Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

# Bakalářská práce

# Analýza a vytvoření automatického systému sledování logů

Plzeň 2013 Pavel Hvězda

# Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 10. května 2013

Pavel Hvězda

# Abstract

This work is the result of a project that has been announced by my supervisor, Michal Švamberg. The objective was to create a simple and effective filtering system for University of West Bohemia for managing logs and detection system dysfunction. It is designed to be able to use an inexperienced worker. The project is accessible from web interface. The goal of this work was not to create event viewer, but a short list of anomalies, which the user wishes to display. The project was successfully put into operation on 28 April 2013.

Tato práce je výsledkem projektu, který mi byl zadán mým vedoucím, Michalem Švambergem. Cílem bylo vytvořit jednoduchý a účinný filtrovací systém pro potřeby Západočeské univerzity pro správu logů a odhalování disfunkcí systému. Je navržena aby ji mohl využívat i nezkušený pracovník. Uživatelské prostředí je přístupné pomocí webového rozhraní. Účelem této práce nebylo vytvořit event viewer, nýbrž naopak krátký seznam anomálií, které si sám uživatel přeje zobrazovat. Projekt byl úspěšně nasazen do provozu dne 26.4.2013.

# Obsah

**1 Úvod** **1**

**2 Technologie** **2**
2.1 Logování . . . . . 2
2.1.1 Co je to log . . . . . 2
2.1.2 Zdroje / generátory logů . . . . . 4
2.1.3 Druhy logování . . . . . 4
2.1.4 Správa logů . . . . . 5
2.1.5 Linux . . . . . 6
2.1.6 Analyzátory logů . . . . . 7
2.1.7 Archivace logů . . . . . 7
2.2 Syslog . . . . . 8
2.3 Databáze . . . . . 9
2.4 Webové servery . . . . . 12
2.5 Webové rozhraní . . . . . 13
2.5.1 Zabezpečení webového rozhraní . . . . . 13
2.5.2 Skripty na straně serveru . . . . . 14
2.5.3 Skripty na straně klienta . . . . . 15
2.5.4 jQuery . . . . . 16
2.5.5 dataTables . . . . . 16

**3 Instalace a konfigurace** **18**
3.1 Syslog-ng . . . . . 18
3.1.1 Instalace . . . . . 18
3.1.2 Konfigurace serveru . . . . . 18
3.1.3 Konfigurace klientských stanic . . . . . 20
3.2 MySQL . . . . . 20
3.2.1 Instalace . . . . . 20
3.2.2 Konfigurace . . . . . 21
3.2.3 Tabulky databáze . . . . . 21
3.2.4 Trigger . . . . . 22

3.3 Apache . . . 23
3.3.1 Instalace . . . 23
3.3.2 Konfigurace . . . 23
3.4 WebAuth . . . 23
3.4.1 Instalace . . . 23
3.4.2 Konfigurace . . . 24
3.5 Firewall . . . 27
3.6 PHP 5 . . . 27
3.6.1 Instalace . . . 27

**4 Zpracování logů 28**
4.1 Logy v systému Swatcher . . . 28
4.1.1 Zpracování logů kernelu . . . 29
4.1.2 Práce s logy v systému Swatcher . . . 30

**5 Statistiky dat 32**

**6 Filtry 33**
6.1 Vstupní filtry . . . 33
6.2 Uživatelské filtry . . . 33
6.3 Tvar filtrů . . . 33

**7 Regulární výrazy 35**

**8 Webové rozhraní 36**
8.1 Advanced filtering . . . 36
8.2 Options . . . 38
8.3 Tabulka výpisů . . . 39
8.4 Edit input filter . . . 39
8.5 Oprávnění a přístup . . . 41

**9 Moduly 42**
9.1 addrow.php . . . 42
9.2 applydef.php . . . 42
9.3 config.php . . . 42
9.4 input.php . . . 43
9.5 delete.php . . . 43
9.6 deletefil.php . . . 44
9.7 form.php . . . 44
9.8 main.php . . . 44
9.9 print.php . . . 45
9.10 save.php . . . 45

9.11 save2.php . . . . . 45
9.12 selected.php . . . . . 45
9.13 table.php . . . . . 45
9.14 tablsubm.php . . . . . 46
9.15 timestamp.php . . . . . 47

**10 Závěr** **48**

**11 Přílohy** **51**

**Uživatelská příručka** **51**

# Seznam obrázků

2.1 WebAuth . . . . . 14

3.1 Era model . . . . . 22

4.1 Infrastruktura logování ZČU . . . . . 28
4.2 Kontextové menu . . . . . 30
4.3 Editace vstupního filtru . . . . . 31

5.1 Statistiky velikostí logů pro rok 2012 . . . . . 32

8.1 Webové rozhraní . . . . . 36
8.2 Administrátorské rozhraní . . . . . 41
8.3 Uživatelské rozhraní . . . . . 41

# 1 Úvod

Téma bakalářské práce bylo vypsáno především proto, že dosud neexistuje žádný komplexní nástroj pro správu logů, který by odpovídal požadavkům Západočeské univerzity, konkrétně pracovišti CIV. Sledování logů generovaných počítači nebo servery je důležitá činnost, která nám dává možnost predikce událostí nebo naopak možnost nalézt odpovědi, proč se události staly a může pomoci předejít problémům ovlivňujících koncové uživatele využívající služeb serverů. Vyvíjený nástroj je tedy zaměřen nikoli na koncové uživatele, ale především na správce rozlehlých sítí, jako je například síť Západočeské univerzity, aby jim ulehčil supervizi. Valná většina serverů využívaných na ZČU funguje na různých distribucích systému Linux. Projekt by měl výrazně ulehčit kontrolu událostí a čtení logů, protože bude shromažďovat důležité informace a data (logy) o celé počítačové síti na jednom místě a bude umožňovat rychlé a efektivní filtrování a vyhledávání s přehledným webovým rozhraním. Také bude umožňovat filtrovat, které příchozí logy nechceme do systému zaznamenat trvale. Účelem nebude shromažďovat veškeré logy a nahrazovat funkci event vieweru (nicméně, k této funkci může také velice dobře sloužit). Bude sloužit především jako nástroj, který zaškolený a znalý pracovník nastaví a méně zkušený kolega nebo externista využije ke kontrole a vyhodnocení logů (událostí), které byly do systému uloženy. Předpokládané množství denních příchozích logů do systému by se mělo pohybovat v únosné míře pro správu, tj. v řádech desítek. Projekt byl pojmenován Swatcher.

# 2 Technologie

## 2.1 Logování

### 2.1.1 Co je to log

Termínem log se označuje v počítačovém názvosloví soubor se záznamem událostí, které nastaly v monitorovaném systému. Každá událost je zaznamenána jako jeden log a obsahuje informace, které jednoznačně popisující danou událost (čas, původce, text atp.). Logy jsou záznamy z antivirových programů, firewallů, modulů a dalších aplikací v systému. V dnešní době má každá propracovanější aplikace vlastní logovací systém a prohlížeč. Speciálním typem logu, který například eviduje změny oprávnění nebo vlastníky souborů v operačním systému je audit, tohoto druhu logů využívá Microsoft Windows. Díky tomuto typu logu jsme v případě incidentu či jiné události schopni dokázat konkrétnímu uživateli jeho činnost, popřípadě konkrétní změny v systému. V minulosti byly logy využívány převážně k ladění aplikací či problémů, ale rozvojem výpočetní techniky získaly na důležitosti.

Standard pro záznam programových zpráv se nazývá Syslog. Každý log tohoto standardu kromě vlastního textu obsahuje i další informace: `priority` (anglický termín pro prioritu), `facility` (anglický termín pro kategorii logu), `processID` (anglický název, prakticky se jedná o program, který záznam vygeneroval, ale není to pravidlem), `datetime` (anglický složený termín pro čas a datum spojených v jeden záznam), `host` (zdroj, myšleno server (stroj) který tento log vygeneroval). Přijatý log vypadá pro příklad následovně: měsíc | den | čas | počítač | název procesu[pid] | text zprávy.

```
Apr 21 06:45:25 karkinos syslog-ng[14462]: Suspending write
operation because of an I/O error; fd='30', time_reopen='10'
```

Priorita udává významnost události. Takovýchto úrovní je 9 a jsou označeny následovně od nejnižší k nejvyšší prioritě:

- debug - ladící zpráva,
- info - informační zpráva,

- notice - upozornění, jedná se o normální stav systému,
- warning - varovné upozornění, od této úrovně je běžné logy zapisovat,
- err - hlášení o chybě,
- crit - kritický stav,
- alert - očekává se urychlená reakce,
- emerg - systém se stává nepoužitelným.

Facility (kategorie) udává, od jaké služby pochází. Facility obsahuje 12 předdefinových kategorií:

- auth - autentizace, například zprávy týkající se přihlašování / odhlašování uživatelů,
- auth-priv - autentizace, vyhrazeno pro zprávy které by z bezpečnostních důvodů měly být odděleny od ostatních a které jsou určeny pouze administrátorovi systému,
- cron - zprávy od crontabu - démona, který zajišťuje pravidelné spouštění akcí,
- daemon - blíže neurčené zprávy systémových aplikací,
- kern - zprávy jádra,
- lpr - zprávy týkající se tiskového systému (například lpd apod.),
- mail - zprávy MTA (obecně pošta - například sendmail apod.),
- mark - vyhrazeno pro tzv. „timestamps“. Značky, které se periodicky zapisují do logu,
- news - zprávy NNTP serveru (diskusní skupiny - Usenet news),
- security - znamená totéž co „auth“ - synonymum,
- syslog - zprávy syslogu,
- user - blíže neurčené zprávy uživatelských aplikací,
- uucp - zprávy aplikací UUCP (Unix to Unix Copy Protocol, dnes se již téměř nepoužívá).

Dále je ještě definováno 8 dalších kategorií určených k libovolnému použití. Mají označení local0 až local7.

### 2.1.2 Zdroje / generátory logů

Za zdroje (generátory) logů považujeme systém, který generuje logy. Mohou to být například zařízení jako switche, routery, firewally, hardwarová zařízení, servery, stanice, popřípadně softwarové aplikace. Jednotlivé zdroje, ač stejného druhu mívají rozdílný způsob logování. Výborným příkladem jsou operační systémy. Logování systému MS Windows je odlišné od systému Linux. MS Windows má pro každou událost, která se v systému stane přidělené speciální a unikátní ID, dle kterého si můžeme danou událost vyhledat, například přímo na webu Microsoft[1], zatímco systémy typu Linux nikoliv.

V síti Západočeské univerzity se jedná především o systémy typu Linux. Kromě samotných generátorů logů a logovacích serverů se využívá takzvaných přeposílatelů (forwarders). Forwarders mohou být v lokální síti a data pouze přeposílají na logovací server (i mimo lokální síť).

### 2.1.3 Druhy logování

Ukládání logů (logování) lze provádět více způsoby. Kromě klasického logování na lokální diskové médium jsou i jiné možnosti, jak logovat a následně uchovávat logy.

Jedním z těchto způsobů je logování přes síť: log je odesílán dle nějakého protokolu (standardně TCP a UDP) po síti na externí úložiště, kde jsou logy uchovávány. Jedná se tedy o způsob logování, kdy jsou logy ukládány na jiný stroj. Mějme tedy počítač nebo server, který centrálně přijímá logy z ostatních počítačů na nějakém portu a archivuje je. Na stanici, odkud chceme logy přeposílat, stačí v konfiguračním souboru uvést jméno nebo IP adresu vzdáleného serveru a zadat port (totožný s portem nastaveným na serveru, kam logy chceme odesílat). Logovacími servery nazýváme stanice (servery), které přijímají logy od generátorů logů a na základě konfigurace je dále zpracovávají. Logovací servery mohou logy ukládat do souborů, databáze, jiných datových úložišť nebo je přeposílat na další logovací servery (především z

[1] `http://www.microsoft.com`

důvodu bezpečnosti, popřípadě další analýzy). V praxi se lze setkat s různou topologií logovacích serverů, jednotlivé firmy mají svůj postup pro práci s logy. Je potřeba si uvědomit, že syslog jako takový nemá žádná opatření pro omezení přístupu, čehož lze využít k zahlcení či zaplnění diskového prostoru (DoS, útok způsobující odepření služby). Proto by měl být server, který vzdáleně přijímá logy chráněn například užitím firewallu tak, aby přijímal spojení na UDP (či jiném protokolu, typicky ještě TCP) a portu pouze z těch strojů, které jsme určili nebo pouze z lokálního či určitého segmentu sítě. Pokud je logovací server přímo v lokální síti, omezuje se nebezpečí zahlcení sítě. Pro zajištění redundance, či rozložení zátěže může být logovacích serverů hned několik.

