Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Webové rozhraní k VOIP hovorům

Viktor Robejšek

Vedoucí práce: Mgr. Jan Starý, Ph.D.

25. června 2015

Poděkování

Děkuji Mgr. Janovi Starému, Ph.D. za rady a vedení práce. Dále děkuji rodině a také všem, co mě podporovali.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 25. června 2015 .....

Odkaz na tuto práci

Robejšek, Viktor. *Webové rozhraní k VOIP hovorům*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Tato bakalářská práce pojednává o části projektu zaměřeného na zachytávání a zpracování hovorů vedených pomocí internetové telefonie. Konkrétně pojednává o databázi PostgreSQL a webovém rozhraní psaném pomocí jazyka PHP, a to o návrhu, implementaci i testování. V úvodní kapitole je popsán projekt jako celek, i jeho jednotlivé části. V návrhu jsou popsány požadavky a případy užití. Kapitola implementace popisuje specifické databázové funkce i funkce rozhraní a výsledné databázové schéma. Také popisuje bezpečnostní prvky aplikace. Testování popisuje kompatibilitu a dokazuje funkčnost kódu. Výsledkem práce jsou zdrojové kódy webového rozhraní a skript vytvářející databázové tabulky a funkce.

Klíčová slova VoIP telefonie, Webové uživatelské rozhraní, PHP, databáze PostgreSQL, SIP.

Abstract

This bachelor thesis discusses part of a project aimed at capturing and processing audio calls obtained with Internet telephony. Specifically, it deals with the design, implementation and testing of a PostgreSQL database and a PHP web user interface. The opening chapter describes the project as a whole and

its individual parts. The design chapter describes the database and interface requirements and the use cases. The implementation chapter describes specific database and interface features and functions, the resulting database schema, and security features. The testing chapter describes compatibility and demonstrates the functionality of the code. The results of this thesis are source codes of the web interface and a script creating the database tables and functions.

Keywords VoIP telephony, Web user interface, PHP, PostgreSQL database, SIP.

Obsah

Seznam obrázků

Úvod

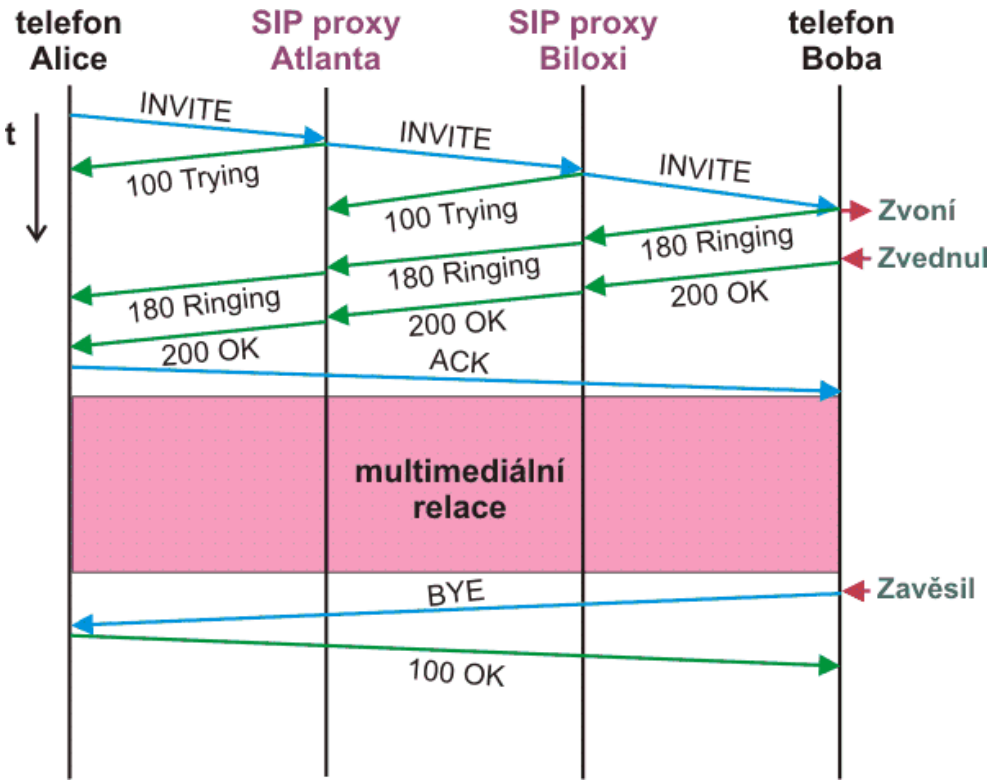
Voice over Internet Protocol, zkratkou VoIP, je sada technologií, které umožňují přenos digitalizovaného zvuku, v tomto případě hlasu, těly paketů UDP[1] nebo TCP/IP[2], přenášených prostřednictvím počítačové sítě, případně jiného média, na kterém lze využít protokol IP. Využívá se tedy k telefonování prostřednictvím takovéhoto datového spojení. Samotný průběh hovoru je pak definován různými standardy, např. H.323 nebo SIP[3]. Typicky pak hovor probíhá tak, že se mezi oběma stranami naváže spojení hovoru, a poté se zvuk přenáší pomocí paketů UDP, kde každý paket obsahuje data protokolu RTP[4], ve kterých se nachází část hlasových informací, typicky 20-30 ms části hovoru. Tyto hlasové informace jsou zakódovány podle určitých kodeků, např. G.711 nebo G.729 [2].

Síťové pakety můžeme odchytit a posléze přečíst. Pokud nejsou data, která pakety přenášely, zašifrována, lze hovor opět dekódovat bez toho, aby o tom věděla jakákoliv ze stran účastnících se hovoru.

Projekt, jehož se účastním, je zaměřen na zachytávání a zpracovávání hovorů vedených pomocí SIP. Hovor se podle SIP naváže tak, že volající pošle volané straně požadavek na hovor. Druhá strana pak zpátky posílá řídící zprávy, oznamující, že telefon vyzvání, a poté, že byl hovor přijat, nebo odmítnut. Poté, co se spojení úspěšně naváže, probíhá již další komunikace pomocí zmíněných UDP paketů, nesoucích data RTP. Po ukončení hovoru se opět pošlou řídící zprávy určující, že byl hovor ukončen [3].

Příklad průběhu SIP hovoru viz. obrázek 0.1. Na obrázku je menší chyba, a to sice popis posledního kódu 100 OK. Jedná se samozřejmě o kód 200 OK.

[1] User Datagram Protocol
[2] Transmission Control Protocol/Internet Protocol
[3] Session Initiation Protocol
[4] Real-time Transport Protocol



Obrázek 0.1: Ukázka průběhu hovoru při využití SIP [1]

Kapitola 1

Popis problému

1.1 Projekt

Cílem projektu jako celku je návrh a implementace sady programů, které budou odchytávat a zpracovávat data hovorů, které na síti probíhají. Práce na projektu je rozdělena do tří částí: zachycení a analýza paketů, extrakce hlasových informací, a zobrazení a správa dat. Jednotlivé části používají sdílenou databázi a dohromady tvoří funkční celek.

Cílem první části, zpracovávané Filipem Šusterem v [4], je vytvořit program, který bude v komunikaci v počítačové síti rozpoznávat a zaznamenávat pakety obsahující data hovoru. Důležité je zajistit jak pakety s řídícími daty, tak pakety s daty multimediálními. Po ukončení hovoru pak zachycené pakety analyzuje a ve vhodném formátu uloží pro další zpracování.

Druhou částí projektu, kterou zpracovává Josef Kučera v [5], je analýza a implementace programu, který ze zkompletovaných informací o hovoru dokáže vytvořit zvukový soubor, který lze později přehrát. K tomu potřebuje data správně seřadit a dekódovat. Musí také vyloučit případné duplikátní a redundantní části hovoru, které vznikají kvůli použití UDP v komunikaci, a vyplnit případná chybějící data.

Poslední částí, již mám na starosti já, je pak vytvoření a správa databáze, která se využívá pro předání dat mezi jednotlivými částmi, a je tedy pro všechny společná. Kromě databáze mám na starosti také návrh a vytvoření webového uživatelského rozhraní, které bude umožňovat výpis a přehrávání hovorů, sledování statistik, a sledování a řízení chování programů zpracovaných v ostatních částech projektu.

