Asterisk
Open source PBX

Technická zpráva 12/2005

Issue date: 15.12.2005
Authors: Tomáš Wija, David Zukal, Miroslav Vozňák

Cesnet, z.s.p.o.
Zikova 4
160 00 Praha

O dokumentu

Tato verze dokumentu označená v.0.3 je rozšířena a upravena o následující:

- revize teoretické části dokumentu
- změna struktury jednotlivých kapitol
- rozšíření popisů kanálů a kodeků
- bližší popis protokolu IAX2
- aplikace oh323 kanálu pro implementaci H.323 terminálů do systému
- použití GnuGk jako externího gatekeepera pro H.323 kanál
- popis Asterisk Command Line Interface - CLI

Obsah

1. Softwarová ústředna ASTERISK

Oficiálně je Asterisk open source hybrid TDM a packet voice PBX, jedná se o IVR (Interactive Voice Response) platformu s funkčností Automatic Call Distribution (ACD). Neoficiálně jde možná o jedno z „nejsilnějších“, flexibilních a rozšířitelných řešení v oblasti integrovaného telekomunikačního softwaru. Jde tedy o kompletní open source softwarovou PBX, běžící na platformách Linux a Unix, poskytující veškeré vlastnosti, které byste očekávali od PBX. **Asterisk Vám poskytne real-time konektivitu jak do sítí PSTN tak do sítí VoIP.** Jedná se o obecnou distribuci pod podmínkami GNU (General Public Licence). Povolenou výjimku tvoří spojení s OpenH323 projektem a to za účelem dostupnosti H.323 podpory. Systém je navržen tak, aby vytvořil rozhraní telefonnímu hardwaru, softwaru a libovolné telefonní aplikaci.

Asterisk může být mimo jiné použit v těchto aplikacích:

- Různorodá VoIP gateway (MGCP, SIP, IAX, H.323)
- Pobočková ústředna (PBX)
- Voicemail služby s adresářem
- Interaktivní hlasový průvodce (IVR) server
- Softwarová ústředna (Softswitch)
- Konferenční server
- Packet voice server
- Šifrování telefonních nebo faxových volání
- Překlad čísel
- Aplikace Calling card
- Prediktivní volič (Predictive dialer)
- Řazení volání do front se vzdáleným zprostředkovatelem
- Vzdálené „kanceláře“ pro existující PBX

1.1 Požadavky na OS Linux

Asterisk běží na mnoha platformách OS, nicméně hlavní vývojářskou platformou je Linux. V případě, že bude Asterisk používán pouze pro VoIP, nebo jste-li příliš pohodlní používat externí media gateway určené pro připojení konvenčních telefonních zařízení, pak můžete vybírat z většího množství systémů (například FreeBSD, Mac OS X a Solaris).

Systém je navržen tak, aby pracoval na linuxovském jádře verze 2.4, existuje i podpora pro jádro 2.6. S jádrem verze 2.4 pracují například RedHat (7.x, 8, 9 a Enterprise), Debian (i386 a PPC architektura), Fedora, Gentoo, Suse (verze 8 a 9), Mandrake (9.0) a Slackware (9.0, 9.1 a 10.0). Pokoušíte-li se vybudovat stabilní systém, je doporučeno použití jádra 2.4.

1.2 Podporované technologie

Systém je navržen tak, aby povoloval použití nových rozhraní a umožňoval snadno přidávat nové technologie. Jeho cílem je podpora veškerých možných typů současných i budoucích telefonních technologií. Obecně jsou rozhraní rozdělená do tří základních skupin:

- Zaptel hardware
- non-Zaptel hardware
- packet voice.

1.2.1 Zaptel pseudo TDM rozhraní

Tvorba nenákladných rozhraní nebyla vůbec jednoduchým úkolem. Tradiční TDM hardware (např. Dialogic, později majetkem Intelu) byl patentován a také byl příliš drahý. Pro dosažení vymezeného cíle bylo přistoupeno ke zcela nové myšlence. Místo, aby zpracování TDM probíhalo hardwarově, byl přidán hostitelský procesor a Asterisk pracoval s tímto procesorem. Jak se CPU stávaly stále rychlejšími a rychlejšími, začalo být rozumnější pro toto TDM zpracování ponechat software využívat hlavní CPU počítače. Po přidání TDM podpory do Asterisku začala firma Zapata Telephony s výrobou pseudo TDM rozhraní, které nazvala Zaptel. Pseudo TDM architektura poskytuje téměř stejnou kvalitu a real-time schopnosti jakou má hardware TDM. Podstatným rozdílem je však podstatně nižší cena a vyšší flexibilita. Zaptel rozhraní dodává firma Digium (http://www.digium.com) a to pro různé varianty síťových rozhraní (včetně PSTN, POTS, T1, E1, PRI, PRA, E&M a mnoho dalších).

1.2.2 Non-Zaptel rozhraní

Tato rozhraní poskytnou konektivitu směrem k tradičním telefonním službám, nepodporují však pseudo-TDM komutování.
Zahrnují následující typy rozhraní:

- ISDN4Linux - základní ISDN rozhraní pro Linux
- OSS/Alsa - rozhraní zvukové karty
- Linux Telephony Interface (LTI)
- Phonejack/Linejack
- Dialogic hardware - standardně není Asteriskem podporován, ale je k dispozici za poplatek a to jako doplněk pro zákazníky s hardwarem Intel/Dialogic

1.2.3 Packet voice protokoly

Zatímco rozhraní Zaptel pro spojení serveru přímo k PSTN již pracovalo spolehlivě, bylo potřeba vyřešit komunikaci čistě přes IP (později Frame Relay). Jednalo se o případ, kdy začátek nebo konec hlasové relace vznikl někde jinde. Jedná se o standardní protokoly pro komunikaci přes paketové sítě (IP a Frame Relay) a jsou to jediná rozhraní, která nepožadují specializovaný hardware. Jelikož autor Asterisku (Mark Spencer) neměl v přílišné oblibě patentovaný protokol H.323, rozhodl se navrhnout a realizovat svůj vlastní protokol. Výsledkem je protokol IAX (Inter Asterisk eXchange) jenž se stará o signalizaci a transport packet voice mezi dvěmi připojenými uzly. Ačkoliv jméno naznačuje přítomnost Asterisku na obou koncích komunikace, IAX může ve skutečnosti spojit každé dva koncové body podporující tento protokol. Následně byla přidána podpora součinnosti s ostatními VoIP systémy a podpora pro další packet voice protokoly. Jedná se o protokoly SIP, H.323, MGCP (Media Gateway Control Protocol) a VoFR (Voice over Frame Relay). S takovouto podporou se Asterisk stává v současné době ideální PBX na bázi IP pro prostředí s rozmanitými typy používaných telefonů.

1.3 Podporované kodeky

Asterisk podporuje následující kodeky:

- ITU G.711 a-law – 64 kbps, použití Evropa
- ITU G.711 u-law – 64 kbps, použití US
- ITU G.723.1 – 5.3/6.3 kbps, frame size 30 ms

- ITU G.726 – 16/24/32/40 kbps
- ITU G.729 – 8 kbps, frame size 10 ms
- GSM - 16 kbps Full Rate, frame size 20 ms
- ADPCM
- iLBC – internet Low Bitrate Codec. Volně dostupný kodek vhodný pro hlasovou komunikaci přes IP. Je vytvořen pro úzkopásmový hovor přenášený v užitečné zátěži přenosovou rychlostí 13.33 kbps s frame size 30 ms (399 bitů patetizováno v 50 bytech) a přenosovou rychlostí 15.2 kbps s frame size 20 ms (303 bitů patetizováno v 38 bytech). Kodek umožňuje degradaci hlasové kvality v případě ztráty přenosových rámců, která nastane v případě spojení se ztrátou nebo zpožděním IP paketů.
- LPC10 – Linear Predictive Coding. Stejné jako kodek LPC ale s počtem 10 výpočtů předvídaných koeficientů. Používáno pro úzkopásmová spojení.
  Linear Predictive Coding (LPC). Používá techniky zdrojového kódování. Tedy místo vysílání hovorového signálu je vypočítáván nejvhodnější signál mezi originálním signálem a filtrem. Poté jsou nejlépe odpovídající parametry tohoto filtru vyslány na dekóder. Tento LPC dekóder použije tyto parametry ke generování syntetického hovoru, který je více či méně podobný originálnímu signálu. Výsledek je srozumitelný, ale jakoby „strojově“. Výstupní přenosová rychlost je 2.4 kbps.
- Speex – open-source hlasový kodek (http://www.speex.org). Cílem je snížení bariér aplikací záznamu hovoru a to poskytnutím alternativ k drahým patentovaných hlasových kodeků. Kodek je dobře adaptovatelný na Internetové aplikace a poskytuje užitečné vlastnosti, které nejsou dostupné ve většině ostatních kodeků (funkce VAD, podpora VBR, funkce Ukrytí ztráty paketů, úzkopásmová 8kHz – širokopásmová 16 kHz – ultra-širokopásmová 32 kHz komprese ve stejném bitovém toku).

1.4 Architektura

Architektura systému je v podstatě velmi jednoduchá, ale zcela odlišná od nejčastěji používaných telefonních produktů. Asterisk vystupuje v podstatě jako středový prvek spojující telefonní technologie na jedné straně s telefonními aplikacemi na straně druhé (**Obr. 1-1**). Vytváří konzistentní prostředí pro rozmístění smíšeného telefonního prostředí.

Telefonní technologie mohou zahrnovat VoIP služby (jako SIP, H.323, IAX a MGCP jak gatewaye tak telefony), stejně jako tradiční TDM technologie (jako T1, ISDN PRI, analogové POTS a PSTN služby, ISDN BRI, atd.).

Telefonní aplikace zahrnují například přemostění volání, konference, hlasovou poštu, automatickou obsluhu, uživatelské IVR skriptování, parkování hovoru, intercom a další.

Obr. 1-1: Obecné umístění Asterisku v systému

Asterisk je pečlivě navržený za účelem poskytnutí maximální flexibility. Okolo systému centrálního jádra PBX jsou definovány specifické API (**Obr. 1-2**). Toto pokročilé jádro ovládá vnitřní propojení PBX, myšleno specifické protokoly, kodeky a hardwarové rozhraní telefonních aplikací. To dovolí v Asterisku použít každou vhodnou technologii a hardware

(nyní i v budoucnu) za účelem vykonávání základních funkcí - propojování hardwaru a aplikací.

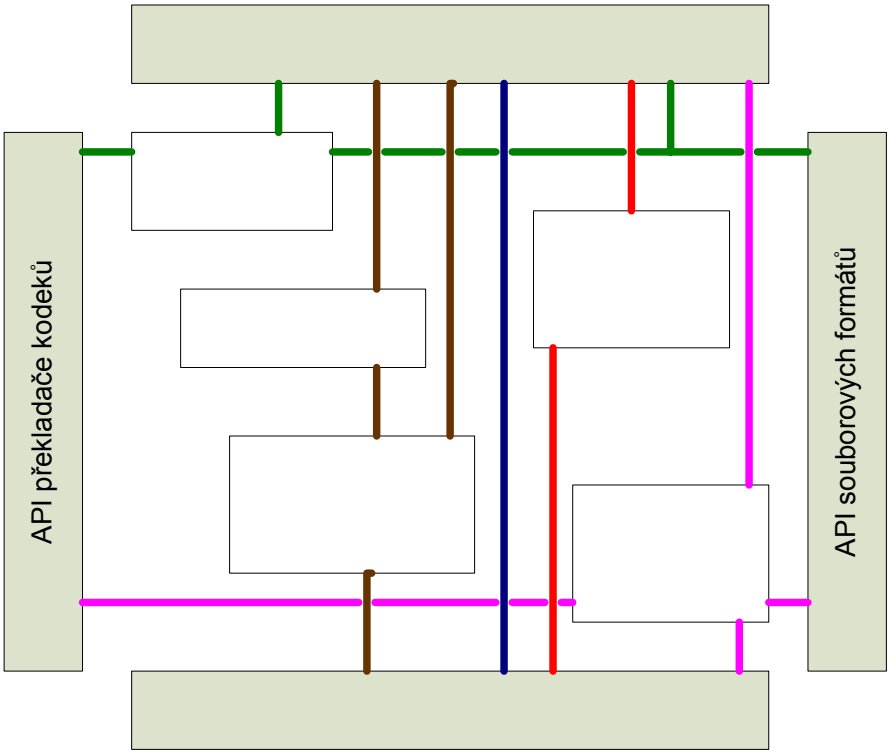
Asterisk jádro vnitřně ovládá tyto položky:

- PBX přepojování (PBX Switching) - podstatou Asterisku je samozřejmě přepojovací systém pobočkové ústředny, spojovací volání mezi různorodými uživateli a automatizovanými úlohami. Přepojovací jádro transparentně spojuje příchozí volání na různých hardwarových a softwarových rozhraních.

- Spouštěč aplikací (Application Launcher) - spouští aplikace zajišťující služby jako jsou například hlasová pošta, přehrání souboru a výpis adresáře.

- Překladač kodeků (Codec Translator) - používá moduly kodeků pro kódování a dekódování různých zvukových kompresních formátů používaných v telefonním prostředí. Množství dostupných kodeků je vhodné pro různorodé potřeby a docílení stavu rovnováhy mezi zvukovou kvalitou a použitou šířkou pásma.

- Scheduler a I/O manažer (Schedule and I/O manager) - ovládání rozvrhování nízkoúrovňových úloh a systémového řízení pro optimální výkon podle stavu zatížení.



Obr. 1-2: Blokové zobrazení architektury

Zaváděné moduly API:

Pro zaváděné moduly jsou definovány čtyři API, usnadňující oddělení hardwaru a protokolů. Použitím tohoto zaváděného modulového systému si jádro Asterisku nemusí dělat starosti s detaily o skutečnosti, jak se volající připojí, co používá za kodeky, atd.

- Kanálové API - ovládá typ spojení příchozího volání, tedy jedná-li se o VoIP spojení, ISDN, PRI, Robbed bit signaling nebo nějakou další technologii. Dynamické jednotky jsou zavedené pro ovládání detailů nižších vrstev těchto spojení.

- Aplikační API - aplikační API bere v úvahu různorodé jednotky úkolů , které mají běžet za účelem vykonávání různorodých funkcí. Conferencing, Paging, Výpis adresáře, Hlasová pošta, Přímý přenos dat a každý další úkol, který by PBX systém mohl vykonávat (teď nebo v budoucnu) je ovládán prostřednictvím těchto oddělených modulů.

- API překladače kodeků - zavádí moduly kodeků pro podporu různých audio formátů kódování a dekódování jako GSM, μ - law, A - law a také MP3.

- API souborových formátů - ovládá čtení a zápis různých souborových formátů pro ukládání dat v souborovém systému.

Používáním těchto API, Asterisk docílí kompletního oddělení mezi jádrem, pracujícím jako PBX server systém a různorodými existujícími technologie (nebo vyvíjenými). Modulární forma dovolí systému hladkou integraci a to jak aktuálně implementovaného hardwaru přepojované telefonie, tak rychle rostoucí hlasové technologie na bázi paketů. Schopnost zavádět moduly kodeků, umožní podporovat extrémně kompaktní kodeky, nutné pro paketový hlas přenášený přes pomalá spojení (např. telefonní modem) za předpokladu, že stále poskytují vysokou kvalitu zvuku přes méně omezená spojení.

Aplikační API poskytuje pružné použití aplikačních modulů pro vykonávání každé funkce a to flexibilně na požádání. Bere v úvahu otevřený rozvoj nových aplikací za účelem přizpůsobení unikátních uživatelských potřeb a situací. Navíc zavádění všech aplikací jako samostatných modulů umožní velmi pružný systém, dovolující administrátorovi navrhovat nejlepší vhodnou cestu pro volající na PBX systému a modifikovat směry volání vhodné pro změnu komunikačních potřeb výnosného podniku.

1.5 IAX protokol

Tvůrcem protokolu IAX stejně jako SW open-source ústředny Asterisk je Mark Spencer. V současné době vyšel nový internet-draft týkající se protokolu IAX nazvaný „IAX: Inter-Asterisk eXchange Version 2“. Tento byl vydán 11/2005 a nahradil předchozí verzi, která již měla prošlou platnost. Vlastní dokument je typu Internet-Draft, tedy pracovním dokumentem IETF (Internet Engineering Task Force) skupiny. Tyto dokumenty jsou platné maximálně 6 měsíců a mohou být kdykoliv nahrazeny nebo upraveny.

Dokument popisuje protokol určený pro řízení aplikační vrstvy a přenášené médium, vytvoření, modifikaci a ukončení relaci prostřednictvím sítě s IP protokolem. IAX je primárně určen pro řízení přenosu hlasu prostřednictvím VoIP volání, může být však použit také pro streaming různorodých dat (audio, video, …).

Primárním cílem protokolu bylo minimalizovat nezbytnou šířku pásma určenou pro signalizaci a vlastní přenášené médium (v našem případě „hlas“). Dalším důležitým cílem bylo poskytnou přirozenou podporu pro NAT transparentnost (Network Address Translation). Pro vynucení průchodu přes NAT firewally není již při použití IAX potřebná žádná další konfigurace. Protokol je vytvářen s ohledem na pozdější rozšiřitelnost a vylepšení.

1.5.1 Základní vlastnosti

IAX je signalizační a přenosový protokol typu peer-to-peer. Může zaznamenávat umístění, vytvoření, modifikaci a ukončení multimediální relace. Přenáší aktuální média toky specifické pro relace, které ovládá. Protokol je navržen a optimalizován pro popis a transport hlasových volání používajících Internet protokol.

Základním přístupem k návrhu IAX je multiplexace signalizace a vícenásobných média toků do jediného UDP spojení mezi dvěma hostiteli. V praxi je toto provedeno prostřednictvím stejného „dobře známého“ UDP portu (4569) a to pro všechny typy IAX provozu. Sjednocení signalizační cesty a cesty přenosu média vytvoří NAT transparentnost, která je hlavní výhodou IAX protokolu oproti alternativním transportním protokolům.