### 2.1.4 Správa logů

Správa logů, anglicky log management, je popis událostí, kterými projde log od jeho vygenerování. Je to soubor pravidel pro práci s logy (získávání, zpracování, ukládání, prohledávání, analýza, popřípadě jejich smazání). Důsledná správa logů uspokojuje zvyšující se potřeby zaručit bezchybný chod systémů v organizacích a umožňuje dokonalý přehled o událostech. Lze ji rozdělit na několik samostatných částí, ale v praxi se jedná o zaběhlý způsob manipulace s logy, který má každá firma/instituce atp. své vlastní a z části unikátní.

Pokud se jedná o důležité logy a hrozí nebezpečí napadení hackerem, tak se proti čištění stop může využít síťová tiskárna, která každý příhozí log vytiskne, popřípadě logování přez síť. Logy mohou přicházet od různých generátorů logů a nemusí být vždy v jednotném formátu. Proces, který logy sjednocuje (provádí normalizaci logů) je výpočetně poměrně náročný, a proto je vždy lepší zvolit pokud možno jednotný systém.

Pokud někdo zaútočí na počítač (server), systémové logy tyto aktivity zaznamenají a aktivity útočníků se dají dohledat. Je běžnou praxí, že se útočníci snaží záznamy buď smazat nebo poupravit.

V rámci ZČU je na stanicích, které přeposílají logy do logovacího systému nainstalován `Syslog-ng` (mluvíme pouze o systémech typu Linux), který přeposílá logy na centrální a záložní logovací server (na obou serverech jsou tedy uchovány totožné informace). Tento systém snižuje pravděpodobnost ztráty dat v případě hardwarové poruchy, útoku a zároveň zvyšuje robustnost.Do systému Swatcher budou odesílány veškeré logy, které přijme právě jeden z těchto logovacích serverů.

### 2.1.5 Linux

Jak již bylo zmíněno, projekt Swatcher je zaměřen především na systémy typu Linux. Operační systém Linux používá unixové jádro, které vychází z myšlenek Unixu a respektuje příslušné standardy POSIX a Single UNIX Specification. Jádro Linuxu je víceúlohové, takže může najedou běžet více oddělených procesů (spuštěných programů), které se v rychlém sledu střídají na procesoru, čímž vzniká dojem jejich současného běhu (tzv. multitasking, stejného efektu lze dosáhnout vícejádrovým procesorem). Zároveň se jedná o víceuživatelský operační systém, na kterém může pracovat více uživatelů zároveň. Proto jsou zavedeny uživatelské účty, ke kterým je přístup chráněn autentizačním mechanismem (klíč, jméno + heslo). K tomu jsou též zavedena přístupová oprávnění, která umožňují omezit přístup jednotlivých uživatelů (resp. jejich procesů) k souborovému systému (tj. souborům a adresářům).

V roce 1983 byl založen projekt GNU, jehož cílem bylo vytvořit nový operační systém unixového typu, který by byl složen jen ze svobodného software. Zakladatelem projektu je Richard Matthew Stallman. Za tímto účelem sepsal licenci GNU GPL, pod kterou jsou šířeny všechny části systému GNU. GNU se stal postupným vývojem kompatibilním s komerčními unixy. K dispozici byly všechny důležité aplikace, systémové knihovny, překladač GCC, textový editor a další, chybělo jen jádro, které by zajistilo samotný běh systému a komunikaci s hardware. Proto byl v roce 1990 zahájen vývoj jádra Hurd, který díky moderní a složité architektuře není stále dokončen.

V roce 1991 Linus Torvalds začal pracovat na vývoji vlastního unixového jádra. Ve škole se totiž seznámil s unixovým operačním systémem Minix. Minix byl ovšem až příliš jednoduchý a navíc k němu nebylo možno získat zdrojové kódy. Po vydání Linusova jádra si okamžitě našlo řadu příznivců, kteří s ním začali spolupracovat, vyvíjet. Linus později uvolnil zdrojové kódy pod licencí GNU GPL. GNU se začal používat společně s jádrem Linux. Vznikl tak produkt se správným názvem GNU/Linux.

Spojením GNU, Linuxu a dalších projektů vznikají takzvané distribuce, jež jsou kompilací jednotlivých částí, a tvoří tak komplexní operační systém. Nikdy proto doopravdy nepracujeme jen s Linuxem nebo GNU, ale s konkrétní distribucí.

### 2.1.6 Analyzátory logů

Analyzátory logů jsou velice užitečným nástrojem pro správce systémů. Vyskytují se od jednoúčelových aplikací přez specializované nástroje na určitý program (nástroj) až ke komplexnějším programům, kde můžeme příchozí data dle možností filtrovat, nastavovat vyjímky, informace atd. Analýza může probíhat v reálném čase na přijímaných a zpracovávaných datech - tuto analýzu označujeme jako real-time (RT). Další možností je data přijímat, ukládat na nějaké úložiště a následně z nich pak generovat například celodenní statistiky. Tuto analýzu nazýváme off-line analýzou. RT analýza se hodí zejména na odhalování kritických stavů, kdy musí být prakticky ihned informován správce (tzv. alert). Díky off-line analýze můžeme sledovat vytížení zařízení, jejich jednotlivých částí a služeb, vytvářet grafy, detekovat anomálie a podobně.

### 2.1.7 Archivace logů

Archivace logů probíhá ve většině případů přímo na logovacím serveru (na serveru který logy shromažďuje od ostatních stanic). Archivaci lze provádět buď ukládáním do databáze, kompresí nebo rotací logů. Pokud by jsme logy nijak nemazali respektive neomezovali, stále by se zvětšovaly a hrozilo by riziko zaplnění disku. Této hrozbě může zabránit právě rotace logů.

Rotace logů nám může usnadnit vyhledávání. Pokud jsou logy rotovány každý den a potřebujeme dohledat informace z určitého dne, nemusíme prohledávat jeden velký soubor, ale stačí nám dohledat soubor jeden soubor. Pokud chceme data uchovávat pouze určitý čas, nastavíme jednoduše počet rotování logů na požadovaný limit a starší soubory jsou při rotaci smazány.

Postup rotace je následující:

1. Soubor s názvem soubor.log je přejmenován na soubor.log.1 aby mohl být vytvořen nový log soubor.log.
2. Při následující rotaci je soubor.log.1 přejmenován na soubor.log.2, soubor.log přejmenován na soubor.log.1 aby mohl být vytvořen nový log soubor.log.
3. Dle našeho nastavení smaže soubory určitého stáří (popřípadě indexu).

Nejčastěji používaný nástroj na OS Linux pro rotaci logů je nástroj `Logrotate`. Lze v něm nastavit:

- jak často se bude provádět rotace (denně, týdně, měsíčně nebo po překročení určité velikosti logu),
- kolik rotací zpětně chceme zachovat,
- typ komprimace a výběr algoritmu,
- práva pro přístupy,
- nastavení akcí před a po rotaci (restart Apache atd.).

## 2.2 Syslog

Pro sbírání událostí na úrovni jednotlivých počítačů / serverů slouží typ aplikace zvaný syslog. Původní syslog vznikl v roce 1980 jako součást projektu Sendmail[2]. Syslog si vedl tak dobře, že se stal takovým nepsaným standardem v unixovém světě. Nakonec ho standardizovala skupina IETF[3]. Jelikož má poměrně dlouhou tradici, většina programů generujících záznamy ho podporuje. Také se kolem jeho protokolu vytvořila řada implementací: Syslogd, Syslog-ng[4], Msyslog[5] nebo Rsyslog[6]. Syslog-ng a Rsyslog jsou k dostání v balíčcích většiny distribucí.

Autorem Syslog-ng je Balázs Scheidler, který v roce 1998 portoval Nsyslogd na Linux. Na Nsyslogd byla založena první vydání, ale později Balázs přepsal jádro celého programu a vznikl tak Syslog-ng, jak ho známe a používáme dnes. Za takto krátkou dobu se stal velmi populární a například Debian, Archlinux nebo SuSE ho mají jako výchozí syslogger.

Rainer Gerhards, primární autor Rsyslog se rozhodl napsat syslog démona který by se mohl srovnat se Syslog-ng. Jméno „Rsyslog“ pochází ze slov syslog a spolehlivý(reliable).

[2] `http://www.sendmail.com/sm/open_source/`
[3] `http://www.ietf.org/`
[4] `http://www.balabit.com/network-security/syslog-ng`
[5] `http://sourceforge.net/projects/msyslog/`
[6] `http://www.rsyslog.com/`

Všechni výše vyjmenovaní daemoni se liší převážně v maličkostech a to především v množství podpůrných balíčků a modulů, jako jsou možnosti ukládat logy do databáze, možnosti filtrování logů a dalších. Můj výběr byl ale zúžený na daemony, kteří umějí zajistit logování do databáze bezplatně (viz bod 2.3), z důvodu další manipulace s daty. Pro projekt Swatcher zvolil Syslog-ng z důvodu, že se stal standardně dodávaným balíčkem pro systém Debian, který je v síti Západočeské univerzity nejvyužívanější. K ukládání do databáze je použit modul `syslog-ng-mod-sql`, který je od verze Debian Wheezy dostupný jako standardní balíček a je tedy bezplatný.

## 2.3 Databáze

Dalším důležitým krokem pro přípravu na efektivní filtrování je výběr databáze a návrh databázového modelu. Databázových systémů je hned několik, například MySQL[7], PostgreSQL[8], Oracle[9], Firebird[10], Microsoft SQL Server[11].

- **MySQL** obsahuje 2 vrstvy. První vrstva, která je úplně nahoře obsahuje služby, jež nejsou jedinečné pro MySQL. Ve druhé vrstvě se nachází valná část mozku MySQL, včetně kódu pro rozbor (parsing), analýzu, optimalizaci a pro všechny zabudované funkce. Na této úrovni se nachází veškerá funkcionalita, která se poskytuje prostřednictvím úložných enginů. Třetí vrstva obsahuje úložné enginy. Ty mají na starosti ukládání a získávání všech dat uložených v MySQL. Server komunikuje s úložnými enginy prostřednictvím API úložných enginů. Toto rozhraní skrývá rozdíly mezi jednotlivými úložnými enginy a činí je na vrstvě dotazů velmi transparentními. API obsahuje několik desítek nízkoúrovňových funkcí, které provádějí operace jako „zahájit transakci“ nebo „získat řádek, který má tento primární klíč“.
- **PostgreSQL** je plnohodnotným relačním databázovým systémem s otevřeným zdrojovým kódem. PostgreSQL umožňuje běh uložených

[7] `http://www.mysql.com/`
[8] `http://www.postgresql.org/`
[9] `http://www.oracle.com/index.html`
[10] `http://www.firebirdsql.org/`
[11] `https://www.microsoft.com/sqlserver/cs/cz/`

procedur napsaných v několika programovacích jazycích, v Perlu, v Pyhon, v jazyku C. PostgreSQL je šířen pod BSD licencí, která je nejliberálnější ze všech open source licencí. Tato licence umožňuje neomezené bezplatné používání, modifikaci a distribuci PostgreSQL a to at' pro komerční nebo nekomerční využití. Plně podporuje cizí klíče, operace JOIN, triggery a uložené procedury.

- **Oracle** je multiplatformní databázový systém s velice pokročilými možnostmi zpracování dat, vysokým výkonem a snadnou škálovatelností. Tento systém podporuje nejen standardní relační dotazovací jazyk SQL podle normy SQL92[12], ale také proprietární firemní rozšíření Oracle (například pro hierarchické dotazy), imperativní programovací jazyk PL/SQL[13] rozšiřující možnosti vlastního SQL (v tomto jazyce je možné tvořit uložené procedury, uživatelské funkce, programové balíky a triggery), dále podporuje objektové databáze a databáze uložené v hierarchickém modelu dat (XML databáze, jazyk XSQL). Dále též obsahuje širokou paletu nástrojů pro podporu.

- **Firebird** je relační multiplatformní databáze. Má plnou podporu procedur a spouští, referenční integritu, možnost externích uživatelských funkcí, nástroje třetích stran včetně grafických administrativních nástrojů.

- **Microsoft SQL Server** je relační databázový a analytický systém. Má integrovanou podporu jazyka XML. Byl navržen pro zvládnutí velkého objemu transakcí. Nabízí možnost zvolit verzi zdarma (SQL Server Express), popřípadě jeho placenou verzi.

Databáze představuje nástroj pro shromáždění a uspořádání informací. Počítačová databáze je kontejnerem objektů. Jedna databáze může obsahovat více než jednu tabulku.