1.2 Využití

Projekt je zaměřen na firmy, případně instituce, které hojně využívají VoIP. Systém jim umožní sledovat, který zaměstnanec kam a jak dlouho volá a

po extrakci i obsah telefonátu. Některé firmy mají ze zákona povinnost tyto záznamy pořizovat a uchovávat. Podle konfigurace lze odposlouchávat i na menších částech sítě, případně jen vybraná telefonní čísla nebo jen v daný den či čas.

Před odposlechem je ale podle § 12, odst. 1 ObčZ nutné získat souhlas odposlouchávaného, případně na něj musí být nejprve uvaleno trestní řízení (viz. zákon č. 141/1961 Sb.).

1.3 Současný stav řešení problematiky

V současnosti lze sledovat VoIP hovory v síti několika způsoby.

1.3.1 Ruční zachytávání a analýza

Existuje několik programů na zachytávání paketů putujících sítí, např. tcpdump, který k zachytávání využívá knihovny libpcap. Do těchto paketů lze poté jednotlivě nahlížet, a vyčíst z nich data o spojení. Z paketů lze pomocí dalších nástrojů vyextrahovat hlasová data, a ta pak dekódovat a přehrát. Toto řešení je ale značně složité a náročné.

1.3.2 Wireshark

Dalším nástrojem využívající knihovnu libpcap je Wireshark. Je to sofistikovaný nástroj pro analýzu síťových paketů. Dokáže pakety zachytávat, a zobrazit data v nich ukrytá. V kontextu VoIP dokáže zjistit informace o proběhlých H.323, ISUP[5], SIP, MGCP[6] a UNISTIM[7] hovorech, vykreslit grafy jejich průběhů, a některé hovory i přehrát. Dokáže ale dekódovat hovory jen v několika kodecích.[6]

1.3.3 HOMER

Systém HOMER je komplexní řešení pro zachytávání a monitorování SIP hovorů. Umožňuje hovory analyzovat a poté v nich vyhledávat, filtrovat je, a zobrazovat grafy jejich průběhů a různé statistiky. Umožňuje také centralizovat data zachycená v různých částech sítě. Hovory ale nedokáže přímo přehrávat.[7]

1.3.4 Další nástroje

Dále existuje několik nástrojů, které umí zachytávat SIP pakety, případně z nich pak zjistit základní informace o hovoru. Neumí ale z paketů vytěžit hlasová data. Mezi tyto nástroje patří např. SIP Analyzer, SipSpy nebo ngrep.

[5] Integrated Services Digital Network User Part
[6] Media Gateway Control Protocol
[7] Unified Networks IP Stimulus

Kapitola 2

Analýza a návrh

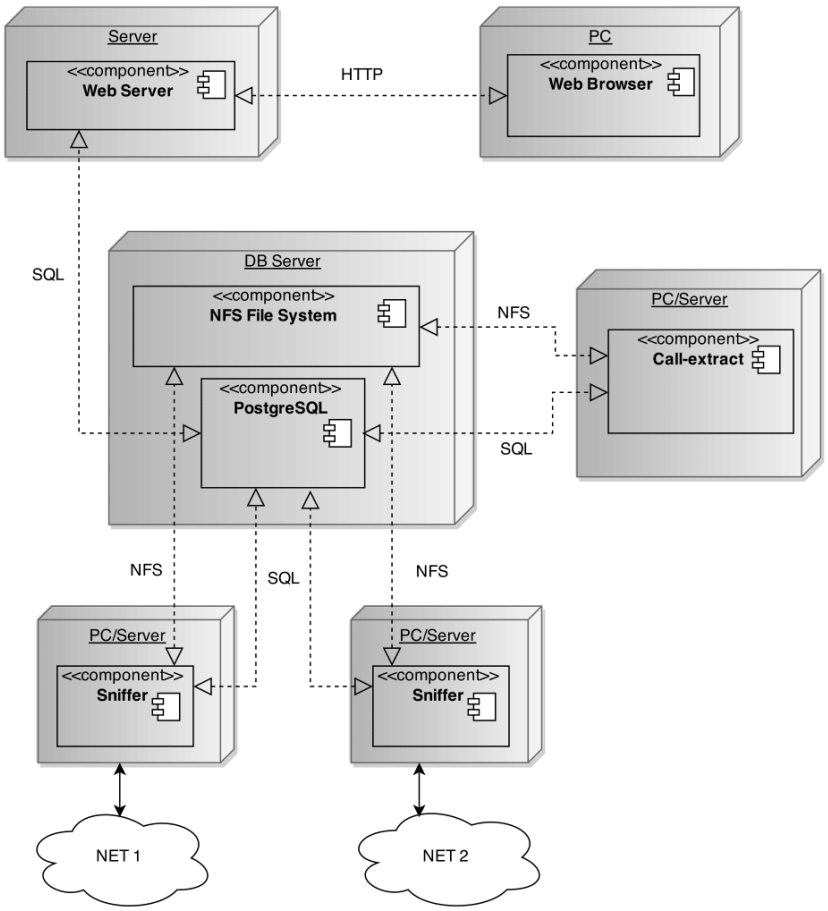
2.1 Propojení prvků projektu

Nutným prvkem propojení jednotlivých částí projektu je databáze. V databázi jsou uložena veškerá data o hovorech. Samotná data hovoru jsou pak uložena na souborovém systému, který nemusí být nutně součástí stejného serveru, na kterém je spuštěna databáze. V takovém případě se musí všude uvádět absolutní cesty.

Dalším prvkem je rekordér, zvaný též sniffer. Ten z databáze zjišťuje pravidla, která mu určují, jaké hovory má nahrávat a posléze do databáze zapsat. Rekordérů může být spuštěno několik najednou, typicky v různých částech sítě.

Další částí je dekodér, zvaný též extraktor, případně call-extract. Dekodér si z databáze bere nezpracované hovory a tvoří z nich přehratelné zvukové soubory, které pak zapíše zpátky. Dekodérů může být spuštěno také více najednou.

Nakonec nad databází běží webový server, který umožňuje data z databáze zobrazovat a upravovat je. Kromě informací o hovorech, rekordérech, pravidlech a dekodérech využívá databázi také ke správě uživatelů webu.



Obrázek 2.1: Diagram nasazení

2.2 Analýza požadavků na webové rozhraní

2.2.1 Funkční požadavky

- Registrace a přihlašování uživatelů
  - Vytvoření uživatelského účtu po registraci
  - Ověření správnosti přihlašovacích informací při přihlašování
- Správa uživatelů
  - Přidávání a odebírání práv uživatelům
- Správa rekordéru a dekodérů
  - Vytváření identifikátorů pro nové rekordéry a dekodéry
- Správa nahrávacích pravidel
  - Vytváření, úprava a mazání pravidel, kterými se rekordéry řídí
- Zobrazení hovorů
  - Zobrazení informací o nahraných hovorech
  - Filtrování a řazení hovorů
- Zobrazení statistik
  - Zobrazení statistik rekordérů, dekodérů a hovorů

2.2.2 Nefunkční požadavky

- Bezpečnost
  - Zabezpečení proti běžným typům útoků.
- Spolehlivost
  - Stabilita a rychlost i při vyšší zátěži
- Využití jazyka PHP
  Ke tvorbě webového rozhraní má být použit jazyk PHP[8] [8]. Tímto jazykem lze za pomoci funkcí a objektových tříd tvořit dynamické webové stránky a aplikace.

[8]PHP: Hypertext Preprocessor

2.3 Analýza požadavků na databázi

Ze zadání plyne nutnost použití databáze typu PostgreSQL [9]. Protože jsou databáze tohoto typu značně sofistikované, rozhodl jsem se toho využít co nejvíce. Databáze umí sama optimalizovat vnitřní dotazy podle frekvence jejich používání. Také má pokročilé možnosti co se týče tvorby funkcí. Veškerá komunikace, která mezi databází a jednotlivými částmi našeho systému probíhá, je tedy vedena za použití tzv. stored procedures, tedy mnou definovaných funkcí, ne pomocí obyčejných dotazů klasického SQL.

2.3.1 Funkční požadavky

- Správa rekordérů a dekodérů
  - Vytváření nových identifikátorů pro rekordéry a dekodéry.
  - Označování rekordérů a dekodérů za aktivní nebo neaktivní.
- Správa uživatelů
  - Vytváření účtů
  - Ověřování přihlašovacích dat
  - Správa práv jednotlivých uživatelů
- Vytváření hovorů a jejich editace
  - Vytváření hovorů rekordéry
  - Dospecifikace pomocí nových informací rekordéru po ukončení hovoru
  - Přidání dalších informací po extrakci hlasových dat hovoru dekodérem
- Správa pravidel a jejich předávání
  - Vytváření platných pravidel pro rekordéry
  - Předání pravidel jednotlivým rekordérům
  - Editace a mazání pravidel
- Předávání práce dekodérům
  - Předání žádaného počtu nezpracovaných hovorů dekodéru.
- Vytváření možných filtrů
  - Předání informací o použitelných filtrech pro výpis hovorů.
- Výpis hovorů
  - Výpis hovorů s volitelnými filtry a řazením.

