku stránky 770px. Rozhodol som sa tak s ohľadom na užívateľov ešte stále používajúcich rozlíšenie obrazovky 800x600. Chcel som predíšť nutnosti horizontálne pohybovať obrazovkou. Stránku som rozdelil do troch hlavných častí. Hlavičku, stred sránky a pätu. Pričom dve z nich sú fixné prvky, hlavička a päta.

Obrázok 4.1: Hlavička stránky



Obrázok 4.2: Päta stránky

Úvodná stránka je prihlasovacia. Po zadaní korektných prihlasovacích údajov sa pod hlavičkou stránky objavý menu. Menu obsahuje tlačítka v závislosti či je prihlásený užívateľ alebo administrátor. Podľa zvolenej položky menu sa zobrazuje obsah strednej časti.



Obrázok 4.3: Prihlasovanie

Obrázok 4.4: Administrátorské menu

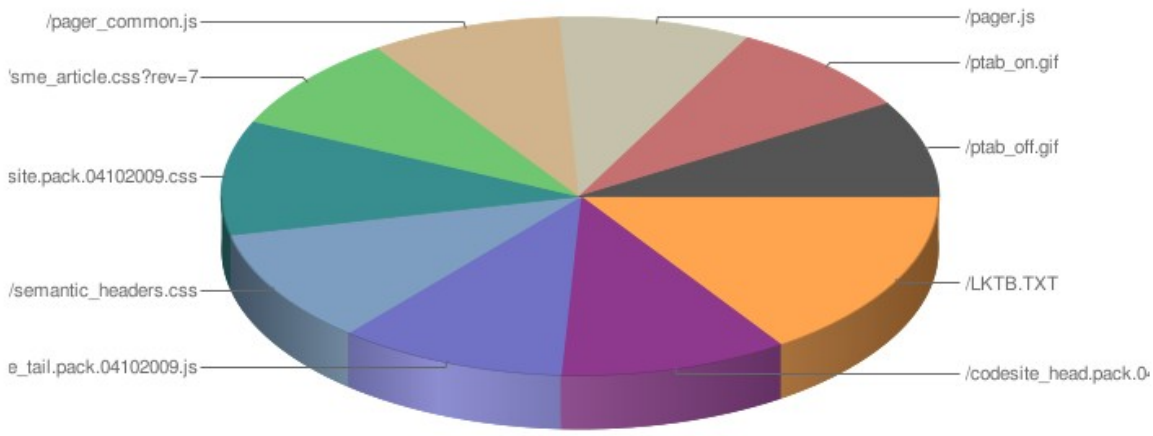
Obrázok 4.5: Užívateľské menu

Presná špecifikácie jednotlivých podstránok:

- **Predpripravené Štat.** - zobrazuje štatistiky v tabuľkách a grafoch podľa zadaného filtra
- **Definovateľné Štat.** - zobrazuje nadafinové štatiskity v tabuľkách podľa vybraných hodnôt
- **Databáza** - zobrazovanie a mazanie riadkov z tabuľky log
- **Užívatelia** - pridávanie, zmena práv, zrušenie uživateľov
- **Zmeň heslo** - zmena hesla
- **Ohlásiť** - odhlásenie zo systému

4.4.2 Zobrazenie štatistík

Zobrazovanie štatistík z uložených logov je veľmi významnou časťou práce, preto som jej venoval značné úsilie. Štandardným prostriedkom zobrazovania je tabuľa. Pomocou kaskádových štýlov som docielil prehľadné zobrazenie aj väčšieho množstva dát. Z hľadiska čitateľnosti a lepšej názornosti je príjemnejšie grafové zobrazenie. Prostredníctvom Google Chart API som vygeneroval grafy. Bolo potrebné vyselektovať, úpraviť dáta. Pripraviť ich do štandardnej podoby pochopitelnej google serverom. Server vrátil vygenerovaný obrázok, ktorý som vložil do stránky. Farby som volil zo širokej škály farieb. Ako vidieť na obrázku 4.6, graf je jasne čitateľný.