- Tabulka – základní stavební kámen databáze. Skládá se z řádků a ze sloupců. Řádkům říkáme záznam. Řádek nese informace o jednom objektu. Sloupcům říkáme pole) nebo také atributy. Každý sloupec je určen pro jinou vlastnost objektů.

- Primární klíč – jednoznačný (unikátní) identifikátor záznamů, podle kterého můžeme záznamy vybírat a také setřídit. Primární klíč by měl být stálý v čase.

[12] `http://www.contrib.andrew.cmu.edu/ shadow/sql/sql1992.txt`
[13] `http://infolab.stanford.edu/ ullman/fcdb/oracle/or-plsql.html`

- Sekundární klíč – jedno nebo několik polí tabulky, podle kterých můžeme vybírat nebo třídit záznamy v tabulce. Sekundární klíč není jednoznačný (ve více záznamech může být tentýž).
- Cizí klíč – sekundární klíč tabulky, který ukazuje do tabulky jiné, kde je zpravidla klíčem primárním.

Nejvíce využívané typy databází jsou: hierarchická databáze, síťová databáze, relační databáze, objektová databáze, objektově relační databáze, dokumentově orientovaná databáze. Relační databáze má následující parametry:

- základním stavebním kamenem relační databáze je tabulka,
- každá tabulka musí mít pevnou strukturu – každý záznam téže tabulky se musí skládat z polí (atributů) stejného významu a stejného datového typu,
- každá tabulka musí mít jednoznačný identifikátor záznamů – primární klíč,
- tabulka by neměla obsahovat datové duplicity (pole, která obsahují stejná data),
- tabulky v relační databázi propojujeme relacemi (vazbami) - tyto relace nám propojují související záznamy ve dvou tabulkách.

Trigger (česky spoušť) v databázi definuje činnosti, které se mají provést v případě definované události nad databázovou tabulkou. Triggery mají stejnou syntaxi jako uložené procedury, ale není možné předávat žádné vstupní parametry, vracet sadu záznamů.

Popis uživatelské aplikace za účelem specifikovat následně strukturu databáze se nazývá ERA model. Používají se zkratky ERA model, E-R-A model, ER model a E-R model. E = Entita (množina dat), R = Relace (vztah mezi entitami), A = Atribut (jednotlivé položky popisující množinu dat). ERA model představuje relace entit včetně popisných atributů. V modelu mohou existovat různá integritní omezení (IO): kardinalita vztahu, povinnost členství ve vztahu. Na obrázku 4.1 je zobrazena ukázka Era modelu, konkrétně tohoto projektu.

Pro svůj projekt jsem zvolil databázový systém MySQL především kvůli mé předchozí pozitivní zkušenosti a také proto MySQL jsou podporovány jazykem PHP, o kterém se zmíním níže. Zvolil jsem relační databázi z důvodu možnosti provázání tabulek a jejich atributů. InnoDB je jednou z mnoha možných úložišť dat v MySQL, byl navržen pro zpracování transakcí - konkrétně pro zpracování mnoha krátkodobých transakcí, které se minimálně anulují. Tento popis odpovídá požadavkům mého projektu.

## 2.4 Webové servery

Výběr správného webového serveru je velice důležitý, každý z webových serverů, které jsou zmíněny níže jsou provozovatelné na systémech Linux. V této kapitole se tedy nezmiňuji o IIS web server od společnosti Microsoft, který je dostupný jako jedna ze základních funkcí Windows serveru.

- **Lighttpd** je velmi výkoný jednoprocesový (single-process) a jednovláknový (single-threaded) webový server. Veškerá pokročilá funkcionalita je rozdělena do modulů. Díky tomu lze webový server optimalizovat přesně pro ty účely, kterým má sloužit. Jedná o jeden z nejrychlejších webových serverů. Lighttpd svoji rychlost nabírá díky masivnímu internímu cachování a také díky mod_cache. Veškerá konfigurace lighttpd může být vygenerována skriptem, respektive výstup skriptu bude vložen do konfiguračního souboru.

- **Nginx** je lehký webový server/reverzní proxy vydaný pod licencí BSD. Mimo protokolů http/https zvládá i POP3 a IMAP, takže se ve skutečnosti nejedná jen o webový server. Umí zpracovat veškerý statický obsah (statické HTML, CSS, iso, ...)
  a určitý typ požadavků směřovat jinam. Nativně webserver neumí PHP, ani Python, Ruby. S použitím FastCGI nebo jiného webového serveru (který už mod_php nebo cokoli jiného), můžeme Nginx použít jako proxy pro zprostředkování tohoto obsahu.

- **Apache** je webový server s otevřeným kódem. Apache podporuje velké množství funkcí, mnoho z nich je implementováno jako kompilované moduly rozšiřující jádro. Mohou to být funkce podpory programovacích jazyků jako Perl, Python, Tcl nebo PHP. Apache poskytuje mnoho tzv. MultiProcessing modulů (MPM) což mu dovoluje přizpůsobit se

potřebám systému na kterém běží. Apache dále obsahuje externí modul pro kompresi dat webových stránek posílaných protokolem HTTP (mod_gzip), open source modul pro ochranu a prevenci webových aplikací před napadením (mod_security). Je stejně jako Lighttpd rozdělen do modulů.

Pro svůj projekt jsem zvolil webový server Apache z důvodu mých předchozích zkušeností s ním, ale také kvůli stabilitě, kompatibilitě s PHP a MySQL, které jsou základními prvky webového rozhraní projektu.

## 2.5 Webové rozhraní

Webové rozhraní bylo jedním za základních požadavků bakalářské práce. Pro jeho implementaci jsem zvolil HTML jazyk, JavaScript a jeho knihovny zvanými JQuery a skriptovací jazyk PHP.

### 2.5.1 Zabezpečení webového rozhraní

Zabezpečení webového rozhraní je dalším důležitým bodem. První možností jak webový server zabezpečit je napsat vlastní systém ověřování uživatelů, další pak využít již existující způsob (řešení). Všechny webové aplikace na Západočeské univerzitě jsou zabezpečeny pomocí WebAuth.

WebAuth je autentizační systém pro webové stránky a webové aplikace. Je to vstupní brána do domény jednotného přihlašování Single Sign On (SSO). Single Sign-on řešení WebAuth je založeno na autentizačním systému Kerberos. Celý systém WebAuth je pak tvořen třemi spolupracujícími moduly do WWW serveru Apache. Srdcem celého systému je login-server běžně nazývaný WebKDC (KDC je převzato z názvosloví systému Kerberos[14], kde KDC je zkratka pro Key Distribution Center – centrum výdeje kerberos ticketů). Na WebKDC běží modul mod_webkdc. Jeho úkolem je přejímat požadavky od aplikačních serverů, zpracovávat je a ověřovat identitu přistupujícího uživatele u autentizační autority. Na obrázku 2.1 je znázorněné schéma práce WebAuth při pokusu o přístup na spravovanou stránku.

[14] `http://web.mit.edu/kerberos/`


Obrázek 2.1: WebAuth

Díky této technologii odpadá řešení neoprávněného přístupu na webový server. Není tak nutné řešit správu uživatelů a hesel.

### 2.5.2 Skripty na straně serveru

Skriptování na straně serveru je technologie, ve které je uživatelův požadavek vykonán webovým serverem. Slouží k dynamickému generování obsahu. Převážně se využívá k přístupu k databázi či jiným datovým úložištím. K tomuto účelu se využívají například jazyky: PHP, Python, Ruby, ASP, Perl ad.

PHP neboli hypertextový preprocesor, původně Personal Home Page je skriptovací programovací jazyk, určený především pro programování dynamických internetových stránek a webových aplikací například ve formátu HTML, XHTML či WML. Interpret PHP skriptu je možné volat pomocí pří-

kazového řádku, nebo s využitím webových služeb. Syntaxe jazyka je inspirována několika programovacími jazyky (Perl, C a Pascal). PHP je nezávislý na platformě, rozdíly v různých operačních systémech se omezují na několik systémově závislých funkcí a skripty lze většinou mezi operačními systémy přenášet bez jakýchkoli úprav. PHP podporuje mnoho knihoven pro různé účely - například zpracování textu, grafiky, práci se soubory, přístup k většině databázových systémů.

Výhody PHP:

- PHP je specializované na webové stránky,
- rozsáhlý soubor funkcí v základní knihovně PHP (přes pět a půl tisíce),
- nativní podpora mnoha databázových systémů,
- multiplatformost (zejména Linux a Microsoft Windows),
- podpora na hostingových službách – PHP je fakticky standardem, který najdeme všude,
- svobodná licence, která (v protikladu k například GPL) neobsahuje copyleft.

Nevýhody PHP:

- nekonzistentní pojmenování funkcí, například: strpos(), strchr(), ale str_replace(), str_pad(),
- ve standardní distribuci chybí ladící (debugovací) nástroj.

Pro svůj projekt jsem zvolil PHP z důvodu jednoduchosti skriptování a podobnosti s programovacím jazykem C a Java.

### 2.5.3 Skripty na straně klienta

Skriptování na straně klienta je technologie, kde se skript umístěný na webových stránkách vykonává ve webovem prohlížeči uživatele. Jedná se především o jazyky AJAX, JavaScript a Flash.

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk, jehož autorem je Brendan Eich z tehdejší společnosti Netscape. Používá jako interpretovaný programovací jazyk pro WWW stránky, vkládaný přímo do HTML kódu stránky. Jsou jím obvykle ovládány různé interaktivní prvky GUI (tlačítka, textová políčka) nebo tvořeny animace a efekty obrázků. Jeho syntaxe patří do rodiny jazyků C/C++/Java. Slovo Java je však součástí jeho názvu pouze z marketingových důvodů. Program v JavaScriptu se spouští až po stažení WWW stránky z Internetu (tzv. na straně klienta), na rozdíl od ostatních jiných interpretovaných programovacích jazyků (například PHP a ASP). Z toho plynou jistá bezpečností omezení, JavaScript například nemůže pracovat se soubory, aby tím neohrozil soukromí uživatele.

Pro svůj projekt jsem si vybral JavaScript pro jeho rozsáhlý framework jQuery.

### 2.5.4 jQuery

JQuery je JavaScriptová knihovna (framework) s vlastní, neustále se rozrůstající sadou funkcí, která usnadňuje práci s JavaScriptem. Klade důraz na jednoduchost, čitelnost a rychlost. Je multiplatformní a je dostupná zdarma. Má mnoho funkcí:

- jednoduchou manipulace s CSS,
- selektory,
- efekty (pomocí předdefinovaných funkcí),
- jednoduchá tvorba animací,
- spoustu pluginů,
- utility – například informace o prohlížeči.

### 2.5.5 dataTables

DataTables je plug-in jQuery. Je to velmi flexibilní nástroj, založený na základech HTML tabulky, díky kterému můžeme přidat pokročilé interakce ovládacích prvků a další funkce. Klíčové vlastnosti:

- variabilní délka stránkování,
- sloupce s možností třídění,
- inteligentní manipulace šířek sloupců,
- zobrazení dat z téměř jakéhokoli zdroje dat.

Těchto výhod bylo v projektu využito prakticky pro veškeré výpisy z databáze, především kvůli přehlednosti, možnosti rychlého filtrování nad výsledky a volby počtu výsledků na stránku.

# 3 Instalace a konfigurace

Pro potřeby projektu mi byl přidělen virtuální stroj, na kterém jsem provedl instalaci operačního systému Debian Wheezy. Následující popis instalací bude tedy odpovídat této verzi Linuxu.

Pro funkčnost projektu Swatcher je nutné nainstalovat a nakonfigurovat tyto komponenty:

- Syslog-ng - logovací daemon,
- Syslog-ng-mod-sql - modul Syslog-ng umožňující ukládání příchozích logů do databáze,
- MySQL - databázový systém,
- Apache - HTTP server a nastavit zabezpečení serveru,
- WebAuth - zabezpečení přístupů na webové stránky,
- Firewall - slouží k řízení a zabezpečení síťového provozu,
- PHP 5 - hypertextový procesor.

## 3.1 Syslog-ng

### 3.1.1 Instalace

Instalaci provedeme s přihlášeným uživatelem `root` příkazem:`apt-get install syslog-ng-mod-sql`. Nainstaluje se nám nejen balíček pro ukládání do databáze, ale zároveň i všechny balíčky potřebné pro jeho fungování, takže Syslog-ng.

### 3.1.2 Konfigurace serveru

Syslog-ng uchovává své konfigurační soubory ve složce `/etc/syslog-ng/`. Pro odpovídající funkčnost bude potřeba upravit soubor:

`/etc/syslog-ng/syslog-ng.conf`. Konfigurační soubor má segmenty: `source`, `destination`, `filter` a `log path`. Určíme tedy odkud budeme přijímat logy vložením následujících řádků do `source`:

```
source src {
 unix-dgram("/var/run/log");
 unix-dgram("/var/run/logpriv" perm(0600));
 udp(
  ip(192.168.1.155)
  port(514)
 );
   internal();
 file("/dev/klog");
};
```

Tím jsou nastaveny port a IP adresa, na které bude Syslog-ng očekávat spojení od zdrojů.