2.3.2 Nefunkční požadavky

- Spolehlivost
  - Stabilita a rychlost i při vyšší zátěži
- Bezpečnost
  - Zabezpečení proti běžným typům útoků.

2.4 Případy užití

2.4.1 Případy užití webového rozhraní

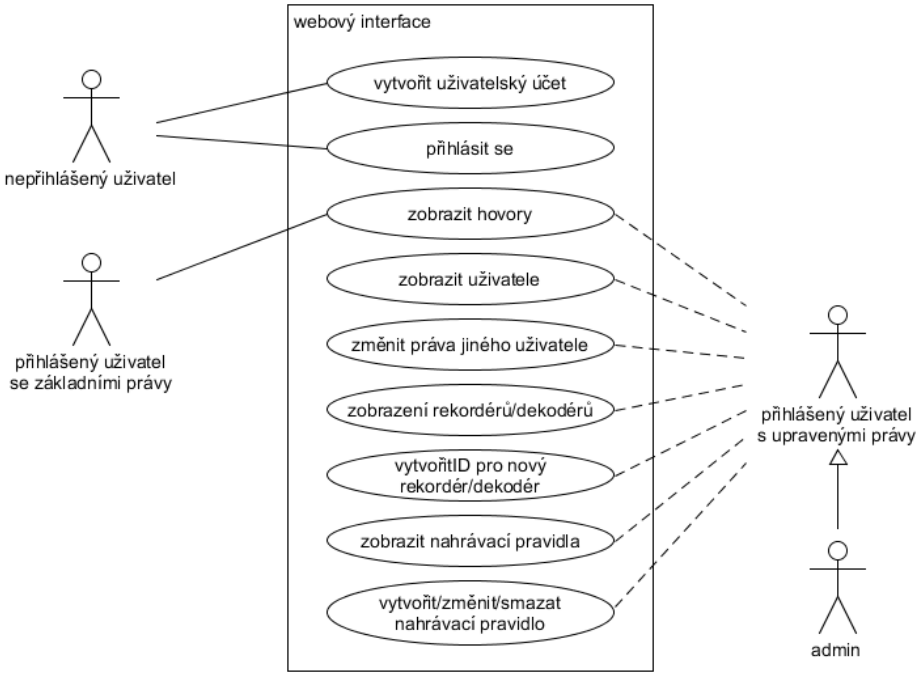
Aktéry webového rozhraní jsou uživatelé. Nepřihlášený uživatel se může přihlásit, nebo zaregistrovat. Po přihlášení jsou mu povoleny pouze akce, ke kterým má oprávnění. Nový uživatel si smí pouze prohlížet informace o hovoru a případně hovor přehrát. Každému uživateli lze ale přidělit nebo odebrat oprávnění, a to jmenovitě:

- Zobrazovat hovory
- Zobrazovat pravidla
- Zobrazovat uživatele
- Spravovat pravidla
- Spravovat oprávnění uživatelů
- Zobrazovat rekordéry a dekodéry
- Vytvářet identifikátory pro rekordéry a dekodéry

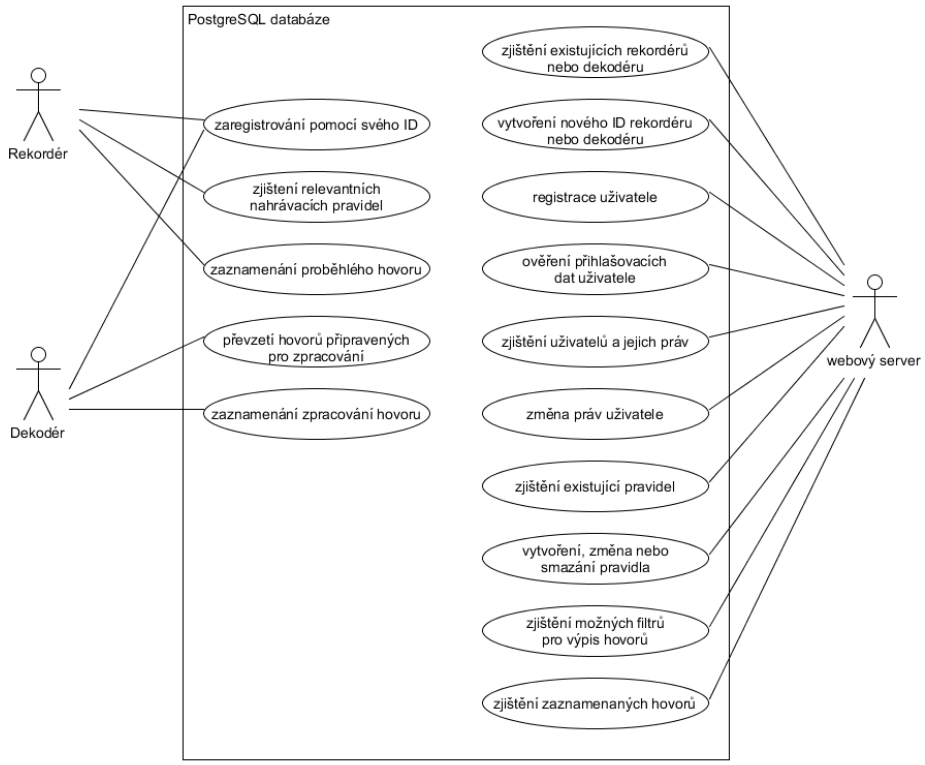
Speciálním případem uživatele je admin. Ten trvale vlastní oprávnění ke všem akcím.

2.4.2 Případy užití databáze

Do databáze přistupují rekordéry, dekodéry a uživatelé prostřednictvím webového rozhraní. Rekordéry a dekodéry se před prací v databázi zaregistrují. Rekordéry pak vytvářejí záznamy o hovorech a dekodéry tyto záznamy dále doplňují o informace o extrakci hlasových dat. Webové rozhraní pak používá databázové funkce pro všechny dostupné akce.



Obrázek 2.2: Diagram užití webového rozhraní



Obrázek 2.3: Diagram užití databáze

Kapitola 3

Implementace

3.1 Databáze

3.1.1 Konfigurace databáze

Z důvodu kompatibility kódu je nutné použít databázi PostgreSQL verze alespoň 9.2 nebo vyšší. Verze PostgreSQL 9.2 je potřeba kvůli využití rozsahových datových typů `int4range` v tabulce Rules.

Do souboru `pg_hba.conf` je nutné povolit přístup z IP adres všech rekordérů a dekodérů a IP adresy webového serveru. Další konfigurace databáze (soubor `postgresql.conf`) je standardní a měla by být přenechána serverovému administrátorovi. Při tomto nastavení databáze nepřijímá žádná spojení zvenčí, a je tedy bezpečná proti útokům zvenčí.

Na vytvořené databázi je nutno spustit soubor `db.sql` (součást této práce), který inicializuje tabulky a funkce.

3.1.2 Schéma databáze

Samotnou databázi tvoří několik tabulek:

- `Calls`
  Tabulka uchovávající informace o hovorech
  - `callid` - Unikátní identifikátor hovoru
  - `note` - Poznámka
  - `srcaddr, dstaddr` - IP adresy volajícího a volaného
  - `srcsip, dstsip` - SIP adresy volajícího a volaného
  - `srcnum, dstnum` - Telefonní čísla volajícího a volaného
  - `dumpfile` - Cesta k souboru se síťovými daty hovoru
  - `snifferr` - Chybová hláška psaná rekordérem při nahrávání tohoto hovoru

  - `invited, rejected, started, crashed, finished` - Časy vzniku a zániku hovoru
  - `calllength` - Délka hovoru
  - `idrecorder` - Identifikátor rekordéru, který hovor nahrál
  - `format` - Kodek, ve kterém byl hovor uskutečněn.
  - `totalpkts, lostpkts, ooopkts, latepkts (in/out)` - Počet paketů celkem, ztracených, out of order, pozdních
  - `callfile` - Cesta ke zvukovému souboru hovoru
  - `extracterr` - Chybová hláška psaná dekodérem při extrakci zvuku
  - `assigned, extracted` - Časy přidělení hovoru dekodéru a úspěšného dekódování
  - `timeassigned` - Čítač, kolikrát byl již hovor přidělen
  - `iddecoder` - Identifikátor dekodéru, který hovor zpracoval

  K zajištění správné funkčnosti musí být správně nakonfigurovány i rekordéry a dekodéry, a to kromě korektního napojení na databázi i dohoda, zda-li se budou do polí `dumpfile` a `callfile` zapisovat cesty absolutní, či relativní, případně vůči čemu relativní. Při špatném nastavení mohou v databázi vzniknout odkazy na soubory, které leží na jiné části souborového systému, nebo dokonce na úplně jiném stroji.

