VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

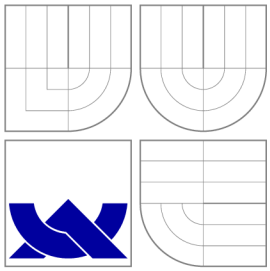
ANALYZÁTOR HTTP PROVOZU S WEBOVÝM ROZHRANÍM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

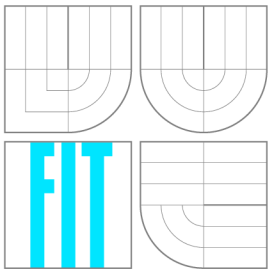
AUTOR PRÁCE
AUTHOR

MARTIN ŽIŽKA

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

ANALYZÁTOR HTTP PROVOZU S WEBOVÝM ROZHRANÍM

HTTP ANALYZER WITH WEB USER INTERFACE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MARTIN ŽIŽKA
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ TOBOLA
SUPERVISOR

BRNO 2010

Abstrakt

Tato práce se zabývá problematikou analýzy HTTP provozu. Popisuje princip a činnost HTTP protokolu. Dále se věnuje struktuře a funkčnosti existujícího nástroje pro analýzu HTTP provozu HTTPry. Zajímá se též o rozšíření nástroje HTTPry o ukládání záznamů do databáze MySQL, podporu IPv6 protokolu a technologie VLAN. Popisuje také metody testování nástroje a porovnává dosažené výsledky upravené verze s původní verzí nástroje. Uvádí návrh a implementaci informačního systému vzniklého pro zobrazení zachycených informací. V závěru diskutuje výsledky práce a možnosti dalšího rozšíření této práce.

Abstract

This thesis provides a issue of the analysis HTTP traffic. It describes the principle and the procedure of HTTP protocol. It is interested in the structure and function of the existing tool for the analysis HTTP traffic HTTPry. It is also interested in the extension of the HTTPry tool about saving records in the database MySQL, IPv6 protocol maintenance and the VLAN technology. It also describes the methods of the testing tool and compares achieved results of modified version with the original tool version. It mentions the proposal and implementation of the information system, which views the captured informations. In the end, there are the results and another opinions of the extension discussed.

Klíčová slova

Httpry, HTTP analyzátor, HTTP protokol, zachytávání HTTP toku

Keywords

Httpry, HTTP analyzer, HTTP protocol, capture HTTP flow

Citace

Martin Žižka: Analyzátor HTTP provozu s webovým rozhraním, bakalářská práce, Brno, FIT VUT v Brně, 2010

Analyzátor HTTP provozu s webovým rozhraním

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jiřího Toboly. Uvedl jsem všechny zdroje, ze kterých jsem čerpal.

.....
Martin Žižka
18. května 2010

Poděkování

Tímto bych rád poděkoval vedoucímu práce Ing. Jiřímu Tobolovi za trpělivost, rady a poskytnutou pomoc při tvorbě této práce. Dále bych chtěl poděkovat své rodině za podporu při psaní této práce.

Obsah

Kapitola 1

Úvod

V dnešní době má téměř každý počítač přístup na internet. Největší využití internetu pro běžného uživatele je navštěvování webových serverů a prohlížení webových stránek. Webové stránky nabízejí nepřeberné množství informací, služeb a zábavy. Internet mají často dostupný např. zaměstnanci ve firmách, úřadech, ale též učitelé a žáci škol. Z pohledu zaměstnavatelů je vhodné hlídat použití internetu ze strany zaměstnanců.

Snadným řešením je webové služby zakázat, to však vždy není možné. Někteří z výše uvedených potřebují přístup na web např. za účelem online technické dokumentace, mediálních prostředků nebo vzdělávání. Bylo by možné nastavit, na které stránky uživatelé smí a na které ne, ale takové nastavení by bylo velice zdlouhavé. Proto je vhodné použít nástroj pro analýzu HTTP provozu, kde zodpovědná osoba má přehled nad využitím internetu.

Cílem této bakalářské práce je seznámit se s technologiemi pro tvorbu webového rozhraní a s nástroji pro zachytávání HTTP provozu. Vybraný nástroj je třeba upravit pro ukládání záznamů do relační databáze. Důležitým bodem je testování upraveného nástroje pro zachytávání provozu. Pro zobrazení zachycených záznamů a statistik je třeba navrhnout a implementovat informační systém.

V kapitole 2 shrnuje teoretické poznatky o síti a HTTP protokolu. Vysvětluje princip činnosti protokolu, jeho vývoj a popis dotazu, odpovědi, metod a návratových kódů. V kapitole 3 uvádí informace o části práce zachytávající HTTP provoz. Obsahuje odůvodnění volby nástroje HTTPry. Seznamuje s činností nástroje a principem analyzování jednotlivých paketů obsahujících HTTP komunikaci. Popisuje také provedené úpravy a rozšíření tohoto nástroje o ukládání do databáze `MySQL` v rámci zadání této práce. Popisuje také implementaci dalších rozšíření nástroje. Jedná se o podporu IPv6, VLAN a další. HTTPry. Kapitola 4 seznamuje s principy testování nástroje HTTPry a obsahuje výsledky jednotlivých testů. Dále popisuje návrh a implementaci dalších rozšíření provedených z důvodu vylepšení vlastností provedených úprav nástroje. Testování je zaměřeno na rychlost a spolehlivost ukládání záznamů.

V kapitole 5 jsou uvedeny informace o použitých technologiích a návrh informačního systému. Popisuje návrh jednotivých statistik zobrazovaných v informačním systému a naznačuje princip výpočtu těchto statistik. Jedná se o statistiky nejnavštěvovanějších serverů, nejaktivnějších uživatelů, intenzita HTTP provozu, záznamy provozu a realtime analýza. Kapitola 6 ukazuje již implementovaný informační systém pomocí obrázků jednotlivých statistik, případně pro každou statistiku uvádí zajímavosti či složitější části z implementace. Závěrečná kapitola zhodnocuje dosažené výsledky a diskutuje možná rozšíření této práce.

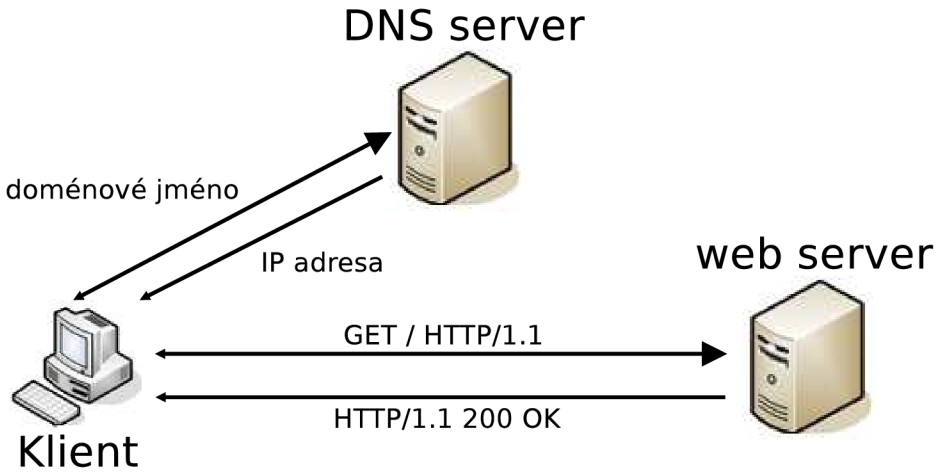
Kapitola 2

HTTP protokol

Tato kapitola seznamuje s protokolem HTTP používaným WWW službou. Obsahuje základní informace o HTTP protokolu, jeho vývoj a způsob komunikace.

2.1 Charakteristika

Hypertext transfer protocol, zkráceně HTTP, je protokol aplikační vrstvy. Používá se pro distribuované informační systémy a pro službu WWW. Pracuje s URL adresami a je určen pro hypertextová prostředí, zpracovává informace různých typů a pomocí hypertextových odkazů tyto informace poskytuje uživateli. Jedná se o protokol textový a bezstavový, komunikuje pomocí zpráv. HTTP protokol je typu klient/server. Klientem HTTP protokolu je většinou webový prohlížeč, který podle zadaného URL vyšle dotaz na server se žádostí o uvedený dokument. Server po přijetí dotaz zpracuje a odešle klientovi odpověď.



Obrázek 2.1: Činnost HTTP protokolu

Spolehlivý přenos dat zajišťuje protokol transportní vrstvy TCP, a to převážně na portu 80. Jedná se o implicitní port pro HTTP protokol. Do URL je však možné zadat i jiný port. Dalšími možnými porty, na kterých se vyskytuje HTTP komunikace, jsou porty 8080,8008 a 519 [24].

HTTP protokol tedy předpokládá, že na jednom konci předá data pro přenos a na druhém konci obdrží data v nezměněné podobě.

První verze HTTP protokolu s označením 0.9 zajišťovala pouze přenos dat bez informací o přenášených datech. Klient musel typ dat odhadnout např. podle koncovky souboru.

Další verze s označením 1.0 doplnila informaci o typu dat do dotazu a odpovědi. Použila pro to existující formát MIME (Multipurpose Internet Mail Extension). Datový přenost poté připomíná emailovou komunikaci. Skládá se z metadat, která tvoří hlavičky zpráv obsahující informace potřebné pro přenos dat mezi účastníky HTTP komunikace, a vlastních dat, která tvoří tělo zprávy. Verze HTTP protokolu 1.0 je popsána v [1]. S rozvojem WWW služby vznikaly další požadavky na protokol HTTP. Jednalo se např. o hierarchickou strukturu proxy, využití cache, virtuální servery a trvalé spojení mezi klientem a serverem.

Tyto požadavky řeší další verze protokolu HTTP označená 1.1, která je popsána v [7], později upravena v [8]. Trvalé spojení umožňuje přes jedno TCP spojení zasílat všechny své požadavky na server, který posílá své odpovědi po tomto spojení. Při použití této verze protokolu je implicitní trvalé spojení. Není-li tedy určeno jinak, klient předpokládá, že server udržuje trvalé spojení. Spojení je ukončeno až odesláním hlavičky `Connection: close`. Zpráva obsahující tuto hlavičku je chápána jako poslední. Pokud tuto hlavičku odešle server, nesmí již klient odeslat další požadavky na server. Všechny zprávy zasílané přes trvalé spojení musí mít v hlavičce `Message-length` definovanou délku. Trvalým spojením se sníží zátěž serveru, který jinak musí navázat několik spojení během krátkého časového intervalu. Zřetězené zpracování umožňuje posílat více dotazů na server, aniž by klient čekal na odpovědi. Server je poté povinnen zasílat odpovědi ve stejném pořadí, jako přijal dotazy. Verze 1.1 zavádí hlavičku `Host`. Definuje zpracování této hlavičky jak na straně serveru, tak na straně klienta. Podle této hlavičky může server na různá jména počítačů vracet různé odpovědi. Vyjednávání o obsahu umožňuje poslat co nejlepší variatu dokumentu, pokud jich existuje více. Serverem řízené dohadování vychází z informací v hlavičkách dotazu Accept, Accept-Charset, Accept-Encoding, Accept-Language, User-Agent a existujících variant dokumentu. Klientem řízené dotazování spočívá ve výběru nejlepší varianty klientem na základě první odpovědi od serveru, kde jsou uvedeny v hlavičce Alternates nebo v těle odpovědi jako seznam URL jednotlivých variant. Je však potřeba druhý dotaz pro získání samotného dokumentu.