Dále určíme cílové úložiště přijímaných logů přidáním následujících řádků do `destination`:

```
destination d_mysql {
 program("/usr/bin/mysql --uswatcher Syslog --ppassword"
 template("INSERT INTO Hosts (host) VALUES ( '$HOST' ) ON
 DUPLICATE KEY UPDATE last_update = NOW();
 INSERT INTO Logs (id_host, facility, priority, level, tag,
 datetime, program, msg)
 (SELECT id, '$FACILITY', '$PRIORITY', '$LEVEL', '$TAG',
 '$YEAR-$MONTH-$DAY $HOUR:$MIN:$SEC', '$PROGRAM', '$MSG'
 FROM Hosts WHERE host = '$HOST');\n")
 template-escape(yes));
}
```

Nakonec určíme že Syslog-ng má logovat ze zdroje `src` do úložiště `d_mysql` následujícími řádky do `log paths`:

```
log {
 source(src);
 destination(d_mysql);
};
```

Pokud nechceme dále nastavovat jaké informace chceme od partnerů přijímat, a jaké ne, je nastavení filters v tomto konfiguračním souboru je pro nás nedůležité. Po uložení konfiguračního souboru je nutné restartovat službu Syslog-ng daemona, to provedeme příkazem: `/etc/init.d/syslog-ng restart`.

### 3.1.3 Konfigurace klientských stanic

Konfigurace klientských stanic spočívá v nastavení logování systému na vzdálenou IP. Určíme tedy jaké logy budeme vzdálenému serveru odesílat (poskytovat) vložením následujícího kódu do `sources`:

```
source s_all {
  # message generated by Syslog-NG
  internal();
  # standard Linux log source (this is the default place
  for the syslog()
  # function to send logs to)
  unix-stream("/dev/log");
  # messages from the kernel
  file("/proc/kmsg" program_override("kernel: "));
  # use the above line if you want to receive remote UDP
  logging messages
  # (this is equivalent to the "-r" syslogd flag)
  # udp();
};
```

Dále určíme cílové úložiště přijímaných logů přidáním následujících řádků do `destination`:

```
destination net_remote_logServer1 { tcp("192.168.200.230"); };
```

Nakonec určíme, že Syslog-ng má logovat ze zdroje `src` na vzdálený logovací server následujícím řádkem do `log paths`:

```
log { source(s_all); destination(net_remote_logServer1); };
```

## 3.2 MySQL

### 3.2.1 Instalace

Instalaci provedeme s přihlášeným uživatelem `root` příkazem: `apt-get install mysql-server mysql-client`. Tímto nainstalujeme jak MySQL server, tak klienta potřebného pro připojení a další knihovny.

### 3.2.2 Konfigurace

Pro možnost připojení do databáze musíme provést konfiguraci. Konfigurační soubory MySQL serveru se nachází ve složce `/etc/mysql/my.cnf`. Nastavením, které nás bude primárně zajímat bude `bind-address`, která určuje na jaké adrese bude poslouchat příchozí spojení služba MySQL serveru, hodnotu si můžeme zvolit dle potřeby.

### 3.2.3 Tabulky databáze

Databáze projektu obsahuje tabulky (viz obr. 3.1):

- `Logs`

  Tabulka `Logs` slouží k uchovávání logů, které jsou vloženy do systému. Struktura tabulky vychází z potřeb modulu `syslog-ng-mod-sql`.

- `Hosts`

  Tabulka `Hosts` slouží k uchovávání generátorů logů (původců jednotlivých logů).

- `Filters`

  Tabulka `Filters` slouží k uchování filtrů, ať už uživatelských, nebo vstupních (viz bod 6). Zda je filtr vstupní nebo uživatelský se ukládá do relační tabulky `Primitive_in_filter`.

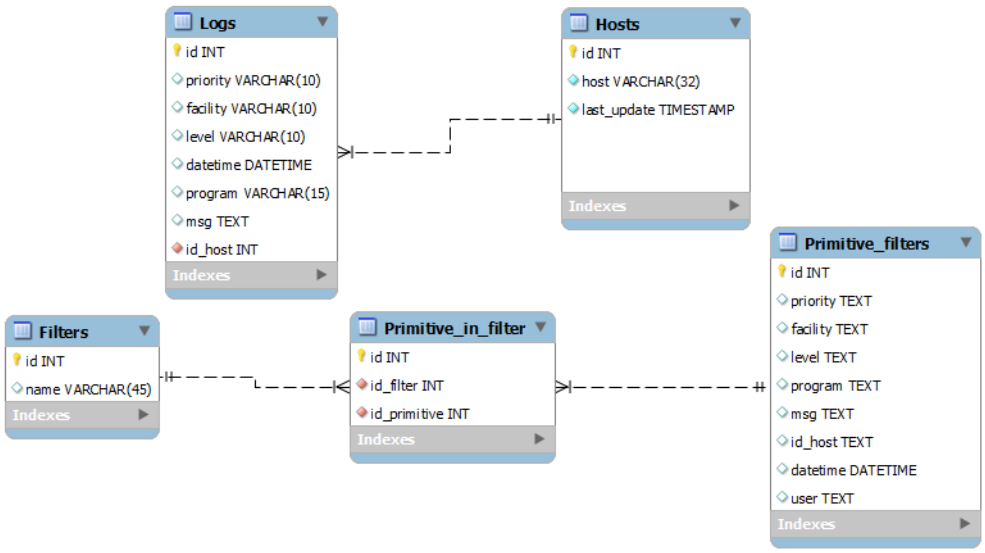
- `Primitive_filters`

  Tabulka `Primitive_filters` obsahuje jednotlivé filtry.

- `Primitive_in_filter`

  Tabulka `Primitive_in_filter` je relační tabulka, která uchovává informace o relacích mezi tabulkami `Filters` a `Primitive_filters`.

Stuktura jednotlivých tabulek je vidět na obrázku 3.1.




Obrázek 3.1: Era model

### 3.2.4 Trigger

Trigger funguje na principu, že vybere všechny vstupní filtry z tabulky `Primitive_filters`. Postupně prochází po sloupcích veškeré záznamy a porovnává jestli se obsah ve sloupci příchozí zprávy nenachází v některém sloupci, nebo je nulový. Je tedy prakticky využit princip logického součinu. Pokud tedy příchozí záznam nebyl nalezen v záznamech vstupního filtru, je uložen do databáze.

Výsledný trigger použitý v systému má tvar:

```
delimiter //
CREATE TRIGGER inputFilter
before insert
on Logs
for each row
begin
 declare msg varchar(255);
 IF (SELECT COUNT(*) FROM Primitive_filters, Primitive_in_filter, Filters
    WHERE Filters.name = "input"
      AND Filters.id =  Primitive_in_filter.id_filter
      AND Primitive_in_filter.id_primitive =  Primitive_filters.id
      AND Primitive_filters.id_host LIKE  CONCAT('%',(SELECT host FROM
          Hosts WHERE id = new.id_host),'%')
        OR Primitive_filters.id_host IS NULL)
```

```
      AND (Primitive_filters.facility LIKE CONCAT('%',new.facility,'%')
         OR Primitive_filters.facility IS NULL)
      AND (Primitive_filters.priority LIKE CONCAT('%',new.priority,'%') OR
         Primitive_filters.priority IS NULL)
      AND (Primitive_filters.tag LIKE  CONCAT('%',new.tag,'%') OR
         Primitive_filters.tag IS NULL)
      AND (Primitive_filters.program LIKE CONCAT('%',new.program,'%')
         OR Primitive_filters.program IS NULL)
      AND (new.msg REGEXP Primitive_filters.msg
         OR Primitive_filters.msg IS NULL)) > 0
      THEN
         CALL raise_error;
   END IF;
END //
```

## 3.3 Apache

### 3.3.1 Instalace

Instalaci provedeme s přihlášeným uživatelem `root` příkazem: `apt-get install apache2`. Tímto nainstalujeme HTTP server Apache.

### 3.3.2 Konfigurace

Konfiguraci HTTP serveru můžeme ponechat v defaultním nastavení, pouze se musí zajistit konfigurace WebAuth, které zajišťuje zabezpečení.

## 3.4 WebAuth

### 3.4.1 Instalace

Instalaci provedeme na systému linux, s přihlášeným uživatelem `root` příkazem: `apt-get install libapache2-webauth`. Tímto nainstalujeme moduly WebAuth do HTTP serveru Apache (konkrétně mod_webauth,

mod_webauthldap a mod_webkdc). Dále budeme potřebovat OpenSSL. Ten nainstalujeme stejně jako moduly k Apache2.

### 3.4.2 Konfigurace

Při konfiguraci WebAuth jsem vycházel z návodu[1], který vypracovalo středisko CIV Západočeské univerzity.

Pro plnou funkčnost WebAuth na straně naší aplikace se musí provést nastavení jednotlivých modulů.

Definice nastavení pro modul `mod_webauth`:

```
WebAuthLoginURL "https://webkdc.zcu.cz/login.fcgi"
   #URL odkazující na přihlašovací formulář
WebAuthWebKdcURL "https://webkdc.zcu.cz/webkdc-service/"
   #URL odkazující na adresu, kam aplikační servery přímo posílají
   kryptované požadavky
WebAuthWebKdcPrincipal webkdc/webkdc
   #obsahuje název krb principalu služby WebKDC
WebAuthKeyring /etc/webauth/keyring
   #souvisí s lokálně uloženým privátním AES klíčem aplikačního
    serveru, sděluje kde daný klíč leží
WebAuthKeyringAutoUpdate on
   #souvisí s lokálně uloženým privátním AES klíčem aplikačního
   serveru, povoluje modulu automatickou změnu tohoto klíče
WebAuthKeyringKeyLifetime 30d
   #souvisí s lokálně uloženým privátním AES klíčem aplikačního
   serveru, určuje jak dlouho může být jednou vygenerovaný klíč
   používán před další výměnou
WebauthKeytab /etc/webauth/keytab
   #sděluje, kde na disku je možné najít klíč krb principalu
   aplikačního serveru
WebAuthServiceTokenCache /etc/webauth/service_token.cache
   #slouží k uložení service-tokenu, který se tak stává použitelným
   pro všechny nově vytvářené procesy WWW serveru Apache.
```

Konfigurace modulu `mod_webauthldap`:

```
WebAuthLdapHost ldap.zcu.cz
 #na kterém serveru bude WebAuthLDAP hledat informace pro autorizaci
```

[1] `http://support.zcu.cz/index.php/LPS:WebAuth/Konfigurace_WebAuthu`

```
    podle skupin.
   WebAuthLdapBase ou=rfc2307,o=zcu,c=cz
    #definuje konkrétní úložiště s informacemi o skupinách.
   WebAuthLdapAuthorizationAttribute cn
    #určuje, který atribut koresponduje s položkou v LDAPu obsahující
    skupiny uživatelů
   WebAuthLdapKeytab /etc/apache2/ldapkeytab webauth/karkinos.civ.zcu.cz
    #obsahuje odkaz na soubor s klíčem krb principalu použitého pro
    dotaz na LDAP server
   WebAuthLdapFilter memberUid=USER
    #definuje filtr do LDAPu, který určuje, na kterou z položek je
     mapován login uživatelů
   WebAuthLdapTktCache /tmp/webauthldap.tkt
    #určuje Kerberos cache, která pokud obsahuje validní krb ticket,
    tak je použit, jinak je použit dříve definovaný keytab k získání
    nového krb ticketu
```

Další část konfigurace už se týká přímo chránění přístupu pro definovaný adresář pomocí WebAuth SSO řešení.

```
<Directory />
   SSLRequireSSL
   Options -Indexes +FollowSymLinks
   AllowOverride All
   Order allow,deny
   Allow from all
   DirectoryIndex index.php

   SSLOptions +ExportCertData +StdEnvVars +FakeBasicAuth
    +OptRenegotiate
   AuthType WebAuth
    #definuje, že přístup pro daný adresář je používána služba WebAuth
   Require privgroup lps
    #určuje seznam povolených skupin uživatelů (zde skupinu lps)
   require user phvezda
    #řádek obsahuje explicitní seznam povolených uživatelů, kteří budou
    autorizováni bez ohledu na příslušnost v některé ze skupin.
</Directory>
```

Také je nutné získat SSL certifikát pro webový server. Na webové stránce žádosti o certifikát zvolíme jméno organizace česky, vyplníme doménové jméno (v našem případě `karkinos.civ.zcu.cz`), a kliknutím na OpenSSL tlačítko provedeme vygenerování konfiguračního souboru OpenSSL. Ten si uložíme například do souboru `server-req.cfg`. Žádost o certifikát vygenerujeme příkazem:

```
openssl req -new -keyout server.key.org out server.csr -config
server-req.cfg
```

Z vygenerovaného kryptovaného souboru s klíčem vygenerujeme ještě nekryptovanou PEM verzi soukromého klíče RSA příkazem:

```
openssl rsa -in server.key.org -out server.key
```

Výslednou žádost vygenerujeme příkazem:

```
openssl req -in server.csr -text
```

Takto vygenerovanou žádost (soubor `server.csr` zašleme na e-mailovou adresu `aaa-req@service.zcu.cz`. Po schválení a vygenerování certifikátu nám administrátoři zašlou e-mailem zpět hotový certifikát. E-mail by měl být následovně strukturovaný:

```
Název serveru
Popis a účel
Administrátor web. serveru, e-mail
Používané aliasy
Délka platnosti cert (standardně 2 roky)
```

Po obdržení certifikátu je nutné v konfiguračním souboru Apache pro příchozí spojení na portu 443 vložit následující řádky:

```
SSLEngine on
SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:
   +SSLv2:+EXP:+eNULL
SSLCertificateFile /etc/apache2/ssl/server.crt
SSLCertificateKeyFile /etc/apache2/ssl/server.key
SSLCertificateChainFile /etc/apache2/ssl/ca-chain.pem
```

Nakonec přesměrujeme příchozí spojení na portu 80 na SSL (port 443, tj. na adresu `https://karkinos.civ.zcu.cz`.