- `Formats`
  Tabulka kodeků hovorů, tedy jejich kódů a názvů. Informace jsou přejaty z [10].
  - `code` - RTP Payload type kodeku
  - `name` - Název kodeku

- `Rules`
  Tabulka uchovávající informace o pravidlech pro rekordéry. Inspirací formátu této tabulky je `crontab(5)`.
  - `id` - Identifikátor pravidla
  - `idrecorder` - Identifikátor rekordéru, kterému je pravidlo určeno. Pokud je vyplněno `null`, rozumí se, že je pravidlo určeno všem rekordérům.
  - `ord` - Priorita pravidla (nižší číslo znamená vyšší prioritu).
  - `yesno` - Určení, zda je pravidlo pozitivní, či negativní (nahrávej hovory splňující podmínky, nebo nenahrávej hovory splňující podmínky)
  - `srcnum` - Číslo volajícího

  - `dstnum` - Číslo volaného
  - `min, hour, day, month, wday` - Rozsah, v jakých minutách, hodinách, dnech, měsících a/nebo dnech v týdnu (ne)nahrávat

- `Recorders`
  Tabulka uchovávající informace o rekordérech a jejich statistiky.
  - `id` - Identifikátor rekordéru
  - `active` - Určuje, zda je rekordér právě aktivní (zda je zaregistrován)
  - `invited, started, rejected, crashed, finished` - Počet pozvaných, začatých, odmítnutých, přerušených a dokončených hovorů, které rekorér zapsal do databáze.

- `Decoders`
  Tabulka uchovávající informace o dekodérech a jejich statistiky.
  - `id` - Identifikátor dekodéru
  - `active` - Určuje, zda je dekodér právě aktivní (zda je zaregistrován)
  - `assigned, extracted, exwitherror` - Počet, kolik hovorů bylo dekodéru přiděleno, kolik dekodér extrahoval a kolik extrahoval s chybou.

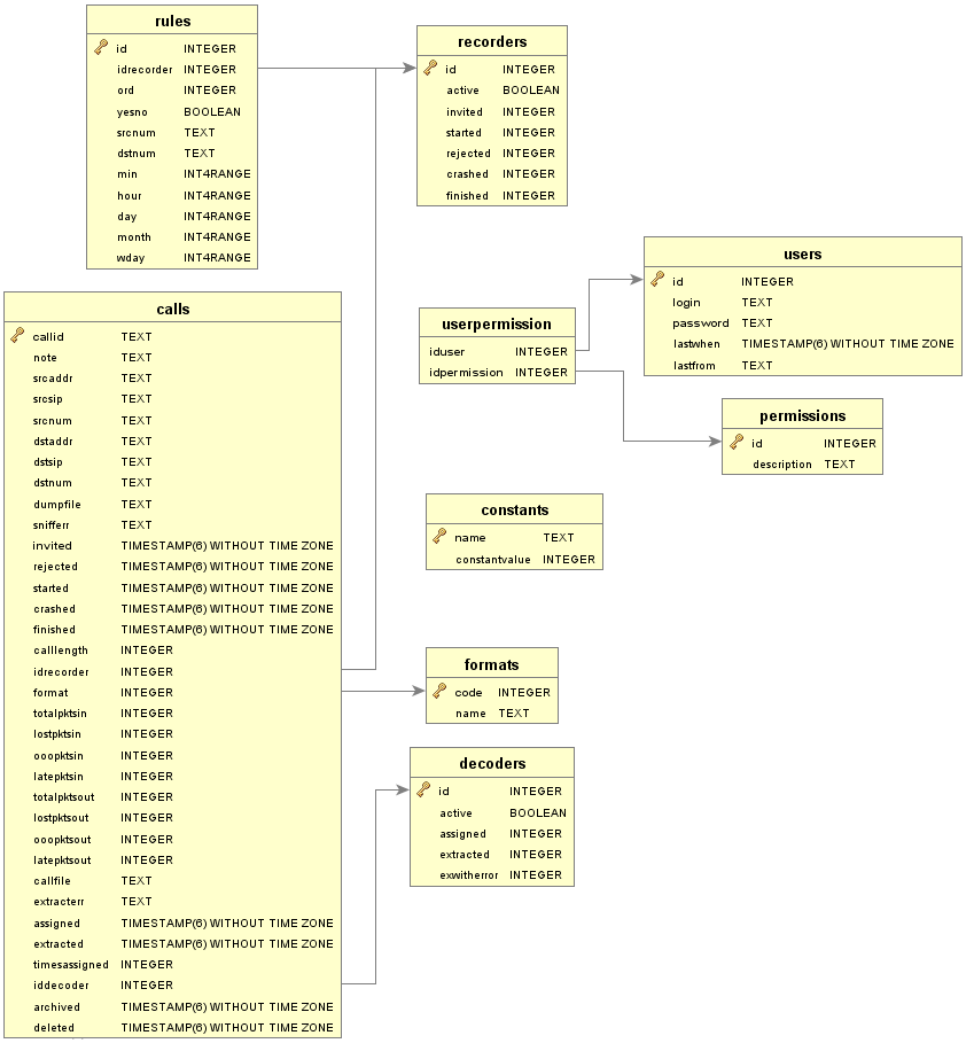
- `Users`
  Tabulka uživatelů webového rozhraní.
  - `id` - Identifikátor uživatele
  - `login` - Jméno uživatele pro přihlašování
  - `password` - Hash hesla uživatele (se solí)
  - `lastwhen` - Čas poslední aktivity uživatele na webu
  - `lastfrom` - IP adresa, ze které se uživatel naposledy přihlásil

- `Permissions`
  Tabulka oprávnění, kterých můžou uživatelé nabývat.
  - `id` - Identifikátor oprávnění
  - `description` - Slovní popis oprávnění

- `UserPermission`
  Propojovací tabulka popisující vztah mezi uživateli a oprávněními.
  - `iduser` - Identifikátor uživatele
  - `idpermission` - Identifikátor oprávnění

- `Constants`
  Tabulka konstant používaných v databázových funkcích. Důvod vytvoření takovéto tabulky je ten, že PostgreSQL nemá místo, kam se dají zapisovat globální proměnné tak, aby byly dostupné z více funkcí.
  - `name` - Název konstanty
  - `constantvalue` - Hodnota konstanty

Vztahy mezi tabulkami můžeme vidět na obrázku 3.1.



Obrázek 3.1: ER diagram databáze

3.1.3 Popis funkcí

Komunikace s rekordéry, dekodéry i webovým uživatelských rozhraním je vedena za použití funkcí (stored procedures) psaných v jazyce PL/pgSQL. Následuje popis funkcí, které jsou něčím důležité nebo zajímavé.

3.1.3.1 Funkce pro rekordéry

- `registerRecorder`(id rekordéru)
  Označí rekordér s daným identifikátorem jako aktivní. Tato funkce musí být zavolána při každém startu rekordéru.

- `unregisterRecorder`(id rekordéru)
  Označí rekordér jako neaktivní. Tato funkce musí být zavolána při každém ukončení rekordéru, včetně neočekávaných.

- `getRulesByOrder`(id rekordéru)
  Vrátí záznamy všech pravidel pro rekordéry, které jsou určeny rekordéru s identifikátorem určeným parametrem funkce, a pravidel, která jsou určeny všem rekordérům. Záznamy jsou seřazeny podle jejich priority.

- `insertCall`(id rekordéru, základní informace o hovoru)
  Vytvoří záznam o hovoru obsahující informace, které jsou známy při vzniku hovoru: číslo a SIP volaného a volajícího, IP adresa volajícího, čas a callid hovoru.