2.2 Syntaxe HTTP protokolu

Protože jsou zprávy HTTP protokolu v textovém tvaru, musejí být syntakticky analyzovány. Jednotlivé hlavičky mají obecně tvar:

```
název hlavičky : hodnota CRLF
```

Výjimku tvoří první řádek, který se liší v dotazu a odpovědi. Přesný tvar je rozebrán v kapitolách 2.3 a 2.4 pojednávajících o formátu dotazu a odpovědi. Hlavička je od zprávy oddělena dvojicí `CRLF`, poté již následuje samotná zpráva podle protokolu MIME. Význam všech hlaviček pro jednotlivé verze protokolu vychází z [1] a [7].

2.3 Formát dotazu

Většina komunikace přes HTTP protokol je iniciována ze strany klienta a sestává se z požadavku na webový server. Požadavek se skládá z metody protokolu, umístění požadovaného

dokumentu na serveru a verze protokolu. V dotazu klienta není specifikovaná adresa ani jméno webového serveru. IP adresu nese protokol nižší vrstvy, a to IP protokol. Podobně protokol TCP specifikuje porty.

V praxi je používáno několik verzí HTTP protokolu. Proto server musí vyhovět požadavkům klienta a odpovědět v požadované nebo v nižší verzi protokolu . Příklad dotazu je uveden zde:

```
GET /index.html HTTP/1.1
Accept: text/html
Accept: image/gif
User-Agent: Mozilla
/* prázdný řádek */
```

První řádek se nazývá dotazový. První parametr určuje metodu protokolu, kterou klient požaduje. Uvádí se vždy velkými písmeny. Server nemusí všechny metody podporovat, v případě požadavku na nepodporovanou metodu vrací chybové hlášení. Většina WWW serverů podporuje metody GET,HEAD,POST a PUT. Druhý parametr udává URL požadovaného dokumentu. Může být v dotazu uvedeno ve třech tvarech.

1. `*`
2. absolutní URL
3. absolutní cesta

Tvar URL závisí vždy na typu požadavku. Hvězdička znamená požadavek na server jako celek.

```
OPTIONS * HTTP/1.0
```

Absolutní URL se používá, pokud je požadavek směrován na proxy.

```
GET http://www.w3.org/pub/WWW/TheProject.html HTTP/1.1
```

Absolutní cesta udává umístění souboru na originálním serveru, se kterým je navázáno spojení pomocí nižších vrstev. Verze HTTP 1.1 posílá v hlavičce doménové jméno nebo IP adresu v hlavičce `Host:`. Absolutní cesta pro kořenový adresář musí obsahovat `/`.

```
GET /pub/WWW/TheProject.html HTTP/1.1
Host: www.w3.org
```

Třetí parametr udává požadovanou verzi HTTP protokolu, udává se vždy ve tvaru `HTTP/x.x`, kde `x.x` udává číslo verze protokolu.

2.3.1 Metody protokolu HTTP

CONNECT Pro použití s proxy serverem

DELETE Požadavek na zrušení dokumentu na serveru.

GET Metoda GET představuje požadavek na zaslání dokumentu určeného pomocí URL adresy. Může být také použita pro odeslání formuláře na server. Je však omezena délkou.

HEAD Metoda HEAD je identická s metodou GET s tím rozdílem, že server nemusí posílat tělo odpovědi. Používá se pro získání dodatečných informací o dokumentu nebo testování dostupnosti hypertextových odkazů a jejich poslední modifikace

OPTIONS Představuje dotaz na možnosti komunikace s uvedeným URL, pomáhá klientovi určit omezení spojení a možnosti serveru.

POST Metoda POST se používá v případě, kdy má cílový server přijmout data z požadavku. Používá se většinou pro odeslání vyplněného formuláře. Výsledkem POST metody může být odeslání mailu, předání dat či uložení do databáze. Takto odesílaná data nejsou nijak omezena, tělo zprávy je možné popsat v hlavičkách.

PUT Představuje požadavek na uložení dat na serveru pod specifikované URL. Takto uložená data jsou poté dostupná pomocí ostatních požadavků. Metoda předpokládá, že uložení dat do souboru provádí přímo server, nikoli CGI skript.

TRACE Slouží k testování omezení nebo testování originálního serveru. Originální server vrací kladnou odpověď bez dat.

2.4 Formát odpovědi

Odpověď HTTP protokolu je zahájena stavovým řádkem, který udává verzi HTTP protokolu. Dále následuje číselný návratový kód a posledním parametrem je textová reprezentace návratového kódu. Položky stavového řádku jsou odděleny mezerami. Stavový řádek je ukončen znaky `CRLF`. Za stavovým řádkem následují jednotlivé hlavičky protokolu ve stejném tvaru jako při dotazu. Po hlavičkách následuje prázdný řádek, za kterým jsou již samotná data. Příklad HTTP odpovědi je uveden níže:

```
HTTP/1.0 200 OK
Server: Netscape-Enterprise/2.0a
Date: Thu, 06 Mar 1997 16:23:43 GMT
Accept-ranges: bytes
Content-length: 1032
Content-type: text/html
```

2.4.1 Návratové kódy

Návratové kódy jsou rozděleny do pěti skupin podle své první číslice a významu.

1xx - informační

Udávají, že požadavek byl obdržen.

- **100 Continue** Meziodpověď od serveru, klient může pokračovat v posílání dotazu. Po obdržení celého požadavku server reaguje odpovídajícím návratovým kódem.

- **101 Switching Protocols** Server porozuměl požadavku a podle specifikace v hlavičce Upgrade mění protokol.

2xx - úspěch

Značí, že server dotaz akceptuje.

- **200 OK** Dotaz byl obsloužen bez chyb a server vrací odpověď.
- **201 Created** Výsledek zpracování dotazu je nový objekt, který lze identifikovat pomocí URL. URL vytvořeného objektu je posláno v těle odpovědi.
- **202 Accepted** Dotaz byl přijat, ale ještě není dokončen. Klient nemusí čekat na dokončení.
- **203 Non-Autorirative Information** Vrácené metainformace nejsou poslané z originálního serveru.
- **204 No Content** Dotaz byl akceptován a obsloužen, ale nevznikla na serveru žádná data pro odesílání klientovi.
- **206 Reset Content**

3xx - přesměrování

- **300 Multiple Choices** Požadovaný dokument je dostupný na několika místech. Klient musí vybrat jeden z těchto dokumentů a znovu vyslat dotaz na vybraný dokument.
- **301 Moved Pernamently** Objekt byl trvale přestěhován na nové URL. Server jej udává v hlavičce Location. Klient musí opakovat dotaz na novém URL.
- **302 Moved Temporarily** Objekt byl dočasně přesunut jinam. Klient se musí dotazovat na jiném místě. Neměl by však přepisovat URL objektu ve svých záložkách, protože se jedná pouze o dočasný přesun.
- **303 See Other** Odpověď na dotaz klienta je dostupná na jiném URL. Nejedná se o náhradu za původní URL.
- **304 Not Modified** Dokument zůstal od posledního stažení beze změny
- **305 Use Proxy** Požadavek musí být opakován prostřednictvím proxy serveru uvedeném v URL.

4xx - klientova chyba

- **400 Bad Request** Chybný dotaz.
- **401 Unauthorized** Požadavek musí obsahovat určité identifikační požadavky, které klient nesplnil.
- **402 Payment Required** Pro budoucí použití.
- **403 Forbidden** Server nemá právo na odeslání požadovaného dokumentu.
- **404 Not Found** Objekt s požadovaným URL neexistuje.
- **405 Method Not Allowed** Požadovaná metoda není na serveru implementována.

5xx - chyba serveru

- **500 Internal Server Error** Při zpracování dotazu došlo na serveru k blíže neurčené chybě.
- **501 Not Implemented** Server nerozpoznal požadovanou metodu
- **502 Bad Gateway** Tuto chybu posílá zprostředkující server, pokud na dotaz klienta dostal od původního serveru špatnou odpověď.
- **503 Service Unavailable** Server nedokáže na klientův dotaz odpovědět. Např. je přetížen nebo právě probíhá jeho údržba. Tato chyba je pouze dočasná, pokud později klient zopakuje svůj dotaz, může být obsloužen.
- **504 Gateway Timeout** Server pracující s proxy nebo gateway nedostal včas odpověď, aby mohl obsloužit požadavek klienta.
- **505 HTTP Version Not Supported** Verze protokolu uvedená v požadavku není na serveru podporována.

Kapitola 3

HTTPry

Vzhledem k cílům této práce jsem zkoumal princip zachytávání HTTP provozu na síti a existující open source nástroje pro tuto činnost. Zvolil jsem možnost úpravy již existujícího nástroje HTTPry pro jeho efektivní a přitom jednoduchou implementaci většiny potřebných funkcí a možnost snadného rozšíření, které je důležité pro tuto práci. Velkou výhodou nástroje je také jeho rychlost potvrzená v kapitole 4.

V této kapitole se seznámíme s nástrojem pro analýzu HTTP provozu HTTPry, jeho původem a funkčností. Uvedeme zde činnost a strukturu nástroje, popíšeme si jednotlivé funkce využívané pro zachytávání provozu HTTP protokolu. Řekneme si fakta o problematice a řešení rozšíření nástroje o ukládání záznamů do databáze. Dále si popíšeme uvažovaná rozšíření nástroje a jejich následnou implementaci, konkrétně překlad doménových jmen sloužící ke snadnější identifikaci webového serveru, podporu IPv6 protokolu a možnost sledování sítí VLAN.

3.1 Základní informace

HTTPry slouží pro zachycení a zobrazení údajů o HTTP provozu. Neslouží přímo pro analýzu provozu, nýbrž pro sledování provozu pro pozdější analýzu. Může být spuštěn v reálném čase a zobrazovat aktuální provoz, nebo být spuštěn jako daemon a záznamy ukládat do výstupního souboru. Může být však upraven i pro použití v jiných aplikacích. Nezachytává přenášená HTTP data, ale zpracovává a zobrazuje vybraná hlavičková pole v dotazu a odpovědi. Nástroj je licencován GNU GENERAL PUBLIC LICENSE, zkráceně GPL, která umožňuje volné použití, šíření a úpravy nástroje. Bližší informace o nástroji jsou k dispozici na [2].

3.2 Struktura a činnost nástroje

3.2.1 Knihovna libpcap

Nástroj využívá knihovny `Pcap`, nezkráceně Packet Capture Library. Knihovna definuje API, pomocí kterého lze přistoupit k rutinám sloužícím pro čtení dat na síťovém rozhraní. V promiskuitním režimu umožňuje přistupovat ke všem datům na síti, tedy i k datům určeným pro jinou stanici. Knihovna umí zapsat na síťové rozhraní binární data s linkovou hlavičkou, neobsahuje ale rutiny na vytvoření TCP či IP hlavičky, proto se používá většinou pouze pro čtení. Také neposkytuje nástroje pro analýzu paketů, kterou je třeba zajistit v aplikaci využívající této knihovny. Knihovna je dostatečně popsána v [20]

3.2.2 Příprava pro logování

Aplikace při spuštění nejprve nalezne a připraví síťové zařízení pro zachytávání provozu. Nástroj je schopný detekovat a automaticky použít aktivní síťové zařízení. Využívá k tomu knihovní funkci `pcap_lookupdev`, která vrátí název aktivního zařízení. Použitím příslušného parametru můžeme nástroji název síťového zařízení manuálně nastavit.

Pro vlastní zachytávání komunikace slouží funkce `pcap_open_live`. Tato funkce vyžaduje jako parametry popis zařízení získaný pomocí `pcap_lookupdev`. Po nastavení zařízení je zavolána funkce `set_header_offset`, která podle použité technologie na linkové vrstvě ISO/OSI modelu určí posun začátku datagramu vyšší IP vrstvy vůči počátku přijatého rámce.

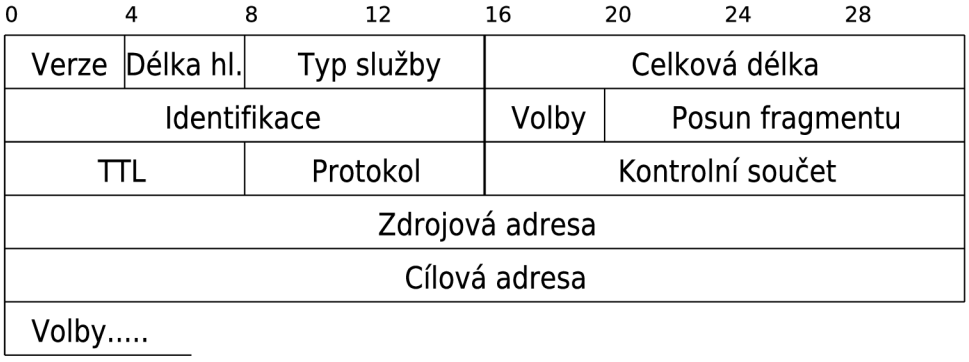
Poté dojde ke kompilaci a nastavení filtru zaznamenávané komunikace. Kompilaci filtru vykonává funkce `pcap_compile`, nastavení filtru funkce `pcap_setfilter`. Pro sledování HTTP provozu je filtr nastaven na zachycení provozu využívajícího TCP protokol na portech 80 a 8080.

Po dokončení této činnosti je síťové zařízení připraveno zachytávat HTTP provoz.

3.2.3 Zachycení komunikace

Samotné zachytávání paketů zajišťuje funkce `pcap_loop`. Parametrem funkce je ukazatel na callback funkci, která bude zavolána v případě zachycení paketu. V tomto případě se jedná o funkci `parse_http_packet`, ukazatel na zachycená data a ukazatel na strukturu `pcap_pkthdr` charakterizující hlavičku paketu v dump file. Jelikož jsou data zapouzdřena dalšími vrstvami TCP/IP modelu, je třeba tyto vrstvy přeskočit, případně z nich získat požadovaná data.

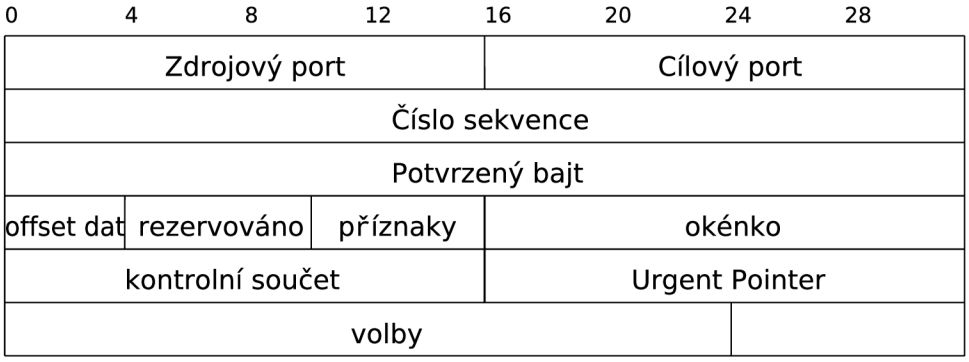
Pro uchování hodnot v IP datagramu slouží struktura `ip_header`, která je definována v hlavičkovém souboru `tcp.h`. V HTTPry je proměnná nesoucí tuto strukturu definována jako ukazatel, který je nastaven na adresu počátku zachyceného IP datagramu, a to pomocí předaného ukazatele na zachycená data a posunutí hlavičky IP datagramu vůči nižší, linkové vrstvě. Z IP datagramu je přečtena zdrojová a cílová adresa. Také je přečtena velikost hlavičky IP datagramu, pomocí které je určen začátek TCP paketu. Velikost hlavičky IP protokolu verze 4 je uvedena v položce `Délka hlavičky`. Hlavička je zobrazena na obrázku 3.1.



Obrázek 3.1: Hlavička IP protokolu verze 4

Pomocí této velikosti je určen posuv TCP hlavičky oproti IP hlavičce. Údaje v TCP

hlavičce uchovává struktura `tcp_header` definovaná v souboru `tcp.h`. Z TCP protokolu je možno zachytit zdrojový a cílový port komunikace. Pomocí položky `Délka dat` je určena pozice samotných dat, v tomto případě HTTP protokolu. Hlavička je zobrazena na obrázku 3.2.



Obrázek 3.2: Hlavička TCP protokolu

Záznamy se dočasně ukládají do hashovací tabulky. Ta je vytvořena při spuštění nástroje. Klíčem tabulky jsou názvy sledovaných položek. Díky tomuto principu může nástroj zaznamenávat libovolnou HTTP hlavičku, kdy klíče tvoří právě názvy HTTP hlaviček. Kromě tohoto zaznamenává zdrojovou a cílovou IP adresu pod položkami `source_ip` a `dest_ip`, porty pod položkami `source_port` a `dest_port`, doménový název cíle pod položkou `host`, verzi protokolu a další. Přehled všech položek je uveden v manuálu nástroje. Informace použité v této kapitole jsou obsaženy v publikacích [11] a [12].

3.3 Úpravy nástroje

V rámci práce je třeba rozšířit HTTPry o ukládání zachycených informací do databáze. Pro snadnější tvoření SQL příkazů byl implementační jazyk změněn z C na C++, které umožňuje pracovat s řetězci `string`. Díky podobnosti obou jazyků není třeba dělat rozsáhlé úpravy nástroje. Dalším krokem je získat informace o dostupných knihovnách pro komunikaci s databázovým serverem MySQL. Vhodným řešením je vytvořit vlastní knihovnu, která využívá knihovnu pro přístup k databázi. Důležitou částí je navrhnout vhodné datové typy pro jednotlivé položky záznamu. Rozbor tohoto problému je uveden v kapitole 3.3.2. Také je vhodné vytvořit uživatelský účet pro nástroj s povolením provádět pouze příkazy `INSERT`. I při zachycení jména a hesla tohoto účtu nebude možné záznamy v databázi měnit či mazat.

Dále bylo provedeno několik volitelných rozšíření. Prvním rozšířením je podpora IPv6 protokolu, který má v budoucnu nahradit starší verzi IP protokolu, a to verzi 4. Dalším rozšířením je podpora sledování HTTP provozu v sítích VLAN. Pro identifikaci webových serverů pomocí doménového jména slouží hlavička `Host`. U starší verze HTTP protokolu však tato hlavička není, proto je vytvořena funkce pro překlad IP adresy na doménové jméno právě pro tyto případy.

3.3.1 Směr komunikace

Pro odlehčení analýzy a ukládání dat jsou sledovány pouze dotazy klientů. Pro účel sledování aktivity uživatelů na síti je dostačující sledovat hodnoty obsažené v dotazech klientů. Dotazy na nový dokument jsou odeslány i v případě, že server při dotazu na původní dokument odpověděl návratovou hodnotou 3xx, což znamená přesměrování. Odpovědi 4xx a 5xx není třeba sledovat, v tomto případě jde o sledování uživatelů a návratové hodnoty v tomto případě nehrají roli. Logováním odpovědí serverů bychom mohli sledovat i další statistiky jako např. používané webové servery, verze HTTP protokolu a hlavičky posíláné v odpovědích. Další možností je zaznamenávat pouze dotazy serverů. V tomto případě bychom mohli sledovat různé statistiky webových serverů.

Pro určení směru komunikace došlo k úpravě filtru pro zachytávání vhodných paketů na tvar:

```
tcp dst port 80 or 8080
```

Ve filtru je uveden pouze cílový port, protože číslo zdrojového portu si klient generuje náhodně. Server naopak pro komunikaci používá stejný port.

3.3.2 Ukládání záznamů do databáze

Pro práci s databází jsem vytvořil knihovnu `db_mysql.h`, která umožňuje pomocí tzv. abstraktní databázové vrstvy (dále jen ADL) použít libovolnou databázi. Pomocí ADL dosáhneme toho, že veškeré změny v komunikaci s databázovým serverem mohou být prováděny pouze v této knihovně, nemusíme tedy upravovat všechny výskyty funkcí pro komunikaci s databází. Podle zadání byly využity v knihovně `db_mysql.h` funkce rozhraní `MySQL.h` pro práci s databází `MySQL`. Implementovány byly pouze funkce využívané v rámci úpravy nástroje.

Knihovna MySQL.h

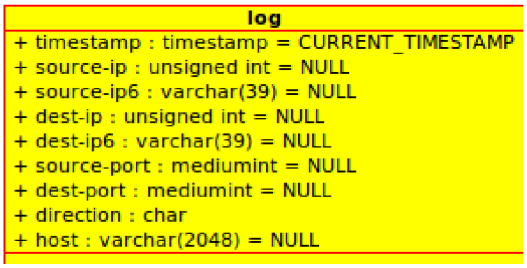
C API kód je distribuován společně s MySQL databází. Je zahrnut do knihovny mysqlclient a umožnuje aplikacím napsaným v jazyce C přistupovat k MySQL databázovému serveru. Pro použití této knihovny je třeba program překládat s připojeným adresářem `-L/usr/lib/mysql` a parametrem `-lmysqlclient`. Dokumentace této knihovny je uvedena v dokumentaci MySQL databáze [19].

Návrh databáze

Pro ukládání vybraných položek HTTP provozu byla vytvořena jednoduchá databáze. Tabulka pro uchování záznamů o provozu obsahuje sloupce pro uložení časového údaje doby zachycení záznamu, ip adresu klienta a serveru a doménové jméno serveru. Časový údaj je uložen v datovém typu timestamp, který je vhodný pro použití s HTTPry, který časový údaj zobrazuje právě v textové reprezentaci timestamp hodnoty. Uložení IP adresy se liší podle verze použitého IP protokolu. Pro verzi 4 jsou adresy zdroje a cíle uloženy jako 32 bitový `unsigned integer`. Pro převod z textové reprezentace IPv4 adresy na číslo je využita MySQL funkce `INET_ATON()` [1], opačný převod zajišťuje funkce `INET_NTOA()`[2]. Číselná

[1] http://dev.mysql.com/doc/refman/5.1/en/miscellaneous-functions.html#function_inet-aton
[2] http://dev.mysql.com/doc/refman/5.1/en/miscellaneous-functions.html#function_inet-ntoa

reprezentace je zvolena z důvodu vyhledávání podsítě, pokud uživatel zadá např. IP adresu a masku sítě. Vyhledávat se poté bude interval mezi dvěma čísly reprezentujícími rozsah IP adres, což je rychlejší. Výhodou je také menší paměťová náročnost než při uložení řetězce. Stejně tak je uložen i doménový název webového serveru. Z tohoto údaje se bude získávat jak název serveru, tak i název domény. Přehlednější popis struktuty databáze poskytuje obrázek 3.3.



Obrázek 3.3: ER diagram pro záznamy

Pro podporu transakcí bylo zvoleno úložiště dat `InnoDB`. Transakce jsou třeba např. když jsou vkládány nové záznamy do databáze a zároveň generovány statistiky webovým rozhraním. Nevyužití transakcí, např. při použití úložiště `MylSAM`, může způsobit nepřesnosti ve statistikách. `InnoDB` také není omezeno maximální velikosti souboru operačního systému, může se, oproti `MylSAM`, skládat z více souborů. V databázi nejsou využity indexy, které by zajistily rychlejší vyhledávání, ale na úkor zpomalení při vkládání. Bližší informace, pomocí kterých byla navržena databáze, jsou uvedeny v publikaci [6].

Z této databáze bude webové rozhraní čerpat data pro analýzu a zobrazení.

3.3.3 Podpora protokolu IPv6

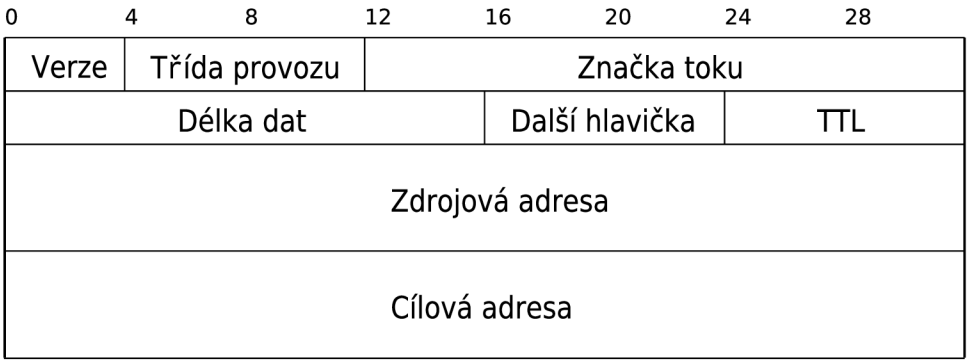
Rozšíření o možnost pracovat s protokolem IPv6 reaguje na současnou situaci se zmenšujícím se počtem IPv4 adres na světě, tudíž s blížícím se nasazením protokolu IPv6 do praxe.

Stručný popis IPv6 protokolu

Formát IPv6 datagramu je definován v [4]. Ostatní mechanismy a datové formáty, související s IPv6, jsou definovány v dalších RFC specifikacích. Hlavním rozdílem oproti protokolu verze 4 je zvýšení velikosti zdrojové a cílové adresy na 128 bitů a také změna koncepce hlaviček protokolu. Verze 4 má proměnlivou délku hlavičky a jednotliví účastníci komunikace si mohli připojovat další nepovinné volby podle potřeby. Verze 6 standartní hlavičku minimalizuje a omezuje její prvky pouze na nejnutnější minimum. Povinná hlavička má konstantní velikost 40B, je uvedena na obrázku 3.4.

Doplňující údaje byly přesunuty do nepovinných rozšiřujících hlaviček. Rozšiřující hlavičky mají přesně definované pořadí, v jakém následují po povinné hlavičce. Všechny rozšiřující hlavičky mají buď konstantní velikost, nebo mají svou velikost uvedenu v sobě. Zde jsou uvedeny nejvýznamnější rozšiřující hlavičky:

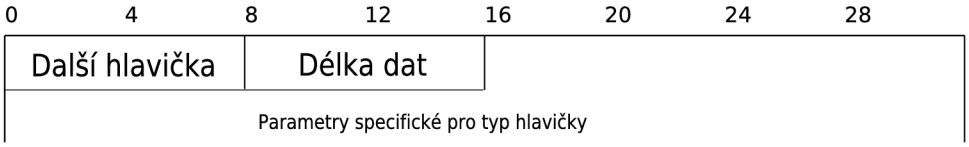
1. volby pro všechny (hop-by-hop options)
2. volby pro cíl (destination options)



Obrázek 3.4: Povinná hlavička IPv6 protokolu

3. směrování (routing)
4. fragmentace (fragment)
5. autentizace (authentication)
6. mobilita (mobility)

Hlavička fragmentace má konstantní velikost, ostatní jmenované mají proměnnou velikost udanou v položce `Délka hlavičky`. Na obrázku 3.5 je ukázka prvních dvou položek rozšiřujících hlaviček s proměnnou délkou.



Obrázek 3.5: Rozšiřující hlavička proměnné délky

Jako vhodný zdroj informací pro toto rozšíření posloužila publikace [16].

Implementace podpory IPv6 protokolu

Pro zachycení komunikace, která využívá na síťové vrstvě protokol IPv6, je třeba využít knihovny `netinet/ip6.h`[17]. Z této knihovny dojde k použití struktur `ip6_hdr` sloužící k uchování povinné hlavičky, a `ip6_ext`, která slouží k uložení obecné rozšiřující hlavičky s proměnnou délkou.

Verze protokolu je identifikována v položce `Verze`. Pokud je nastavena na 6, jedná se o IPv6 protokol. Z povinné hlavičky je uchována zdrojová a cílová IP adresa. Pro tyto položky je třeba vytvořit v tabulce sloupce, v tomto případě jsou datového typu `varchar(39)`. Poté mohou následovat volitelné hlavičky, které je třeba projít a zjistit posunutí začátku HTTP protokolu od začátku IPv6 protokolu. Zachytáváme pouze nejvýznamnější rozšiřovací hlavičky uvedené v kapitole 3.3.3.

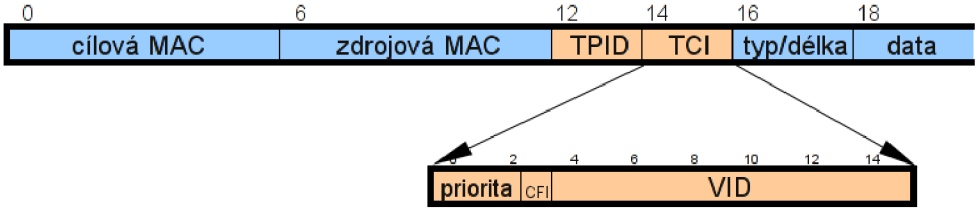
3.3.4 Podpora technologie VLAN

Technologie VLAN slouží k provozování více virtuálních sítí na jedné fyzické síti. Tato kapitola obsahuje stručný popis, kompletní popis VLAN technologie je uveden v publikaci [10].

Stručný popis 802.1Q

Standart 802.1Q obsahuje zásady tvorby VLAN, formát přídavných hlaviček ethernetového rámce a směr dalšího vývoje. Význam spočívá právě ve standartizaci přídavných hlaviček. Definuje vytváření VLAN sítí na první a druhé vrstvě ISO/OSI modelu.

Přepínače musí při směrování rámců znát příslušnost každého rámce k jednotlivým VLAN sítím. Podle toho poté rozhodnou, na které výstupní porty budou rámce posílány. Informace o příslušnosti jednotlivých rámců k jednotlivým VLAN sítím jsou přidávány do rámců pomocí 4 bajtových značek. Tyto značky jsou vloženy za zdrojovou MAC adresu.



Obrázek 3.6: Ethernetový rámec s VLAN značkou

TPID Definovaná hodnota 8100 hex značí, že rámec nese informace IEEE 802.1Q

Priority Obsahuje prioritu rámce

CFI MAC adresy v kanonickém formátu nebo RIF pole

VID Jednoznačná identifikace VLAN, do které rámec přísluší

Implementace podpory VLAN

Pro zavedení podpory VLAN sítí je třeba detekovat, zda-li ethernetový rámec neobsahuje VLAN značkování. Pro snadnou detekci VLAN značky slouží knihovna `vlan.h`. Dojde tedy k otestování hodnoty `tpid` v analyzovaném rámci. Pokud obsahuje hodnotu $0x8100$, jedná se o označení paketu pro VLAN síť. K posunutí IP hlavičky je třeba přičíst velikost VLAN značky, tedy 4 bajty. Je třeba neopomenout pořadí bajtů na síti (big endian) a architektuře procesoru (např. Intel little endian). Tento rozdíl odstraňuje funkce `ntohs()`.