## 3.5 Firewall

V operačním systému Debian, který jsem instaloval na server je nastaven jako defaultní firewall `iptables`. Firewall byl nastaven mým vedoucím dle norem střediska ZČU - CIV.

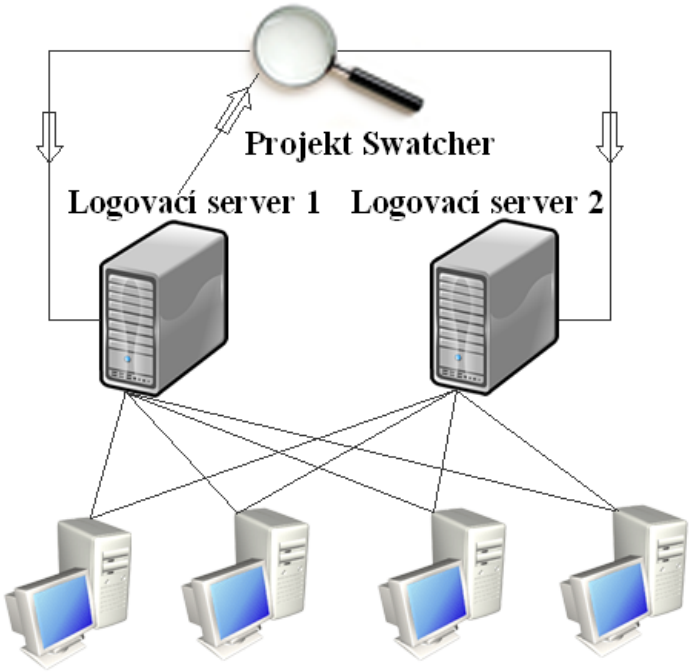
## 3.6 PHP 5

### 3.6.1 Instalace

Instalaci provedeme s přihlášeným uživatelem `root` příkazem:`apt-get install php5 php5-cli`. Tímto nainstalujeme PHP 5 a jeho modul pro spoutění skriptů z příkazové řádky.

# 4 Zpracování logů

V síti Západočeské univerzity se logování provádí na 2 centrální logovací servery, jeden ze těchto serverů přesměrovává příchozí logy na můj projekt. Swatcher přeposílá své logy na oba logovací servery. Přibližná zjednodušená infrastruktura tedy vypadá viz obr. 4.1.




Obrázek 4.1: Infrastruktura logování ZČU

## 4.1 Logy v systému Swatcher

Příchozí logy do systému Syslog-ng rozdělí na jednotlivé části (datetime, program, priority atd.) a provede vložení do tabulky `Logs`, pokud nejsou obsaženy ve vstupním filtru (viz bod 6.1). O uložení či neuložení příchozího logu rozhoduje trigger, který je nastaven nad tabulkou `Logs`, kam se příchozí logy ukládají na akci `before insert`. Logy které projdou tímto filtrováním, jsou

uloženy do výše zmíněné tabulky *Logs* do jednotlivých sloupců odpovídajích jednotlivým částem informací z logu. Při otevření webového prohlíže a načtení úvodní stránky jsou načteny veškeré logy obsažené v systému, s limitem zobrazení 1000 zpráv, což je vzhledem k účelu projektu naprosto dostačující (možnost nastavení viz modul `config.php`).

### 4.1.1 Zpracování logů kernelu

Logy generované samotným jaderným modulem (kernelem), jsou poněkud odlišné od ostatních logů. Jsou generovány při startu systému, spušnění/reakce nějakého modulu jádra, popřípadě při kritických chybách. Tyto logy jsou pro správce systému velice důležité, řekněme prvořadé, jelikož jádro je základním stavebním kamenem celého systému, tudíž jeho stabilita ovlivňuje chování celku. Logy generováné kernelem jsou unikátní v tom, že na počátku zprávy obsahují timestamp, který přesně na milisekundy určuje čas události od spuštění systému, což u jiných logů nenajdeme (tato funkčnost bohužel chybí ve starších jádrech). Takovýto záznam vypadá přibližně takto:

```
: [3999154.040920] svc: 147.228.52.200, port=59439: unknown version
(0 for prog 100003, nfsd)
```

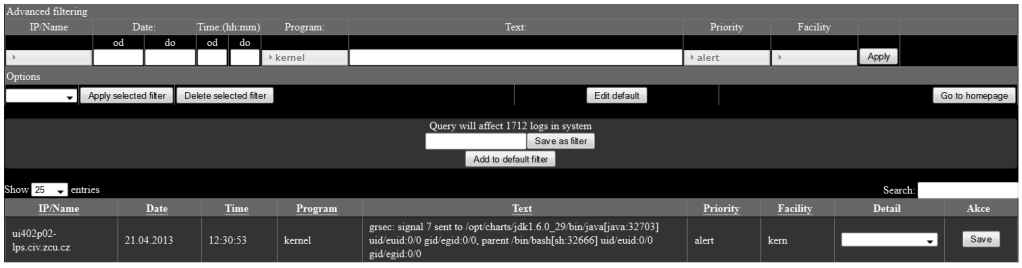
Prvních sedm čísel vyjadřuje vteřiny, zbylých 6 číslic pak nižší časové řády. Syslog-ng na koncové stanici sbírá logy systémem čekání na změnu v souboru, kam program (modul či jiná součást) logy ukládá. Kernel ale vygeneruje několik „logů“ nebo spíše záznamů za nepatrný čas, respektive s nepatrným časovým rozdílem (v rámci mikrosekund až jedné vteřiny), ale tyto zaznamenané jednotlivé logy jsou vázány k jedné události, popřípadě by měly být brány jako jedna událost a tak i jako jeden výpis, to se ale neděje (díky systému ukládání logů do souboru a následného čtení). Výsledkem tedy je několik záznamů v databázi, které je nutno spojit dle určitého pravidla. Rozhodl jsem se nastavením naplánované úlohy v crontabu systému, na kterém běží Swatcher, spustit každou minutu php skript (viz modul `timestamp.php`), který provádí následující operace: Nejprve si vypíše veškeré logy z databáze, kde se `programID` rovná programu `kernel` a seřadí je vzestupně dle času a id stanice, od které byl log obdržen. Poté po jednom záznamu porovná s následujícím záznamem id stanice (id v tabulce `Hosts`), čas ve vteřinách (tj. prvních 7 číslic timestampu) a následně odečte čas v nižších řádech (zbylá šestice čísel) první a nádledující události, pokud je tento rozdíl záporný či nulový (k

tomuto nastavení slouží proměnná `timestamp` v modulu `config.php`), dojde k meziuložení textu první události a připojení textu z události druhé. Takto skript projde celou databázi. Logy, které byly spojeny, smaže a vytvoří jeden nový log, kterému na začátek textové zprávy vloží timestamp.

### 4.1.2 Práce s logy v systému Swatcher

Práce s logy v systému Swatcher je rozdělena do několika možných scénářů.

1. Uživatel nebo administrátor hledá konkrétní záznamy v databázi.
   - Využije tabulku `Advanced filtering`. Textové pole `Text` je vyhodnocováno regulárními výrazy. Pokud tedy chceme využít nějaký ze speciálních znaků při vyhledávání holého textu, musíme před něj vložit zpětné lomítko.
   - Po stisknutí tlačítka `Apply` je vypsáno až 5000 logů (z důvodu efektivity a rychlosti dataTables, možnost nastavení pomocí proměnné v modulu `config.php`), které odpovídají našim kritériím.
   - Zároveň se zobrazí kontextové menu (viz obr. 4.2), které vypíše počet logů ovlivněných našim výběrem, nabízí volby `Save as filter` a pouze administrátor pak `Add to input filter` (viz obr. 4.2).




Obrázek 4.2: Kontextové menu

2. Administrátor ve výpisu logů nalezne záznam, který nechce v systému uchovávat.
   - Využije rolovacího menu `Detail` v tabulce výsledků a zvolí akci:

**ignore event** přidej tento záznam do vstupního filtru pro tento stroj (tato událost od tohoto stroje nebude uložena do systému).

**ignore this host** přidej tento stroj dovstupního filtru (žádná událost, která přijde od tohoto stroje nebude uložena do systému).

**ignore event for all** přidej tento záznam do vstupního filtru (tato událost od tohoto stroje nebude uložena do systému).

- Po stisknutí tlačítka `Save` se provede uložení do vstupního filtru.

3. Administrátor chce smazat či upravit uložené vstupní filtry, popřípadě přidat nový.

   - Kliknutím na tlačítko `Edit input filter` na hlavní stránce se dostane na stránku s možností editování vstupního filtru (viz obr. 4.3).
   - Pokud chce vytvořit nový záznam, stiskne tlačítko `Add new row`, do databáze je tedy vložen nový záznam, který je neprázdný (sloupec `Text` je vyplněn upozorněním, že se filtr nesmí uložit prázdný).
   - Pokud chce upravit již existující záznam, může provést libovolné změny v jednom řádku a stisknutím tlačítka `Save` záznam uložit.
   - Tlačítkem `Delete` libovolný záznam smaže.
   - Stisknutím tlačítka `Test`, lze zkontrolovat vyplněné údaje, systém vypíše které sloupce jsou v pořádku a které ne. Dále vyčíslí počet ovlivněných logů.


Obrázek 4.3: Editace vstupního filtru

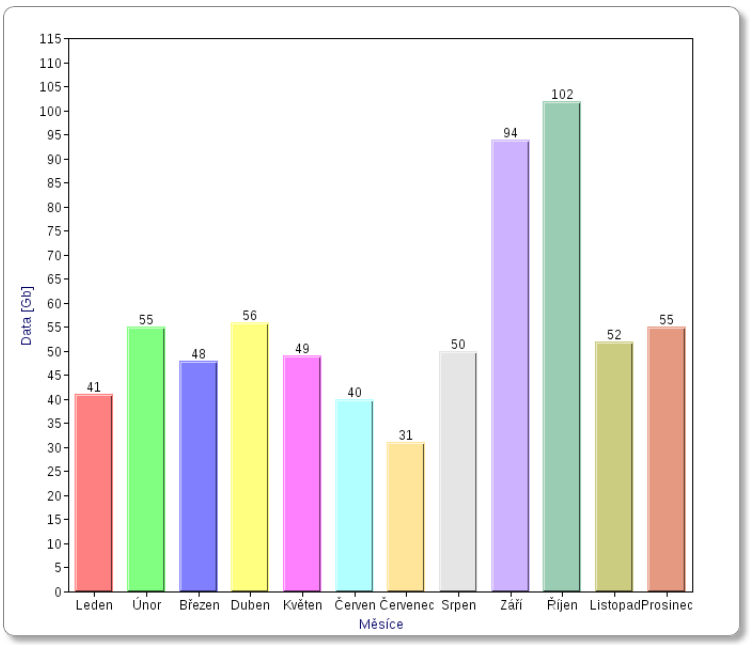
# 5 Statistiky dat

V síti Západočeské univerzity se v roce 2012 nacházelo 349 generátorů logů. Tyto zdroje vygenerovaly ve zmíněném roce 2012 - 673 GB (viz obr. 5.1), což je při průměrné délce odesílané zprávy (logu) 147.332B cca 4897260864 logů. Jednoduchým výpočtem dostaneme počet příchozích logů na centrální servery za sekundu:

$$4897260864/365/24/60/60 = 155.29 \text{ zaznamů/s}$$

Výsledek udává pouze průměrnou hodnotu logů, které jsou ukládány do systému.