- `updateCallStarted`(id rekordéru, `callid` hovoru, další informace o hovoru)
  Doplní do záznamu o hovoru nové informace zjištěné při úspěšném navázání telefonátu: IP adresu volaného, kodek a čas.

- `updateCallRejected`(id rekordéru, `callid` hovoru, další informace o hovoru)
  Doplní do záznamu o hovoru nové informace zjištěné při odmítnutí hovoru volaným: čas, poznámka.

- `updateCallCrashed`(id rekordéru, `callid` hovoru, další informace o hovoru)
  Doplní do záznamu o hovoru cestu k souboru s paketovými daty přerušeného hovoru, chybovou hlášku rekordéru a čas.

- `updateCallFinished`(id rekordéru, `callid` hovoru, další informace o hovoru)
  Doplní do záznamu o hovoru cestu k souboru s paketovými daty úspěšně dokončeného hovoru a čas. Dále se dopočítá a uloží délka hovoru.

3.1.3.2 Funkce pro dekodéry

- `registerDecoder`(id dekodéru)
  Označí dekodér s daným identifikátorem jako aktivní. Tato funkce musí být zavolána při každém startu dekodéru.

- `unregisterDecoder`(id dekodéru)
  Označí dekodér jako neaktivní. Tato funkce musí být zavolána při každém ukončení dekodéru, včetně neočekávaných.

- `getFreshCalls`(id dekodéru, počet)
  Vrací zadaný počet hovorů připravených pro zpracování dekodérem. Nejprve zavolá funkci obnovující seznam hovorů, které jsou dokončeny, ale ještě nebyly úspěšně zpracovány. Pokud byl hovor již přidělen jinému dekodéru, zkontroluje se, zda nebyl překročen čas na zpracování, daný konstantou v databázi. Pokud byl čas překročen, hovor lze přidělit znovu. Hovor může být postupně přidělen vícekrát, dokud počet přidělení nedosáhne počtu daného další konstantou. Soubor s hovorem je v takovém případě pravděpodobně chybný, a proto přestane být přidělován dalším dekodérům.

- `updateExtractedCall`(id dekodéru, `callid` hovoru, další informace hovoru)
  Do záznamu o hovoru doplní nové informace získané zpracováním a cestu k výslednému zvukovému souboru. Dále se doplní čas zpracování, a případně se v databázi odstraní cesta k souboru s paketovými daty hovoru, pokud byl tento soubor po zpracování smazán.

- `updateFailedExtract`(id dekodéru, `callid` hovoru, chybová hláška)
  Doplní hlášku o chybě při zpracování hovoru dekodérem.

3.1.3.3 Funkce pro webové rozhraní

- `newRecorder()`
  Vytvoří nový identifikátor pro využití novým rekordérem.

- `newDecoder()`
  Vytvoří nový identifikátor pro využití novým dekodérem.

- `createUser`(jméno, hash hesla)
  Zkontroluje, jestli je zadané uživatelské jméno volné, a pokud ano, vytvoří záznam o novém uživateli. Také uživateli přidělí základní oprávnění prohlížet hovory.

- `loginUser`(id uživatele, IP adresa)
  Aktualizuje IP adresu posledního přihlášení a jeho čas.

- `checkUser`(id uživatele, IP adresa)
  Porovná současnou IP adresu uživatele s IP adresou, ze které se přihlásil. Toto slouží jako ochrana proti útoku typu session hijacking. Navíc zkontroluje čas poslední aktivity. Pokud se IP adresy liší, nebo uplynulo příliš dlouho od poslední akce (kde časové rozmezí je nastavitelná konstanta), funkce vrátí false. V opačném případě nastaví čas poslední akce na současný a vrátí true.

- `createRule`(informace o pravidle)
  Zkontroluje platnost vstupu a v kladném případě vytvoří záznam o novém pravidle pro rekordéry.

- `getCalls`(aktivní filtry, řazení, offset)
  Vrátí záznamy o hovorech za využití zadaných filtrů, seřazené podle zadaných kritérií. Pro účely stránkování je využit offset záznamů.

- `getFilter`(sloupec, aktivní filtry)
  Vrátí použitelný filtr hovorů, za použití filtrů, které jsou již aktivní.

Ukázka volání funkce `getcalls`:

- `getcalls('12345678',NULL,NULL,NULL,NULL,'week','calllength', 'ASC',NULL)`
  Vypíše hovory z čísla 12345678 za poslední týden seřazené podle délky od nejkratšího.

3.1.3.4 Typický průběh hovoru

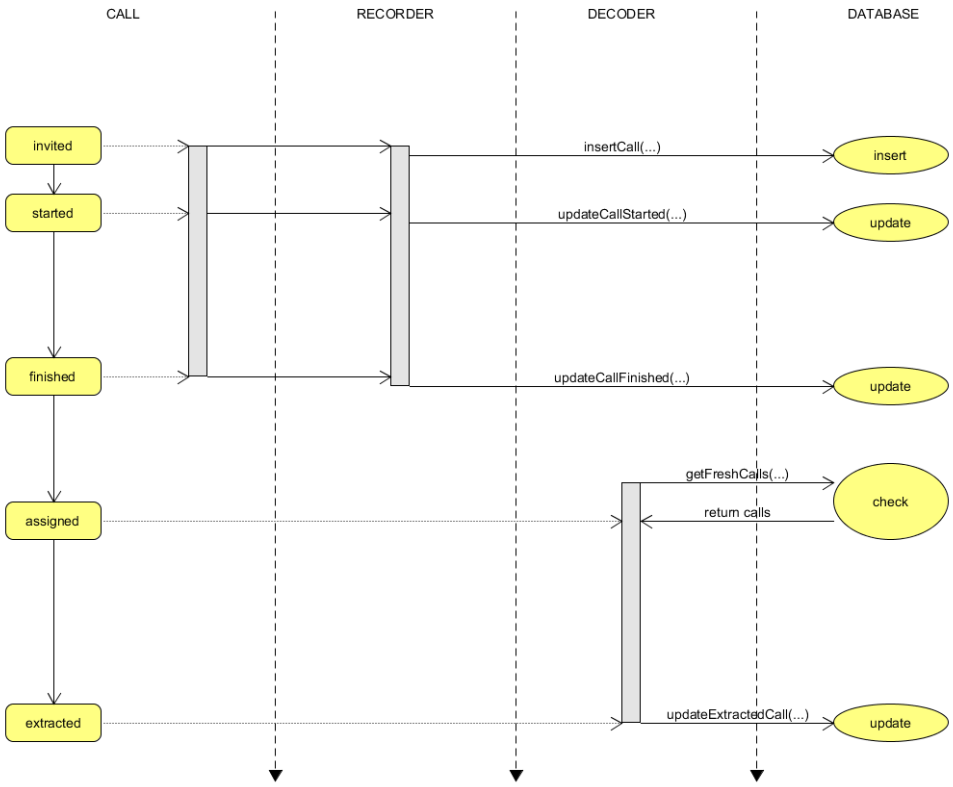
Hovor vznikne vysláním signálu `INVITE` po síti. Hned v této chvíli vzniká v databázi první zmínka o novém hovoru, a to tím, že rekordér po zachycení signálu zavolá funkci `insertCall`.

Rekordér dále naslouchá a volá funkci odpovídající stavu hovoru, tedy `updateCallStarted` nebo `updateCallRejected`. Budeme předpokládat, že hovor úspěšně začal.

Rekordér dále naslouchá a zaznamenává pakety. Hned po zachycení ukončovacího signálu hovoru `BYE` volá příkaz `updateCallFinished`. Případně pokud se tento signál nevyskytl a hovor dále nepokračuje, nebo vznikla nějaká chyba při naslouchání, volá se funkce `updateCallCrashed`. V této chvíli je v databázi informace o hovoru, včetně cesty k souboru s pakety přenosu.

Čeká se do chvíle, než nějaký dekodér zavolá funkci `getFreshCalls`. Ta mu vrací seznam hovorů ke zpracování, mezi kterými může být i tento nový hovor. Dekodér hovor zpracovává, a poté zavolá funkci `updateExtractedCall` pokud byl úspěšný, nebo `updateFailedExtract` v případě neúspěchu. Pokud je hovor úspěšně zpracován, práce na něm je dokončena.

Graficky je tento proces zachycen na obrázku 3.2.