IAX je binární protokol. Hlavní výhodou této skutečnosti je efektivita využití šířka přenosového pásma a z důvodu, že kvalita hlasových volání často souvisí s velikostí využitého frekvenčního pásma. Protokol je specificky optimalizovaný pro vytvoření účinného využití šířky frekvenčního pásma pro individuální hlasová volání. Obecně lze říci, že efektivita využití šířka frekvenčního pásma ostatních druhů přenášených toků je obětována právě pro hlasová volání. Další výhodou tohoto tvaru protokolu jsou robustnost proti útokům typu „přeplnění bufferu“ a kompaktní implementace schopností, které redukují problémy s interoperabilitou související se syntaktickou analýzou.

Základní komunikační jednotkou v IAX je rámec. Existují rozmanité třídy rámců, které jsou popsány v dokumentu. Obecně Full rámce přenášejí signalizaci nebo řídicí data, zatímco Mini rámce přenášejí data média toků. Full rámce rovněž poskytují možnost použití připojenýxh Informačních elementů (IE). IE mohou popisovat různé typy uživatelsky specifických dat a dta specifických pro volání. Meta rámce jsou používané pro síťování volání nebo přenos video toků.

IAX podporuje opatření pro zabezpečení prostřednictvím umožnění vícenásobných metod uživatelské autentifikace a autorizace, stejně tak jako možnost peer registrace.

1.5.2 Struktura použitých komunikačních rámců

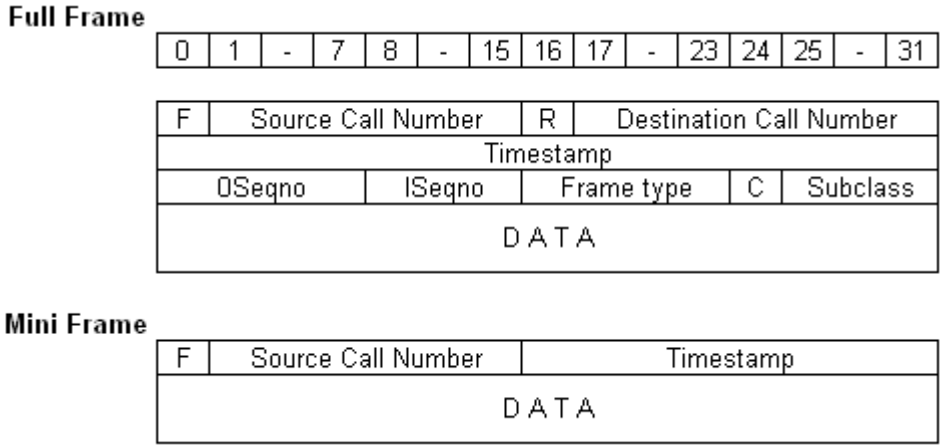
Full Frame je používán pro přenos signalizační, audio nebo video informace. Tyto rámce jsou používány pro inicializaci, setup a ukončení IAX volání. Full rámce jsou doručovány spolehlivě – ze strany příjemce je vyžadována příslušná odezva a to buď zprávou ACK nebo odezva založená na obdržení příslušné odpovědi na vyslaný Full rámec. Standardní velikost tohoto rámce je 12 oktetů.

Mini Frame. Název tohoto rámce plyne ze skutečnosti, že jejich záhlaví má minimální velikost 4 oktety. Jejich účelem není přenos signalizačních nebo řídících dat, ale přenos média toků již dříve sestavených IAX volání. Neexistuje zpětné potvrzován, jedná se tedy o nespolehlivé doručování rámců. Předpokladem pro využití tohoto typu přenosu je vlastní přenos VoIP provozu, který může ztratit několik přenosových rámců a to bez podstatné degradace kvality přenášeného hlasového signálu. Protože jsou tyto data typicky přenášena v reálném čase, je opětovný přenos ztracených rámců zcela nesmyslný. Daný okamžik již na straně příjemce již proběhl a starší data jsou tak k ničemu. Použitý Timestamp je zkrácen na 16 bitů, jedná se o 16 nižších bitů 32 bitové hodnoty udržované koncovým bodem (přenášeno ve Full Frame).

Meta Frame. Meta rámce slouží jednomu ze dvou definovaných účelů. Meta video rámce umožní přenos video toků s optimalizovaným záhlavím. Tyto jsou svým účelem podobné Mini rámcům. Meta trunk rámce jsou používané pro síťování vícenásobných IAX

média toků mezi dvěmi peer a to do jednoho záhlaví. Důvodem je další minimalizování využité šířky frekvenčního pásma.

Information element. IAX zprávy přenášené pomocí Full rámců mohou přenášet informační elementy ke specifickým uživatelským datům nebo datům příslušných volání. Informační element je připojený k záhlaví rámce ve daném datovém poli. První oktet informačního elementu se skládá z IE pole, což je identifikační číslo které definuje tento specifický informační element (typy viz. Internet-draft). Druhý oktet je pole „délka dat“, která specifikuje počet oktetů přenášených daným IE. Ostatní oktety, které následují, přenáší již konkrétná data jejichž tvar je závislý na typu přenášených dat.



Obrázek 1-3: Binární forma používaných rámců

- F - bit určující zda se jedná o Full Frame (1) nebo Mini Frame (0)
- Call number - je to 15-ti bitové číslo používané pro sledování média toku konkrétního koncového bodu. Pokud je hodnota „0“ pak toto znamená, že číslo je neznámé.
- Timestamp - 32 nebo 16 bitová hodnota.
- R - tento bit má hodnotu 1 pokud je rámec opětovně přenášen.
- OSeqno (Outbound Stream Sequence Number) – začíná vždy 0 a je inkrementován. Je používáno příjemcem pro sledování řazení rámců.
- ISeqno (Inbound Stream Sequence Number) - příchozí rámce.
- Frame type - identifikuje třídu zprávy (bližší popis a seznam v dokumentaci [1])
- C - určuje jak bude interpretován parametr Subclass
- Subclass - jestliže C = 1, pak je Subclass interpretován jako mocnina 2.
  - jestliže C = 0, pak je Subclass interpretován jako 7 bitová hodnota.

1.5.3 Typy používaných rámců

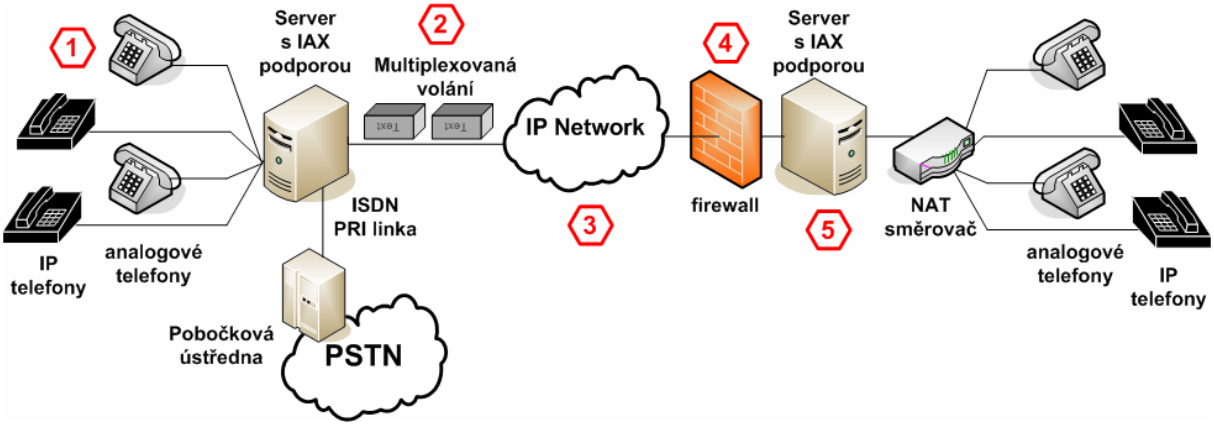
IAX protokol specifikuje 10 typů rámců, které jsou definovány prostřednictvím pole „frametype“ IAX Full rámce. Jedná se o tyto typy:

- DTMF frame – přenáší jednotlivé číslice DTMF volby (RFC2833)
- Voice frame – přenos audio dat
- Video frame – přenos video dat
- Control frame – přenos řídících dat dané relace
- Null frame – rámec s nulovou hodnotou (nepovinný přenos)

- IAX frame – přenos řídících dat, která poskytnou IAX protokolu specifický management koncových bodů. Tyto rámce ovládají IAX signalizaci (např. call setup, udržování a ukončení volání). Mohou také ovládat přímý přenos média dat, což však není volbou pro VoIP volání.
- Text frame – přenos ne-kontrolních textových zpráv
- Image frame – přenos jednotlivých grafických souborů
- HTML frame – přenos dat typu HTML
- Comfort Noise frame

1.5.4 IAX a struktura sítě

IAX protokol přenáší audio pakety pouze s 4 bytovým záhlavím a vlastní příkazy používají pouze velmi malou část přenosového pásma. Pro více současných volání redukuje IAX režii každého kanálu prostřednictvím kombinování dat několika kanálů do jednoho paketu, neredukuje tím tedy pouze počet záhlaví, ale také počet paketů.



Obrázek 1-4: Principiální schéma možností IAX protokolu

(1) Generovaný hovor je vyslán skrze Server s IAX podporou (Asterisk), který plní funkce pobočkové ústředny a gatewaye. Server může zakódovat hovor do IAX formátu a to jak hovor z klasické PSTN připojené prostřednictvím PRI linky, tak hovor ve formátu VoIP protokolu.

(2) Server s IAX podporou zapouzdří volání do IAX protokolu a multiplexuje vícenásobná volání do jednoho kanálu. Eliminuje také nadbytečné paketové záhlaví.

(3) Volání je směrováno přes IP síť.

(4) Volání prochází bez problémů přes firewall. Důvodem je skutečnost, že komunikace probíhá přes jediný UDP port (jednoduchý požadavek na jeden otevřený UDP port).

(5) Server s IAX podporou rozliší, na který telefon má dané volání směrovat (multiplexované IAX médium obsahuje také signalizaci).

1.5.5 Budoucnost ?

Přestože jsou všechny implementace protokolu IAX totožné, hlavní překážkou pro masivnější rozšíření protokolu je v současnosti jeho nepřijetí výrobci. Hlavním problémem je skutečnost, že vlastní protokol není stále standardizován (IAX je pouze typu Internet-Draft). Na druhé straně se i bez této standardizace objevují nejen softwarové telefony, ale již také první hardwarové telefony (například námi testovaná Atcom AT-320).

Tvůrci protokolu IAX se vydali směrem zjednodušení použitého komunikačního protokolu, který již nebude zažívat problémy na úrovni firewall (H.323) a na úrovni NAT (SIP). Významným krokem by byla bezesporu standardizace protokolu (například prostřednictvím RFC), což by znamenalo pevný dokumentační základ pro výrobce v oblasti VoIP technologií.

V předchozím uvedeném textu je přiblížen pouze nástin vlastností protokolu, nejsou například vůbec popsány jednotlivé typy komunikačních zpráv. Původní verze draftu vydaná 1/2005 byla značně rozšířena současnou verzí 11/2005 a nová verze je již podstatně rozšířena a pro jednodušší přiblížení všech vlastností již není příliš prostoru a navíc by to ani v tomto textu nemělo smysl. Pro podrobnější informace je proto vhodnější přímo použít daný internet-draft 11/2005.

1.6 Kanály Asterisku

Kanály jsou logická spojení s různými signalizačními a přenosovými cestami, které může Asterisk využívat k vytváření a spojování jednotlivých hovorů. Kanál by mohl představovat spojení s obyčejným telefonním přístrojem (telefonní linkou) nebo Internetové telefonní hovory („logické volání“). Asterisk nedělá žádný rozdíl mezi typem kanálu „FXO (Foreign eXchange Office)“ a „FXS (Foreign eXchange Station)“, nerozlišuje tedy mezi telefonními linkami a telefony. Každé volání je umístěno na odlišném kanále.

Prostřednictvím kanálů vstupují do systému různé formáty komunikace; fyzické telekomunikační okruhy (jako FXO, FXS, PRI, BRI), softwarově založené spojení, síťově připojitelné entity (jako SIP a IAX) a exkluzivní vnitřní kanály Asterisku určené pro všechny dodatečné prostředky (jako Agent, Console, Local). Asterisk se všemi těmito kanály zachází jako s přípojnými body, jejichž vzájemnou interakci provádíte v Dialplan (extensions.conf). Je důležité si uvědomit, že i kdyby se kanály lišily v rámci použité technologie a konektivity, Asterisk vám dovolí zacházet se všemi jakoby byly téměř stejné.

Díky způsobu zacházení s kanály je Asterisk extrémně flexibilní a silnou PBX. V mnoha klasických PBX mají různé kanály zcela jiné způsoby komunikace. Asterisk vám umožní jednat s daným kanálovým typem stejným způsobem jako s ostatními kanálovými typy.

Asterisk ve standardní distribuci poskytuje následující typy kanálů:

- Agent – ACD Agent kanál
- Console – konzolový klientský ovladač pro zvukové karty
- H.323
- IAX nebo IAX2 – Inter-Asterisk Exchange Protocol
- Local – Loopback do dalšího kontextu
- MGCP – Media Gateway Control Protocol
- Modem – pro připojení ISDN linek
- phone – Linuxový telefonní kanál
- SIP – Session Initiation Protocol
- Skinny – ovladač pro Cisco Skinny Client Control Protocol
- VOFR – Voice over Frame Relay
- VPB – pro připojení obyčejného telefonu a telefonní linky používající Voicetronix karty
- Zap – pro připojení obyčejného telefonu a telefonní linky používající Digium karty

Ovladače kanálu nabízejí další technologie, které mohou být volitelně instalovány:

- bluetooth – dovolí používat bluetooth zařízení pro změnu směrování
- CAPI – ISDN CAPI kanál
- mISDN – mISDN kanál
- SCCP – alternativní Skinny/SCCP kanál
- UNISTIM – Nortel UNISTIM kanál

1.6.1 Kanál Agent

chan_agent je pseudo-kanál pro směrování volání k ACD agentovi. Agenti jsou definováni v agents.conf a aktivováni speciální log-in procedurou. To dovolí agentovi přizpůsobit se, aby byl k dispozici pro odebrání hovoru z fronty pomocí přihlášení se k telefonu. Agent může později zastavit odebírání hovorů pomocí odhlášení se od telefonu. Agenti mají smysl pouze v ACD uspořádání. Normálně jsou definováni jako účastníci fronty (hovory čekající ve frontě) uvnitř queues.conf.

1.6.2 Kanál H.323

V současnosti existuje několik nezávislých implementací kanálu H.323 do systému Asterisk (h323, oh323, ooh323c, woomera). V našem případě jsme zvolili kanál oh323.

- **h323**

Jedná se o kanál obsažený v Asterisk zdrojové distribuci v adresáři /channels/h323. Tento kanál funguje pouze jako H.323 Gateway, ne však jako Gatekeeper.

- **oh323**

Implementace H.323 kanálu (ve skutečnosti vůbec první implementace) nazvaný Asterisk-oh323, která je stále aktivně vyvíjena a to projektem firmy InAccess Network (http://www.inaccessnetworks.com/projects/asterisk-oh323).

- **ooh323c**

Předpokládá se, že tento kanál bude brzy součástí asterisk-addons balíčku. Jedná se o nový kanálový driver založený na open source H.323 stack (ooh323c) firmy Objective systéme (http://www.obj-sys.com/open). Tento stack je vyvinutý v jazyku C a obsahuje pouze zdrojový kód potřebný pro sestavení H.323 signalizačního kanálu. Všechny ostatní procesy jsou ovládány samotným Asteriskem. Tato skutečnost poskytne škálovatelnost pro H.323 volání, která budou primárně závislá pouze na kapacitě Asterisku s ohledem na ovládání média streamů.

- **woomera**

Jedná se o protokol (Woomera), který umožní dát Váš VoIP systém na jeden systém server/proces a Vaši PBX na jiný systém a následně je propojit pomocí jednoduchého UDP protokolu. chan_woomera je Asterisk driver kanálu navržený pro propojení Asterisk PBX se systémem Woomera. V současnosti je systém ve verzi Beta a podporuje pouze H.323. Předpokladem je podpora abstraktní vrsty OPAL VoIP, která umožní komunikaci s ostatními protokoly. Prostřednictvím Woomera bude možno propojit Asterisk s H.323 serverem (openh323 code), který bude podporovat H.323 přes Ipv6.

Poznámky:

- h323 – neobsahuje jitter buffer. Tato implementace používá Asterisk RTP Stack.
- oh323 – používá RTP/RTCP stack a implementaci adaptivního jitter buffer z OpenH323. Nepoužívá kodeky systému OpenH323, ale kodeky Asterisku.
- Použití oh323 odstraňuje problémy plynoucí z použití h323 (stabilita). Nicméně přibližně 10-15 krát zvyšuje využití procesoru.

1.6.3 Kanál IAX

IAXtel, služba nabízená firmou Digium, od prosince 2003 podporuje pouze protokol IAX2. Protokol IAX je od ledna 2004 totožný s protokolem IAX2.
IAX2 může být použit pro:

- vzájemné spojení Asteriskových serverů (IAX Trunking)
- připojení ke službě IAXtel
- připojení IAX klientů

Konfigurace kanálů se provádí modifikací souboru iax.conf.Funkce IAX Trunking umožní vícenásobným hlasovým tokům sdílet jediný „trunk“ k jinému serveru, redukováním režie vytvořené IP pakety.

Formát „IAX channel name“ používaný pro odchozí spojení:

```
IAX/[<user>[:<secret>]@]<peer>[:<portno>][/<exten>[@<context>][/<options>]]
```

<user> - UserID na vzdáleném serveru nebo jméno klienta konfigurované v iax.conf (optional)
<secret> - Password (optional)
<peer> - jméno serveru ke kterému je prováděno připojení
<portno> - číslo portu určeného pro spojení na server (optional)
<exten> - extension ve vzdáleném Asterisk serveru (optional)
<context> - kontext který je používán ve vzdáleném Asterisk serveru (optional)
<options> - jediné dostupné option „a“ ve významu „požadovat automatickou odpověď“

Formát „IAX channel name“ používaný pro příchozí spojení:

```
IAX[[<username>@]<host>]/<callno>
```

<username> - uživatelské jméno
<host> - host jméno (IP adresa
<callno> - lokální telefonní číslo

1.6.4 Kanál Local

Kanál chan_local je pseudo-kanál. Používá se pro vytvoření smyčky, která volá zpět do Dialplan v různých context. Užitečné je především rekurzivní směrování, které je schopno vracet se do Dialplan po ukončení volání.

1.6.5 Kanál Modem

Kanál Modem je použit pouze pro ISDN karty ovládané i4l (isdn4linux) ovladačem. Existuje zde také alternativa této metody, která používá ISDN hardware uvnitř Asterisku (použití CAPI kanálu - chan_capi).

1.6.6 Kanál SIP

SIP kanálový modul umožní Asterisku VoIP komunikaci se SIP telefony a ústřednami. Konfigurace SIP kanálů/klientů se provádí modifikací souboru sip.conf. Asterisk může působit jako:

- SIP klient – to znamená, že se Asterisk zaregistruje jako klient k jinému SIP serveru a přijímá a umisťuje volání k tomuto serveru.
- SIP server – Asterisk může být nakonfigurován tak, že se SIP klienti (telefony, softwaroví klienti) zaregistrují k serveru Asterisk a sestaví SIP relaci se serverem. Asterisk není však plnohodnotný SIP server (jako např. SIP Express Router). V případě že počítáte s několika sty SIP telefony v síti je vhodnější použít SER a propojit jej na Asterisk za účelem poskytnutí voicemailu a přístupu do PSTN.
- SIP gateway – Asterisk vystupuje jako media gateway mezi SIP, IAX, MGCP, H.323 a PSTN spojením.

Základní vlastnosti:

- Podpora ENUM (enum.conf)
- Podpora DNS SRV záznamů (SIP srvlookup)
- Podpora SIP přes UDP, ale ne SIP přes TCP.
- Možnost uložení SIP zařízeních v databázi pro dynamickou konfiguraci (SIP Mysql Peer)

1.6.7 Kanál vpb

Jestliže máte Voicetronix kartu rozhraní (např. OpenLine4, OpenSwitch8, OpenSwitch16) budete používat jejich linuxové ovladače. Asterisk komunikuje s těmito ovladači použitím vpb kanálového modulu. Konfigurace vpb kanálového modulu se provádí modifikací souboru vpb.conf.

1.6.8 Kanál Zap

Zap kanálový modul poskytne mezivrstvu (interface layer) mezi Asteriskem na jedné straně a Zaptel a/nebo ZapHFC ovladači rozhraní na straně druhé. Tyto ovladače poskytují schopnost používat karty rozhraní pro připojení vaší PBX k tradičnímu digitálnímu nebo analogovému telefonnímu vybavení:

Asterisk < -- > ZapModule < -- > ZaptelDriver < -- > Digium Interface card < -- > Phone/switch/PSTN
ZapHFCDriver < -- > ISDN Card with HFC-Chipset < -- > PSTN

Konfigurace ZAP kanálů se provádí modifikací souboru zapata.conf.

2. Dialplan – číslovací plán

Dialplan je konfigurován v souboru extensions.conf. Je to nejdůležitější konfigurační soubor systému. Řídí způsob ovládání a směrování příchozích a odchozích hovorů. Toto je místo, kde kontrolujete chování všech spojení provedených prostřednictvím PBX.

2.1 Organizace dialplan

2.1.1 [general]

V této sekci definujete některá obecná nastavení. Na začátku konfiguračního souboru by měla být sekce s označením:

[general]

, kde definujete několik hlavních voleb, týkajících se dialplan.

- static: v tomto stupni tato volba ovlivňuje pouze operace příkazů *save dialplan*. Default hodnota je *no*, ale vzorový soubor extension.conf instalovaný explicitně nastaví static = yes.
- writeprotect: Jestliže je nastaveno writeprotect=yes a static=yes, pak můžete ukládat aktuální dialplan prostřednictvím CLI (Command Line Interface) příkazu „save dialplan“. (definice globálních proměnných v sekci [globals] zůstane nezměněna). Deafult hodnota je *no*.

Upozornění:

- „save dialplan“ přepíše váš existující soubor extensions.conf nově generovaným z aktuálního dialplan. Kopie vašeho starého souboru extensions.conf nebude ponechána. Všechny vaše komentáře v souboru budou ztraceny.
- Neplatí skutečnost, že „*no*“ je default hodnota jestliže není nic specifikováno, ale explicitně instalovaný soubor extensions.conf nastavuje hodnoty writeprotect=no a static=yes. Tato skutečnost umožňuje zranitelnost souboru extensions.conf v default instalaci.

2.1.2 [globals]

Definice globálních proměnných dialplan a jejich počáteční hodnoty následují [general] nastavení. Sekce globálních proměnných začíná hlavičkou:

[globals]

V současnosti nejsou globální proměnné Asterisk obvykle používány jako proměnné, ale jako konstanty. Obvykle jsou použity pro možnost mít samostatné místo v dialplan, kde můžete specifikovat hodnoty, které máte možnost v budoucnu změnit, jestliže budete chtít změnit konfiguraci PBX.

Aktuální hodnota globální proměnné může být v dialplan změněna použitím příkazu *SetGlobalVar*, může být uvnitř dialplan vyjádřena syntaxí:

${VariableName}

Poznámka: Jména globálních proměnných nejsou citlivé na velikost písmen, tedy ${MYGLOBAL} a ${myGLObal} jsou ekvivalentní.

Příklad: Definování některých globálních proměnných.

```
[globals]

; Určení které extension budou vyzvánět při příchozím volání
        INCOMING => Zap/3&Zap/4
; Určení jak dlouho budou tyto extension vyzvánět než volání místo toho přejde do voicemail
        RINGTIME => 3
; Určení jaký zvukový soubor bude přehrán jako úvodní pro voicemail
        VMANNOUNCE => mysounds/my-vm-announce
; Definování kanálů na které jsou jednotlivé extension připojeny
        KITCHEN => Zap/3
        STUDY => Zap/4
        HALL => Zap/5
; V případě že chceme provést odchozí volání, které linky použijeme
      OUTGOING => Zap/1&Zap/2
```

Vytvoření těchto definicí ve skutečnosti neprovádí vůbec nic. Systém ve skutečnosti například ví co má dělat s proměnnou nazvanou „INCOMING“. Je na vás definovat požadované proměnné a poté je použít v dialplanu požadovaných způsobem.

V tomto případě byly všechny proměnné napsány s použitím velkých písmen, ačkoliv globální proměnné nejsou citlivé na velikost písma. Použití velkých písmen je pouze konvencí pro odlišení od proměnných kanálů, které jsou obvykle psány ve smíšeném tvaru.

2.1.3 Používání proměnných

Asterisk může používat globální nebo kanálově specifické proměnné jako argumenty příkazů. Proměnné jsou odkazovány použitím následujícího syntaxu:

```
${foo}
```

Kde „foo“ je jméno dané proměnné. Jméno proměnné může být libovolný alfanumerický řetězec začínající písmenem. Uživatelsky definovaná jména proměnných nejsou citlivá na velikost písma. Oproti tomu proměnné definované Asteriskem (globální) jsou na tuto velikost citlivá. Například ${EXTEN} bude funkční, ale ${exten} nebude.

V systému Asterisk existují tři typy proměnných: globální proměnné, kanálové proměnné a proměnné prostředí.

- *Globální proměnné* mohou být nastaveny buď v části [globals] souboru extensions.conf nebo použitím příkazu *SetGLobalVar*. Jakmile jsou definované, mohou být odkazovány kdykoliv jakýmkoliv kanálem.
- *Kanálové proměnné* jsou nastavovány použitím příkazu *SetVar*. Každý kanál obdrží vlastní prostor proměnných, takže neexistuje možnost kolize mezi rozdílným voláním a proměnná je automaticky zničena po zavěšení kanálu.
- *Proměnné prostředí* poskytují možnost přístupu k proměnným prostředí unix v Asterisk.

Jestliže definujete proměnnou prostředí se shodným jménem jako globální proměnnou, pak odkazy na toto jméno vrátí hodnotu kanálové proměnné. Například předpokládejme, že definujete context „FooTest“ s jednoduchým extension 100 s následující definicí:

```
[FooTest]
exten => 100,1,SetGlobalVar(FOO=5)
exten => 100,2,NoOp(${FOO})
exten => 100,3,NoOp(${foo})
exten => 100,4,SetVar(foo=8)
exten => 100,5,NoOp(${FOO})
exten => 100,6,NoOp(${foo})
```

Příkaz NoOp použijeme z důvodu pomoci zkoušení programu. Jestliže vytočíte extension 100 v context FooTest a máte spuštěný Asterisk s konzolou generující hlášení, můžete obdržet podobné hlášení systému:

```
- Executing SetGlobalVar("Zap/1-1", "FOO=5") in new stack
- Setting global variable 'FOO' to '5'
- Executing NoOp("Zap/1-1", "5") in new stack
- Executing NoOp("Zap/1-1", "5") in new stack
- Executing SetVar("Zap/1-1", "foo=8") in new stack
- Executing NoOp("Zap/1-1", "8") in new stack
- Executing NoOp("Zap/1-1", "8") in new stack
```

Zjistíte, že volání *SetGlobalVar*, ${FOO} a ${foo} vrátí hodnotu globální proměnné (konkrétně 5). Po volání *SetVar*, globální proměnná „foo“ byla zatemněna kanálovou proměnnou „foo“, poté ${FOO} a ${foo} dají hodnotu 8. Nicméně hodnota globální proměnné zůstává nezměněna (5) a každý jiný kanál odkazující na globální proměnnou ${foo} bude stále udávat hodnotu 5.

2.1.4 Proměnné a výrazy

Také existuje podpora pro používání proměnných ve tvaru ${VARIABLENAME}. Rovněž můžete použít výraz ve tvaru $[EXPRESSION], kde může být výraz regulárním výrazem, porovnáním, přírůstkem, odčítáním atd. Pro důkladnější vysvětlení výrazů a proměnných se podívejte do Asterisk dokumentace.

2.1.5 Funkce manipulace s řetězci

Délka řetězce

${LEN(foo)} - vrací délku řetězce nazvaného „foo“.

Příklad:

```
exten => 100,1,SetVar(Fruit=pear)
exten => 100,2,NoOp(${LEN(Fruit)})
exten => 100,3,NoOp(${LEN(${Fruit})})
```

První NoOp by ukázal hodnotu 5 (délka řetězce „fruit“). Druhý NoOp by ukázal hodnotu 4 (délka řetězce „pear“). Toto je možnost jak kontrolovat NULL nebo prázdný řetězec.

Podřetězce

${foo:offset:length}

- vrací podřetězec řetězce „foo“, začíná s daným offsetem a vrací další znaky

- Jestliže je offset záporný, vezme pozici doleva od pravého konce řetězce
- Jestliže je délka vynechána nebo záporná, poté všechny zbývající části řetězce začínají s offsetem.

Příklad:

${123456789:1} - vrací řetězec 23456789
${123456789:-4} - vrací řetězec 6789
${123456789:0:3} - vrací řetězec 123
${123456789:2:3} - vrací řetězec 345
${123456789:-4:3} - vrací řetězec 678

Příklad použití:

```
exten => _NXX.,1,SetVar(areacode=${EXTEN:0:3})
        - obdrží první 3 čísla z ${EXTEN}
exten => _516XXXXXXX,1,Dial(${EXTEN:3})
        - obdrží všeschny ale první 3 čísla z ${EXTEN}
exten => 100,1,SetVar(whichVowel=4)
exten => 100,2,SetVar(foo=AEIOU:${whichVowel}:1)
        - nastaví ${foo} na samostatné písmeno „U“
```

Zřetězení řetězců

Jednoduché zřetězení dvou řetězců:

```
${foo}${bar}
555${theNumber}
${longDistancePrefix}555${theNumber}
```

2.2 Context a extension

Číslovací plán se skládá ze souboru několika context. Tyto definované context jsou nejdůležitější částí konfiguračního souboru extensions.conf a jsou jednou z nejdůležitějších částí celé konfigurace systému. Context je pouze souborem několika extensions.

Příklad:

Context "mainmenu":

Extension	Description
s	Uvítací zpráva a instukce
1	Prodej
2	Podpora
3	Účtování
9	Adresář
#	Zavěšení

Tento context s názvem „mainmenu“ má pouze jedno číselnou volbu. Bod extension „s“ je počátečním extension kde volající začíná. Tento extension může například přehrát uvítací zprávu. Pro volbu Prodej stiskněte 1, pro Podporu stiskněte 2, pro Účtování volte 3, volba 9 je

pro adresář společnosti, případně volte „#“ pro zavěšení. Ve skutečnosti může každá tato položka buď vytočit něčí skutečné extension nebo může volajícího směrovat do dalšího menu.

Context mohou být používány za účelem implementování čísel důležitých rysů, jako například:

- Security - povolení mezinárodních volání pouze z určitých telefonů
- Routing - směrování hovorů na základě extension
- Autoattendant - uvítání volajících a vyzvání k volbě extension
- Multilevel menus - menu pro prodej, podporu atd.
- Authentication - dotaz na přístupové heslo pro určité extension
- Callback - redukce mezinárodních poplatků
- Privacy - seznam nežádoucích volajících čísel na dané extension
- PBX Multihosting - lze vytvářet „virtuální hosty“ na PBX
- Daytime/Nighttime - lze měnit chování na základě hodin
- Macros - vytváření skriptů pro běžně používané funkce

Když Asterisk obdrží na kanále příchozí spojení, podívá se do context definovaného pro konkrétní kanál a tím určí co má systém provést. Context definuje různé soubory příkazů závisejících na skutečnosti, kterou extension uživatel vytočil. Například může context poskytnout jeden soubor příkazů pro stav, kdy uživatel vytočil „123“ a jiný soubor příkazů pro stav, kdy uživatel vytočil „9“ nebo třeba číslo začínající „555“.

Pro některé druhy spojení (například příchozí volání z venkovní telefonní linky) nevolí uživatel žádnou extension. V tomto případě se systém zachová jakoby uživatel volil speciální extension nazvaný „s“ (jako Start). Asterisk hledá extension se jménem „s“ v definicích context kanálu, kterému jsou instrukce určeny pro ovládání volání.

Pro příklad předpokládejme, že máte kanál „Zap/1“ spojený s telefonním přístrojem v budově. Dále předpokládejme, že v konfiguračním souboru pro Zap kanály (zapata.conf) máte definován context = john pro Zap kanál 1. Takže, použijeme-li telefonní přístroj pro vytočení čísla, Asterisk se podívá na context se jménem „john“ za účelem zjištění co má v tomto případě provést. Definici v souboru extensions.conf začínáte prostřednictvím vložení jména context do hranatých závorek na samostatném řádku, například:

```
[john]
```

Pro každý context potřebujete definovat jednu nebo více extension, které Asterisk používá pro porovnávání s vytáčeným číslem. Pro každý extension musíte také systému definovat skutečnost, co má v daném případě provádět a to prostřednictvím souboru příkazů.

2.2.1 Používání context

Když obdrží Asterisk volání příslušející k dané extension (buď příchozí volání z venkovní nebo z vnitřní extension). Určení kterému context volání přísluší, závisí na skutečnosti, kterým kanálem volání přišlo. Konfigurujete-li kanály prostřednictvím Asterisk PBX, jednou z věcí kterou musíte provést je definovat do kterého context bude příchozí volání na kanále umístěno. K tomuto lze použít následující definici:

```
context = incoming
```

Prvním způsobem jak používat context je přinutit Asterisk provádět rozdílné věci v závislosti na skutečnosti odkud volání přichází. Určitě budete mít definován více než jeden context pro to, jak systém ovládá příchozí volání na Vaši telefonní linku:

- může vyzvánět některý z Vašich extension

- může přehrávat zprávu
- může nahrávat voicemail zprávu
- a tak dále …

Asterisk můžete používat pro ovládání spojení z vnitřních extension - může být povoleno vytáčení různých extension nebo provádění odchozího hovoru - tímto způsobem definujete skutečnost, že volání z různých kanálů směřuje do různých context.

2.2.2 Začlenění context

Jeden extension context může obsahovat jiný. Například uvažujme následující contexty:

Context "default":

Extension	Description
101	Mark Spencer
102	Wil Meadows
0	Operator

Context "local":

Extension	Description
_9NXXXXXX	Local calls
include => "default"	

Context "longdistance":

Extension	Description
_91NXXNXXXXXX	Long distance calls
include => "local"	

Máme definovány následující contexty:

- Context *default* - povoluje volání tří telefonních extension (Mark, Wil a Operator).
- Context *local* - obsahuje jeden vzor extension pro povolení vytáčení pouze 7 číselné volby (místní volání) a také obsahuje context „default“, tedy uživateli zpřístupní volání na extension Mark, Wil a Operator.
- Context *long distance* - má jeden vzor extension pro přístup k meziměstským volání a obsahuje context „local“ což umožňuje místní volání a také volání na extension Mark, Wil a Operator.

Použitím vhodných kontextů extension můžete bezpečně kontrolovat skutečnost kdo má přístup k placeným službám.

2.3 Extensions

Extension může být ve dvou typech: *prostý* nebo *vzor*.

Prostý extension může být číslo (např. 1234) a může také obsahovat standardní symboly * a #, které se objevují na běžných telefonech (např. 12#34*56 je platná extension). Některé telefonní klávesnice mají také speciální DTMF tlačítka označované A, B, C a D, extension může být poté definována i prostřednictvím těchto písmen. Ve skutečnosti může jméno extension obsahovat každé písmeno nebo číslici, právě tak jako některé interpunkční značky.

Některé VoIP telefony jsou schopny „vytáčet“ extension „čísla“, které mohou mít tvar libovolného řetězce (např. „Office“). V Asterisku je také možné definovat extension ve tvaru „Office“.

Odpověď na otázku zda jsou jména extension citlivá na velikost písma není zcela jednoznačná. Na velikost písma jsou citlivá v případě, že Asterisk zkouší porovnat extension vytočené uživatelem proti extension definovaném pro context. V tomto případě musí obě extension souhlasit a to včetně velikosti písma. Takže jestliže uživatel prostřednictvím VoIP telefonu vytočí „OFFICE“, systém nebude provádět příkazy definované pro extension „Office“. Na druhou stranu nejsou jména extension citlivá na velikost písma v případě, že nedefinujete rozdílnou extension (ve stejném context), která má stejné jméno lišící se pouze ve velikosti písma. Tedy nemůžete definovat jeden soubor příkazů pro extension „Office“ a jiný soubor příkazů pro extension „OFFICE“.

2.3.1 Jména Extensions

Extension Dialplan mohou být jednoduchá čísla jako „421“ nebo „0“. Mohou to být také alfanumerické jména jako „james“ nebo „bond007“. Třebaže typický telefon není schopen volit extension nazvaný „james“ (některé i toto umožňují), často může Váš logický Dialplan zahrnovat skoky z jedné extension do jiné extension a pro tyto skoky můžete definovat extension jména s jakýmkoliv jménem chcete, ačkoliv si nepřejete je vytáčet přímo.

Samozřejmě tlačítkové telefony nemají pouze číslice 0 – 9, mají také znaky „*“ a „#“. Některé tlačítkové telefony mají také čtyři extra číselné znaky (A, B, C a D). Jestliže máte v organizaci takovýto handset, nic vám nebrání vytvořit možnost použití těchto dodatečných tlačítek pro vlastní speciální účely.

Poznámka: Pro vytvoření extension, které je spouštěno volení symbolu „#“ musíte použít vzor extension. Asterisk nerozeznává symbol „#“ jako běžný číselný znak, třebaže se tento objevuje na všech telefonech s volbou DTMF.

2.3.2 Předdefinovaná jména extension

Pro některé speciální účely používá Asterisk následující extension:

i :	Invalid	neplatný
s:	Start	začátek
h:	Hangup	zavěšení
t :	Timeout	překročení časového limitu
o :	Operator	operátor

2.3.3 Definování extension

Na rozdíl od tradiční PBX, kde jsou extension asociovány s telefonem, rozhraním, menu atd., jsou v Asterisk extension definovány jako seznam příkazů určených k provedení. Příkazy jsou obecně vykonávány v pořadí v jakém jsou definovány prostřednictvích svých prioritních tag. Přesto však některé příkazy (jako příkazy *Dial* a *GotoIf*) mají schopnost přesměrování na jiné místo na základě definované podmínky.

V případě, že je vytočena extension, je proveden příkaz označený prioritou 1, následně příkaz s prioritou 2 atd. Toto je prováděno dokud nenastane některý z následujících stavů:

- Volání je zavěšeno
- Příkaz vrátí kód výsledku –1 (indikace poruchy)

- Příkaz s některou vyšší prioritou neexistuje (poznamenejme, že systém nepřeskočí chybějící prioritu)
- Volání je směrováno na novou extension

V syntaxi souboru extensions.conf je každý krok příkazu v extension napsán v následujícím formátu:

exten => pattern,priority,Command(parameters)

Syntax pro definování extension v Asterisk konfiguračním souboru extensions.conf je spíše těžkopádná. Naštěstí je zde několik pomocných nástrojů (GUI tool).

Definice jednotlivé extension obsahuje jeden nebo více příkazů. Každý tento příkaz je uveden na samostatném řádku v následujícím formátu:

exten => extension,priority,Command(parameters)

- *extension* je jméno extension, ať už písmenné jméno nebo vzor extension. Toto jméno je přesně opakováno pro každý další příkaz s nižší prioritou pro dané extension.
- *priority* je číslo typu Integer. Pojmenování „priorita“ je částečně zavádějící, protože se ve skutečnosti jedná pouze o číslo příkazu příslušejícímu k extension. Vlastní provádění příkazů začíná systém na místě priority 1. Jestliže se řádek s prioritou 1 nevyskytuje, pak extension nebude odpovídat žádnému volenému číslu. Jestliže po řádku s prioritou 1 není řádek s prioritou 2, pak Asterisk ukončí provádění pro danou extension a to i v případě, že jsou dále uvedeny příkazy s prioritou 3 nebo více. Nicméně při provádění některých příkazů může systém provést skok (v závislosti na logice příkazu) na jinou prioritu, než která logicky následuje.
- *command* je jméno prováděného příkazu (také nazýváno „application“). Více v seznamu Asterisk příkazů.
- *parameters* závisí na daném příkazu. Některé příkazy tyto parametry nevyžadují, lze je vynechat.

Příklad:

```
exten => 123,1,Answer
exten => 123,2,Playback(tt-weasels)
exten => 123,3,Voicemail(44)
exten => 123,4,Hangup
```

Toto je definice jednoduché extension nazvané „123“. V případě, že je prováděno volání na extension „123“, Asterisk na volání odpoví sám, přehraje zvukový soubor nazvaný „tt-weasels“, poskytne uživateli možnost zanechat voicemail zprávu v mailbox (44) a poté zavěsí.

Asterisk se nestará o pořadí ve kterém vkládáte řádky do souboru extensions.conf. Můžete míchat řádky v různém pořadí. Tato skutečnost však na funkci nemá vliv, protože systém pro určení skutečnosti, který příkaz má provést, použije prioritu.

Další volby pro definování extension obsahují volby obvykle označovány jako „logika ex-přítelkyně“. Tato logika odpovídá volené extension, jestli přichází zvenku nebo zevnitř, na základě callerID volající osoby.

Příklad:

```
exten => 123/100,1,Answer()
exten => 123/100,2,Playback(tt-weasels)
exten => 123/100,3,Voicemail(123)
```

```
exten => 123/100,4,Hangup()
```

Toto odpovídá extension 123 a vykoná následující volby POUZE je-li Caller-ID Number volajícího účastníka 100. Toto může být také uděláno odpovídajícím vzorem, jak je vidět dále:

```
exten => 1234/_256NXXXXXX,1,Answer()
```

Toto odpovídá pouze pro extension 1234 a to jestliže Caller ID Number začíná číslem 256. Tento tvar je velmi užitečný pro zachování locals pro vytáčení bezplatných čísel a nákladů za volání.

Příklad:

```
exten => s,1,Answer
exten => s/9184238080,2,SetCIDName(EVIL BASTARD)
exten => s,2,SetCIDName(Good Person)
exten => s,3,Dial(SIP/goodperson)
```

V podstatě volání přichází, v prioritě 2 oddělíte osoby které nemáte rádi. Kdokoliv jiný zůstává ve správném směru a priorita 3 vrací každého do hlavního směru.

2.3.4 Vzory Extension

Jména extension nejsou tedy omezena na jednotlivé specifické extension „čísla“. Jednotlivé extension mohou rovněž odpovídat vzorům. V souboru extension.conf se jméno extension stává vzorem, jestliže začíná znakem „podtržítko“. V extension vzorech mají následující znaky speciální smysl:

X	odpovídá některé z číslic 0 – 9
Z	odpovídá některé z číslic 1 – 9
N	odpovídá některé z číslic 2 – 9
[1237-9]	odpovídá některé číslici nebo písmenu uvedenému v závorkách (v tomto případě tedy 1,2,3,7,8,9)

Příklad:

Context "routing":

Extension	Description
_61XX	Dallas Office
_62XX	Dallas Office
_7[1-3]XX	San Jose Office
_7[04-9]XX	Los Angeles Office

Tento context nazvaný „routing“ posílá volání na různé servery podle jejich extension. Tato společnost rozhodla, že všechny telefonní extension budou dlouhé 4 čísla. Jestliže uživatel volí extension začínající 61 nebo 62 bude směrován na Dallas. Vše začínající na 71, 72 nebo 73 bude směrováno do San Jose a všechno zbývající (70, 74-79) bude směrováno na Los Angeles.

Příklad:

_NXXXXXX	odpovídá obvyklému 7 číselnému telefonnímu číslu

_1NXXNXXXXXX	odpovídá kódu oblasti a telefonnímu číslu předcházejícímu jedničkou
_9011.	odpovídá libovolnému řetězci pěti znaků začínajících 9011, neznamená to však vlastní znakový řetězec 9011
_#	odpovídá jednomu stisknutí tlačítka „#“

Upozornění: Nepoužívejte vzor „_.“ Protože tento tvar znamená všechno a to včetně speciálních Asterisk extension jako „i“, „t“, „h“ atd. Místo toho použijte například „_X.“ nebo „_X“ což nemá význam speciálních extension.

2.4 Pomocné funkce

2.4.1 Znovu zavedení (Reloading)

Pokud požadujete opětovné zavedení dialplan po provedených změnách a to bez reloadu celého Asterisk, použijte Asterisk CLI příkaz *extensions reload*.

2.4.2 Problematika rozdělení souboru

Pomocí příkazu *#include <filename>* v souboru extensions.conf jsou do tohoto souboru zahrnuty další soubory. Tímto způsobem můžete nastavit systém, kde je extensions.conf pouze hlavním souborem, ale ne jediným. V dalším kroku můžete vytvořit například soubor users.conf obsahující lokální uživatele, soubor services.conf obsahující různorodé služby, atd. Touto cestou může být dialplan lépe a snadněji udržován a to bez závislosti na velikosti vašeho nastavení. Příkaz *#include <filename>* není shodný s příkazem *include <context>*. Příkaz *#include* pracuje ve všech Asterisk konfiguračních souborech.

2.4.3 Forwarding na jiný Asterisk

Syntax:

```
[iaxprovider]
switch => IAX2/user:[key]@server/context
```

Specifikuje forwarding na jiný Asterisk. Potřebný *user* a *key* budou definovány v souboru iax.conf volaného serveru. Použije se context v souboru extensions.conf volaného serveru.

2.4.4 Ovládání extensions.conf zvenku

- Asterisk extension z mysql
- Jako u všech souborů typu *.conf můžete použít příkaz *#include* pro zahrnutí jiného souboru

Příklad: Jednoduchý extensions.conf použitím příkazu #include:

```
#include "my-extra-config-file"

[globals]
ALL=Zap/1&SIP/1000&SIP/1001
```

```
[default]
exten => s,1,Answer
exten => s,2,Playback(welcome-message)
exten => s,3,Goto(context-in-include-file,s,1) ; go to context defined in included file
… atd
```

2.4.5 Použití makra pro vytvoření extension

Příklad:

```
[globals]
PHONE1=Zap/1
PHONE2=SIP/6002

[macro-oneline]
exten => s,1,Dial(${ARG1},20,t)
exten => s,2,Voicemail(u${MACRO_EXTEN})
exten => s,3,Hangup
exten => s,102,Voicemail(b${MACRO_EXTEN})
exten => s,103,Hangup

[local]
exten => 6601,1,Macro(oneline,${PHONE1})
exten => 6602,1,Macro(oneline,${PHONE2})
```

2.5 Předdefinované kanálové proměnné

Existují některé kanálové proměnné nastavené prostřednictvím systému, které můžete použít v definici vašeho dialplan. Tyto proměnné jsou, oproti uživatelem definovaným proměnným, citlivé na velikost použitého písma.

- ${ACCOUNTCODE}: Kód účtu, jestliže je definován (viz. Asterisk billing)
- ${ANSWEREDTIME}: Čas kdy bude volání odpovězeno.
- ${CALLERID}: Aktuální Caller ID (jméno a číslo)
- ${CALLERIDNAME}: Aktuální Caller ID jméno
- ${CALLERIDNUM}: Aktuální Caller ID číslo
- ${CALLINGPRES}: Proměnná PRI Call ID Presentation pro příchozí volání
- ${CHANNEL}: Aktuální jméno kanálu
- ${CONTEXT}: Jméno aktuálního context
- ${DATETIME}: Aktuální datum a čas ve formátu DDMMYYYY-HH:MM:SS
- ${DIALEDPEERNAME}: Jméno volané strany. V současnosti „rozbité“, viz. DIALEDPEERNAME
- ${DIALEDPEERNUMBER}: Číslo volané strany. V současnosti „rozbité“, viz. DIALEDPEERNUMBER
- ${DIALEDTIME}: Čas kdy bylo číslo voleno.
- ${DIALSTATUS}: Stav volání. Viz. DIALSTATUS
- ${DNID}: Identifikátor voleného čísla (Dialed Number Identifier). Limitované použití viz. DNID
- ${EPOCH}: Aktuální UNIX-style epocha (číslo v sekundách od 1 Jan 1970)

- ${EXTEN}: Aktuální extension
- ${HANGUPCAUSE}: Poslední návratová kód zavěšení na Zap kanále připojeném k PŘI rozhraní
- ${INVALID_EXTEN}: Žádané extension při přesměrování na „i“ (invalid) extension
- ${LANGUAGE}: Aktuální jazykové nastavení
- ${MEETMESECS}: Číslo v sekundách určující jak dlouho se úživatel účastnil MeetMe konference
- ${PRIORITY}: Aktuální priorita
- ${RDNIS}: Aktuální přesměrování DNIS, Caller ID které přesměrovalo volání. Limitované použití viz. RDNIS
- ${SIPDOMAIN}: SIP destination domain volání v pásmu (inbound)
- ${SIP_CODEC}: Použito pro nastavení SIP kodeku pro volání (patrně „rozbité“ ve verzi 1.0.1, v pořádku verze 1.0.3 & 1.0.4, nevíme o verzi 1.0.2)
- ${SIPCALLID}: SIP dialog Call-ID: záhlaví
- ${SIPUSERAGENT}: SIP user agent záhlaví
- ${TIMESTAMP}: Aktuální datum a čas ve formátu YYYYMMDD-HHMMSS
- ${TXTCIDNAME}: Výsledek aplikace TXTCIDName
- ${UNIQUEID}: Aktuální unikátní identifikátor volání (call unique identifier)
- ${TOUCH_MONITOR}: použito pro „jedno dotekový záznam“ (viz. features.conf a wW dial flags). Jestliže je toto nastaveno na obou stranch volání, poté var obsahuje app_args pro app_monitor neboli je použit default WAV||m

2.6 Aplikačně specifické proměnné

Některé aplikace zabírají extra vstupy nebo poskytují extra výstupy používáním kanálových proměnných.

- ChanIsAvail vrací ${AVAILCHAN}: První dosažitelný kanál
- Dial bere vstup z ${VXML_URL}: Posílá XML Url telefonu Cisco 7960
- Dial bere vstup z ${ALERT_INFO}: Nastavuje tempo vyzvánění Cisco telefonů
- Dial vrací ${CAUSECODE}: Jestliže volba selže, toto je errormessage
- Dial vrací ${DIALSTATUS}: Stav návratového textového kódu posledního pokusu vytáčení
- EnumLookup vrací ${ENUM}: Výsledek vyhledávání
- MeetMe bere vstup z {MEETME_AGI_BACKGROUND}: AGI skript pro spuštění
- MeetMe vrací ${MEETMESECS}: Počet sekund jak dlouho byl uživatel v konferenci
- Hangup čte ${PRI_CAUSE}proměnná pronastavení návratových PŘI kódů (PRI return codes)
- TXTLookup vrací ${TXTCIDNAME}: Výsledek DNS vyhledávání
- AgentCallbackLogin vrací ${AGENTBYCALLERID_${CALLERID}}: ID úspěšně přihlášeného agenta

2.7 Proměnné specifické pro makra

V případě, že použijete context makra, jsou dostupné extra kanálové proměnné.

- ${ARG1}: První argument makra
- ${ARG2}: Druhý argument makra (a tak dále)

- ${MACRO_CONTEXT}: Context extension, která spouští toto makro
- ${MACRO_EXTEN}: Extension, která spouští toto makro
- ${MACRO_OFFSET}: Nastaveno prostředniuctvím makra za účelem ovlivnění priority, kde bude vykonávání pokračovat v případě opuštění makra
- ${MACRO_PRIORITY}: Priorita v extension, kde bude makro spuštěno

2.8 Proměnné prostředí unix

Prostřednictvím následující syntaxe můžete zpřístupnit proměnné prostředí unix:

${ENV(foo)}

- ${ENV(ASTERISK_PROMPT)}: Aktuální Asterisk CLI prompt.

${ENV(RECORDED_FILE)}: jméno posledního souboru uloženého prostřednictvím příkazu Record (dostupné v CVS > 2004-03-21)

3. Instalace Asterisku

Instalace Asterisku byla provedena prostřednictvím následujícího příkazu. Tímto byla nainstalována základní distribuce Asterisku společně s konfiguračními soubory a soubory určené pro vývoj.

```
apt-get install asterisk asterisk-config asterisk-dev
```

Dodatečně byla doinstalována podpora pro vzájemné propojení s jinou SW ústřednou Asterisk pomocí protokolu IAX (Inter-Asterisk Exchange) a pro možnost implementace IAX terminálů do systému.

```
apt-get install libiax-dev libiax0
```

Dodatečně byla také doinstalována podpora pro oh323 kanál a podporu H.323 terminálů. Současně je nutno doinstalovat vhodný gatekeeper (v našem případě GnuGk) jehož konfigurace je popsána dále v textu.

```
apt-get install oh323
```

Tímto je Asterisk nainstalován a po nakonfigurování prostřednictvím úpravy konfiguračních souborů je možno jej spustit například prostřednictvím následujícího příkazu, který spustí Asterisk v tzv. konzolovém módu.

```
asterisk -vvvgc
```

Poznámka:

Z důvodu rozšíření dostupných zvukových souborů pro vytváření hlasových menu, přehrávání audio odpovědí a podobně, byly doinstalovány zvukové soubory prostřednictvím systému CVS.

```
cd /usr/src
export CVSROOT=:pserver:anoncvs@cvs.digium.com:/usr/cvsroot
cvs login                                          ; the password is anoncvs.
cvs checkout asterisk-sounds
cd asterisk-sounds
make install
```

3.1 Kontrola Asterisku pomocí CLI

Základním způsobem jak monitorovat a ovládat chování ústředny Asterisk je použití tzv. Command Line Interface (CLI). V následujícím textu je uvedeno pouze několik užitečných příkazů, úplný výčet pomocí příkazu „help“ v CLI.

Příkazem, který nám toto rozhraní CLI zpřístupní je:

```
asterisk – r
CLI>                    ; prompt označující že se uživatel nachází v CLI
```

- Nejužitečnějším příkazem je „help“ zobrazující všechny ostatní dostupné příkazy
- Zadáním počátečního písmene (případně několika) a následným použitím klávesy TAB jsou vypsány příslušné odpovídající příkazy

Následující příkazy předpokládají stav připojení k CLI:

```
iax2 debug                  - povolení IAX debugging (také pro H.323 a SIP)
iax2 set jitter             - nastavení hodnoty jitter buffer
iax2 show channels          - zobrazení aktivních IAX kanálů (také pro SIP)
iax2 show users             - zobrazení definovaných IAX uživatelů (také pro SIP)
reload                      - reload konfiguračních souborů
restart now                 - okamžitý restart
restart when convenient     - restart Asterisk, když jsou všechna volání ukončena
set verbose                 - nastavení úrovně zobrazovaných hlášení systému
show applications           - výpis všech Asterisk aplikací
show application <app>      - výpis využití specifické Asterisk aplikace
show channels               - výpis všech aktivních kanálů
show channel <channel>      - výpis informací o specifickém kanále
show codecs                 - zobrazení informací o kodecích
show dialplan               - zobrazení číslovacího plánu a příslušných akcí
show version                - zobrazení verze Asterisku
sip debut                   - povolení SIP debugging
stop now                    - okamžité zastavení Asterisku
```

4. Konfigurační soubory - Asterisk

Uvedené konfigurační soubory vychází s testovací instalace Asterisku na VŠB-TU Ostrava. V následujícím textu jsou rozebrány pouze některé důležité parametry, ostatní jsou uvedeny buď v teoretické části předchozího textu nebo v dokumentaci.

4.1 Dialplan (extensions.conf)

```
[general]                       ; Testovaci Dialplan (extensions.conf)

static=yes
writeprotect=no                 ; zamezeni prepisovani diaplan z CLI

[globals]

OPERATOR = SIP/421              ; definovany kanal/telefonni cislo
SUPPORT = SIP/424               ; odkazovani na promennou ${OPERATOR}

; Cesty pro dodatecne sound files definovane jako promenne
sounds_exten = /var/lib/asterisk/sounds/
sounds_phonetic = /var/lib/asterisk/sounds/phonetic/
sounds_letters = /var/lib/asterisk/sounds/letters/
sounds_silence = /var/lib/asterisk/sounds/silence/

; Urceni prefixu pro dany protokol - volba
; SIP_PREFIX = _42
; H323_PREFIX = _43
; SKINNY_PREFIX = _44
; IAX2_PREFIX = _45

TUO_TRUNK = IAX2/tuo:testwest@tuo
```

Proměnná TUO_TRUNK definuje propojení s další ústřednou Asterisk. Je použit protokol IAX2, přístupové jméno, heslo a server. Parametry uvedeného serveru jsou definovány v souboru iax.conf (viz. dále).

Makro nazvané „basic“ definuje základní prováděné akce, které jsou provedeny při vytočení daného čísla. Je dále použito při definování konkrétních extension. Pomocí příkazů nastavujeme základní popis volání (CallerID, CallerName). Následně je volba vytočena, v případě obsazení je signalizováno obsazení.

```
[macro-basic]

exten => s,1,SetCallerID(${CALLERID})
exten => s,2,SetCIDName(${CALLERNAME})
exten => s,3,Dial(${ARG1})
exten => s,4,Congestion
exten => s,5,Hangup
```

Makro nazvané „basic-oh323“ definuje základní prováděné akce, které jsou provedeny při vytočení daného čísla. Funkci má podobnou jako makro „basic“. Toto dodatečné makro vzniklo pro H.323 terminály optiPoint 300 advance, které nezobrazují korektně příchozí číslo, nahrazují jej symbolem „*“. Proto je do proměnných CIDNumber, CIDName a CallerID přiřazena hodnota CallerIDNumber. Přesto však konkrétně optiPoint 300 advance číslo korekně nezobrazí. Například SW telefony toto číslo zobrazí korektně (MS Neetmeeting).

```
[macro-basic-oh323]

exten => s,1,SetCIDNum(${CALLERIDNUM})
exten => s,2,SetCallerID(${CALLERIDNUM})
exten => s,3,SetCIDName(${CALLERIDNUM})
exten => s,4,Dial(${ARG1})
exten => s,5,Congestion
exten => s,6,Hangup
```

Kontext „skupina“ používáme k definování telefonních čísel. Použijeme předem definované makro „basic“ s parametrem kanál/číslo (SIP/xxx) respektive (IAX2/xxx). Znak „podtržítko“ označuje skutečnost, že čísla za tímto znakem jsou volena přímo.

```
[skupina]

; extension SIP
exten => _421,1,Macro(basic,SIP/421)
exten => _424,1,Macro(basic,SIP/424)
exten => _428,1,Macro(basic,SIP/428)    ; test SW s cryptovanim md5
exten => _429,1,Macro(basic,SIP/429)    ; test SW

; extension H.323
exten = _430,1,Macro(basic-oh323,OH323/430)
exten = _431,1,Macro(basic-oh323,OH323/431)

; extension IAX
exten => _450,1,Macro(basic,IAX2/450) ; Ethernet Phone AT-320

; extension testovani telefonu
exten => _497,1,Macro(basic,SIP/497)    ; SW phone Messenger
exten => _498,1,Macro(basic,SIP/498)    ; HW phone Grandstream Budge Tone 100
```

Pro směrování volání na telefonní číslo, které je definováno v jiném Asterisku použijeme předcházející tvar. Předpokládáme, že uvedené číslo „420“ je definováno v příslušných konfiguračních souborech jiného Asterisku (extensions.conf, sip.conf).

```
; smerovani na cislo, které není definovane v sip.conf
exten => _420,1,Macro(basic,${TUO_TRUNK}/${EXTEN})
```

Na telefonním čísle 499 je umístěn „echotest“. Jsou použity audio soubory ve formátu *.GSM. Cesta k danému souboru je definována prostřednictvím proměnných (${cesta}).

```
; „echotest“ – „Bravo Charlie Tango is not available“
```

```
exten => _499,1,Answer()
exten => _499,2,Playback(${sounds_phonetic}bravo)
exten => _499,3,Wait(1)
exten => _499,4,Playback(${sounds_phonetic}charlie)
exten => _499,5,Wait(1)
exten => _499,6,Playback(${sounds_phonetic}tango)
exten => _499,7,Wait(1)
exten => _499,8,Playback(${sounds_exten}T-is-not-available)
exten => _499,9,Wait(2)
exten => _499,10,Hangup()
```

Uvedené Voicemenu nejprve přehraje úvodní oznámení, ve kterém je uživatel přivítán a je mu nabídnuta volba „1“ pro volání Podpory a volba „2“ pro volání Operátora. Nejprve je však definován čas 20 sekund pro provedení nabídnuté volby (ResponceTimeout). Pokud je definovaný čas vyčerpán je hovor automaticky ukončen. Po provedení volby uživatelem je pomocí příslušných kontextů volání provedeno.

```
; Jednoduche voicemenu smerujici hovory na zaklade stisknuteho tlacitka
exten => _456,1,Wait(2)
exten => _456,2,Answer()
exten => _456,3,DigitTimeout,10
exten => _456,4,ResponseTimeout,20
exten => _456,5,Playback(${sounds_exten}welcome)
exten => _456,6,Playback(${sounds_exten}thank-you-for-calling)
exten => _456,7,Wait(1)
exten => _456,8,Playback(${sounds_exten}press-1)
exten => _456,9,Playback(${sounds_exten}for-tech-support)
exten => _456,10,Wait(1)
exten => _456,11,Playback(${sounds_exten}press-2)
exten => _456,12,Playback(${sounds_exten}speak-to-the-operator)
exten => 1,1,Goto(context_support,s,1)
exten => 2,1,Goto(context_operator,s,1)
exten => t,1,Hangup

[context_operator]
exten => s,1,Dial(${SUPPORT})
exten => s,2,Hangup()

[context_support]
exten => s,1,Dial(${OPERATOR})
exten => s,2,Hangup
```

4.2 Session Initiation Protocol (sip.conf)

Dále uvedený konfigurační soubor je zkrácen o definování jednotlivých uživatelských čísel, jejichž definice jsou shodné s definovaným číslem „421“.

```
[general]

context=skupina          ; Default context pro prichozi volani
recordhistory=yes        ; Povoleni ukladani SIP history
port=5060                ; UDP Port (pro SIP 5060)
bindaddr=0.0.0.0         ; IP adresa pro navazani spojeni (0.0.0.0 pro vsechny)
srvlookup=yes            ; Povoleni DNS SRV lookup na odchozi volani
tos=lowdelay             ; Nastaveni QoS parametru ToS
                         ; (lowdelay,throughput,reliability,mincost,none)
disallow=all             ; Zakazani vsech kodeku
allow=alaw               ; povoleni vybranych kodeku
allow=gsm
allow=ulaw
musicclass=default       ; Nastaveni hudby pri přidrženi hovoru
rtptimeout=60            ; Ukonceni volani kdyz neni 60 sec RTP aktivita
                         ; (volani není ve stavu hold)
rtpholdtimeout=300       ; Ukonceni volani když není 300 sec RTP aktivita
                         ; (volani ve stavu hold)
nat=yes                  ; NAT nastavení (yes, no, never, route)
```

Základní definice uživatele s číslem „421“ má následující tvar:

```
[421]
type=friend
context=skupina
language=cz
host=dynamic
username=421
secret=heslo
callerid=David <421>
```

Následující definice uživatele s číslem „428“ umožňuje použití kryptovaného zápisu hesla prostřednictvím md5. Pro generování hesla ve tvaru md5 použijeme příkaz (realm je nastaven default na „asterisk“):

```
echo -n "<user>:<realm>:<secret>" | md5sum
```

```
[428]
type = friend
context = skupina
language = cz
host = dynamic
auth = md5                                      ; povoleni kodovani md5
md5secret = efe5082a9b433e7e0d316bc48b911c7f    ; zakodovane heslo
callerid = md5 <428>
```

Pokud chceme omezit IP adresy ze kterých je uživateli umožněno se registrovat je provedeno pomocí příkazů deny a permit.

```
[429]
deny=0.0.0.0/0.0.0.0                ; zamezeni registrace k danemu uctu ze
                                    ; vsech IP adres
permit=158.196.251.96               ; povoleni registrace k uctu z konkretni IP adresy
;permit=158.196.251.1/255.255.255.0    ; povoleni registrace k uctu ze vsech
                                       ; IP adres dane site
type=friend
context=skupina
language=cz
host=dynamic
username=429
secret=heslo
callerid=SJphone <429>
```

4.3 Inter-Asterisk Exchange (iax.conf)

Propojení mezi Asterisky jsme provedli pouze pro některá volená čísla. Na tomto popisovaném jsou definována čísla SIP/421 a SIP/424. Druhý Asterisk je na CVT Technické univerzity Ostrava a je na něm definováno číslo 420/SIP. Obě SW ústředny samozřejmě obsahují více čísel, ale vzájemná kompatibilita je řešena prozatím pouze pro tyto uvedené čísla. Prostřednictvím ústředny na CVT lze dále přistupovat také do ostatních lokalit s ústřednou Asterisk (Karviná, Opava, Orlová).

```
[general]
bindport=4569
iaxcompat=yes
delayreject=yes         ; zpozdeni odpovedi na autorizaci pri pritomnosti hesla
bandwidth=high
jitterbuffer=no

disallow = all          ; zakazani vsech kodeku
allow = alaw            ; povoleni pouzivanych kodeku pro komunikaci
allow = ulaw
allow = gsm
allow = ilbc            ; povoleno kvuli SW klientum
allow = speex           ; povoleno kvuli SW klientum

tos=lowdelay            ; Nastaveni QoS parametru ToS
                        ; (lowdelay,throughput,reliability,mincost,none)
```

Pro nastavení informací o daném připojovaném serveru použijeme následující zápis. Použijeme kódovanou autorizaci prostřednictvím md5, definujeme uživatelské jméno, heslo a host IP adresu.

```
[tuo]
type=friend
auth=md5
```

```
username=asterisk
secret=testwest
host=195.113.113.24
context=virtual
trunk=yes
```

Základní definice uživatele s číslem „450“ je následující:

```
[450]
type=friend
context=skupina
language=cz
host=dynamic
username=450
secret=heslo
callerid=IAX1 <450>
```

Jedná se opět o základní nastavení uživatelského účtu shodné s konfigurací SIP účtů. Je tedy dále možné použít také kryptování hesla nebo omezení IP adresy, z které je možno se k danému účtu přihlásit.

4.4 Asterisk OpenH323 (oh323.conf)

Kapitola se zabývá konfigurací oh323 kanálu v systému Asterisk. Konkrétní instalaci kanálu oh323 do systému Asterisk lze provést pomocí obvyklého příkazu instalace balíčku přímo pomocí internetového připojení *apt-get*.

```
apt-get install oh323
```

Dále bude uvedena základní struktura konfiguračního souboru oh323.conf.

```
[general]

listenAddress=0.0.0.0   ; adresa na BIND pro příchozí spojení
listenPort=1717         ; nastavení portu
tcpStart=10000          ; konfigurace rozsahu TCP portů používaných H.323
tcpEnd=20000
udpStart=10000          ; konfigurace rozsahu TCP portů používaných H.323
udpEnd=20000
fastStart=no            ; povolení metody FastStart
```

```
h245Tunnelling=no       ; H.245 tunnelling (yes,no)
h245inSetup=no          ; povolení H.245 zpráv již ve zprávě SETUP
inBandDTMF=no           ; detekce DTMF uvnitř pásma
silenceSuppression=no   ; potlačení ticha v hovoru
jitterMin=20            ; jitter buffer (jednotkou ms, 20 … 10000)
jitterMax=120
ipTos=lowdelay          ; Nastaveni QoS parametru ToS
                        ; (lowdelay,throughput,reliability,mincost,none)
```

```
outboundMax=10      ; nastavení maximálního počtu ochozích H.323 spojení
inboundMax=10       ; nastavení maximálního počtu příchozích H.323 spojení
simultaneousMax=10  ; nastavení maximálního počtu současných H.323 spojení
```

Následující sekce provádí nastavení voleb sledování systémových hlášení Asterisku. Parametr „libTraceFile“ může být linux „stdout“ nebo úplná cesta k tracefile. Do příslušného výstupu jsou ukládány pouze systémová hlášení příslušná OpenH323.

```
wrapLibTraceLevel=1
libTraceLevel=0
libTraceFile=stdout
```

```
gatekeeper=158.196.251.129      ; IP adresa gatekeepera - GnuGk
gatekeeperTTL=600               ; nastavení timeout pro registraci
```

```
userInputMode=TONE      ; nastavení módu pro vysílání uživatelského vstupu
                        ; (Q931, STRING, TONE, RFC2833)
amaFlags=default        ; AMA flags (default, omit, billing, documentation)
accountCode=H323
context=skupina         ; nastavení default kontextu pro H.323 volání
```

Konfigurace H.323 aliasů, prefixů a souvisejících Asterisk kontextů. Zjednodušeně řečeno se jedná o prefixy, které lze volit z H.323 terminálů

```
[register]
context=skupina
gwprefix=41
gwprefix=42
gwprefix=45
gwprefix=2
gwprefix=3
gwprefix=5
gwprefix=9
```

Specifikace a konfigurace příslušných kodeků podporovaných pro H.323 komunikaci. Je definován povolený typ kodeku a příslušná hodnota frame size. V systému oh323 kanálu je umožněno použití následujících kodeků:

- G711U - G.711 u-Law
- G711A - G.711 A-Law
- G7231 - G.723.1(6.3k)
- G72316K3 - G.723.1(6.3k)
- G72315K3 - G.723.1(5.3k)
- G7231A6K3 - G.723.1A(6.3k)
- G7231A6K3 - G.723.1A(6.3k)
- G726 - G.726(32k)
- G72616K - G.726(16k)
- G72624K - G.726(24k)

- G72632K - G.726(32k)
- G72640K - G.726(40k)
- G728 - G.728
- G729 - G.729
- G729A - G.729A
- G729B - G.729B
- G729AB - G.729AB
- GSM0610 - GSM 0610
- MSGSM - Microsoft GSM Audio Capability
- LPC10 - LPC-10

```
[codecs]
codec=G711A
frames=20
codec=G711U
frames=20
```

5. Konfigurační soubory - GnuGk

Kapitola se zabývá využitím GnuGk ve funkce gatekeepera potřebného pro podporu H.323 kanálu v SW ústředně Asterisk. GnuGk je plně H.323 kompatibilní gatekeeper dostupný na stránkách projektu http://www.gnugk.org. Zde je uvedena pouze konfigurace umožňující přihlášení libovolného terminálu, která je pouze omezena požadavkem na tvar čísla 43x a příslušnou definicí čísla v souboru číslovacího plánu (extensions.conf). Pokročilejší autorizace je řešena například v technické zprávě „GNU Gatekeeper a jeho nasazení v síti CESNET2“.

Konkrétní instalaci gnugk lze provést pomocí obvyklého příkazu instalace balíčku přímo pomocí internetového připojení *apt-get*.

```
apt-get install gnugk
```

5.1 Kontrola GnuGk pomocí Telnet

Nejprve definujeme co pro GnuGk představuje pojem *Status Port*. *Status port* je externí rozhraní pro monitorování a kontrolu gatekeepera. Gatekeeper posílá ven zprávy o ochozích voláních na všechny připojené klienty a může přijímat příkazy skrze toto rozhraní.

Zprávy vysílané GnuGk na Status Port jsou uspořádávány do následujících tří výstupních trace úrovní:

- Level 0 – oznámení a přímé odpovědi na zadávané příkazy
- Level 1 – oznámení, přímé odpovědi na zadávané příkazy, CDR a Route Request
- Level 2 – veškerý výstup. Toto je default výstupní level.

```
gnugk -r                          - spuštění GnuGk v Routed módu pro všechna volání
telnet <ip.address> 7000          - standardní přístup ke GnuGk prostřednictvím telnet a default
                                    nastaveného portu 7000
```

Následující příkazy předpokládají stav připojení ke status portu (např. pomocí telnet):

reload	- znovunačtení konfigurace GnuGk
setlog /var/log/gnugk/trace_1.log	- určení výstupního souboru pro uložení zachycených dat
debug trc 5	- provedení debug s danou úrovní trc 5 pro zachycení signalizace do souboru
version	- zobrazení verze a informací o OS gatekeepera
statistics	- výpis statistických informací o GnuGk
PrintAllRegistrations	- výpis všech registrovaných koncových bodů
PrintAllRegistrationsVerbose	- zobrazení detailů o registrovaných koncových bodech
PrintCurrentCalls	- výpis všech aktuálních volání používajících stejný ACF syntax
PrintCurrentCallsVerbose	- zobrazení detailů aktuálních volání
Find xxx	- nalezení registrovaného koncového bodu, xxx je alias nebo prefix
FindVerbose xxx	- výpis detailů o registrovaném koncovém bodu, xxx je alias nebo prefix
DisconnectCall xxx	- rozpojení volání s daným číslem xxx
DisconnectIP xxx	- rozpojení volání s danou IP adresou xxx
DisconnectAlias xxx	- rozpojení volání s daným aliasem xxx
RouteToAlias	- směrování volání na virtuální frontě na specifický alias
TransferCall	- přesunutí sestaveného spojení z alias A na alias B
exit	- opuštění telnet rozhraní

5.2 Konfigurace gatekeeper.ini

Dále bude uvedena ukázka použitého konfiguračního souboru gatekeeper.ini. Jsou uvedeny pouze námi použité konfigurační parametry a sekce, protože program GnuGk jich poskytuje značné množství (viz. http://www.gnugk.org/gnugk-manual.html).

```
[Gatekeeper::Main]
Fourtytwo=42                ; používáno programem ke zjištění přítomnosti
                            ; konfiguračního souboru
Name=OpenH323GK-VSB         ; identifikace gatekeepera při komunikaci
```

Signalizační zprávy mohou být přenášeny dvěmi způsoby. První metodou je Direct Endpoint Call Signalling. V tomto případě jsou zprávy přenášeny přímo mezi jednotlivými koncovými body. Druhou metodou je Gatekeeper Routed Call Signalling. Při použití této metody jsou jednotlivé zprávy směrovány mezi jednotlivými koncovými body přes daný gatekeeper.

```
[RoutedMode]
GKRouted=0                          ; mód GK Routek Call Signalling
H245Routed=0                        ; přenášení zpráv H.245 přes GK
CallSignalPort=1721                 ; port pro signalizaci (používaný GK)
CallSignalHandlerNumber=1           ; počet „vláken“ určených pro ovládání
                                    ; signalizačních / H.245 kanálů
RemoveH245AddressOnTunneling=0      ; tunelování H.245 zpráv přes Q.931
AcceptNeighborsCalls=1              ; akceptování volání, která nejsou uvedena
```

```
                                    ; v CallRec obsažených v CallTable
AcceptUnregisteredCalls=0           ; volba akceptování neregestrovaných volání
SupportNATedEndpoints=1             ; volba registrace koncových bodů za NAT
DropCallsByReleaseComplete=1
        ; spojení je ukončeno zprávou RAS DisengageRequest. Některé body ji nemusí
        ; akceptovat a proto je poslána zpráva Q.931 Release Complete
SendReleaseCompleteOnDRQ=1
        ; vyslání obou zpráv (Release Complete přes H.225/Q.931 a DRQ přes RAS)
        ; při položení
ForwardOnFacility=1
        ; při přijetí Q.931 Facility s parametrem callForwarded, může GK přímo směrovat
        ; zprávu SETUP k určenému koncovému bodu
ShowForwarderNumber=1
        ; přepsání Calling Party Number na číslo přesměrované
```

Definice H.323 proxy vlastností. Znamená to, že GnuGk bude provoz směrovat mezi volajícím a volaným koncovým bodem. Tedy nebude probíhat přímá komunikace mezi koncovými body. Využití, když máte některé koncové body s privátní IP adresou za NAT prvkem a některé koncové body využívající veřejnou IP adresou za prvkem.

GnuGk může fungovat jako proxy pro logické kanály RTP/RTCP a T.120. Jsou podporovány rovněž procedury fast-connect a H.245 tunelling. Pokud má však GnuGk pracovat jako proxy musí mít přímé propojení do obou sítí (síť volajícího a volaného).

```
[Proxy]
Enable=0
```

Sekce uvádějící skutečnost, která čísla typu E.164 jsou směrována na specifickou gateway. První řádek nám definuje IP adresu konkrétní používané SW ústředny Asterisk včetně příslušného portu. Druhý řádek definuje „proměnnou“ asterisk a příslušná čísla, která lze poté z H.323 terminálu vytáčet směrem na Asterisk. Proměnná asterisk má návaznost na dále uvedenou sekci [EP::asterisk].

2. - jakékoliv číslo začínající 2 + minimálně jedno číslo
!43 - čísla začínající 43 nebudou směrována na GW

```
[RasSrv::GWPrefixes]
158.196.251.129:1717
asterisk=2.,3.,!43,4.,5.,9.
```

Sekce definující parametry RRQ.

```
[RasSrv::RRQFeatures]
OverwriteEPOnSameAddress=1      ; definuje jak zpracovat požadavek RRQ z IP adresy,
                                ; která neodpovídá námi uloženému stavu
AcceptEndpointIdentifier=1      ; akceptování endpointidentifier v úplné RRQ zprávě
AcceptGatewayPrefixes=1
        ; GW může registrovat své prefixy s GnuGk použitím supportedPrefixes v poli
        ; terminalType zprávy RRQ
```

Sekce definující parametry ARQ.

```
[RasSrv::ARQFeatures]
ArjReasonRouteCallToSCN=0      ; zamítnutí volání z GW na sebe
ArjReasonRouteCallToGatekeeper=1
        ; jestliže 1, GK zamítne odpovídajících ARQ bez předem definovaného
        ; CallRecord v CallTable a to v Routed módu. Koncový bod může ihned rozpojit
        ; volání a přeposlat opět na GnuGk zprávu SETUP
CallUnregisteredEndpoints=1    ; GnuGk může akceptovat ARQ z registrovaného
                               ; koncového bodu
RemoveTrailingChar=#           ; specifikuje koncový znak který bude odstraněn
                               ; v parametru destinationInfo
RoundRobinGateways=1           ; volba cyklické obsluhy výběru GW
```

Následující sekce definuje jak pracují různé politiky směrování (routing policie) uvnitř GnuGk. Požadavek příchozího volání může směrován použitím osmi různých způsobů.

explicit - cíl je explicitně specifikován v požadavku směrování
internal - vyhledání cíle v Registration Table
parent - směrování volání použitím informací poslaných prostřednictvím GnuGk v odpovědi na ARQ
neighbor - směrování volání použitím sousedů prostřednictvím výměny LRQ zpráv
dns - cíl je určený z DNS
vqueue - použití virtuálního mechanismu front a generování RouteRequest pro ponechání provedení směrování externími aplikacemi
numberanalysis - poskytnutí podpory pro přesahující číslice poslaných pro zprávy ARQ
enum - podpora metody ENUM (RFC3761) pro použití DNS lookup ke konverzi reálného E.164 čísla na H.323 informaci.

```
[RoutingPolicy]
default=explicit,internal,parent,neighbor
```

Specifikace akce na přijetí RRQ (confirm nebo deny) pro modul AliasAuth. První alias (obvykle H323ID) koncového bodu k registraci je určen v této sekci. Jestliže je parametr přítomen bude aplikována příslušná hodnota jako pravidlo. Pravidlo sestává z podmínek oddělených znakem „&“. Registrace je akceptována, když jsou uplatněny všechny podmínky.

```
[RasSrv::RRQAuth]
default=confirm
```

Definuje počet pravidel, která jsou povolená pro připojení k status portu. Kdokoliv kdo má přístup k status portu má úplnou kontrolu nad GnuGk. Je nutné korektní nastavení. Lze zde například definovat zakázání připojení, ošetření přístupu na základě IP adresy nebo na základě hesla.

```
[GkStatus::Auth]
rule=allow                     ; povolení jakéhokoliv připojení
```

Definice některých vlastností LRQ a LCF. Námi použitý parametr určuje jestli bude přijatý LRQ přesměrován nebo ne. Parametr *always* směruje LRQ bezpodmínečně.

```
[RasSrv::LRQFeatures]
```

```
ForwardLRQ=always
```

Definování mechanismu autentifikace pro gatekeeper. Zde existuje možnost omezení registrace koncového bodu na daný GnuGk. Tato sekce je velmi důležitá pro budoucí použití (http://homel.vsb.cz/~voz29/files/voz54.pdf). Jsou umožněny například následující možnosti:

- Použití hesla (MD5)
- Autentifikace zpráv RegistrationRequest (RRQ)
- Zamezení autentifikace na základě IP volajícího
- Zamezení autentifikace na základě prefixu volajícího
- Autentifikace pomocí RADIUS serveru
- Autentifikace pomocí SQL databáze

```
[Gatekeeper::Auth]
default=allow
```

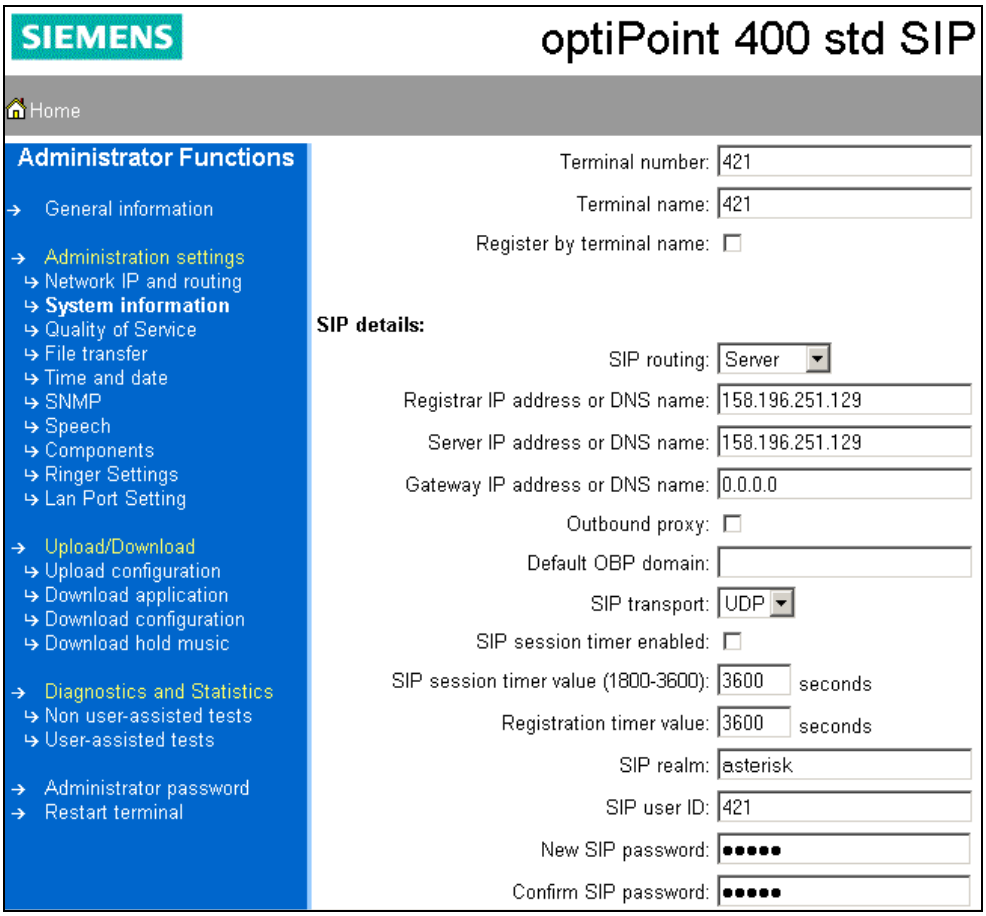
Sekce konfigurace parametrů koncového bodu. V našem případě definujeme pouze jeden koncový bod a to „asterisk“. Alias Asterisk má návaznost na sekci [RasSrv::GWPrefixes] kde je rovněž použit.

```
[EP::asterisk]
Capacity=8              ; počet současných volání, které je možno vyslat na daný
                        ; koncový bod
GatewayPriority=1       ; Povolení na prioritně založeném směrování v případě, že více
                        ; než jedna GW odpovídá volenému číslu
```

6. Konfigurace SIP klientů

6.1 optiPoint 400 Standard - HW

Použili jsme telefon optiPoint 400 standard se SIP image 2.3.14. Základní konfiguraci IP adresy, síťové masky a gateway jsme provedli prostřednictvím klávesnice telefonu. Ostatní konfiguraci lze následně provést pomocí webového rozhraní. Při konfiguraci účtu prostřednictvím klávesnice jsme narazili na problém při zadávání hesla, kdy telefon po zadání a potvrzení na chvíli zatuhnul.



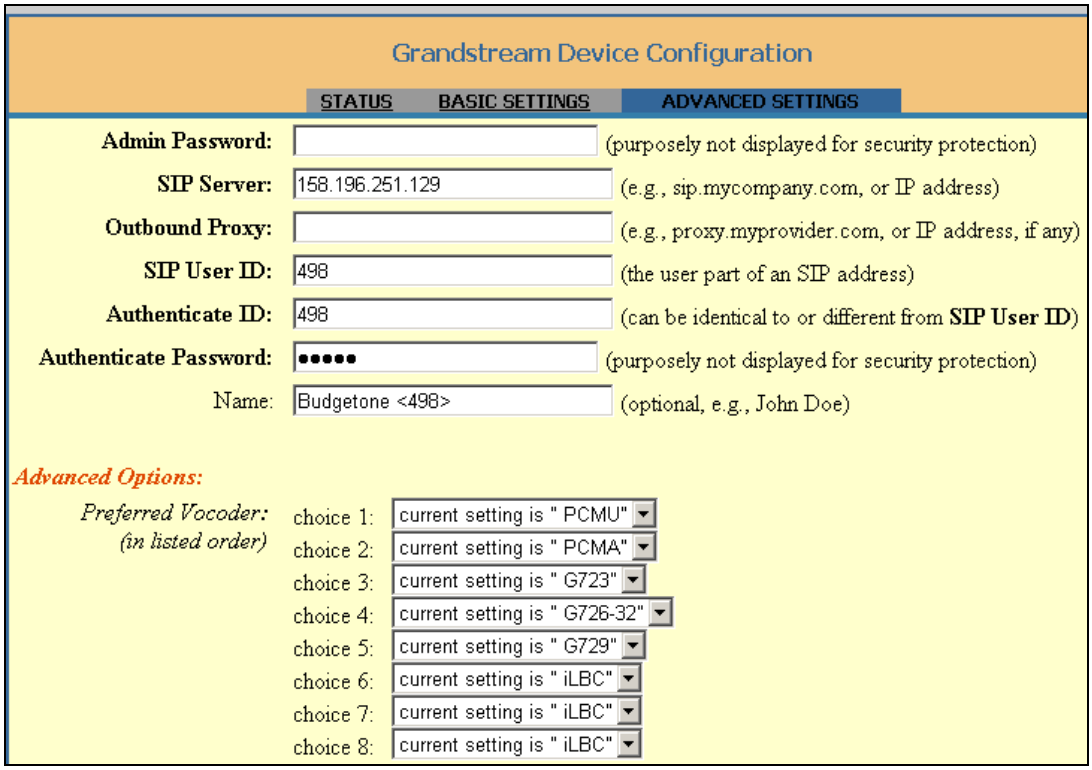
Obr. 6-1: Základní konfigurace telefonu optiPoint 400 Standard

Připojení přes webové rozhraní provedeme následovně – http://ip.adresa:8085 – v okně webového prohlížeče (admin heslo 123456). Na obrázku (**Obr. 6-1**) je zobrazeno konfigurační okno „System information“, což je část konfigurace připojení telefonu k SIP serveru. Nejprve nastavíme položky „Terminal number/name“ které jsou v první fázi shodné a představují telefonní číslo uživatele. Telefon umožňuje také registraci na základě jména terminálu.

„SIP routing“ nastavíme do módu „Server“. IP adresa Asterisku je zadána v položkách „Registar/Server IP address or DNS name“. Uživatelský účet je zadán v položce „SIP user ID“, heslo pak v dalších dvou položkách. Jedná se o jméno a heslo zadané v konfiguračním souboru Asterisku (sip.conf).

6.2 Budge Tone – 100 - HW

Použili jsme SIP telefon Grandstream Budge Tone – 100 s verzi softwaru 1.0.5.23. Základní konfiguraci IP adresy, síťové masky a gateway jsme provedli prostřednictvím klávesnice telefonu. Ostatní konfiguraci lze následně provést pomocí webového rozhraní, což velice doporučujeme. Při konfiguraci účtu prostřednictvím klávesnice se stává konfigurace vzhledem k použitému displeji nepřehlednou.



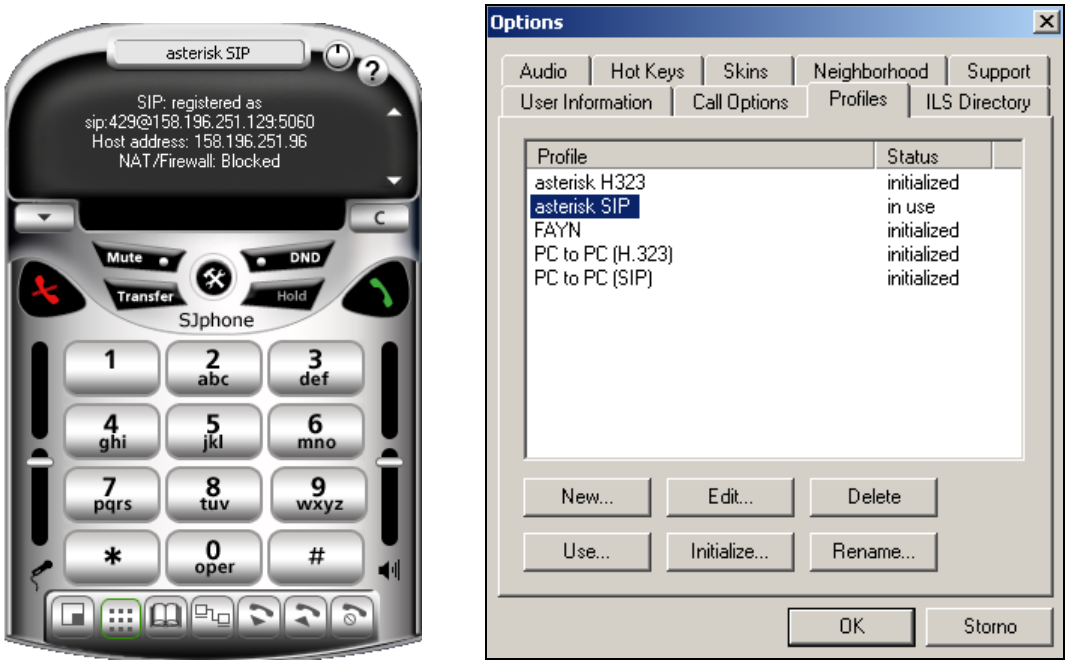
Obr. 6-2: Základní konfigurace telefonu Budge Tone - 100

Na obrázku (**Obr. 6-2**) je zobrazeno konfigurační okno „Advanced Settings“, což je část konfigurace telefonu pro připojení k SIP serveru včetně všech nastavitelných parametrů. Nastavíme adresu Asterisku „SIP Server“. Nastavení uživatelského účtu provedeme v položkách „SIP User ID / Authenticate ID / Password“. Jméno a heslo je opět dané konfigurací Asterisku (sip.conf).

Dále je nutné nastavit v položce „Advanced option“ používané kodeky. Doporučujeme nastavit kodeky na základě obrázku (**Obr. 6-2**). Důvodem je skutečnost, že pro použití v síti požadujeme hlavně podporu kodeku G.711 (položky PCMU a PCMA), dále kodek GSM, který však tento telefon nepodporuje. V konfiguračním souboru (sip.conf) jsou v základní části definované podporované kodeky (v této chvíli jsou to oba kodeky G.711 a kodek GSM), ostatní kodeky jsou zakázané z důvodu jejich omezené podpory v používaným SW a HW telefonech.

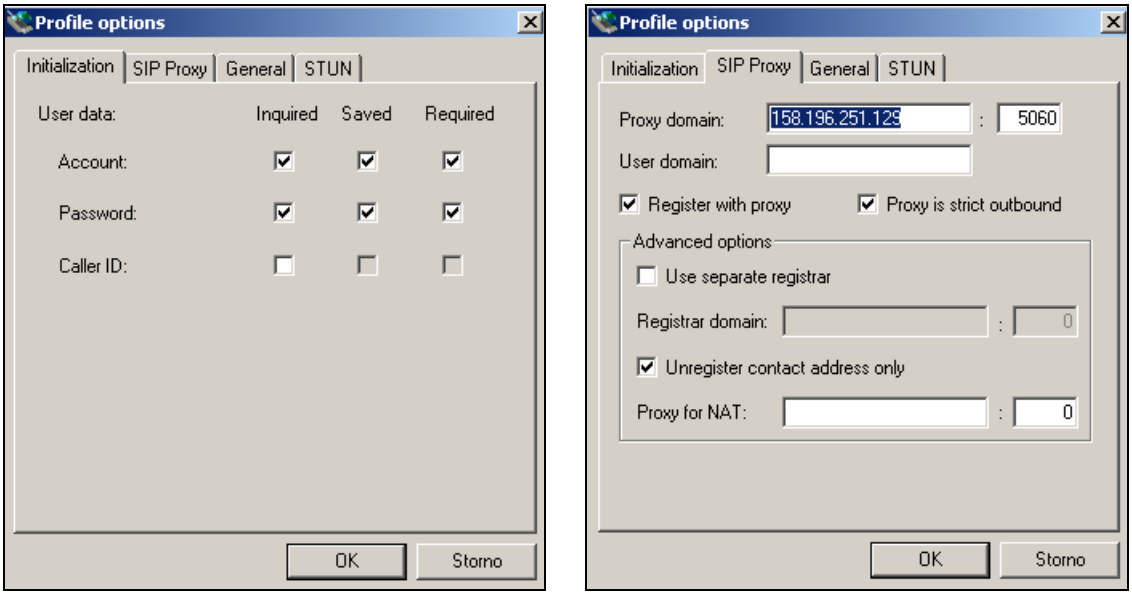
6.3 SJ Phone - SW

Použili jsme program SJ Phone Build 1.40.258 instalovaný pod OS Windows XP. Pro zaregistrování k Asterisku jsme tento program nakonfigurovali do SIP módu. Na obrázku (**Obr. 6-3**) je zobrazeno základní uživatelské prostředí telefonu. Nejprve je nutné vytvořit si nový profil pomocí volby „New“, dále zadáme jméno profilu, případně souboru pod kterým bude uložen. Položkou „Profil“ zvolíme mód „Call through SIP Proxy“.



Obr. 6-3: Prostředí SJ Phone , konfigurační prostředí

Následně konfigurujeme program způsobem, který je ukázán na obrázku (**Obr. 6-4**). Pokud nezaškrtneme políčko „Password – Saved“ pak bude uživatel při každém přístupu dotazován na přístupové heslo k danému účtu. Konkrétně se jedná o heslo nastavené v konfiguračním souboru Asterisk (sip.conf). Položka „Proxy domain“ je IP adresa daného Asterisku.



Obr. 6-4: Konfigurace připojení k Asterisku – mód SIP

Potvrdíme provedené změny, uživatel je následně dotázán na uživatelský účet a heslo příslušející k tomuto účtu (nastaveno v souboru sip.conf). Po zadání se vrátíme do volby „Profile“. Vybereme nakonfigurovaný profil a aktivujeme jej použitím „Initialize…“. Nakonec ještě provedeme restart aplikace a softwarový telefon je zaregistrován a připraven k použití.

Použití kodeků: K výběru kodeků se uživatel dostane prostřednictvím volby „Options – Audio – Compression Settings“. Jsou dostupné kodeky GSM, G.711 a-law, G.711 μ-law a kodeky iLBC (20ms a 30ms). Všechny tyto Asterisk podporuje. Doporučujeme použití kodeku G.711 μ-law.

6.4 Windows Messenger - SW

Nelze nastavit používané kodeky. Komunikoval bez problému s telefonem Grandstream, ale s telefonem Siemens nebylo slyšel příchozí kanál SW telefonu. Možnost registrování k Asterisku i na základě autorizace. Nutno použít službu „Komunikační služba SIP“.

Konfigurace připojení:

- Možnosti – Účty – Přihlašovací jméno (ve tvaru „jméno@Asterisk_IP_addr“)
- Možnosti – Účty – Přihlašovací jméno – Upřesnit – Název serveru nebo adresa IP (ve tvaru „Asterisk_IP_addr:5060“)
- Při přihlašování vyžaduje přihlašovací jméno ve tvaru „jméno@Asterisk_IP_addr“, uživatelské jméno a heslo.

Nutno následně vytvořit seznam kontaktů nebo volat prostřednictvím volby Zahájit hlasovou konverzaci – Jiné – zadat volbu ve tvaru „číslo@Asterisk_IP_addr”.

Použití tohoto SW telefonu nedoporučujeme !!!

6.5 X-lite - SW

Použili jsme program X-Lite v.2.0 Build 1103m pod OS Windows XP (SIP softphone). Pro zaregistrování k Asterisku jsme tento program nakonfigurovali do SIP módu. Na obrázku (**Obr. 6-5**) je zobrazeno základní uživatelské prostředí telefonu. Všechna důležitá nastavení se prováděla v položce „System Settings“.

Obr. 6-5: Základní prostředí X-lite

V nabídce „System Settings“ jsou důležité položky „Network“ a „SIP Proxy“. V nabídce „Network“ (**Obr. 6-6**) se nastavují informace o DNS serveru. V nabídce „SIP Proxy“ (**Obr. 6-7**) nastavíme uživatelské jméno a heslo (definované v Asterisku), doménu a SIP Proxy (IP adresy Asterisku).

Obr. 6-6: Síťové nastavení

Obr. 6-7: SIP Proxy nastavení

Nakonec provedeme restart aplikace a softwarový telefon je zaregistrován a připraven k použití.

Použití kodeků: K výběru kodeků se uživatel dostane prostřednictvím volby „Advance System Settings – Codec Settings“. Dostupné jsou kodeky GSM, G.711 a-law, G.711 μ-law, iLBC a Speex. Všechny tyto Asterisk podporuje. Povolené kodeky jsou zobrazeny na displeji telefonu a při každém volání je zvýrazněn ten, který je právě používán.

7. Konfigurace IAX klientů

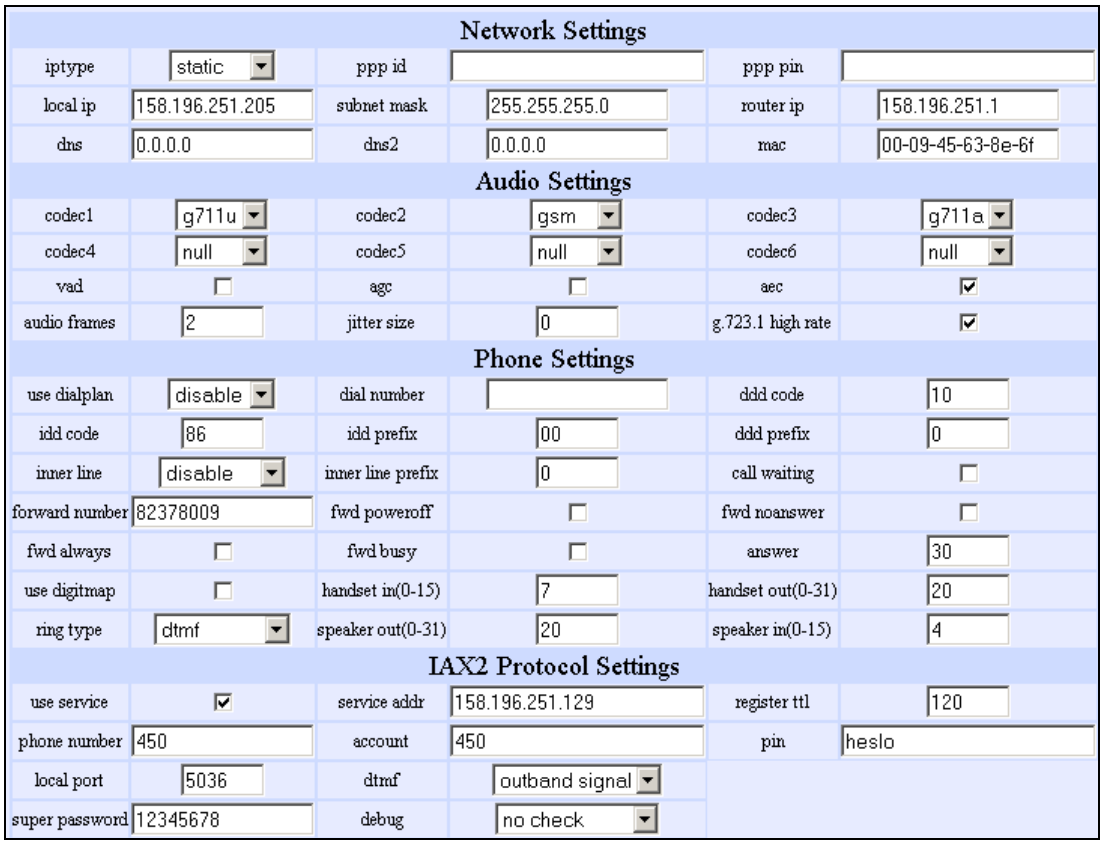
Pro správnou funkci SW klientů je potřeba mít korektně nakonfigurovaný soubor iax.conf (viz. Kapitola 6.3). Kromě klasických kodeků G.711 a-law, G.711 μ-law a GSM je vhodné také povolit kodeky iLBC a Speex, které softwaroví klienti podporují.

7.1 Ethernet Phone AT-320 - HW

IP telefon AT-320 je levným HW řešením pro VoIP. Jeho výhodou je, že podporuje protokoly H.323, SIP, MGCP a nově také protokol IAX2.

Nejprve je nutno prostřednictvím klávesnice telefonu konfigurovat IP adresu, síťovou masku a gateway. Poté je telefon restartován. Následnou konfiguraci doporučujeme provést prostřednictvím webového rozhraní.

K telefonu se přihlásíme pomocí IP adresy a default hesla pro správce „12345678“. Konfigurační menu je zobrazeno na obrázku (**Obr. 6-8**).



Network Settings					
iptype	static	ppp id		ppp pin	
local ip	158.196.251.205	subnet mask	255.255.255.0	router ip	158.196.251.1
dns	0.0.0.0	dns2	0.0.0.0	mac	00-09-45-63-8e-6f
Audio Settings					
codec1	g711u	codec2	gsm	codec3	g711a
codec4	null	codec5	null	codec6	null
vad	☐	agc	☐	aec	☑
audio frames	2	jitter size	0	g.723.1 high rate	☑
Phone Settings					
use dialplan	disable	dial number		ddd code	10
idd code	86	idd prefix	00	ddd prefix	0
inner line	disable	inner line prefix	0	call waiting	☐
forward number	82378009	fwd poweroff	☐	fwd noanswer	☐
fwd always	☐	fwd busy	☐	answer	30
use digitmap	☐	handset in(0-15)	7	handset out(0-31)	20
ring type	dtmf	speaker out(0-31)	20	speaker in(0-15)	4
IAX2 Protocol Settings					
use service	☑	service addr	158.196.251.129	register ttl	120
phone number	450	account	450	pin	heslo
local port	5036	dtmf	outband signal		
super password	12345678	debug	no check		

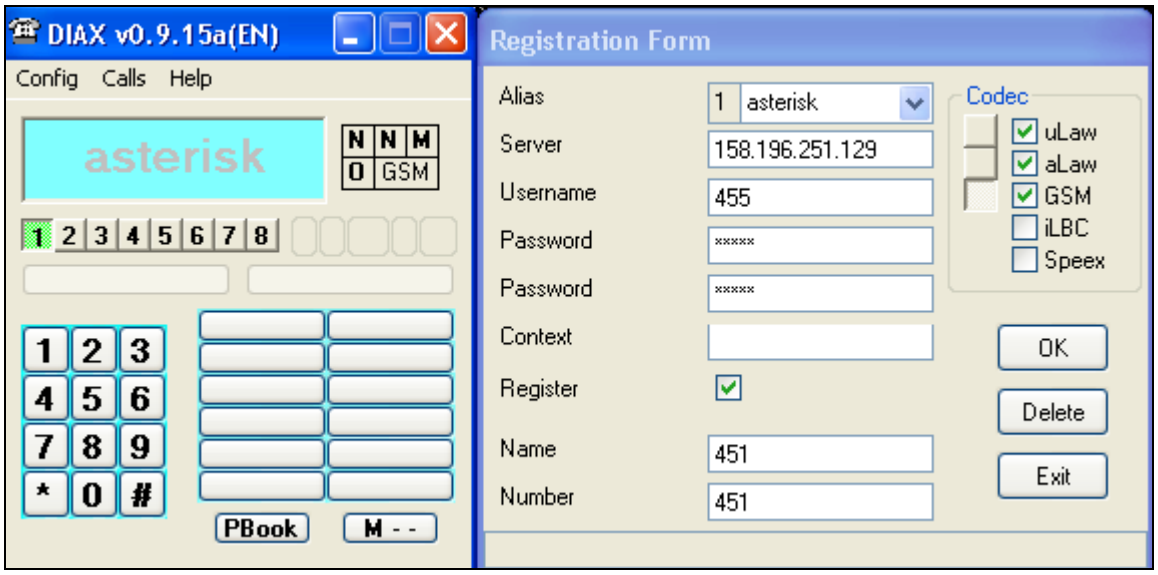
Obr. 6-8: Konfigurační menu telefonu AT-320

Pro zaregistrování se k Asterisku provedeme následující nastavení. Jedná se konkrétně o položky „IAX2 Protocol Settings“.

V telefonu je nahrán image IAX v.1.42.028. „Service addr“ je IP adresou Asterisku. „Phone number“ je telefonní číslo daného účtu. Uživatelský účet / heslo pod kterým se telefon registruje k Asterisku je zadán ve volbě „account / pin“. „Local port“ nastavíme na hodnotu 5036.

7.2 DIAX - SW

Použili jsme program DIAX v.0.9.15a pod OS Windows XP (IAX softphone). Klient podporuje registraci s ústřednou Asterisk, registraci s více servery, protokol IAX2, nezávislá konfigurace kodeků pro jednotlivé servery, umožňuje také připojení USB telefonu, atd. Nastaví registračních údajů pro připojení k ústředně Asterisk je zobrazeno na **Obr. 6-9** (menu „Config – Registration“).



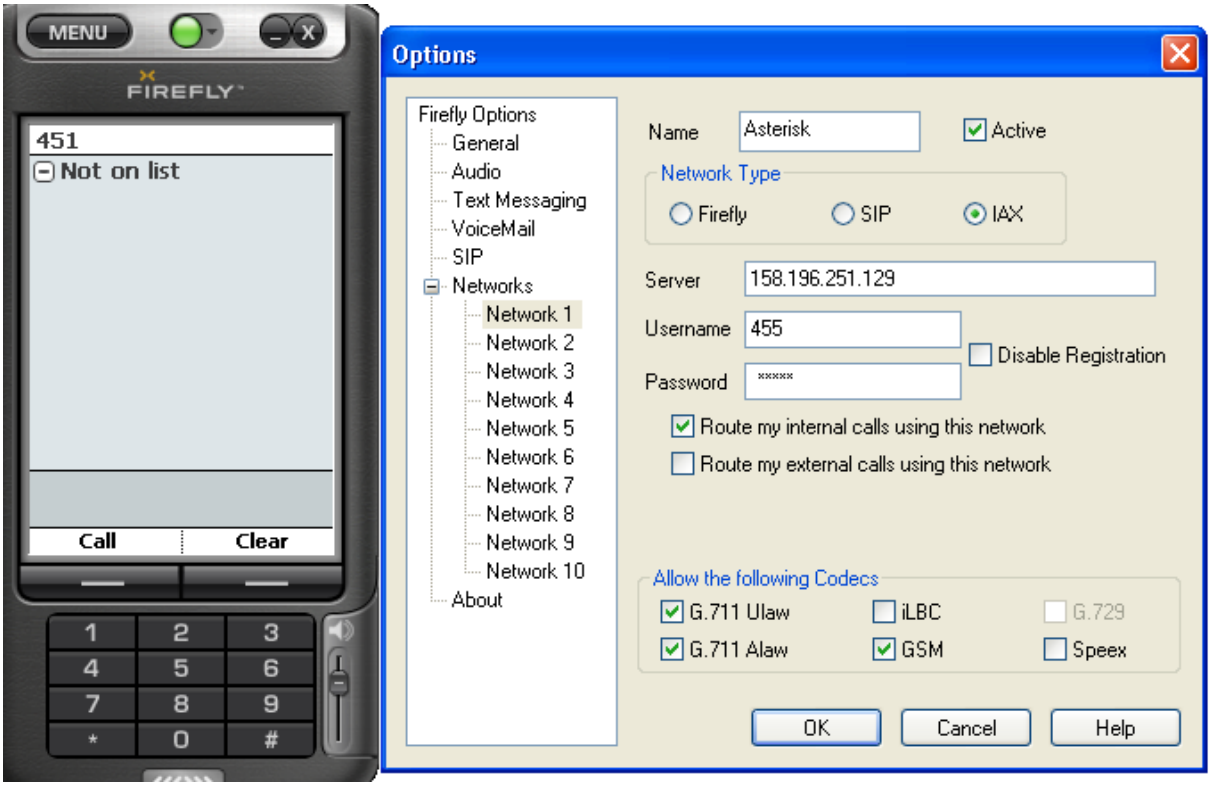
Obr. 6-9: Základní prostředí DIAX, Registrace k ústředně Asterisk

Položka „Alias“ představuje používaný profil připojení k serveru, je možno konfigurovat 8 různých profilů s různými parametry (včetně audio kodeků). Username a Password je shodné s nakonfigurovaným účtem v ústředně. Označením položky Register donutíme SW telefon aby se registroval na základě jména a hesla. Adresu serveru lze v případě potřeby zadat včetně příslušného portu.

Použití kodeků: K výběru kodeků se uživatel dostane prostřednictvím volby „Config - Registration“. Jsou dostupné kodeky GSM, G.711 a-law, G.711 μ-law, kodek iLBC a kodek Speex. Všechny tyto Asterisk podporuje. Doporučujeme použití kodeku G.711 a-law nebo GSM, při použití kodeku G.711 μ-law nebyl hovor několik sekund slyšet. Při výběru více kodeků lze jeden z nich vybrat jako preferovaný. Lze také použít kodeky iLBC a Speex, u těchto však není zaručena kompatibilita s klienty HW. Několikrát se nám také stalo že hovor byl spojen se zpožděním a na straně HW telefonu jej nebylo možné vyzvednout (je to otázkou jestli to byla chyba SW telefonu, HW telefonu nebo Asterisku).

7.3 FireFly - SW

Použili jsme program FireFly v.1.9.9 pod OS Windows XP (IAX, SIP softphone). Klient podporuje registraci s ústřednou Asterisk, protokoly IAX, SIP, nezávislá konfigurace až 10 různých serverů, možnost registrace do sítě FireFly, vytvoření účtu a volání do komerčních sítí po celém světě. Nastavení registračních údajů pro připojení k ústředně Asterisk je zobrazeno na **Obr. 6-10** (menu „Menu – Option – Network – Network 1“).



Obr. 6-10: Základní prostředí FireFly, Registrace k ústředně Asterisk

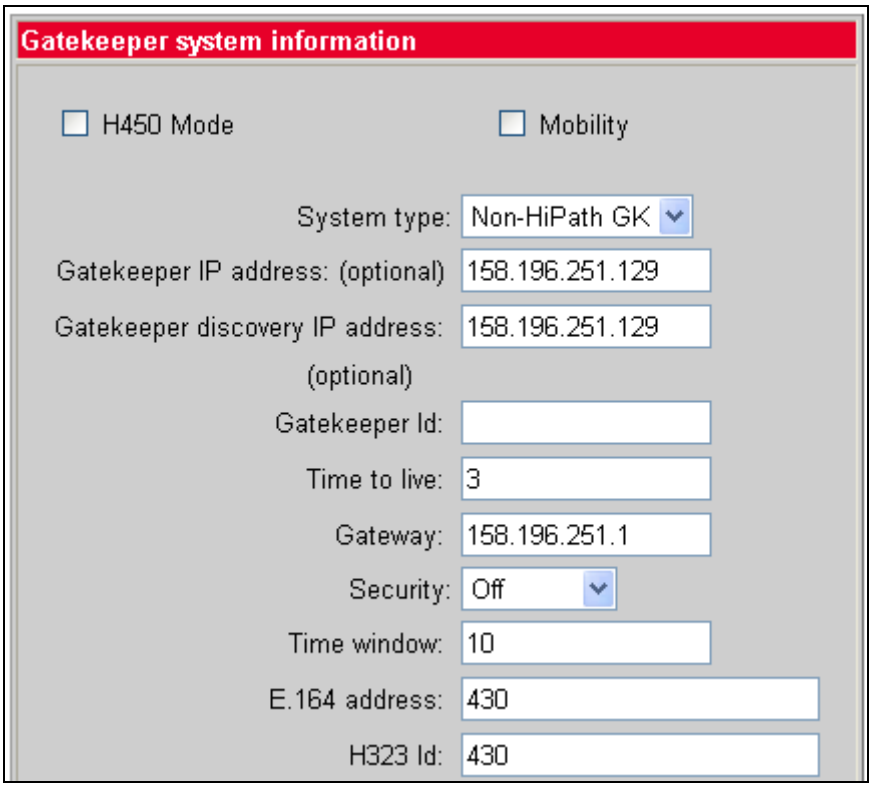
Nejprve zvolíme typ používané sítě, tedy komunikační protokol – IAX. Server, Username a password jsou opět konfigurované parametry v ústředně. Volbu „Disable Registration“ ponecháme neaktivní, telefon se bude tedy automaticky registrovat k ústředně. Volbou „Activate v pravém horním rohu určíme, že právě tento daný profil bude používán.

Použití kodeků: K výběru kodeků se uživatel dostane prostřednictvím shodného postupu jako ke konfiguraci připojení. Jsou dostupné opět kodeky GSM, G.711 a-law, G.711 μ-law, kodek iLBC a kodek Speex. Všechny tyto Asterisk podporuje. Doporučujeme použití některého z kodeků G.711 a-law, GSM nebo G.711 μ-law. Lze také použít kodeky iLBC a Speex, u těchto však není zaručena kompatibilita s klienty HW.

8. Konfigurace H.323 klientů

8.1 optiPoint 300 advance – HW

Použili jsme telefon optiPoint 300 advance s H.323 image 2.5.32. Základní konfiguraci IP adresy, síťové masky a gateway jsme provedli prostřednictvím klávesnice telefonu. Ostatní konfiguraci lze následně provést pomocí webového rozhraní.



Obr. 6-11: Základní konfigurace telefonu optiPoint 300 advance

Připojení přes webové rozhraní provedeme následovně – http://ip.adresa:8085 – v okně webového prohlížeče (admin heslo 123456). Na obrázku (**Obr. 6-11**) je zobrazeno konfigurační okno „Gatekeeper system information“, což je část konfigurace připojení telefonu k H.323 gatekeeperu (Administrator – Administrator Settings – System information). Nejprve nastavíme položky „E.164 address/H.323 ID“, které jsou v první fázi shodné a představují telefonní číslo uživatele. Pro registraci je podstatné E.164 address, pomocí kterého se telefon registruje k danému GnuGk. Podmínkou funkčnosti s Asteriskem je definování příslušného čísla v konfiguračním souboru (extensions.conf).

„Systém Type“ nastavíme do módu „Non-hiPath GK“. IP adresa Asterisku je zadána v položkách „Gatekeeper IP address/Gatekeeper discovery IP address“..

Možnosti používaných kodeků lze nastavit v položce „Speech parameters“. Lze definovat použití kodeků G.711 a G.723. Toto nastavení lze použít ve dvou módech – Always / Preferred. Nedoporučujeme používat řízení kvality služby na vrstvě 3. Tedy nezaškrtávat možnost QoS Layer 3 (Administrator – Administrator Settings – Quality of service).

V souvislosti s tímto telefonem se vyskytl problém korektního zobrazení příchozího telefonního čísla (viz. macro-basic-oh323). Telefon v současnosti toto telefonní číslo nezobrazuje korektně.

8.2 optiPoint 400 standard - HW

Použili jsme telefon optiPoint 400 standard s H.323 image 3.1.24. Základní konfiguraci IP adresy, síťové masky a gateway jsme provedli prostřednictvím klávesnice telefonu. Ostatní konfiguraci lze následně provést pomocí webového rozhraní.

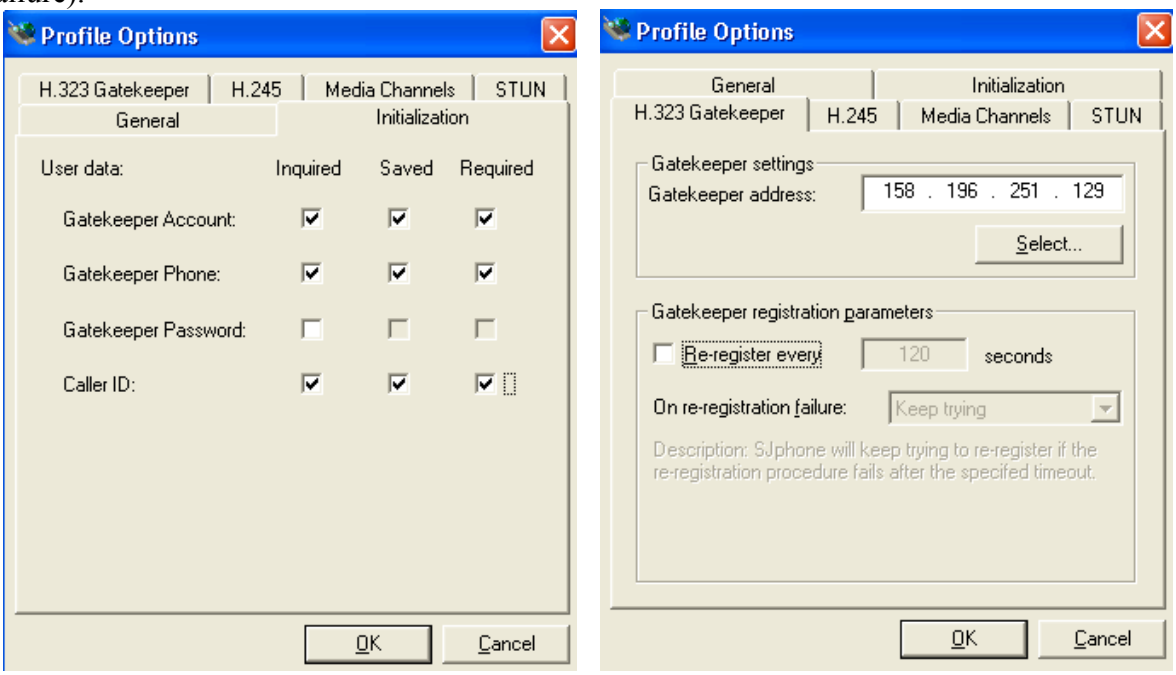
Připojení přes webové rozhraní provedeme následovně – http://ip.adresa:8085 – v okně webového prohlížeče (admin heslo 123456). Grafické rozhraní i jednotlivé parametry jsou totožné s nastavením telefonu optiPoint 300 advance (kapitola. 8.1).

V souvislosti s tímto telefonem se vyskytl problém problém korektního zobrazení příchozího telefonního čísla (viz. macro-basic-oh323). Telefon v současnosti toto telefonní číslo nezobrazuje korektně.

8.3 SJ Phone - SW

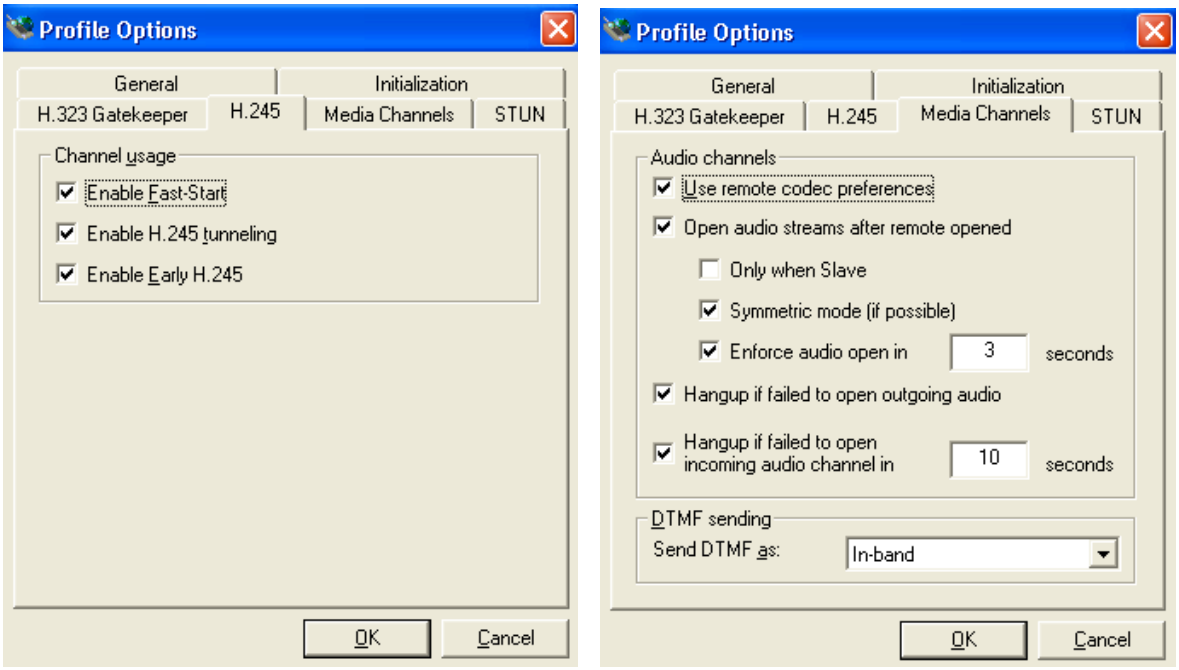
Použili jsme program SJ Phone Build 1.60.289a instalovaný pod OS Windows XP. Pro zaregistrování k Asterisku jsme tento program nakonfigurovali do H.323 módu. Základní uživatelské prostředí je shodné s prostředím uvedeným na obrázku (**Obr. 6-5**). Nejprve je nutné vytvořit si nový profil pomocí volby „New“, dále zadáme jméno profilu, případně souboru pod kterým bude uložen. Položkou „Profil type“ zvolíme mód „Calls through H.323 Gatekeeper“.

Následně konfigurujeme program způsobem, který je ukázán na obrázku (**Obr. 6-12**). Možnost použít heslo pro autentifikaci ke gatekeeperu není v tomto okamžiku podstatná, zatím není pro registraci požadováno heslo. Omezení je pouze požadavkem mít dané telefonní číslo konfigurované v číslovacím plánu (extensions.conf), jinak nebude dané telefonní číslo přístupné z registrovaných ne-H.323 terminálů. Položka „Gatekeeper address“ je IP adresa daného Asterisku. Ve stejném okně může uživatel také nastavit parametry týkající se registrace k danému gatekeeperu. Například časový interval opětovné registrace (Re-register every) nebo určení akce, která se má provádět při selhání registrace (On re-registration failure).



Obr. 6-12: Konfigurace připojení k Asterisku – mód H.323

Mezi další možnosti uživatelské konfigurace patří například uživatelské definování akcí příslušných H.245 (Fast-start, H.245 tunneling, Early H.245), což je samozřejmě pro správné používání nastavit rovněž v konfiguračních souborech Asterisku. Lze také nastavit parametry jakým způsobem je přenášena DTMF volba a několik dalších. Uvedené parametry jsou zobrazeny na obrázku (**Obr. 6-13**).



Obr. 6-13: Rozšířené možnosti konfigurace SJ Phone

Po konfiguraci potvrdíme provedené změny. Poté je uživatel dotázán na parametry uživatelského účtu. Po zadání se vrátíme do volby „Profile“. Vybereme nakonfigurovaný profil a aktivujeme jej použitím „Initialize…“. Nakonec ještě provedeme restart aplikace a softwarový telefon je zaregistrován a připraven k použití.

Použití kodeků: K výběru kodeků se uživatel dostane prostřednictvím volby „Options – Audio – Compression Settings“. Jsou dostupné kodeky GSM, G.711 a-law, G.711 μ-law a kodeky iLBC (20ms a 30ms). Všechny tyto Asterisk podporuje. Doporučujeme použití kodeku G.711 μ-law.

9. Odkazy na Internet

9.1 Zdroje informací

Firma Digium:
http://www.digium.com

Stránky týkající se projektu GnuGk (GNU Gatekeeper):
http://www.gnugk.org
http://www.gnugk.org/gnugk-manual.html - manuál GnuGk
http://homel.vsb.cz/~voz29/files/voz54.pdf - využití GnuGk v síti CESNET

Stránky týkající se projektu Asterisk:
http://www.voip-info.org/ - problematika SW ústředny Asterisk
http://www.asterisk.org/ - problematika SW ústředny Asterisk

Stránky týkající se IP telefonie:
http://www.freevoice.cz/ - zaměřeno na IP telefonie a Asterisk
http://www.terena.nl/library/IPTELEPHONYCOOKBOOK/contents1.html - projekt IP telephony cookbook

Užitečné stránky oebcného Linuxu – Debian:
http://packages.debian.org/stable/ - úplný výčet balíčků zařayených do distribuce „stable“

Zajímavé stránky:
http://www.javvin.com/ - přehled používáných protokolů a jejich základních vlastností
http://community.roxen.com/developers/idocs/rfc/ - možnost stáhnutí všech RFC

9.2 SW klienti

[1]	SJ-Phone	SW telefon	SIP, H.323 http://www.sjlabs.com
[2]	Windows Messenger	SW telefon	SIP http://www.microsoft.com
[3]	X-lite	SW telefon	SIP http://www.xten.com/
[4]	DIAX	SW telefon	IAX http://www.laser.com/dante/diax/diax.html
[5]	FireFly	SW telefon	SIP, IAX http://www.freshtel.net/firefly/