Během měření, které proběhlo dne 25.4., proběhlo vyčištění tabulek v MySQL a spuštění bez zadaných vstupních filtrů. Za 180 minut se do systému uložilo 2 534 300 logů ze 341 zdrojů. Po přidání několika základních vstupních filtrů, se počet logů v databázi zkrátil na 23 723. Dle testu lze usoudit, že pro můj virtuální stroj bylo přiděleno dostatečné množství výpočetního výkonu.




Obrázek 5.1: Statistiky velikostí logů pro rok 2012

# 6 Filtry

Systém filtrů v systému je rozdělen do dvou kategorií: vstupní filtry a uživatelské filtry.

## 6.1 Vstupní filtry

Vstupní filtry v systému představují množinu událostí, které nechceme zaznamenávat. V systému jako takovém reprezentuje tuto množinu pravidel pouze jeden prvek a to je filtr `Input` v tabulce `Filters`, na něj jsou pak navázány další záznamy z relační tabulky `Primitive_in_filter` a z ní jednotlivá pravidla v tabulce `Primitive_filters`. Editovat tato pravidla můžeme na stránce, která se načte po zmáčknutí tlačítka `Edit input filter`, které je dostupné pouze pro administrátory (viz bod 8.5).

## 6.2 Uživatelské filtry

Uživatelské filtry jsou množinou filtrů, které si navolil sám uživatel ve webovém rozhraní. Slouží především pro usnadnění manipulace uživatele se systémem, aby nemusel stále v případě potřeby vyplňovat stejné údaje. Veškeré uživatelské filtry ukládané do systému musí mít unikátní název a nesmí se jmenovat `input`, což je i systémem kontrolováno. Takto uložené filtry si může uživatel aplikovat zvolením filtru v tabulce `Options` a stisknutím tlačítka `Apply selected filter` (viz bod 8.2). Tyto filtry slouží k prohlížení logů, které nebyly odfiltrovány „Input filtrem“.

## 6.3 Tvar filtrů

Tvar filtrů ukládaných do databáze byl během psaní projektu několikrát změněn z důvodu neuceleného pohledu na problematiku. Tabulka
`Primitive_filters` obsahuje prakticky stejné sloupce jako tabulka `Logs`, navíc byl ovšem přidán sloupec `user`, který indikuje jaký uživatel filtr vytvo-

řil a sloupec `datetime`, z důvodu časového údaje o vytvoření filtru (úpravě). Z důvodu možnosti několikanásobného výběru údajů u `program`, `priority`, `facility` a `hosts` jsou jednotlivé údaje ukládány do sloupců ve tvaru:

```
'kernel', 'syslog-ng'
```

Zpráva je ponechána v původním tvaru, jen obsahuje nulování speciálních znaků (viz bod 7).

# 7 Regulární výrazy

Regulární výraz (zkracováno na regexp, regex či jen RE podle anglického regular expression) je řetězec popisující celou množinu řetězců. Představil je americký matematik Stephen Cole Kleene.

V mém projektu byly regulární výrazy zvoleny především pro jejich obecnost a možnosti, kterými se dá definovat celý obsah zprávy. Ukázkou využití je zpráva, kde se objevuje uživatelské jméno a nás zajímá pouze co udělal uživatel root. Tento problém by bez regulárních výrazů nešel vyřešit.

Abych mohl tuto funkčnost implementovat, využil jsem knihovny pro MySQL - REGEXP. REGEXP se nemusí instalovat, je nativní součástí MySQL. Využívá množství speciálních znaků, kterými se výraz definuje. Problém ovšem nastává, pokud chce uživatel označit událost (tj. záznam ve výpisu logů) jako nežádoucí a nechce měnit text, musí se ve zprávě provést nulování těchto speciálních znaků. Pokud chceme například uložit zprávu: `timeout: retrying...`, systém jí uloží do databáze jako: `timeout: retrying\.\.\.` . Aby se zpráva opravdu uložila v tomto formátu, musí být zpětné lomítko duplikováno, jelikož v MySQL znamená zpětné lomítko také pro nulování znaků, ale následně ho neuloží, musíme tedy lomítko vložit třikrát. Další komplikací je ovšem, že jazyk PHP také považuje zpětné lomítko za nulovací znak, tudíž pokud chceme, aby tečky byly vynulovány tímto způsobem, musíme nahradit jedno zpětné lomítko pěti zpětnými lomítky, by se do databáze opravdu uložilo jedno.

Kompletní dokumentaci k REGEXP lze nalézt přímo na webových stránkách MySQL[1] a práci s nimi pak například na tutorialspoint[2].

[1] `http://dev.mysql.com/doc/refman/5.1/en/regexp.html`
[2] `http://www.tutorialspoint.com/mysql/mysql-regexps.htm`

# 8 Webové rozhraní

Webové rozhraní je přizpůsobeno především přehlednosti, s možností efektivního pokročilého filtrování logů. Vzhled stránek je rozdělen (viz obr. 8.1) na několik částí: `Logo`, `Advanced filtering`, `Options` a `Results` a v některých případech, které budou níže specifikované se zobrazí speciální kontextové menu.


Obrázek 8.1: Webové rozhraní

## 8.1 Advanced filtering

Tabulka Advanced filtering slouží k vyhledávání přes celou tabulku logů uložených v systému. Nabízí možnost filtrovat logy podle:

1. IP/Name
   - Slouží k vyhledávání logujících strojů, pokud se nezdaří překlad IP adresy přez DNS na jméno, je zobrazena IP adresa.
   - Možnost vícenásobného výběru z rolovacího zaškrtávacího menu.
   - Řazeno abecedně.
2. Date
   - Slouží k vyhledávání logů k určitému datu.
   - Možnost filtrovat *od-do* určitého datumu.
   - Pro snažší výběr datumu je použita knihovna datepicker (JavaScript), která vyvolá kalendář s možností procházet měsíce, dny a roky.

- Pokud je vyplněno pouze pole *od*, systém automaticky vypíše všechny logy od námi zadaného data až do aktuálního, pokud je vyplněno pouze pole *do*, systém vypíše veškeré logy z databáze až do aktuálního data.

3. Time

   - Slouží k vyhledávání podle času.
   - Možnost filtrovat *od-do* určitého času.
   - Čas musí být zadán ve formátu HH:MM, jinak bude vypsána výstražná hláška o neplatnosti zadání a pole vyčištěno.
   - Pokud je vyplněno pouze pole *od*, systém automaticky vypíše veškeré logy zadaného až do aktuálního času, pokud je vyplněno pouze pole *do*, systém vypíše veškeré logy z databáze až do aktuálního času.

4. Program

   - Slouží k vyhledávání podle programu, který daný log generoval, dle RFC 5424 normy se jedná o procID.
   - Možnost vícenásobného výběru z rolovacího zaškrtávacího menu.

5. Text

   - Slouží k vyhledávání podle textu zprávy, dle RFC 5424 normy se jedná o MSG.
   - Možnost vyhledávání podle hesel, nebo po sobě jdoucích sekvencí slov, které odpovídají vyhledávanému textu. Text je pak interpretován jako regulární výraz (viz bod 7).

6. Priority

   - Slouží k vyhledávání podle priority - závažnosti události, dle RFC 5424 normy se jedná o SEVERITY.
   - Možnost vícenásobného výběru.
   - Řazeno abecedně.

7. Facility

   - Slouží k vyhledávání podle facility - druhi události, dle RFC 5424 normy se jedná o FACILITY.

- Možnost vícenásobného výběru.
- Řazeno abecedně.

8. Tlačítko Apply
    - Tlačítko sloužící k odeslání formuláře a vyhodnocení vybraných hodnot.

## 8.2 Options

Tabulka Options slouží k manipulaci buď již s uloženými filtry, nebo k editaci filtru vstupního.

1. Rolovací menu
    - Jsou-li v systému již uloženy jiné filtry než vstupní, je v rolovacím možnost volby mezi nimi.
2. Tlačítko Apply selected filter
    - Slouží k aplikaci filtru vybraného v rolovacím menu.
    - Pokud je tlačítko stisknuto bez výběru filtru, je načtena hlavní stránka.
3. Tlačítko Delete selected filter
    - Slouží k vymazání filtru vybraného v rolovacím menu.
    - Pokud je tlačítko stisknuto bez výběru filtru, je načtena hlavní stránka.
4. Tlačítko Edit input filter
    - Slouží k editaci vstupního filtru (viz bod 6).
5. Tlačítko Go to homepage
    - Tlačítko slouží pro návrat na hlavní stránku.

## 8.3 Tabulka výpisů

Tabulka výpisů je zobrazována vždy, když se vypisuje nějaká množina logů z databáze. Díky použití dataTables si můžeme výsledky srovnat dle libovolného parametru kliknutím na daný sloupec.

1. Show entries
   - Slouží k vyhledávání v množině vypsaných výsledků.
2. Quick search
   - Slouží k vyhledávání v množině vypsaných výsledků.
3. Rolovací menu Detail
   - Nabízí možnost volby co s daným záznamem chceme provést.
4. Tlačítko save
   - Slouží k provedení akce zvolené v rolovacím menu Detail.

## 8.4 Edit input filter

Stránka `Edit input filter` slouží k administraci uložených filtrů, obsahující svým obsahem množinu dat, které jsou po nás jistým způsobem nepotřebné, běžné. Tabulka obsahuje sloupce:

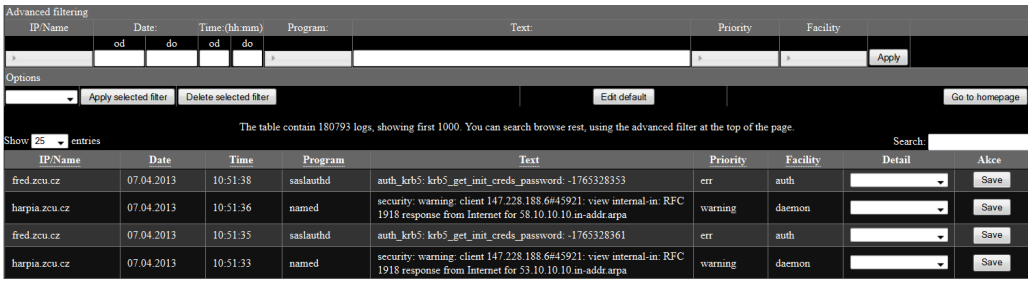
1. Created
   - Datum a čas vytvoření záznamu.
2. Creator
   - Přihlašovací jméno uživatele, který tento záznam přidal.
3. IP
   - Seznam IP adres, pro které je zadaný záznam platný.

- Mnohonásobný záznam se dá vytvořit ve tvaru: 'záznam1', 'záznam2' ...
- Při ukázání myší je vypsána nápověda pro vkládání.

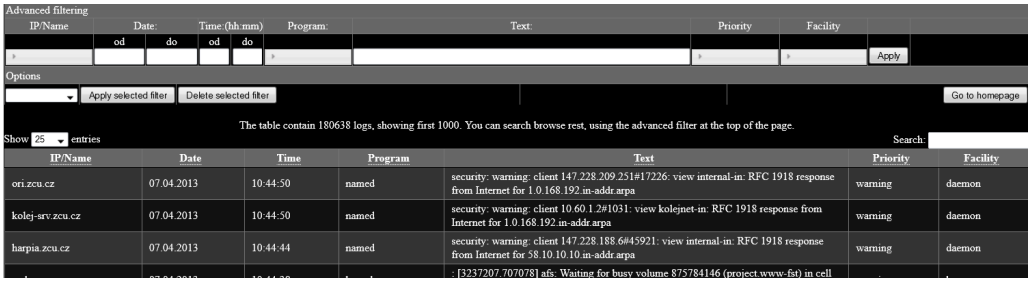
4. Program
   - Seznam programů, pro které je zadaný seznam platný.
   - Mnohonásobný záznam se dá vytvořit ve tvaru: 'záznam1', 'záznam2' ...
   - Při ukázání myší je vypsána nápověda pro vkládání.
5. Text
   - Text zprávy, popřípadě MySQL REGEXP (viz bod 7).
6. Priority
   - Při ukázání myší je vypsána nápověda pro vkládání.
   - Seznam priorit, pro které je zadaný filtr určen.
   - Možnost mnohonásobného záznam se dá vytvořit ve tvaru, kdy každá položka je obklopena apostrofy. Jednotlivé položky jsou odděléné čárkami.
7. Facility
   - Seznam kategorií, pro které je zadaný seznam platný.
   - Mnohonásobný záznam se dá vytvořit ve tvaru: 'záznam1', 'záznam2' ...
   - Při ukázání myší je vypsána nápověda pro vkládání.
8. Tlačítko save
   - Tlačítko sloužící k uložení změn provedených v záznamu.
   - Pokud není dodržen správný zápis, je vypsána chybová hláška MySQL odkazující na příslušné políčko, viz modul `delete.php`.
9. Tlačítko delete
   - Tlačítko sloužící k odstranění záznamu.
10. Add new row
    - Tlačítko sloužící k přidání prázdného záznamu.