Obrázek 3.2: Diagram průběhu zpracování hovoru

3.2 Webový interface

3.2.1 Konfigurace webového serveru

Ke správné funkčnosti je potřeba webového serveru s nainstalovaným PHP. Navíc musí PHP podporovat funkce PostgreSQL databází. K tomu potřebuje být PHP kompilováno pomocí `--with-pgsql[=DIR]`, kde DIR je cesta k hlavnímu adresáři PostgreSQL, nebo musí být knihovny načteny pomocí `extension` v `php.ini` (viz. instalační příručka PHP).

Na určené místo je poté nutno přesunout zdrojové soubory webového rozhraní (součást této práce). Dále je třeba v souboru `/recources/config.php` vyplnit správné přístupové údaje k databázi.

Potřeba je PHP verze 5.5 nebo vyšší, a to kvůli využití funkcí pro hashování, tedy `password_hash` a `password_verify` ve funkcích pro registraci a přihlašování uživatelů. Tyto funkce jsou kompatibilní s funkcí `crypt()`. Výhodou je jejich jednoduchá použitelnost a využití silných hashovacích funkcí, spolu s generátorem solí a nastavitelnou algoritmickou náročností (cenou) funkce.

3.2.2 Požadavky na webový prohlížeč

Webový prohlížeč musí kromě klasického HTML umožňovat zpracování CSS[9] verze 3, jmenovitě vlastnost `display: flex` a funkci `calc(..)` [11].

Více o kompatibilitě viz. sekce 4.2.2 Kompatibilita s webovými prohlížeči.

Snímek obrazovky výsledného webu s použitím CSS viz. obrázek 3.3.

3.2.3 Vzhled a navigace po webovém rozhraní

Webové rozhraní je rozděleno do několika stránek. Mezi stránkami se uživatel přesouvá pomocí odkazů ve statickém navigačním panelu, případně pomocí menu na levém okraji obrazovky, které se mění podle právě zobrazené stránky.

- Home
  Domovská stránka. Slouží jako uvítací obrazovka po přihlášení. Levé menu umožňuje přechod ke stránce s konfigurací, kde lze nastavit, jaké sloupce se budou zobrazovat při výpisu hovorů. Sloupce jsou organizovány do skupin. (Ne)zobrazení je pak nastaveno celé skupině najednou:
  - sloupce zobrazené vždy - volající číslo, volané číslo, čas `invited`, délka hovoru
  - sloupce se SIP adresami - zdrojová SIP, cílová SIP
  - sloupec s poznámkou
  - sloupce s IP adresami - zdrojová IP, cílová IP
  - sloupce s chybovými hláškami - `snifferr`, `extracterr`

[9] Cascading Style Sheets

  - sloupce s dalšími časy - časy `rejected`, `started`, `crashed`, `finished`
  - sloupce s identikikátory - `idrecorder`, `iddecoder`
  - sloupec s formátem (kodekem)
  - sloupce s paketovými statistikami - počet paketů celkem, ztracených, přijatých mimo pořadí, pozdních; rozděleno na jejich směr (in/out)
  - sloupce s informacemi o zpracování hovoru dekodérem - časy `assigned`, `extracted` a počet přidělení

- Calls
  Stránka s tabulkou hovorů. Levé menu je nahrazeno filtry, které lze kliknutím aktivovat. Tabulka navíc obsahuje své listování, kdy počet výsledků zobrazených na jednom listě je dán konstantou. Prostřednictvím odkazů v tabulce lze aktivovat řazení podle sloupců, přehrání hovoru, nebo zobrazení kompletních informací o jednom konkrétním hovoru.

- Rules
  Stránka s tabulkou pravidel. Levé menu umožňuje přechod k formuláři na přidání nového pravidla. V tabulce jsou odkazy na mazání a editaci pravidel.

- Recorders
  Stránka s tabulkou rekordérů, včetně statistik. Levé menu umožňuje vytvořit nový rekordér.

- Decoders
  Stránka s tabulkou dekodérů, včetně statistik. Levé menu umožňuje vytvořit nový dekodér.

- Users
  Stránka s tabulkou uživatelů, včetně jejich oprávnění. Odkazy v tabulce umožňují přechod k formuláři pro změnu oprávnění jednotlivých uživatelů.

- Login/Register
  Stránky s formuláři pro přihlášení/vytvoření uživatele.

Načítání stránek je řešeno přečtením zpracované proměnné `$_GET['p']`. Začátek odkazu tedy může vypadat například `/?p=calls`.

Uživatel má teoreticky přístup ke všem stránkám, a to i v případě, že nevlastní určitá oprávnění, nebo není vůbec nepřihlášen. Namísto obsahu ale uvidí pouze hlášku oznamující, že nevlastní potřebná oprávnění.

Veškerý vzhled webu je řešen pomocí CSS v samostatném souboru. Vzhled stránek viz. obrázek 3.3.

Obrázek 3.3: Ukázka výsledného zobrazení

3.2.4 Popis funkcí

3.2.4.1 Registrace, přihlašování a kontrola uživatelů

Uživatelé se registrují a přihlašují pomocí jednoduchého formuláře typu jméno, heslo. Obsahy textových polí jsou pak předány odpovídajícím funkcím. Data o přihlášeném uživateli jsou uložena v `SESSION` proměnných.

- register($username, $password)
  - Ověří, zda je dané jméno uživatele volné
  - Vytvoří hash hesla funkcí `password_hash`
  - Zavolá databázovou funkci `createUser` na vytvoření záznamu o uživateli v databázi
- login($username, $password)
  - Databázovou funkcí `loginUser` získá z databáze uživatele se zadaným jménem
  - Ověří hash hesla funkcí `password_verify`
  - Uloží do `SESSION` proměnné identifikátor a jméno uživatele
  - Další databázovou funkcí zaznamená čas a IP adresu, ze které se uživatel přihlásil
- isLoggedInCorrectly()
  - Pomocí databázové funkce `checkUser` zkontroluje, zda je uživatel přihlášen stále ze stejné IP adresy a zda neuplynula od jeho poslední akce příliš dlouhá doba
  - Pokud nejsou obě podmínky vyhodnoceny kladně, odhlásí uživatele

3.2.4.2 Stahování dat z databáze

Pro získání dat z databáze existuje několik funkcí. Některé funkce vrací celou databázovou tabulku (např. `getRulesArray`), jiné pouze její část (např. `getCallsArray`), a některé pouze jeden řádek (např. `getRuleById`). Všechny tyto funkce částečně nebo plně sdílí průběh, a tím jest:

- Ověření způsobilosti uživatele k této akci
- Zavolání databázové funkce
- Ověření návratového stavu
- Ověření, zda se vrátila nějaká data (alespoň jeden řádek dat)

- Tvorba objektů nesoucích data, kde každá instance objektu zpracuje jeden řádek tabulky získané z databázové funkce
- Návrat pole objektů

Přiřazování entit do objektů probíhá jednoduchým způsobem, kde řádek získaný funkcí `pg_fetch_array` s parametrem `PGSQL_ASSOC` je vnímán jako pole. Typické přiřazení do objektu pak vypadá např. `$this->id=$row['id'];`.

3.2.4.3 Řazení a filtrování

Řazení a filtrování je řešeno pomocí proměnných typu `GET`. Do nich se zaznamenává jaké hodnoty kterých polí filtrovat, a podle kterého sloupce řadit a jakým směrem.

Dostupné filtry jsou získávány z databázové funkce `getFilter`. Ta vrátí nejčastější hodnoty daného sloupce, ze kterých je pak vytvořeno menu s odkazy, které tyto filtry aktivují. Jinak lze filtr aktivovat vepsáním hodnoty do příslušného textového pole. Filtry jsou dostupné pro kategorie, kde se jeví jako nejvhodnější. Jmenovitě jsou to:

- Volající číslo
- Volané číslo
- Formát (kodek hovoru)
- Ukončené/neukončené hovory
- Zpracované/nezpracované hovory
- Stáří hovoru (méně než den, méně než týden, méně než měsíc)

Řazení je aktivováno odkazy v záhlaví tabulky. Po dalším kliknutí se změní orientace řazení. Řazení je dostupné pro všechny sloupce, které mají časová nebo číselná data. Jmenovitě tedy:

- Časy hovorů - `invited`, `rejected`, `started`, `crashed`, `finished`, `assigned`, `extracted`
- Číselná data - počet paketů (celkem, ztracených, mimo pořadí, pozdních), délka hovoru, počet přidělení dekodérům

Pomocí proměnných typu `GET`, které se vyplní při klikání na odkazy či tlačítka filtrů a na odkazy řazení v tabulce, se vytvoří specifické volání databázové funkce `getCalls`, která poté vrátí vyfiltrovaná a seřazená data ke zpracování funkcí `getCallsArray`.