Pro zachycení VLAN komunikace je třeba upravit filtr nástroje HTTPry. Ten má po této úpravě tvar:

```
(tcp dst port 80 or 8080) or (vlan and (tcp dst port 80 or 8080))
```

Tento filtr zajišťuje zachycení paketů jak při použití VLAN značení tak i bez tohoto značení. Je však použit pouze pokud je na linkové vrstvě detekována technologie Ethernet.

3.3.5 Překlad IP adres na doménová jména

Při verzi protokolu 1.1 je doménové jméno webového serveru uvedeno v dotazu na HTTP server pod hlavičkou `Host`. Jelikož tato hlavička může obsahovat i port, je port oříznut a dále neukládán. Tato hlavička je pro protokol 1.1 povinná, proto s ní lze v dnešní době většinou počítat. U verze 1.0 je situace obtížnější. Protokol informaci o názvu serveru neobsahuje, proto je třeba získat doménové jméno užitím funkce `gethostbyaddr()`, pomocí které získáme z IP adresy doménové jméno. IP adresa serveru je získána z IP protokolu.

Dnes většina webových serverů podporuje HTTP protokol verze 1.1, takže tento algoritmus bude pro většinu záznamů nevyužit. Použití tohoto algoritmu způsobuje určité zpomalení závislé na rychlosti překladu. Další nevýhodou tohoto algoritmu je překlad IP adresy na doménové jméno počítače, který nemusí vždy odpovídat jménu webového serveru.

Kapitola 4

Testování HTTPry

Důležitou částí implementace nástroje pro logování je testování. Testování je zaměřeno především na rychlost a spolehlivost modifikovaného HTTPry v porovnání s původní, neupravenou verzí nástroje. Také dochází k testování dalších provedených rozšíření.

Testy probíhaly na počítači s těmito parametry: procesor Intel Core 2 Duo T5270 1.40GHz, 2048MB RAM pamět, operační systém Kubuntu 9.10 s jádrem 2.6.31-19.

4.1 Porovnání s původní verzí

Testování se zaměřuje na rychlost zpracování určitého vzorku zachycených dat. Pro tento případ došlo nejprve k zachycení vzorku dat pomocí nástroje wireshark. Vzorek dat obsahující 100000 HTTP paketů je uložen v souboru `data2.cap`. Vzorek je předložen původní a nové verzi ke zpracování. V tabulce 4.1 jsou uvedeny průměrné hodnoty měření, pro každou verzi nástroje bylo provedeno celkem 5 měření pro minimalizaci chyby měření. Pro automatizaci testu byl napsán skript `test_speed.sh`. Skript porovná zachycené soubory mezi sebou, aby se odhalily případné ztráty záznamů, ke kterým by ale v tomto případě nemělo docházet. Počet přidaných záznamů do databáze je třeba zkontrolovat manuálně.

	původní verze soubor	verze s úpravami databáze	verze s úpravami soubor
real	2.29s	23.26s	1.14s
user	1.22s	1.30s	0.65s
sys	0.98s	0.74s	0.49s

Tabulka 4.1: Porovnání původní a upravené verze HTTPry

Pro velké zpomalení při ukládání do databáze bylo testováno u upravené verze i ukládání do souboru bez ukládání záznamů do databáze, abychom odhalili, do jaké míry zpomaluje nástroj právě ukládání do databáze a jak další provedená rozšíření. Doba zpracování u upravené verze závisí také na počtu HTTP dotazů ve vzorku dat. Počet zachycených záznamů původní verze je v tomto případě vyšší než počet záznamů upravené, rozbor obou faktů je proveden v kapitole 3.3.1.

Problém zpomalení při ukládání databáze je dán vlastností úložiště dat InnoDB, které v defaultním nastavení provádí autocommit po každém provedení příkazu `INSERT`. Řešením tohoto problému je využití rozšířeného příkazu `INSERT`, kdy do jednoho příkazu `INSERT` lze vložit více řádků pro vložení.

Pro toto rozšíření byl vytvořen buffer, který dočasně uchovává zachycené položky ve tvaru SQL dotazu. Při naplnění bufferu dojde k odeslání rozšířeného příkazu `INSERT` na databázový server. Dojde tak k uložení několika záznamů během jednoho dotazu a hlavně během jedné transakce. Po odeslání je buffer vyprázdněn a situace se opakuje.

Lze také využít spuštění transakce příkazem `START TRANSACTION`, uložit zvolený počet záznamů do databáze a transakci potvrdit příkazem `COMMIT`.

U obou přístupů je otestována jejich rychlost při ukládání do databáze a lepší způsob je poté v nástroji ponechán. Výsledky testů jsou uvedeny v tabulce 4.2. Podobně jako v předchozím testu jsou i zde uvedeny průměrné hodnoty měření.

	Multiinsert	Manuální transakce
real	2.79s	36.62s
user	0.95s	1.27s
sys	0.15s	0.75s

Tabulka 4.2: Porovnání obou přístupů

Další možností vylepšení rychlosti by mohlo být použití jiné databáze. Byla vytvořena knihovna `db_postgre.h` pro práci s PostgreSQL databázovým serverem. Knihovna využívá rozhraní `postgresql/libpq-fe.h` dostupné s PostgreSQL databází. Překládat je nutné s parametrem `-lpq` a adresářem s knihovnami `-L/usr/lib/postgre`. Výsledky testování rychlosti ukládání do této databáze jsou uvedeny v tabulce 4.3. Informace potřebné k použití PostgreSQL databáze jsou k dispozici v dokumentaci[21].

	Autocommit	Multiinsert	Manuální transakce
real	46.20s	16.59s	28.25s
user	4.93s	1.05s	3.52s
sys	1.80s	0.19s	1.36s

Tabulka 4.3: PostgreSQL databáze

Pomocí výsledků těchto testů byla ponechána implementace využívající rozšířený příkaz `INSERT`. Vhodnější databází pro ukládání záznamů je podle těchto měření databáze MySQL.

4.2 Testování rychlosti

Testování při vyšších rychlostech sleduje, zda-li nástroj dokáže uložit všechny zachycené pakety, případně k jaké dochází ztrátovosti. Využívá nástrojů wireshark pro zachycení HTTP komunikace do souboru a následně pomocí `tcpreplay` je zachycená komunikace opakována. Pomocí `tcpreplay` lze nastavit rychlost opakování komunikace, a tudíž i určit, při jaké rychlosti začne vznikat ztrátovost nástroje. Skript nejprve spustí nástroj HTTPry pro zachycení dat a poté nástroj `tcpreplay` pro opakování zachycené komunikace v souboru `data.cap`. Tento soubor obsahuje 20000 HTTP paketů. Pomocí `tcpreplay` je komunikace pětkrát opakována. Počet odeslaných paketů je tedy 100000. Závěrem je porovnán počet odeslaných paketů nástrojem `tcpreplay`, počet parsovaných a zpracovaných paketů a počet záznamů v databázi. Průměrné výsledky měření uvádí tabulka 4.4, výsledky jednotlivých měření jsou uvedeny na CD. Pro porovnání je provedeno testování neupravené verze ná-

stroje. Výsledky jsou v tabulce 4.5. Testy jsou provedeny pro implementaci popsanou v 4.1. Je třeba zohlednit fakt popsaný v kapitole 3.3.1.

Nastaveno[Mb/s]	Průměrná rychlost[Mb/s]	Přijaté	Ztracené	Ztrátovost[%]
10	6.94	50000	0	0
25	11.98	50000	0	0
50	17.31	49859	141	0.28
100	39.04	49215	785	1.57
200	46.67	48871	1129	2.26
300	61.1	48517	1483	2.97
400	83.1	47289	2711	5.42

Tabulka 4.4: Ztrátovost upraveného nástroje

Nastaveno[Mb/s]	Průměrná rychlost[Mb/s]	Přijaté	Ztracené	Ztrátovost[%]
10	6.95	100000	0	0
25	11.7	100000	0	0
50	17.3	100000	0	0
100	39.12	100000	0	0
200	46.7	100000	0	0
300	61.01	99189	811	0.81
400	83.1	61191	38809	38.81

Tabulka 4.5: Ztrátovost původního nástroje

Menší ztrátovost při vyšších rychlostech je dána menším počtem zachycených paketů, diskutovaném v kapitole 3.3.1. Upravená verze nástroje byla během testů méně náročnější na procesor než původní verze. Naopak při vyšších rychlostech procesor zatěžoval MySQL server. Pro poslední řádek tabulky nástroj HTTPry vytěžoval procesor na 17–23%, pro MySQL server bylo zatížení procesoru 30–35%. Původní nástroj vytěžoval procesor na 42–47% při ukládání do souboru.

4.3 Testování dalších rozšíření

4.3.1 Překlad doménových jmen

Pomocí jednoduchého programu `webclient`[1] došlo k dotázání se několika webových serverů na různé webové stránky pomocí protokolu HTTP 1.0 bez hlavičky `Host`. Cílem testu je sledovat schopnost nástroje získat doménový název serveru pomocí jeho IP adresy verze 4 i 6. Dotaz na server vypadal takto:

```
GET / HTTP/1.0\r\n
```

Pro tento test byl vytvořen skript `test_host.sh`. Skript spouští několikrát program webclient a doménová jména zachycená nástrojem HTTPry jsou uložena do souboru. Nevýhodou

[1] 1.projekt kurzu IPK 2008/2009

algoritmu je případ, kdy se doménový název serveru neshoduje s webovou adresou. Např. pro server `www.vutbr.cz` nástroj zachytí `piranha.ro.vutbr.cz`.

4.3.2 Podpora IPv6 protokolu

Pro testování správného ukládání IP adresy verze 6 posloužil nástroj `wget`[2], pomocí kterého jsou s parametrem `-6` generovány požadavky na webové stránky. Podmínkou je, aby klient i server podporoval technologii IPv6. Pro testování byl využit školní webový server[3]. Data jsou zachycena pomocí nástroje wireshark a uložena do souboru `data_ipv6.cap` společně s daty obsahujícími IPv4 protokol pro testování kombinace těchto protokolů. Skript `test_ipv6` provede spuštění HTTPry se zadaným parametrem pro čtení z `CAP` souboru, výstup nástroje je zobrazen na standartní výstup pro snadnou kontrolu zachycených dat. Tento test proběhl úspěšně, zachycené IPv6 adresy odpovídají skutečnosti.

4.3.3 Podpora VLAN

Pro testování zachycení VLAN technologie byl opět využit program wireshark, pomocí kterého došlo k zachycení vzorku HTTP dat, které jsou uloženy v souboru `data_vlan.cap`. Tento vzorek dat je předložen nástroji HTTPry a jsou z něj zachyceny záznamy o HTTP provozu. Pro lepší porovnání je výstup nástroje uložen do souboru `test_vlan1.txt`. Poté je využito nástroje `tcprewrite`, pomocí kterého je do předchozích záznamů vložena zadaná VLAN značka a výsledná data jsou uložena do souboru `data_vlan_2.cap`, který je opět skenován nástrojem HTTPry. Výstup je uložen do souboru `test_vlan2.txt`. Pomocí nástroje diff jsou oba soubory porovnány. Nástroj diff neodhalil žádný rozdíl v souborech, tento test byl tedy rovněž úspěšný.

[2] http://www.gnu.org/software/wget/
[3] www6.fit.vutbr.cz

Kapitola 5

Návrh řešení webového rozhraní

Pro lepší a přehlednější zobrazení záznamů získaných nástrojem HTTPry bylo třeba vytvořit grafické uživatelské rozhraní. Pro svoji dostupnost a multiplatformnost bylo zvoleno rozhraní webové. Tato kapitola lehce seznamuje s použitými technologiemi, knihovnami a hlavně s popisem návrhu informačního systému.

5.1 Použité technologie

Použité technologie se odvíjejí od možností webových hostingů a od zadání této práce.

5.1.1 Server

Na serveru je použita kombinace jazyka PHP a databáze MySQL. PHP je skriptovací jazyk sloužící k tvorbě dynamického webu. Výhodou PHP je jeho jednoduchost, multiplatformnost, spousta již hotových knihoven a aplikací snadná komunikace s databází MySQL a dalšími. Implementovaný informační systém využívá PHP pro generování webových stránek, grafů, tabulek a pro zpracování záznamů v databázi. K tvorbě PHP kódu pomohla publikace [9].

MySQL databáze se vyznačuje multiplatformností, slušnou rychlostí a dostupností. Jedná se o relační databázi. Informace pro tuto práci byly čerpány z on-line dokumentace[19] a také z publikace [6].

5.1.2 Klient

Klientská část aplikace je vytvořena pomocí značkovacího jazyka pro webové dokumenty HTML, kaskádových stylů CSS a skriptovacího jazyka JavaScript. Pomocí kaskádových stylů je vytvořen vzhled informačního systému. Výhodou jazyka JavaScript je jeho integrovanost do webových prohlížečů, je tedy možné interpretovat jeho zdrojový kód na straně klienta. Lze tak naprogramovat různé efekty, dynamicky měnit obsah stránek atd. Velkou nevýhodou je vypnutí JavaScriptu ve webovém prohlížeči, je tedy nutné, pokud je to možné, počítat i s touto variantou. K tvorbě vzhledu aplikace posloužily publikace [18] a [14], k programování JavaScriptu posloužila publikace [25]

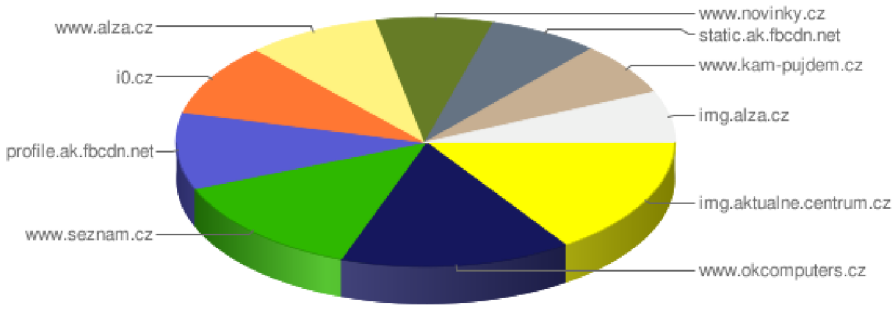
5.1.3 AJAX

AJAX je technologie umožňující měnit obsah webových stránek bez jejich znovunačítání. Je tedy možné komunikovat s webovým serverem na pozadí, přijatá data poté zpracovat a

prezentovat pomocí JavaScriptu. Sníží se tak zátěž na webové servery a zůstanou nastaveny globální proměnné JavaScriptu. Technologie AJAX, její výhody a nevýhody jsou dobře popsány např. v publikaci [3].

5.1.4 Nástroj pro tvorbu grafů

Pro tvoření grafů je využito již existující API rozhraní Google Chart Tools. Graf se generuje pomocí speciálně upravené webové adresy. Lze jej na stránce zobrazit pomocí HTML tagu pro obrázky `<img>`, kde v atributu `src` je uvedená právě adresa pro vygenerování grafu. Podrobný seznam argumentů pro tvorbu grafu je uveden na domovské stránce Google Chart API [23]. Ukázku grafu lze vidět na obrázku 5.1. Při použití PHP je potřebné

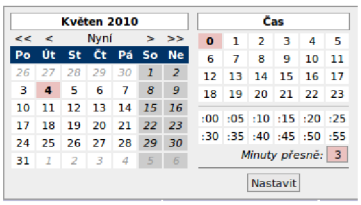


Obrázek 5.1: Příklad grafu

generovat adresu grafu dynamicky v závislosti na požadovaných datech, které chceme zobrazit. K usnadnění generování adresy slouží knihovna GChart. Ta umožňuje nastavit různé parametry grafů a vytvořit webovou adresu pro zobrazení grafu. Možnosti a návod knihovny GChart jsou uvedeny na její domovské stránce [22].

5.1.5 Kalendář

Komponenta kalendář umožňuje uživatelsky přívětivé zadávání data a času pro jednotlivá pole formulářů. Ke své činnosti využívá framework Prototype[1] pro JavaScript. Umožňuje nastavit různé jazyky, případně doimplementovat vlastní jazyk jako v případě češtiny. Více informací o komponentě je uvedeno na stránce autora [13].

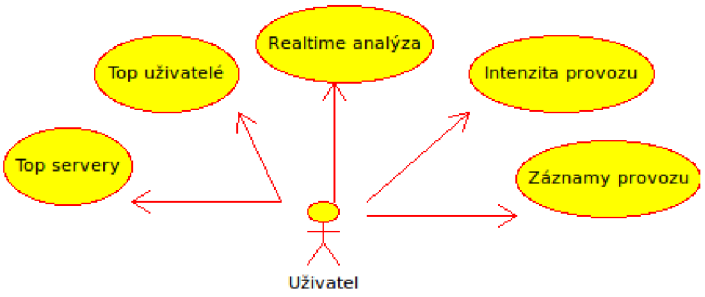


Obrázek 5.2: Příklad kalendáře

[1] www.prototypejs.org

5.2 Návrh řešení

Návrh řešení je důležitou částí vytváření informačního systému. Diagram případů užití 5.3 zobrazuje uvažované možnosti užití systému. Podrobnější popis návrhu je uveden v následujících podkapitolách.



Obrázek 5.3: Diagram případů užití

5.2.1 Realtime analýza

Analýza v reálném čase bude zobrazovat průběžně přibývající záznamy v databázi, konkrétně IP adresu klienta a serveru, časové razítko zachycení a doménové jméno serveru. Vše zobrazené v přehledné tabulce. K dispozici bude možnost detekci nových záznamů zastavit a opět spustit pomocí tlačítek. K detekci bude docházet vždy za určitý časový interval. Při prvním i opětovném spuštění sledování dojde k zobrazení několika posledních záznamů (počet závisí na implementaci). Interval mezi aktualizacemi je třeba volit vhodně tak, aby nedocházelo k přidávání příliš mnoho dat najednou. Z tohoto důvodu bude zavedeno omezení maximalního počtu zaznamů v jedné aktualizaci.

5.2.2 Statistika TOP servery

Statistika TOP servery zobrazuje nastavitelný počet nejaktivnějších serverů ve vybraném dni. Parametry zobrazení lze nastavit pomocí formuláře. Zadání data je usnadněno komponentou kalendář. Počet sledovaných serverů/domén lze nastavit pomocí výběrového pole v rozsahu 10 až 100, a to s krokem po 10ti záznamech.

Výsledky statistiky jsou zobrazeny v koláčovém grafu a tabulce. Jeden graf a tabulka pro webové servery, jeden pro domény 2.řádu. Tabulka obsahuje možnost vyhledat detaily pro každý webový server nebo doménu. Dále bude tabulka obsahovat možnost pomocí odkazu sledovat informace o webovém serveru nebo doméně [2]. Pomocí odkazu vyhledej je možné přepnout do části záznamy provozu s automaticky předvyplněnými parametry pro zvolený server nebo doménu. Pokud záznamy pro vybraný den nejsou k dispozici, zobrazí se informace o nedostupnosti záznamů.

5.2.3 Statistika TOP uživatelé

Pro tuto statistiku lze nastavit počet sledovaných nejaktivnějších uživatelů a pro každého z nich možnost zvolit počet zobrazených nejnavštěvovanějších serverů a domén. Rozsahy

[2] http://whois.domaintools.com/

jsou opět volitelné pomocí výběrového pole od 10 do 100 sledovaných položek. Pomocí komponenty kalendář lze zadat sledovaný den.

Statistika je zobrazena v koláčovém grafu a tabulce, ve které je kromě počtu záznamů konkrétního uživatele přiložen i seznam jeho nejnavštěvovanějších serverů a domén druhého řádu. Pomocí odkazu vyhledej lze přepnout do záznamů provozu s nastavenými příslušnými parametry pro vyhledávání. K dispozici je možnost pomocí odkazu sledovat informace o daném uživateli, serveru nebo doméně jako v případě statistiky top serverů.

Pro tuto statistiku je třeba pro vyhledávání top serverů přidat možnost zadání IP adresy klienta.

5.2.4 Intenzita HTTP provozu

Intenzita HTTP provozu zaznamenává počet záznamů v databázi za určitý interval, přičemž hodnoty jsou vždy rozděleny do tříd právě podle zvoleného intervalu. K dispozici bude výběr intervalů den, týden a měsíc. Význam vybraného data závisí na zvoleném intervalu. Pro interval den značí zvolené datum statistiky sledované právě pro vybraný den. Záznamy jsou tříděny podle jednotlivých hodin během zvoleného dne. Pro intervaly měsíc a týden udává zvolené datum konečnou hodnotu intervalu pro sledování. Záznamy jsou v tomto případě třízeny podle data. Volitelnou možností je zadání IP adresy klienta nebo rozsah IP adres pro více klientů. Pro zadání parametru je vytvořen formulář, pro volbu intervalu slouží `select`, pro nastavení data komponenta kalendář. K zobrazení statistik je použit sloupcový graf, kde na ose x jsou třídy časového intervalu, osa y pak obsahuje počet záznamů HTTP provozu. Statistiky lze vidět i v tabulce, kde pro každý interval je uveden počet příslušných záznamů.

5.2.5 Záznamy provozu

Záznamy provozu slouží pro export záznamů s vybranými parametry z databáze a jejich následné uložení a distribuce v souboru. K dispozici je volba formátu CSV[3] a XML[4]. Pro zadání parametrů bude vytvořen formulář umožňující zadat interval, ve kterém nás komunikace zajímá, pomocí kalendáře. Ke zpřesnění nastavení intervalu je možné nastavit datum i čas. Dále je možné nastavit IP adresu klienta a serveru, a to ve verzi 4 i verzi 6. Také bude možno zadat rozsah IPv4 adres, např. pomocí masky podsítě. Posledním parametrem je doménový název serveru. Pokud bude tato položka zadána, budou zobrazeny všechny záznamy obsahující zadaný podřetězec v položce `Host`. Formulář bude možno odeslat s využitím technologie AJAX i běžnou cestou pomocí metody POST.

Po odeslání formuláře dojde ke zpracování zadaných parametrů a získání záznamů z databáze. Aby uživatel nemohl zadat parametry tak, aby vznikl příliš velký soubor, je velikost souboru omezena při jeho vytváření. Pokud soubor přesáhne danou maximální velikost, je automaticky smazán a uživateli zobrazena chybová hláška. V tomto případě je třeba zvolit více menších intervalů pro sledování.

Na této stránce jsou zobrazeny již vytvořené záznamy komunikace ke stažení. K dispozici je i datum vytvoření a velikost souboru. Pro úsporu místa je možno tyto soubory se záznamy smazat.

[3] http://tools.ietf.org/html/rfc4180
[4] http://www.w3.org/XML/

5.2.6 Zobrazení chyb při získávání dat

Pro uchování chybových hlášení a jejich následné zobrazení na straně klienta bude vytvořena třída `Errors`. Hlavním významem této třídy je možnost zobrazit více chyb, např. pro záznamy provozu špatně zadaná IP adresa klienta i serveru. Třídy uvedené výše tuto třídu zdědí. Některé třídy a metody budou využívat mechanismu výjimek. Ten bude použit v případě, že může nastat pouze jedno chybové hlášení. Každá třída využívající výjimek bude mít vytvořenou vlastní třídu s názvem `nazev_tridyException`, která bude dědit třídu `Exception`. Při použití technologie AJAX může nastat chyba při zpracování dat na straně klienta. Zde však není třeba využívat speciální třídy. Důležité bude zobrazit všechny chyby ve stejném vzhledu a formátu.

5.2.7 Formát zadání IP adresy

Zadávání IP adresy se liší podle verze IP protokolu. Pro verzi 4 je možné zadat samotnou IP adresu ve formátu `xxx.xxx.xxx.xxx`, kdy dojde k filtrování podle přesně zadané IP adresy. Pro sledování celé podsítě je možno zadat libovolnou adresu z této sítě / maska podsítě. Maska může být ve formátu `xxx.xxx.xxx.xxx` nebo číslo od 1-32, které charakterizuje počet jedniček v binárním vyjádření masky sítě [15]. U verze IP protokolu 6 lze IP adresu zadat v úplném nebo zkráceném formátu[5]. Formát s prefixem není podporován. Pro zadání IP adresy obou verzí bude sloužit třída `IP`, která podle zadaného řetězce identifikuje verzi protokolu pro danou IP adresu a případně i rozsah adres, pokud bude zadána i maska podsítě nebo prefix.

Kapitola 6

Implementace informačního systému

Tato kapitola informuje o implementaci informačního systému pro reprezentaci zachycených záznamů komunikace. Uvádí stručný popis částí systému a detailně popisuje nejvýznamnější části zdrojového kódu jednotlivých funkcí systému. Pro detailní zkoumání funkčnosti slouží komentáře přímo ve zdrojových kódech.

Informační systém je rozdělen na tři implementační vrstvy:

- Vrstva komunikující s databází Třída `DB` sloužící k přístupu k databázi MySQL.
- Vrstva tříd pro zpracování parametrů a generování výsledků
- Vrstva tvořící formuláře a prezentující výsledky

6.1 Realtime analýza

Realtime analýza využívá technologii AJAX, pomocí které zobrazuje nové záznamy. V tabulce na obrázku 6.1 lze vidět sledované údaje. Pro omezení počtu zobrazených záznamů je maximální počet nových záznamů omezen. Nevýhodou této statistiky je sledování v rychlém

Pozastavit

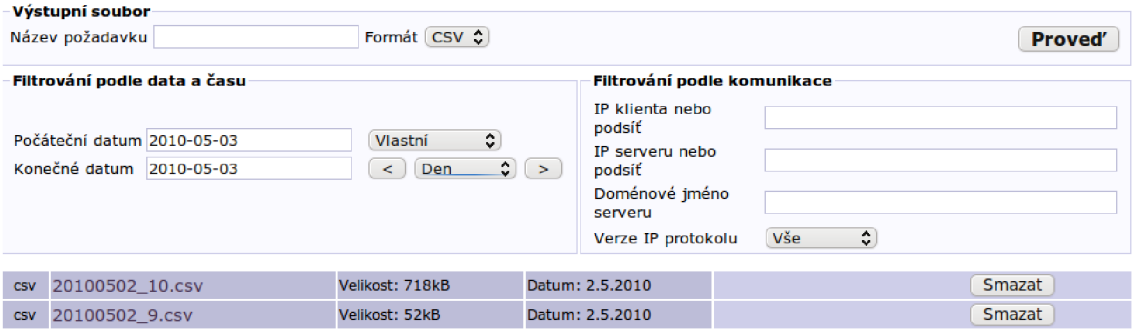
Čas záznamu	Klient	Server	Doménové jméno
2010-04-26 12:46:24	147.229.195.7	209.85.135.138	www.google-analytics.com
2010-04-26 12:46:23	147.229.195.7	209.85.135.138	www.google-analytics.com
2010-04-26 12:46:22	147.229.195.7	91.213.160.12	c1.navrcholu.cz
2010-04-26 12:46:22	147.229.195.7	91.213.160.12	c1.navrcholu.cz
2010-04-26 12:27:54	147.229.195.7	74.125.39.138	www.google-analytics.com
2010-04-26 12:27:52	147.229.195.7	91.213.160.12	c1.navrcholu.cz
2010-04-26 12:27:52	147.229.195.7	91.213.160.12	c1.navrcholu.cz
2010-04-26 12:27:52	147.229.195.7	74.125.39.138	www.google-analytics.com

Obrázek 6.1: Realtime analýza

provozu, které významněji zatěžuje prohlížeč (testováno na prohlížeči Firefox). Při použití bufferu v HTTPry nejsou zobrazeny aktuálně zachycené záznamy, ale ty záznamy, které jsou již uložené na serveru.

6.2 Záznamy provozu

Na obrázku 6.2 lze vidět formulář pro nastavení parametrů vyhledávání jednotlivých záznamů. Pod formulářem je seznam již dříve vygenerovaných souborů se záznamy, který se pomocí technologie AJAX pravidelně aktualizuje pro případ vygenerování souborů jiným klientem. Soubory jsou vhodné pro další zpracování uživatelem a není třeba řešit stránkování záznamů či zobrazení v tabulce.



Obrázek 6.2: Záznamy provozu

6.3 Top servery

Na obrázku 6.3 je zobrazen formulář pro výběr parametrů této statistiky. SQL dotaz zajišťující tuto statistiku je uveden níže:

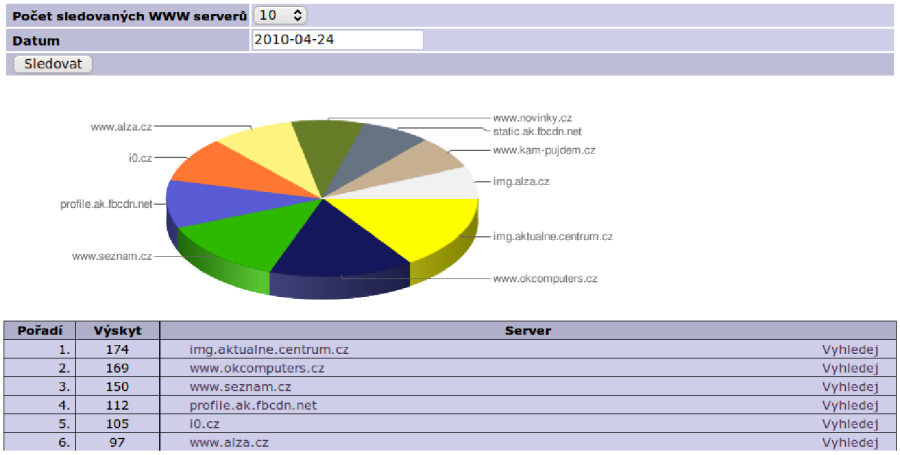
```
SELECT count(*),IF(NOT host = '', host,
IF(`dest-ip` IS NOT NULL,INET_NTOA(`dest-ip`),`dest-ip6`))
as server, IF(NOT host = '', 'name', 'addr') as ident
FROM log dalsi_parametry GROUP BY server ORDER BY count(*)
DESC LIMIT N
```

V případě prázdného sloupce host je ve statistice uvedena jedna z použitých IP adres. Druh identifikace (host nebo addr) slouží pro možnost vyhledání záznamu. N udává počet sledovaných záznamů, `dalsi_parametry` reprezentují podmínky dotazu, např. datum.

Statistika domén je založena na podobném dotazu. Rozdíl nastane v případě identifikace IP adresou, doména je v tomto případě označena jako -. Pro zjištění domény druhého řádu je využito MySQL funkce `SUBSTRING_INDEX()`. Tato funkce je k dispozici i v databázi PostgreSQL.