## 8.5 Oprávnění a přístup

Jelikož mají projekt Swatcher využívat spíše zaškolení pracovníci (nikoliv sám administrátor), byly vytvořeny 2 druhy oprávnění. První, administrátorské (viz obr. 8.2), které má všechny výše zmíněné funkce a druhé, uživatelské (viz obr. 8.3), kde jsou odstraněny veškeré možnosti pro mazání příchozích logů a stejně tak je odebrán přístup do úpravy vstupního filtru. Nastavení oprávnění se provádí přes modul `config.php`, ve kterém se nachází proměnná `PrivUsers`. Přidáním uživatelského jména do této proměnné se definuje administrátorský účet.




Obrázek 8.2: Administrátorské rozhraní




Obrázek 8.3: Uživatelské rozhraní

# 9 Moduly

## 9.1 addrow.php

Modul slouží k přidání nového vstupního filtru při stisku `Add new row` na stránce editace vstupních filtrů.

## 9.2 applydef.php

Modul sloužící k aplikaci všech vstupních filtrů. Využíván pro promazání tabulky logů při ukládání nebo editaci pravidla vstupního filtru. Je tedy volán pouze po použití tlačítka `Save` v oblasti tabulkového výpisu a úpravě vstupního filtru. Modul tedy nejprve zjistí id `Input filtru` v tabulce `Filters`, hledá všechny záznamy v `Primitive_in_filter` kde je dané id obsaženo a nakonec skládá MySQL dotaz pro vymazání obsahu vycházejícího z každého nalezeného záznamu.

## 9.3 config.php

Modul, ve kterém jsou uloženy veškeré konfigurační údaje jako například přihlašovací jméno do databáze, názvy tabulek a samotné připojení k databázi. Tento modul je tedy nejzávažnější bezpečností mezera v systému a nesmí tedy být odcizen / stažen. Je přiložen ve všech ostatních modulech, kde je potřeba přístupu k databázi, popř. autentizace uživatele. Obsahuje důležité proměnné: `$privUsers` (pole uživatelů, kteří jsou v systému administrátory), `$timestamp` (proměnná pro určení maximálního času spojování kernel logů), `$printNormal` (proměnná pro určení počtu výpisů) a `$printMax` (proměnná pro určení počtu výpisu při filtrování).

## 9.4 input.php

Modul je tabulka složená z více HTML textarea. Každý řádek odpovídá jednomu vstupnímu filtru, každý sloupec pak tedy určitému sloupci v téže tabulce.

## 9.5 delete.php

Modul je volán modulem `input.php` v případě uživatelova kliknutí buď na tlačítko delete nebo save. Modul tedy podle stlačeného tlačítka buď odstraní záznam z tabulek `Primitive_in_filter` a `Primitive_filters` nebo po jednom textovém poli zkusí select z databáze - pro ověření validnosti zadaných dat, tj. jestli jsou v pořádku dle MySQL jazyka, např:

```
$Test = "SELECT * FROM Logs WHERE ";
$Sql2 = "Insert into Primitive_filters ( datetime, user, ";
$Msg2 = " VALUES ('".$_GET['date']."', '".$_GET['host']."', ";
$array = array("id_host", "program", "msg", "priority", "facility");
$i = 0;
foreach ($array as $pole){
   if(!empty($_GET[""].$i.""])){
      if($pole != "msg"){
        $mess = str_replace("'","\'",$_GET[""].$i.""]);
        $fin = str_replace('"','\"',$mess);
        $Sql2 .= "".$pole.",";
        $Msg2 .= "'".$fin."',";
        $count++;
        $Sql = MySQL_Query("".$Test."".$pole." IN(".$_GET[""].$i.""].");");
        if (!$Sql) {
            die("Syntax error - ".$pole.": " . mysql_error());
        }
      }
....
...
```

## 9.6 deletefil.php

Modul slouží k vymazání uživatelského filtru, který se zvolí v rolovacím menu v tabulce Options. Dle id, které je v tabulce Filters unikátní provede vymazání záznamu.

## 9.7 form.php

Modul slouží k vyhodnocení tabulky `Advanced filtering`, z vyplněných polí se složí SQL dotaz, který je následně vypsán modulem `table.php`, viz níže. Zároveň slouží na uložení vyplněných hodnot do proměnných pojmenovaných dle obsahu, tj. např: `$msg`, `$program` a další, které jsou pak využity v níže vkládaných modulech.

## 9.8 main.php

Obsahuje zdrojový kód html stránky doplněný o funkce jazyka PHP. Dle stisknutého tlačítka prování vkládání jednotlivých modulů, které se načtou na hlavní stránce.

```
if( isset($_GET['Submit'])){
...
else if( isset($_GET['Submit3'])){
   include("selected.php");
}
else if( isset($_GET["Editdef"])){
   include(input.php");
}
else if( isset($_GET['Submit33'])){
   include("deletefil.php");
}
```

## 9.9 print.php

Modul sloužící jako univerzální výpis, obsahuje výběr všech dat z tabulky `Logs` a zároveň je tento výběr limitována na 1000 výsledků z důvodu omezení velikosti základního výpisu (z důvodu efektivity a rychlosti dataTables, možnost nastavení pomocí proměnné v modulu `config.php`).

## 9.10 save.php

Modul slouží k uložení filtru v kontextovém menu. Modul je ve své podstatě pouze vkládání do tabulky `Filters`, které se provede po stisknutí tlačítka `Apply` pokud je vyplněný alespoň jeden sloupec v tabulce Advanced filtering a dále jméno filtru v kontextovém menu.

## 9.11 save2.php

Modul sloužící k uložení speciálních proměnných z modulu formular.php (`$program`, `$msg` atd.) do tabulky `Primitive_filters`. Modul je volán vždy v souvislosti s modulem `formular.php` a slouží k uložení zvolených kriterií jako vstupní filtr.

## 9.12 selected.php

Modul sloužící aplikování uživatelem vybraného filtru. Vybere tedy z tabulky `Filters` odpovídající id a díky záznamu v relační tabulce vypíše a spojí MySQL dotaz z záznamů v tabulce `Primitive_filters`.

## 9.13 table.php

Modul sloužící k vypisování veškerých dat, které jsou mu předány v proměnné `$final`.

## 9.14 tablsubm.php

Modul sloužící k zpracování formuláře vyplněného na stránce výpisu, odeslaného tlačítkem `Save`. Modul projde všechny pole, v poli `TEXT` vynuluje všechny řídící znaky(viz bod 7).

## 9.15 timestamp.php

Modul sloužící ke spojování kernel logů z tabulky `Logs`, jak již bylo zmíněno v kapitole Kernel logy najde logy se stejným zdrojem a pokud odpovídají dle UNIX timestampu je sloučí. Modul je volán z crontabu systému každou minutu.

```
while ($data = MySQL_Fetch_Array($Sql)){
   $pos = strpos($data[0], '[');
   $end = strpos($data[0], ']');
   $rest = substr($data[0], $pos+1, $end-3);
   if(is_numeric($rest)){
      $pos2 = strpos($rest, '.');
      $first = substr($rest, 0, $pos2);
      $second = substr($rest, $pos2+1, strlen($rest));
      if ($i != 0){
         if(($tmps - $first) == 0 && ($tmpid == $data[2]) && ($tmpm  -
         $second) <= $set_me){
            $fnl = substr($data[0], $end+1, strlen($data[0]));
            if($tmp == 0){
               $tmpmsg .= ": [".$rest."]";
            }
            $tmpmsg .= $fnl;
            $tmpmsg .= "\n";
            if($tmp != 0){
                MySQL_Query("DELETE FROM Logs where id = '".$tmpidu."'");
   ...
   ..
```

# 10 Závěr

Primárním cílem této bakalářské práce bylo vytvořit systém filtrování logů, generovaných v počítačové síti Západočeské univerzity. Úkolem bylo vytvořit uživatelské prostředí, ve kterém si sám uživatel může zvolit které druhy záznamů jsou běžné a které ne.

Vznikl systém, který běží na operačním systému Debian Wheezy a k dispozici má 1GB operační paměti. Webové rozhraní je napsáno jazykem HTML s využitím JavaScriptu a jeho knihovnou jQuery. Manipulaci s MySQL zajišťují PHP scripty. Pokud je v systému uloženo do 100 000 logů, webové rozhraní běží plynule, reakce jsou v zanedbatelných časech. Se zvyšujícím se počtem záznamů se zvyšuje doba odpovědí, což je ale i dáno hardwarovou konfigurací. Systém jako takový nevykazuje extrémní zátěže.

Během měření, které proběhlo dne 25.4., proběhlo smazání obsahu všech tabulek v MySQL a spuštění Syslog-ng bez zadaných vstupních filtrů. Za 180 minut se do systému uložilo 2 534 300 logů ze 341 zdrojů. Po přidání několika základních vstupních filtrů, se počet logů v databázi zkrátil na 23 723. Tento počet by se měl dlouhodobě po nastavení většího množství vstupních filtrů držet v řádu desítek.

Projekt by šel rozšířit například o propojení se systémem Nagios, popřípadě by šel upravit na jednoduchý eventviewer (muselo by se ale upravit vypisování výsledků, jelikož jQuery plugin dataTables není vhodný pro velké objemy dat). Dalším vhodným rozšířením by bylo například přidání administrace uživatelských účtů (zvolit vhodnější řešení než mít jména vyjmenované v PHP poli, například vytvořit v MySQL další tabulku s názvem Users).

Systém byl následně dne 26.4. nasazen do běžného provozu, čímž byly splněny cíle této bakalářské práce.

# Literatura


[1] *Evi Nemeth, Garth Snyder, Trent R. Hein:* **Linux - Kompletní příručka administrátora**
Computer Press, Brno 2004. ISBN 80-722-6919-4

[2] *Mark Maslakowski:* **Naučte se MySQL za 21 dní**
Computer Press, Praha 2001. ISBN 80-7226-448-6

[3] *Michal Brandejs:* **UNIX - LINUX - Praktický průvodce**
GRADA Publishing, Praha 1996. ISBN 80-7169-170-4

[4] *Musciano Kennedy:* **HTML & XHTML - Kompletní průvodce**
Computer Press, Praha 2000. ISBN 80-7226-407-9

[5] *Jaroslav Pokorný:* **Dotazovací jazyky**
Univerzita Karlova v Praze, 1994. ISBN 80-901475-2-6

[6] *Karel Žák:* **Historie relačních databází**
http://www.root.cz/clanky/historie-relacnich-databazi/

[7] *Petr Krčmář:* **Historie operačního systému GNU/Linux**
http://www.root.cz/texty/historie-operacniho-systemu-gnulinux/

[8] **Apache Web Server**
http://www.root.cz/specialy/linux-na-serveru/webserver/

[9] *CIV, ZČU:* **WebAuth**
http://support.zcu.cz/index.php/LPS:WebAuth

[10] *Adam Štrauch:* **Lighttpd: lehký webserver**
http://www.root.cz/clanky/lighttpd-lehky-webserver/

[11] *experti komunity jQuery:* **jQuery - Kuchařka programátora**
Computer Press, a.s Brno, 2010. ISBN 978-80-251-3152-7

[12] *Ian Gilfillan:* **Knihovna programátora - myslíme v MySQL 4**
GRADA Publishing, Praha 2003. ISBN 80-247-0661-X

[13] *Jiří Bráza:* **PHP 5 - Začínáme programovat**
GRADA Publishing 2005. ISBN 80-247-1146-X.