Proměnné jsou načítány a zpracovávány z odkazu stránky. Ukázka odkazu stránky zobrazující vyfiltrované hovory:

- `/?p=calls&srcnum=123&age=week&order=invited&ordertype=ASC`
  Vypíše hovory z čísla 123 mladší než týden, seřazené podle času vzniku vzestupně.

Další info viz. předchozí sekce 3.2.4.2 Stahování dat z databáze.

3.2.4.4 Bezpečnostní funkce

Bezpečnostní prvky aplikace budou dále rozebrány v sekci 3.3.
Ke své funkčnosti potřebují tyto funkce:

- sanitize($input, $type)
  Tato funkce odstraňuje ze zadaného vstupu nepovolené znaky. Rozlišuje se mezi několika typy vstupů
  - num
    Ponechá pouze čísla.
  - alfanum
    Ponechá čísla a malá i velká písmenka abecedy.
  - message
    Ponechá čísla, písmenka, mezery, dvojtečky, tečky, pomlčky a hranaté závorky. Proměnné ošetřené tímto vstupem slouží k předání chybových zpráv, které mohou tyto znaky obsahovat.
  - callid
    Ponechá čísla, písmenka, zavináč, tečky a pomlčky. Jsou to znaky, které se mohou vyskytnout v callid hovoru.
  - phplink
    Ponechá čísla, písmenka, zavináč, tečky, pomlčky, ampersandy, rovnítka a lomítka. Tyto se mohou vyskytnout v odkazech, které jsou ukládány do proměnných.
  - page
    Ponechá čísla, písmenka a podtržítka. Tyto znaky jsou použity v názvech souborů, které web potřebuje ke své funkčnosti.
- isValidPage($p)
  Tato funkce porovná hodnotu proměnné se seznamem souborů, které jsou dostupné k zobrazení. Znemožňuje zobrazení souborů, které by uživatel (v tomto případě spíše útočník) vidět nikdy neměl.

3.3 Bezpečnost

3.3.1 Session hijacking

Session hijacking je typ útoku, při kterém se útočník pokouší získat neoprávněnéný přístup k informacím nebo službám využitím spojení uživatele, který má k těmto přístup.

Má aplikace se proti těmto typům útoku brání tím, že při přihlášení uživatele na webové rozhraní se zaznamená jeho IP adresa. Při každé uživatelské akci se kontroluje, zda se jeho adresa nezměnila, a pokud ano, je uživatel donucen přihlásit se znovu. Navíc se kontroluje prodleva mezi jeho akcemi, kde je po dlouhé prodlevě (nastavitelné konstantou v databázi) uživatel také donucen přihlásit se znovu.

Tato opatření mají za následek to, že je nemožné nepozorovaně aktivně využívat spojení přihlášeného uživatele k účelům útočníka. Pokud útočník nesdílí s uživatelem IP adresu, zbývá mu v tomto případě pouze odposlech spojení. Uživateli se navíc při každém přihlášení zobrazí informace o posledním přihlášení, jmenovitě čas poslední aktivity a IP adresa přihlášení. Bystrý uživatel má tedy možnost zjistit, zda jeho účet někdo využíval v době, kdy byl odhlášen.

3.3.2 Cross-site scripting

Cross-site scripting (XSS) je typ útoku, při kterém se útočník podstrčením vlastního počítačového kódu snaží získat informace nebo přístup do aplikace, případně se snaží omezit funkčnost webových stránek či aplikace. Využívá k tomu části systému, ve kterých se dynamicky načítá obsah, tedy například uživatelské vstupy.

Jako ochranu proti těmto útokům používá aplikace funkcí ošetřujících veškeré vstupy z webového rozhraní. Těmi jsou pole formulářů a různé odkazy na stránkách. Dále je ošetřen obsah proměnných, do kterých by uživatel neměl ručně zasahovat, ale lze toho docílit nastavením těchto proměnných pomocí modifikace webového odkazu.

Ošetření vstupů probíhá konkrétně tak, že se pomocí regulárních výrazů odstraní veškeré neplatné znaky a speciální symboly kromě těch, které jsou potřeba. Některá pole mají povoleny pouze alfanumerické znaky, jiné mají ale povoleny i další znaky (např. callid může obsahovat navíc zavináč, tečky a pomlčky). Více viz. předchozí sekce 3.2.4.4 Bezpečnostní funkce.

Načítání jednotlivých stránek je řešeno také pomocí proměnné. Zde by tedy útočník mohl požádat o soubor, který není normálně dosažitelný. Proti tomuto je aplikace chráněna tak, že se obsah této proměnné před načtením souboru porovná se seznamem souborů, ke kterým mají uživatelé přístup. Pokud žádaná stránka není na seznamu platných stránek, je uživatel přesměrován na domovskou stránku aplikace.

3.3.3 SQL injection

SQL injection je typ útoku, kde se útočník snaží spustit na databázi svůj vlastní SQL kód. Typicky k tomu využívá neošetřené vstupy.

Proti tomuto útoku je aplikace chráněna podobně jako proti útokům typu XSS, tedy ošetřením vstupů. Navíc jsou pro komunikaci s databází použity parametrizované dotazy využívající databázové funkce, které jsou podle dokumentace PostgreSQL proti tomuto ošetřené.

Databáze je navíc nastavena tak, aby nekomunikovala s nikým jiným než webovým rozhraním, rekordéry a dekodéry. Přímý útok na databázi by tedy musel být veden z jejich IP adres.

Kapitola 4

Testování

4.1 Testy databáze

Databázové funkce byly otestovány pomocí jednotkových testů frameworku plpgunit [12]. Tento framework připraví testovací rozhraní, tedy tabulky pro výsledky a potřebné funkce, včetně funkcí typických pro jednotkové testování v jiných jazycích, tzn. funkce typu `assert.is_equal(have,want)`. Pro správnou funkci musí být testy psány specifickým stylem.

Testy lze spustit všechny najednou a pokud jsou dobře napsány, vytvoří se přehledná tabulka s jejich úspěšností, včetně případných chybových hlášek. Aby se zabránilo problémům, musí být testy spouštěny na obrazu databáze, nebo při její odstávce. Pokud budou spuštěny na živé databázi, může se stát, že data, která vznikla při běhu testů, budou nějakým způsobem zpracována jinými částmi systému.

Databáze má funkcí sice mnoho, ale jsou často velmi jednoduché, nebo velmi podobné. Proto jsou demonstrativně testovány jen vybrané funkce. Těmi jest:

- Vytváření uživatelů
  Při vytváření uživatelů je otestována nemožnost vytvoření uživatele s neunikátním jménem. Nejprve se vytvoří uživatel s platnými údaji a ověří se, že byl skutečně vytvořen. Poté se test pokusí vytvořit nového uživatele za použití stejných údajů a očekává, že databáze vyhodí výjimku.

- Vytváření pravidel
  Vytváření pravidel otestuje validátor polí pravidel. Test se nejprve pokusí vytvořit pravidlo za použití platných parametrů a očekává úspěch, poté se snaží vytvořit pravidlo za použití parametrů neplatných, a očekává výjimku. Téměř stejným způsobem by se otestovala i editace pravidel.

- Vytvoření nového hovoru
  Zde se testuje nutnost použití platného a aktivního id rekordéru a ne-

možnost přidat hovor s duplicitním callid. Nejprve se pokusí přidat hovor bez registrace rekordéru a očekává výjimku. Poté se zaregistruje, a zkusí hovor přidat znovu. Zde očekává úspěch. Nakonec se pokusí opět přidat stejný hovor, tentokrát ale čeká výjimku. Velmi podobně by se otestovaly téměř všechny ostatní funkce pro rekordéry a dekodéry.

Testy některých funkcí nebyly provedeny z důvodu, že se jedná o funkce silně závislé na datech. Protože se testuje na databázi, u které před testem nemůžeme zaručit obsah, musela by se vzorová data řešit jinak.

Testy mohou být spuštěny skriptem v souboru `/test/db/run_tests.sh` na CD, který nainstaluje plpgunit, spustí testy, zapíše výstup do souboru a odinstaluje plpgunit.

4.2 Testy rozhraní

4.2.1 Automatizované testy

Webové rozhraní bylo částečně otestováno automatickými testy. K tomu bylo využito Selenium IDE [13]. Jedná se o balík nástrojů, který automatizuje ovládání prohlížečů.