Obrázok 4.6: Vygenerovaný Google Chart graf

4.4.3 Administrácia užívateľov

Administrácia užívateľov zahrňuje úkony: vytváranie nového užívateľa webového rozhrania, zmenu práv, zrušenie užívateľa a zmenu hesla. Vymenované úkony plne postačujú k štadardnej administrácii. Heslá sú prenášané v šifrovanej md5 podobe, čo výrazne zvyšuje ochrany pred odcudzením.

4.4.4 Inštalácia

Pre nainštalovanie webového rozhrania je treba prekopírovať adresár *PHP* do adresára web servera.

5 Testovanie

Rýchlosť bezchybného spracovania je kľúčovým faktorom správneho funguvania celej práce. Rýchlosť spracovania boli otestované na každej časti zlášť a nakoniec spolu ako celoku. Testovanie HTTPry prebiehalo pomocou dvoch testovacích skriptov. Testovacie skripty som napísal v pythone. Pracujú na princípe cyklického vytvárania HTTP požiadavok. V jednej som využil funkcii **wget**, ktorá sťahuje len indexové súbory webovej stránky. Webová stránka je jediný parameter wget. Vybral som osem stránok. Volil som stránky s najmenším index súborom, to z dôvodu, čím kratšie sťahovanie tým viac index súborov stiahne to je tým viac HTTP správ sa vymení počas komunikácie. Druhý pracuje na podobnom princípe s tým rozdielom, že v ňom volám program webclient (prvý projekt do predmetu IPK). Webclient šťahuje okrem index súboru aj obrázky, ktorých cesta sa nachádza v index súbore. Testovacie skripty je možné nájsť na priloženom CD v adresáry *testery zataze*. HTTPry počas testovania dvoma testovacími skriptami vykazovalo zaťaženie procesora 0,5% až 1% na procesore Centrino Duo T2250 1.73GHz. Využitá pamäť bola 0.1% pri 1GB RAM čo predstavuje okolo 5MB. Pri všetkých vykonaných testoch skončil HTTPry so 100% úspešnosťou spracovania HTTP paketov.

Databázu som testoval na rýchlosť vkladania do databáze dotazom INSERT. Pripravil som jednoduchého klienta na vkladanie do databazy. Test úspešne odskúšať rýchlosť vkladania. Test som opakoval tri krát. Čas som meral C programom *time*. V cykle som ukladal dáta do databazy (tabuľka bola vždy prázdna):

100	INSERTOV	=	real	0m0.078s	real	0m0.116s	real	0m0.065s
			user	0m0.003s	user	0m0.001s	user	0m0.003s
			sys	0m0.004s	sys	0m0.004s	sys	0m0.005s
1000	INSERTOV	=	real	0m0.655s	real	0m0.616s	real	0m0.629s
			user	0m0.021s	user	0m0.021s	user	0m0.022s
			sys	0m0.025s	sys	0m0.028s	sys	0m0.023s
10000	INSERTOV	=	real	0m6.361s	real	0m6.882s	real	0m7.160s
			user	0m0.211s	user	0m0.206s	user	0m0.162s
			sys	0m0.195s	sys	0m0.219s	sys	0m0.209s

Predpokladal som, že rýchlosť bude postačujúca. No nakoniec sa ukázalo, že nie je. Primárne databáza PostgreSQL vykonáva auto-commit po každom vložení. Čo veľmi spomaluje vkladanie. Nakoniec sa mi podarilo dosiahnuť väčšiu rýchlosť vkladania obnovovaním (dotaz COMMIT) dát v tabuľke po každom päťdesiatom vložení. Čo prinieslo výrazné zlepšenie rýchlosti.

Dôkladne som otestoval funkčnosť webového rozhrania, so zameraním na prenos a kontrolu šifrovaných hesiel. Zachytávanie logov analyzátorom bolo bez problémov vyskúšané na troch sietiah vrátane školskej sieti kolejnet. Celú prácu som implementoval a testoval na svojom notebooku.

6 Záver

Základným cieľom bakalárskej práce bolo analyzovať HTTP tok odchytávaním HTTP správ. Čoho som dosiahol aplikovaním programu HTTPry. Do programu HTTPry som naimplementoval klienta na pripájanie a ukladanie zachytených a spracovaných HTTP správ do databázy PostgreSQL. Vytvorením webového rozhrania som umožnil užívateľovi jednoducho a prehľadne kontrolovať uložené dáta. Webové rozhranie obsahuje správu užívateľov webového rozhrania, rovnako ako jednoduchú správu databáze. Rýchlost samotného HTTPry a PostgreSQL databázy boli dôkladne otestované, napriek tomu sa vyskytli problémy pri ukladaní logov z HTTP analyzátora do databázy, kde v určitých prípadoch HTTP analyzátor nestíha ukladať spracované logy. To znamená, že štatistiky nie sú 100% spoľahlivé. Tento problém sa mi podarilo z časti vyriešiť vypnutím autocommit. Naimplementoval som manuálne potvrdzovanie po päťdesiatich vloženiach (INSERT). Táto úprava výrazne zvýšila rýchlosť procesu ukladania dát do databázy. Dosiahol som hodnotu 950 HTTP paketov za minútu. Podľa testovania sa straty začínajú objavovať pri prijatí viac ako 950 HTTP paketov za minútu. V praxi to znamená, že pre bezstratové ukladanie logov je nasadenie analyzátora odporúčané v sieti s maximálne piatimi užívateľmi. V literatúre [21] sa uvádzajú nastavenia konfiguračných súborov, no bez predchodzích znalostí PostgreSQL databázy som nebol schoný za zostávajúci čas vyladiť výkon databázy na úroveň, ktorá by bola aplikovateľná pre väčší počet užívateľov siete.

Celkovo môžem zhodnotit výsledok mojej práce za úspešný a prínosný. Podarilosa mi splniť všetky požiadavky vyplývajúce zo zadania. Z návrhu sa mi podarilo splniť dva body úplne a jeden čiastočne.

Navrhnutý a implementový systém umožňuje jednoduché monitorovanie webovej komunikácie pomocou webového rozhrania. Práca nájde využitie v podnikoch aj domácnostiach, kde je požiadavka na analyzovanie HTTP toku. V podnikoch je vhodná aplikácia je na Proxy server, cez ktorý pristupujú počítače z LAN siete na internet. Rozširuje možnosti správy siete. Administrátor získava väčšiu prehľadnosť chovania klientov na sieti tým pádom môže preventívne zakročiť v prípade hrozby narušenia bezpečnosti. V domácnosti je využiteľný napríklad ak rodičia majú záujem kontrolovať domény, ktoré navštevujú ich deti.

Rozšírenia by sa týkali HTTPry analzyzátora a aj webového rozhrania. HTTPry spracúva a ukladá HTTP požiadavky a odpovede v poradí akom sú spracované. HTTPry by som upravil aby ukladalo HTTP požiadavku a HTTP odpoveď spolu do jedného riadka tabuľky. Toto vylepšenie by dovolilo vytvárať presnejšie štatistiky. Webové rozhranie umožňuje vytvárať štatistiky, mazať údaje z databázy, správu užívateľov. Ako ďaľšie rožšírenie vidím možnosť väčšej kontroly nad databázou. Ucelenejší grafický dizajn by prispel k väčšej prehľadnosti webového rozhrania.

Literatúra

[1] *Aplikačná vrstva.* [online]. Dostupný z URL: <http://cs.wikipedia.org/wiki/Aplikační_vrstva>

[2] *List of web browsers.* [online].Dostupný z URL: <http://en.wikipedia.org/wiki/List_of_web_browsers#Text-based>

[3] *Hypertext Transfer Protocol.* [online]. Dostupný z URL: <http://cs.wikipedia.org/wiki/HTTP#HTTP_1.1_Aktualizovan.C3.A1_hlavi.C4.8Dka>

[4] *Hypertext Transfer Protocol.* [online]. Dostupný z URL: <http://en.wikipedia.org/wiki/HTTP>

[5] *Hypertext Transfer Protocol -- HTTP/1.1.* [online],. Dostupný z URL: <http://URL.ietf.org/rfc/rfc2068.txt>

[6] *Hypertext Transfer Protocol -- HTTP/1.1*. [online]. Dostupný z URL: <http://URL.ietf.org/rfc/rfc2616.txt>

[7] *Hypertext Transfer Protocol -- HTTP/1.0*. [online]. Dostupný z URL: <http://tools.ietf.org/html/rfc1945>

[8] *Prednášky predmetu IPK.* [online].

[9] *PCAP*. [online]. Dostupný z URL: <http://URL.tcpdump.org/pcap3_man.html>

[10] *The Sniffer's Guide to Raw Traffic*. [online]. Dostupný z URL: <http://yuba.stanford.edu/~casado/pcap/section1.html>

[12] *Converted the site from MySQL to PostgreSQL*. [online]. Dostupný z URL: <http://jamonation.com/node/734>

[13] *Benchmarking Drupal on PostgreSQL vs. MySQL*. [online]. Dostupný z URL: <http://2bits.com/articles/benchmarking-postgresql-vs-mysql-performance-using-drupal-5x.html>

[14] Welling, L., Thompsonová, L.: *PHP a MySQL rozvoj webových aplikací*. Praha, SoftPress s.r.o, 2004. ISBN 80-86497-60-7

[15] Krejčí, L.: *PHP Kapesní přehled*. Brno, Computer Press, 2006. ISBN 80-251-0808-2

[16] Hugh, E. W., Lane, D.: *PHP a MySQL*. Brno, Compruter Press, 2002. ISBN 80-7226-760-4

[17] Staníček, P.: *CSS Hotová řešení*. Brno, Compruter Press, 2003. ISBN 80-251-1031-1

[18] Kroužek, J., Domes, M.: *CSS Kapesní přehled*. Brno, Computer Press, 2006. ISBN 80-251-0773-6

[19] Croft, J., Lloyd, I., Rubin, D.: *Mistrovství v CSS*. Brno, Computer Press, 2007. ISBN 978-80-251-1705-7

[20] *PostgreSQL Reference Manual - Programing Guide*. [online]. Dostupný z URL: <http://URL.network-theory.co.uk/docs/postgresql/vol2/MainFunctions.html>

[21] *PostgreSQL Documentation*. [online]. Dostupný z URL: <http://URL.postgresql.org/docs/8.3/interactive/index.html>

[22] Stěhule, P.: *Desatero*. [online]. Dostupný z URL: <http://URL.postgres.cz/index.php/Desatero>

[23] *Google Chart API*. [online]. Dostupný z URL: <http://code.google.com/apis/chart/>

[24] Kerninghan, B. W., Ritchie, D. M.: *Programovací jazyk C*. Brno, Computer Press, 2006. ISBN 80-251-0897-X

[25] *jQuery*. [online]. Dostupný z URL: <http://jquery.com/>

Dodatek A

Obsah CD

Priložené CD obsahuje:

- adresár ***httpry-0.1.4*** - HTTP analyzátor so všetkými súbormi vrátane Makefile
- adresár ***testery zataze*** - obsahuje dva zaťažovacie testy a program webclient
- adresár ***PHP*** - obsahuje webové rozhranie
- adresár ***postgresql*** - súbory prislúchajúce k databáze
- súbor ***README*** - obsahuje popis obsahu CD, postup inštalácie práce