[14] *Vladimír Pošmura:* **Apache: Příručka správce WWW serveru**
Computer Press, 2002. ISBN: 80-7226-696-9

[15] *David Flanagan:* **JavaScript - Kompletní průvodce**
ComputerPress, Praha 2002. ISBN 80-7226-626-8

[16] *R. Gerhards:* **RFC 5424. The Syslog Protocol**
http://www.ietf.org/rfc/rfc5424.txt

[17] *Karen Kent, Murugiah Souppaya :* **Recommendations of the National Institute of Standards and Technology**
Gaithersburg : NIST , 2006

[18] **Syslog-ng**
http://www.balabit.com/network-security/syslog-ng

[19] **Logrotate**
http://linuxcommand.org/man_pages/logrotate8.html

[20] *David Procházka:* **PHP 6**
GRADA publishing, Praha 2012. ISBN 978-80-247-3899-4

# 11 Přílohy

# Uživatelská příručka

# Swatcher

Plzeň 2013 Pavel Hvězda

## A.1 O projektu

Tato práce je výsledkem projektu, který mi byl zadán mým vedoucím, Michalem Švambergem. Cílem bylo vytvořit jednoduchý a účinný filtrovací systém pro potřeby Západočeské univerzity pro správu logů a odhalování disfunkcí systému. Je určen pro různé distribuce systému Linux. Projekt by tedy měl výrazně ulehčit kontrolu událostí a čtení logů, protože bude shromažďovat důležité informace a data (logy) o celé počítačové síti na jednom místě a bude umožňovat rychlé a efektivní filtrování a vyhledávání s přehledným webovým rozhraním. Také bude umožňovat filtrovat, které příchozí logy nechceme do systému zaznamenat trvale. Účelem tedy nebude shromažďovat veškeré logy a nahrazovat funkci event vieweru (nicméně, k této funkci může také velice dobře sloužit). Bude sloužit především jako nástroj, který zaškolený a znalý pracovník nastaví a méně zkušený kolega nebo externista využije ke kontrole a vyhodnocení logů (událostí), které byly do systému uloženy.

## B.1 Filtry

Systém filtrů v systému je rozdělen do dvou kategorií: vstupní filtry a uživatelské filtry.

## B.2 Vstupní filtry

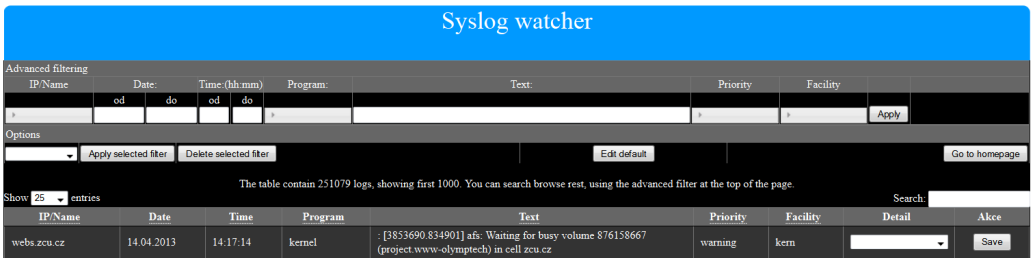
Vstupní filtry v systému představují množinu událostí, které nechceme zaznamenávat. V systému jako takovém reprezentuje tuto množinu pravidel pouze jeden prvek a to je filtr `Input`. Editovat tato pravidla můžeme na stránce, která se načte po zmáčknutí tlačítka `Edit input filter`, které je dostupné pouze pro administrátory (viz bod 8.5). Systém je nastaven tak že všechna pravidla uložená do vstupního filtru jsou aplikována jako množina výsledků, které nechceme do systému zaznamenávat.

## B.3 Uživatelské filtry

Uživatelské filtry jsou množinou filtrů, které si navolil sám uživatel ve webovém rozhraní. Slouží především pro usnadnění manipulace uživatele se systémem, aby nemusel stále v případě potřeby vyplňovat stejné údaje. Veškeré uživatelské filtry vkládané do systému musí mít unikátní název a nesmí se jmenovat `input`, což je i systémem kontrolováno. Takto uložené filtry si může uživatel aplikovat zvolením filtru v tabulce `Options` a stisknutím tlačítka `Apply selected filter` (viz bod 8.2). Tyto filtry slouží k prohlížení logů, které nebyly odfiltrovány „Input filtrem“.

# C.1 Webové rozhraní

Webové rozhraní je přizpůsobeno především přehlednosti, s možností efektivního pokročilého filtrování logů. Vzhled stránek je rozdělen (viz obr. C.1) na několik částí: Logo, Advanced filtering, Options a Results a v některých případech, které budou níže specifikované se zobrazí speciální kontextové menu.




Obrázek C.1: Webové rozhraní

## C.2 Advanced filtering

Tabulka Advanced filtering slouží k vyhledávání přes celou tabulku logů uložených v systému. Nabízí možnost filtrovat logy podle:

1. IP/Name
    - slouží k vyhledávání logujících strojů, pokud se nezdaří překlad IP adresy přez DNS na jméno, je zobrazena IP adresa,

- možnost vícenásobného výběru,
- řazeno abecedně.

2. Date
    - slouží k vyhledávání logů k určitému datu,
    - možnost filtrovat *od-do* určitého datumu,
    - pro snažší výběr datumu je použita freeware knihovna datepicker, která vyvolá kalendář s možností procházet měsíce, dny a roky,
    - pokud je vyplněno pouze pole *od*, systém automaticky vypíše všechny logy od námi zadaného data až do aktuálního, pokud je vyplněno pouze pole *do*, systém vypíše veškeré logy z databáze až do aktuálního data.
3. Time
    - slouží k vyhledávání podle času,
    - možnost filtrovat *od-do* určitého času,
    - čas musí být zadán ve formátu HH:MM, jinak bude vypsána výstražná hláška o neplatnosti zadání a pole vyčištěno,
    - pokud je vyplněno pouze pole *od*, systém automaticky vypíše veškeré logy zadaného až do aktuálního času, pokud je vyplněno pouze pole *do*, systém vypíše veškeré logy z databáze až do aktuálního času.
4. Program
    - slouží k vyhledávání podle programu, který daný log generoval, dle RFC 5424 normy se jedná o procID,
    - možnost vícenásobného výběru.
5. Text
    - slouží k vyhledávání podle textu zprávy, dle RFC 5424 normy se jedná o MSG,
    - možnost vyhledávání podle hesel, nebo po sobě jdoucích sekvencí slov, které odpovídají vyhledávanému textu. Text je pak interpretován jako regulární výraz (viz bod 7).

6. Priority
    - slouží k vyhledávání podle priority - závažnosti události, dle RFC 5424 normy se jedná o SEVERITY,
    - možnost vícenásobného výběru,
    - řazeno abecedně.
7. Facility
    - slouží k vyhledávání podle facility - druhi události, dle RFC 5424 normy se jedná o FACILITY,
    - možnost vícenásobného výběru,
    - řazeno abecedně.
8. Tlačítko Apply
    - tlačítko sloužící k odeslání formuláře a vyhodnocení vybraných hodnot.

## C.3 Options

Tabulka Options slouží k manipulaci buď již s uloženými filtry, nebo k editaci filtru vstupního.

1. Rolovací menu
    - jsou-li v systému již uloženy jiné filtry než vstupní, je v rolovacím možnost volby mezi nimi.
2. Tlačítko Apply selected filter
    - slouží k aplikaci filtru vybraného v rolovacím menu,
    - pokud je tlačítko stisknuto bez výběru filtru, je načtena hlavní stránka.
3. Tlačítko Delete selected filter
    - slouží k vymazání filtru vybraného v rolovacím menu,
    - pokud je tlačítko stisknuto bez výběru filtru, je načtena hlavní stránka.

4. Tlačítko Edit input filter
    - slouží k editaci vstupního filtru (viz bod 6).
5. Tlačítko Go to homepage
    - tlačítko slouží pro návrat na hlavní stránku.

## C.4 Tabulka výpisů

Tabulka výpisů je zobrazována vždy, když se vypisuje nějaká množina logů z databáze. Díky použití pluginu JQuery DataTables si můžeme výsledky srovnat dle libovolného parametru kliknutím na daný sloupec.

1. Show entries
    - slouží k vyhledávání v množině vypsaných výsledků.
2. Quick search
    - slouží k vyhledávání v množině vypsaných výsledků.
3. Rolovací menu Detail
    - nabízí možnost volby co s daným záznamem chceme provést.
4. Tlačítko save
    - slouží k provedení akce zvolené v rolovacím menu Detail.

## C.5 Edit input filter

Tabulka `Edit input filter` slouží k administraci uložených filtrů, obsahující svým obsahem množinu dat, které jsou po nás jistým způsobem nepotřebné, běžné. Tabulka obsahuje sloupce:

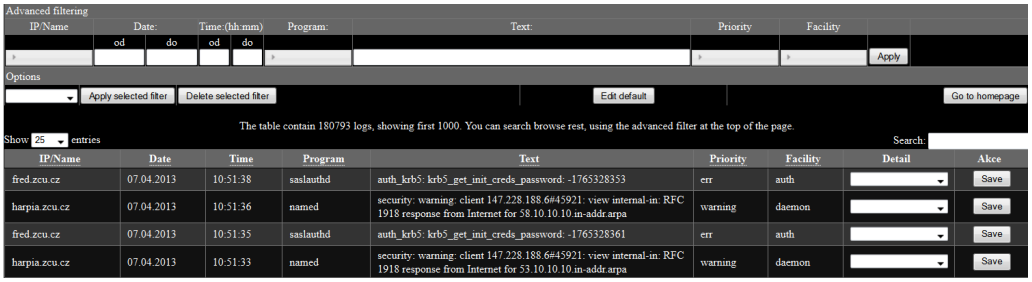
1. Created
    - datum a čas vytvoření záznamu.

2. Creator
    - přihlašovací jméno uživatele, který tento záznam přidal.
3. IP
    - seznam IP adres, pro které je zadaný záznam platný,
    - mnohonásobný záznam se dá vytvořit ve tvaru: 'záznam1', 'záznam2' ...,
    - při ukázání myší je vypsána nápověda pro vkládání.
4. Program
    - seznam programů, pro které je zadaný seznam platný,
    - mnohonásobný záznam se dá vytvořit ve tvaru: 'záznam1', 'záznam2' ...,
    - při ukázání myší je vypsána nápověda pro vkládání.
5. Text
    - text zprávy, popřípadě MySQL REGEXP (viz bod 7).
6. Priority
    - při ukázání myší je vypsána nápověda pro vkládání,
    - seznam priorit, pro které je zadaný filtr určen,
    - možnost mnohonásobného záznam se dá vytvořit ve tvaru, kdy každá položka je obklopena apostrofy. Jednotlivé položky jsou oddělené čárkami.
7. Facility
    - seznam kategorií, pro které je zadaný seznam platný,
    - mnohonásobný záznam se dá vytvořit ve tvaru: 'záznam1', 'záznam2' ...,
    - při ukázání myší je vypsána nápověda pro vkládání.
8. Tlačítko save
    - tlačítko sloužící k uložení změn provedených v záznamu,
    - pokud není dodržen správný zápis, je vypsána chybová hláška MySQL odkazující na příslušné políčko, viz modul `delete.php`.

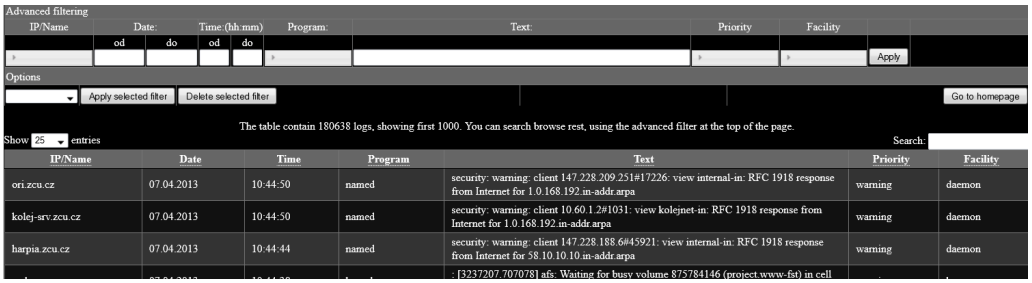
9. Tlačítko delete
    - tlačítko sloužící k odstranění záznamu.
10. Add new row
    - tlačítko sloužící k přidání prázdného záznamu.

## D.1 Oprávnění a přístup

Jelikož mají systém využívat spíše zaškolení pracovníci (nikoliv sám administrátor), byly vytvořeny 2 druhy oprávnění. První, administrátorské (viz obr. D.2), které má všechny výše zmíněné funkce a druhé, uživatelské (viz obr. D.3), kde jsou odstraněny veškeré možnosti pro mazání příchozích logů a stejně tak je odebrán přístup do úpravy vstupního filtru. Nastavení oprávnění se provádí přes modul `config.php`, ve kterém se nachází proměnná `PrivUsers`. Přidáním uživatelského jména do této proměnné se definuje administrátorský účet.




Obrázek D.2: Administrátorské rozhraní




Obrázek D.3: Uživatelské rozhraní