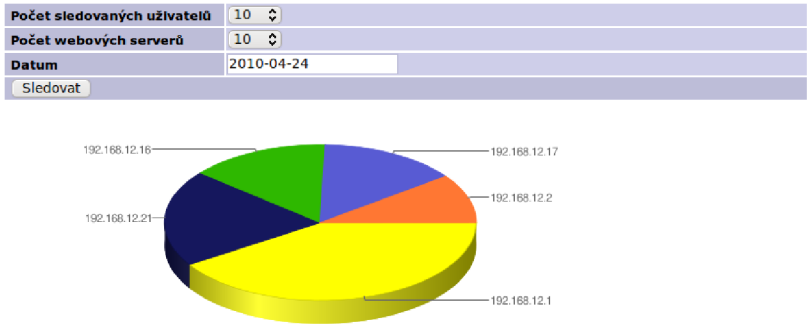
6.4 Top uživatelé

Statistika top uživatelé zobrazuje formulář umožňující nastavit datum, počet sledovaných uživatelů a pro každého z nich počet webových serverů a domén. K tomuto využívá předchozí statistiky pro TOP servery s podmínkou IP adresy příslušného klienta. Opět je zde možnost vyhledat konkrétní záznamy a detailní informace o serveru a doméně. Tuto část lze vidět na obrázku 6.4



Pořadí	Výskyt	Server	
1.	174	img.aktualne.centrum.cz	Vyhledej
2.	169	www.okcomputers.cz	Vyhledej
3.	150	www.seznam.cz	Vyhledej
4.	112	profile.ak.fbcdn.net	Vyhledej
5.	105	i0.cz	Vyhledej
6.	97	www.alza.cz	Vyhledej

Obrázek 6.3: Top servery



Pořadí	Výskyt	Uživatel	Servery		Domény	
			160	img.aktualne.centrum.cz	180	centrum.cz
			96	www.seznam.cz	168	fbcdn.net
			77	www.novinky.cz	114	seznam.cz
			76	i0.cz	95	novinky.cz

Obrázek 6.4: Top uživatelé

6.5 Intenzita provozu

Zajímavostí v této části je použití asociativního pole pro generování grafu. Indexy musí v tomto případě odpovídat popiskám grafu, proto jsou v případě intervalu den indexy v rozsahu 0-23, pro intervaly týden a měsíc jsou to den a měsíc ve tvaru `dd.mm.`.

Hodina	Výskyt	Hodina	Výskyt	Hodina	Výskyt	Hodina	Výskyt
0:00:00 - 0:59:59	0	1:00:00 - 1:59:59	0	2:00:00 - 2:59:59	0	3:00:00 - 3:59:59	0
4:00:00 - 4:59:59	0	5:00:00 - 5:59:59	0	6:00:00 - 6:59:59	0	7:00:00 - 7:59:59	0
8:00:00 - 8:59:59	0	9:00:00 - 9:59:59	0	10:00:00 - 10:59:59	0	11:00:00 - 11:59:59	0
12:00:00 - 12:59:59	0	13:00:00 - 13:59:59	0	14:00:00 - 14:59:59	0	15:00:00 - 15:59:59	0
16:00:00 - 16:59:59	982	17:00:00 - 17:59:59	1299	18:00:00 - 18:59:59	0	19:00:00 - 19:59:59	0
20:00:00 - 20:59:59	0	21:00:00 - 21:59:59	0	22:00:00 - 22:59:59	0	23:00:00 - 23:59:59	0

Obrázek 6.5: Intenzita provozu

Kapitola 7

Závěr

7.1 Shrnutí výsledků

Cílem práce bylo vytvořit nástroj pro analýzu HTTP provozu s přehledným webovým rozhraním. Po analýze dostupných open source nástrojů byl vybrán HTTPry, který byl rozšířen o možnost ukládání do MySQL databáze. Dalším rozšířením nástroje pro logování je podpora IPv6 protokolu, podpora technologie VLAN a překlad doménových jmen pro protokol HTTP verze 1.0. Upravený nástroj byl otestován v reálném provozu a jeho vlastnosti porovnány s původní verzí. Nástroj byl optimalizován pro co možná nejrychlejší vkládání záznamů do MySQL databáze. Ani takto však nedokáže zachytit všechny pakety. Nástroj byl testován také na PostgreSQL databázi, může tedy pracovat i s touto databází. Pro přehlednou prezentaci záznamů bylo navrženo a vytvořeno webové rozhraní, které zároveň provádí analýzu záznamů a tvorbu předdefinovaných statistik.

Výsledek práce splňuje všechny požadavky plynoucí ze zadání. Navíc jsou přidána i další rozšíření nástroje pro logování. Navržený a implementovaný informační systém umožňuje sledovat záznamy v přehledných statistikách. Nástroj je vhodné instalovat např. v domácnosti nebo v menší počítačové síti na proxy serveru, přes který ostatní počítače v LAN síti přistupují k internetu. Kvůli bezpečnosti je dobré mít nástroj spuštěný na stejném počítači, na kterém je databázový server s databází pro ukládání logů.

7.2 Budoucnost projektu

Vhodným dalším rozšířením nástroje je možnost sledovat pro jednotlivé pakety v rámci VLAN sítí jejich příslušnost do virtuální sítě pomocí VID položky, která obsahuje identifikátor VLAN sítě. Pro tuto položku je třeba v databázi vytvořit sloupec vhodného datového typu. Položka VID je 12ti bitová (přesný rozsah je 1-4094, 0 značí prioritní rámce, 4095 je rezervovaná hodnota), vhodným datovým typem je typ `UNSIGNED SMALLINT`. Filtrování záznamů podle VID by v tomto případě mohlo být přidáno do formulářů statistik TOP servery a TOP uživatelé, intenzita provozu i záznamy provozu. Pro zadání IPv6 adresy by mohla být zavedena podpora zadávání prefixu.

Vhodnou ochranou databáze je nastavení omezeného počtu příkazu pro vložení dat `INSERT`, aby i v případě útoku nebylo možné zadat příliš mnoho záznamů do databáze. Konkrétní hodnota by záležela na intenzitě provozu sledované sítě.

Velice potřebným rozšířením informačního systému je jeho bezpečnost. Návrh zvýšení bezpečnosti zahrnuje použití protokolu HTTPS a autentizace uživatele. Zde jsou dvě mož-

nosti, a to autentizace pomocí vlastního formuláře a použití session. S tímto rozšířením je třeba přidat funkce pro spravování uživatelských účtů, rovněž v databázi je třeba vytvořit tabulku pro účty.

Statistiky záznamů v tomto projektu vykonává přímo webový server a databáze, což může být při velkém počtu záznamů či statistik časově i výpočetně náročné. Tento problém by mohlo vyřešit vytvoření statistik průběžně během dne, přičemž webové rozhraní by zobrazovalo již tyto statistiky předpočítané. Vytvoření statistik je možné realizovat přímo v MySQL databázi pomocí událostí. V určitém intervalu by aktivovaná událost přepočítala statistiky z nových záznamů v databázi a uložila je do dalších tabulek pro jednotlivé statistiky.

Literatura

[1] Berners-Lee, T.; Fielding, R.; Irvine, U.; aj.: RFC1945 - Hypertext Transfer Protocol – HTTP/1.0 [online]. `http://www.w3.org/Protocols/rfc1945/rfc1945`, 1996.

[2] Bitel, J.: HTTPry [online]. `http://dumpsterventures.com/jason/httpry/`.

[3] Darie, C.: *AJAX a PHP*. Zoner Press, 2006, iSBN 80-86815-47-1.

[4] Deering, S.; Hinden, R.: RFC 2460 - Internet Protocol, Version 6 (IPv6) [online]. `http://www.ietf.org/rfc/rfc2460.txt`, 1998.

[5] Deering, S.; Hinden, R.: RFC2373 - IP Version 6 Addressing Architecture [online]. http://www.faqs.org/rfcs/rfc2373.html, 1998.

[6] DuBois, P.: *MySQL profesionálně*. Mobil Media, 2003, iSBN 80-86593-41-X.

[7] Fielding, R.; Gettys, J.; Mogul, J.; aj.: RFC2068 - Hypertext Transfer Protocol – HTTP/1.1 [online]. `http://www.faqs.org/rfcs/rfc2068.html`, 1997.

[8] Fielding, R.; Gettys, J.; Mogul, J.; aj.: RFC2616 - Hypertext Transfer Protocol – HTTP/1.1 [online]. `http://www.w3.org/Protocols/rfc2616/rfc2616.html`, 1999.

[9] Gutmans, A.: *Mistrovství v PHP 5*. Computer Press, 2005, iSBN 80-251-0799-X.

[10] IEEE: Virtual Bridged Local Area Networks [online]. `http://standards.ieee.org/getieee802/download/802.1Q-2005.pdf`, 2006, iSBN 0-7381-4877-6.

[11] Information Sciences Institute University of Southern California: RFC791 - Internet Protocol [online]. `http://www.faqs.org/rfcs/rfc791.html`, 1981.

[12] Information Sciences Institute University of Southern California: RFC793 - Transmission Control Protocol [online]. `http://www.faqs.org/rfcs/rfc793.html`, 1981.

[13] Jongsma, J.: Date/Time Picker [online]. `http://home.jongsma.org/software/js/datepicker`, 2010 [cit. 2010-02-23].

[14] Meyer, E.: *Eric Meyer o CSS*. Zoner Press, 2007, iSBN 978-80-86815-64-0.

[15] Pummill, T.; Manning, B.: RFC1878 - Variable Length Subnet Table For IPv4 [online]. ftp://ftp.rfc-editor.org/in-notes/rfc1878.txt, 1995.

[16] Satrapa, P.: *Internetový protokol IPv6 [online]*. CZ.NIC, 2008, iSBN 978-80-904248-0-7.

[17] Stevens, W.; Thomas, M.; Nordmark, E.; aj.: RFC3542 - Advanced Sockets Application Program Interface (API) for IPv6 [online]. `http://tools.ietf.org/html/rfc3542`.

[18] Wempen, F.: *HTML a CSS: Krok za krokem*. Computer Press, 2007, iSBN 978-80-251-1505-3.

[19] WWW stránky: MySQL Documentation [online]. `http://dev.mysql.com/doc/`.

[20] WWW stránky: PCAP Documentation [online]. `http://www.tcpdump.org/pcap3_man.html`.

[21] WWW stránky: PostgreSQL Documentation [online]. `http://www.postgresql.org/docs/`, 2010.

[22] WWW stránky: GChart knihovna [online]. `http://gchart.sourceforge.net/`, 2010 [cit. 2010-02-17].

[23] WWW stránky: Google Chart API [online]. `http://code.google.com/intl/cs/apis/charttools/`, 2010 [cit. 2010-02-17].

[24] WWW stránky: PORT NUMBERS. http://www.iana.org/assignments/port-numbers, 2010 [cit. 2010-05-13].

[25] Zakas, N. Z.: *JavaScript pro webové vývojáře*. Computer Press, 2009, iSBN 978-80-251-2509-0.

Dodatek A

Obsah CD

- Upravený nástroj HTTPry - složka /httpry
- Původní nástroj HTTPry - složka /httpry_old
- Informační systém - složka /is
- Skripty pro vytvoření MySQL a Postgre databáze - složka /sql
- Testovací skripty a vzorky dat - složky /test a /test/data
- Zdrojové kódy technické zprávy - složka /zprava
- Programová dokumentace HTTPry a informačního systému - složka /dokumentace
- Pokyny pro instalaci systému - soubor readme.txt