Selenium obsahuje funkce, které dokáží ovládat webový prohlížeč. Dokáže vyhledávat různé prvky na stránce a zjišťovat jejich vlastnosti. Dále umí vyplňovat formuláře, odesílat je, klikat na odkazy a podobně. V případě potřeby dokáže vytvořit i snímek obrazovky.

Pro web našeho rozsahu je ale takovýto druh testování zbytečně komplikovaný. Proto je tímto způsobem otestováno jen několik funkcí. Ostatní funkce by vypadaly velmi podobně. Vždy se jedná o akce, které vypisují formuláře, klikají na odkazy a zjišťují, zda byl prohlížeč úspěšně přesměrován na správné místo, a podobně. Testy jsou psány v jazyce Java a ke spuštění vyžadují balíky `org.openqa.selenium` a `org.junit`. Testy jsou uloženy v souboru `/test/www/Tests.java` a dají se spustit jako běžné `junit` testy.

4.2.2 Kompatibilita s webovými prohlížeči

Web byl manuálně testován na prohlížečích:

- Chrome verze 43
- Firefox verze 28 a 38
- Internet Explorer verze 10 a 11
- Safari verze 8
- Safari iOS verze 8.3
- Opera verze 29

- Android Browser verze 4.2 a 4.4
- Chrome for Android verze 42
- Opera Mini verze 8
- Lynx verze 2.8.8

Jediný Android Browser verze 4.2 měl problém stránky zobrazit bez omezení funkčnosti. Nepodporuje totiž vlastnost `display: flex` jazyka CSS3. Na prohlížečích Android Browser 4.4 a Opera Mini verze 8 se vyskytly problémy se zobrazením, které ale nemají vliv na funkčnost. Konkrétně se jednalo o přetečení šířky stránky tabulkou, kde si tabulku sice lze prohlédnout, ale ostatní prvky webu nejsou stejně široké. Tento problém vznikl absencí funkce `calc(..)` jazyka CSS3 v těchto prohlížečích. Tato funkce je použita pro výpočet šířky části stránky, ve které se zobrazuje obsah stránky, v tomto případě tabulky.

Jiné prohlížeče žádný problém neměly. Díky využití meta tagu `viewport` se stránka zobrazuje bez problémů i na mobilních zařízeních s menšími displeji. Kompatibilita s několika oblíbenými prohlížeči, včetně verzí, od kterých mají požadovanou funkcionalitu, je k nahlédnutí v tabulce 4.1.

Nejnovější použité funkce - `display: flex` (resp. `calc(..)`), jsou podle statistik [14] kompatibilní s 95,03% (resp. 93,27%) browserů používaných v Česku.

platforma	browser	verze	funkčnost	zobrazení
Windows/Linux	Chrome	19+	ok	ok
	Firefox	28+	ok	ok
	IE	10+	ok	ok
	Opera	29+	ok	ok
	Lynx		ok	n/a
Android	Android Browser	4.4+	ok	špatná šířka stránky
	Android WebView	40+	ok	ok
	Chrome for Android	42+	ok	ok
	Opera Mini	8+	ok	špatná šířka stránky
	Opera Mobile	24+	ok	ok
Mac OS	Safari	6+	ok	ok
iOS	Safari	6.1+	ok	ok

Obrázek 4.1: Kompatibilita s webovými prohlížeči

4.3 Testy projektu jako celku

Testování úspěšného propojení všech prvků projektu probíhalo manuálně.

Na serveru byl spuštěn rekordér, databáze a webový server. Byly uskutečněny testovací hovory, které rekordér zachytil a úspěšně zapsal do databáze a na souborový systém. Dále byla prostřednictvím webového rozhraní vytvořena nahrávací pravidla, která byla posléze dalšími hovory otestována. Poté byl spuštěn dekodér, který postupně zpracoval všechny uskutečněné hovory a extrahoval z nich hlasová data. Všechny informace o hovorech a statistiky byly postupně zobrazovány na webu. Systém tedy fungoval tak, jak měl.

V dalším testování byl rekordéru předložen ke zpracování soubor se záznamem síťové komunikace obsahující několik set hovorů. Rekordér tyto hovory postupně zajistil a nahrál do databáze. Dekodér poté hovory dále zpracovával. Hovory se daly zobrazit pomocí webového rozhraní. I při zátěži tedy systém fungoval bez problémů.

Dále byl otestován případ, kdy databáze a webový server nebyly spuštěny na stejném serveru. Po správné konfiguraci fungovalo vše podle očekávání.

Závěr

Cílem bakalářské práce bylo navrhnout a implementovat uživatelské rozhraní a databázi pro systém, jež umožní zachytávání a zpracování síťových dat internetové telefonie.

Z mého pohledu dopadl projekt celkem úspěšně. Funkcionalita by se ale dala dále rozšířit, a to např. o možnost zpracování hlasových dat vedených ve více různých kodecích, nebo o možnost zpracovat hovory, které mají více než dva účastníky. Základní účely a požadavky ale plní.

Má část projektu, tedy databáze a webové rozhraní, požadavky splňuje také. Databáze by ale mohla být více optimalizována. Webové rozhraní by se dalo dále rozšířit o další operace s hovory, např. mazání a archivace. Dále by na webu mohlo přibýt přímé ovládání rekordérů a dekodérů, které v současnosti fungují téměř samostatně a jsou ovládány příkazovou řádkou. K tomuto by ale bylo třeba implementovat vlastní komunikační protokol, který by jednotlivé části využívaly. Dále by na webu mohlo přibýt více statistik, včetně grafů a podobně. Možností rozšíření je tedy určitě alespoň několik.

Literatura

[1] [obrázek] SIP transakce. Online, [vid. 2015-06-17]. Dostupné z: https://cs.wikipedia.org/wiki/Session_Initiation_Protocol#/media/File:SIP_transakce.png

[2] Bazala, D.: *Telekomunikace a VoIP telefonie*. BEN - technická literatura, 2006, ISBN 80-7300-201-9.

[3] Rosenberg, J.; Schulzrinne, H.; Camarillo, G.; aj.: SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), Červen 2002. Dostupné z: http://www.ietf.org/rfc/rfc3261.txt

[4] Šuster, F.: *Extrakce VOIP dat ze síťového provozu*. Bakalářská práce, České vysoké učení technické, 2015.

[5] Kučera, J.: *Extrakce a zpracování hlasu z VOIP paketů*. Bakalářská práce, České vysoké učení technické, 2015.

[6] Combs, G.: Wireshark. 1998-, online, [vid. 2015-04-21]. Dostupné z: http://www.wireshark.org/about.html

[7] Mangani, L.: HOMER. 2014, online, [vid. 2015-04-21]. Dostupné z: http://www.sipcapture.org/

[8] The PHP Documentation Group: PHP Manual. 1997-2015, online, [vid. 2015-06-01]. Dostupné z: https://php.net/manual/en/index.php

[9] The PostgreSQL Global Development Group: PostgreSQL 9.4.4 Documentation. 1996-2015, online, [vid. 2015-06-01]. Dostupné z: http://www.postgresql.org/docs/9.4/static/index.html

[10] Schulzrinne, H.; Casner, S.: RTP Profile for Audio and Video Conferences with Minimal Control. RFC 3551, Červenec 2003. Dostupné z: http://www.ietf.org/rfc/rfc3261.txt

[11] Bos, B.: Cascading Style Sheets. 1994-2015, online, [vid. 2015-06-01]. Dostupné z: http://www.w3.org/Style/CSS/

[12] plpgunit. Online, [vid. 2015-06-20]. Dostupné z: https://github.com/mixerp/plpgunit

[13] Selenium. Online, [vid. 2015-06-21]. Dostupné z: http://www.seleniumhq.org/

[14] Can I use... Support tables. Online, [vid. 2015-06-10]. Dostupné z: http://caniuse.com/

Příloha A

Seznam použitých zkratek

VoIP Voice over Internet Protocol

UDP User Datagram Protocol

TCP/IP Transmission Control Protocol/Internet Protocol

SIP Session Initiation Protocol

RTP Real-time Transport Protocol

RFC Request for Comments

PHP PHP: Hypertext Preprocessor

SQL Structured Query Language

CSS Cascading Style Sheets

XSS Cross-site scripting

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `impl` ..... zdrojové kódy implementace
    - `www` ..... zdrojové kódy webového rozhraní
    - `db.sql` ..... skript inicializující databázi
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `test`
  - `www` ..... testy webového rozhraní
  - `db` ..... testy databáze
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF