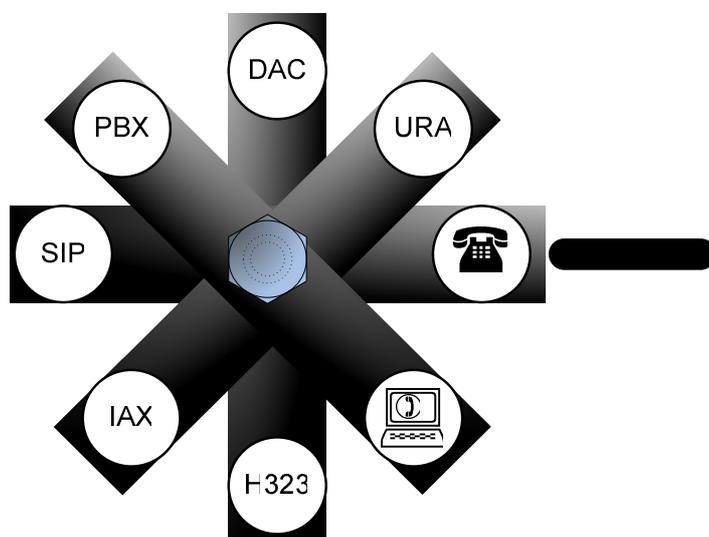


Asterisk PBX

Guia de Configuração



Como construir e configurar um PABX com Software Livre

Por: Flávio Eduardo de Andrade Gonçalves
flavio.goncalves@voffice.com.br

Todos os direitos reservados. É proibida a reprodução total ou parcial deste livro.

Prefácio

O Asterisk PBX é, em minha opinião, uma revolução nas áreas de telefonia IP e PABX baseado em software. Durante muitos anos o mercado de telefonia foi ligado a equipamentos proprietários fabricados por grandes companhias multinacionais. Apesar de termos equipamentos de baixo custo nestas arquiteturas eles também apresentam baixa funcionalidade. Com a entrada do Asterisk, mais e mais empresas vão poder experimentar recursos como URA - unidade de resposta audível, DAC – distribuição automática de chamadas, mobilidade, correio de voz, e conferência, antes restritos à grandes companhias devido ao alto custo.

A telefonia IP quando atingir massa crítica fará com que o PABX de qualquer empresa possa falar com o PABX de qualquer outra através da Internet. O protocolo DUNDI é um primeiro ensaio nesta área. Na hora de avaliar os benefícios do Asterisk é preciso enxergar este horizonte futuro que são operadoras IP como a VONAGE, GVT, FreeWorldDialup e interligação automática com outros PABX. A economia em DDD e DDI é só a ponta do iceberg.

Este livro foi criado com o objetivo de facilitar a adoção do Asterisk PBX em países de língua portuguesa. Um dos primeiros problemas que encontrei tentando aprender e implementar o Asterisk foi a falta de documentação. Apesar do Asterisk handbook, o asteriskdocs.org e do Wiki (www.voip-info.org) que foram as principais fontes de referência para este material, as informações estão espalhadas aqui e ali o que torna difícil o aprendizado.

Apesar de usar alguns exemplos com equipamentos de mercado, este material não recomenda especificamente nenhum equipamento ou provedor de serviços. Use-os por sua conta e risco.

Não tivemos a pretensão de ensinar tudo que existe sobre o Asterisk PBX neste livro, pois isto seria uma missão quase impossível, novos recursos estão sendo adicionados todos os dias e o Asterisk têm muitos. Nossa principal pretensão neste material é de que o leitor possa ter acesso aos principais recursos e a partir deles possa descobrir e implementar recursos mais avançados.

Eu espero que vocês se divirtam tanto aprendendo o Asterisk quanto eu me diverti escrevendo sobre ele, tempo e paciência são requisitos indispensáveis para testar todos os recursos deste material.

Flávio Eduardo de Andrade Gonçalves
Diretor Geral
V.Office Networks
flavio.goncalves@voffice.com.br

Sobre o Autor

Flávio Eduardo de Andrade Gonçalves é engenheiro de redes sênior da V.Office Networks. Com certificações da Cisco Systems (CCNP/CCDP/CCSP), Microsoft (MCSE) e Novell (MCNE) dirige uma empresa especializada em redes de computadores em Florianópolis desde 1996. Desde 1992 ministra treinamento, cria projetos e auxilia na resolução de problemas com redes Novell, Microsoft, Linux e Cisco. Nos últimos cinco anos tem se dedicado integralmente à implantação de redes com VPN e redes com Voz sobre IP.

Agradecimentos

Tenho aqui de agradecer a minha família pela paciência de me ver trabalhando as madrugadas e fins de semana para que este material pudesse ser escrito. Agradeço à Clarice minha esposa e companheira pelo incentivo e apoio e a Ana Cristina Gonçalves por resolver todos os entraves como publicação, distribuição, capa, marketing que possibilitaram que este material chegasse aos usuários.

Sumário

INTRODUÇÃO AO ASTERISK	1
1.1 OBJETIVOS DO CAPÍTULO	1
1.2 O QUE É O ASTERISK	1
1.2.1 QUAL O PAPEL DA DIGIUM?	2
1.2.3 O PROJETO ZAPATA	3
1.4 PORQUE O ASTERISK?	5
1.4.1 REDUÇÃO DE CUSTOS EXTREMA	5
1.4.2 TER CONTROLE DO SEU SISTEMA DE TELEFONIA	5
1.4.3 AMBIENTE DE DESENVOLVIMENTO FÁCIL E RÁPIDO	6
1.4.4 RICO E ABRANGENTE EM RECURSOS	6
1.4.5 É POSSÍVEL PROVER CONTEÚDO DINÂMICO POR TELEFONE.	6
1.4.6 PLANO DE DISCAGEM FLEXÍVEL E PODEROSO	6
1.4.7 RODA NO LINUX E É CÓDIGO ABERTO	6
1.4.8 LIMITAÇÕES DE ACESSO À REDE PÚBLICA NO BRASIL	6
1.4.9 LIMITAÇÕES DA ARQUITETURA DO ASTERISK	7
1.5 ARQUITETURA DO ASTERISK	7
1.5.1 CANAIS	8
1.5.2 CODECS AND CONVERSÕES DE CODEC	10
1.5.3 PROTOCOLOS	10
1.5.4 APLICAÇÕES	11
1.6 CENÁRIOS DE USO DO ASTERISK	12
1.6.1 VISÃO GERAL	12
1.6.2 TELEFONIA DO JEITO ASTERISK	15
1.6.3 O CLÁSSICO PABX 1x1	16
1.6.4 CRESCENDO O SEU PABX USANDO UM BANCO DE CANAIS	17
1.6.5 INTERLIGAÇÃO DE FILIAIS À MATRIZ	18
1.6.6 UNIDADE DE RESPOSTA AUTOMÁTICA	20
1.7 INTERFACE DE GERENCIAMENTO DO ASTERISK.	21
1.7.1 COMPORTAMENTO DO PROTOCOLO	21
1.7.2 TIPOS DE PACOTE	21
1.7.3 AUTENTICAÇÃO	22
1.8 ASTERISK GATEWAY INTERFACE – AGI.	22
1.8.1 USANDO O AGI	23
1.9 SUMÁRIO	23
1.10 QUESTIONÁRIO	25
BAIXANDO E INSTALANDO O ASTERISK	29
2.1 OBJETIVOS DO CAPÍTULO	29
2.2 INTRODUÇÃO	29
2.3 HARDWARE MÍNIMO	29
2.3.1 MONTANDO O SEU SISTEMA	30

2.3.2 QUESTÕES DE COMPARTILHAMENTO DE IRQ	31
2.4 ESCOLHENDO UMA DISTRIBUIÇÃO DO LINUX.	32
2.4.1 REQUISITOS DO LINUX	32
2.4.2 PACOTES NECESSÁRIOS.	32
2.5 INSTALANDO O LINUX PARA ATENDER AO ASTERISK.	33
2.6 OBTENDO E COMPILANDO O ASTERISK	40
2.6.1 O QUE É CVS?	40
2.6.2 DRIVERS PARA AS PLACAS DE TELEFONIA	40
2.6.3 COMPILANDO O ZTDUMMY	42
2.7 INSTALANDO E CONFIGURANDO O HARDWARE	43
2.7.1 PASSOS NECESSÁRIOS PARA INSTALAÇÃO DO HARDWARE.	43
2.7.2 INSTALAR O HARDWARE NO PC	43
2.7.3 CARREGAR OS DRIVERS DE KERNEL	44
2.7.4 CONFIGURANDO O ARQUIVO ZAPTEL.CONF	45
2.8 OBTENDO E COMPILANDO O ASTERISK	46
2.7 INICIANDO E PARANDO O ASTERISK	47
2.7.1 PARÂMETROS DE LINHA DE COMANDO DO ASTERISK.	47
2.7.2 ABAIXO OS PARÂMETROS DISPONÍVEIS	48
2.8 INICIANDO O ASTERISK EM TEMPO DE INICIALIZAÇÃO.	48
2.9 CONSIDERAÇÕES SOBRE A INSTALAÇÃO DO ASTERISK	50
2.9.1 SISTEMAS EM PRODUÇÃO	50
2.9.2 CONSIDERAÇÕES SOBRE A REDE	50
2.10 SUMÁRIO	51
2.11 QUESTIONÁRIO	52
CONFIGURAÇÃO DO ASTERISK	55
<hr/>	
3.1 OBJETIVOS DO CAPÍTULO	55
3.2 INTRODUÇÃO	55
3.3 ARQUIVOS DE CONFIGURAÇÃO DO ASTERISK	55
3.3.1 GRUPO SIMPLES	56
3.3.2 FORMATO DE OBJETO COM HERANÇA DE OPÇÕES	57
3.3.3 OBJETO ENTIDADE COMPLEXA	57
3.4 DRIVERS DE CANAL ASTERISK	58
3.4.1 ZAPATA.CONF	58
ANDAMENTO DA CHAMADA	61
OUTRAS OPÇÕES	65
3.4.2 NOMENCLATURA DOS CANAIS ZAP	65
3.4.3 EXEMPLO DE ARQUIVO COMPLETO	65
3.4.4 CONFIGURAÇÃO DOS TELEFONES IP SIP	66
3.4.5 ARQUIVO EXEMPLO DO SIP.CONF SEÇÃO GERAL [GENERAL]	67
3.4.6 OPÇÕES PARA CADA TELEFONE	67
3.4.7 EXEMPLO COMPLETO DO SIP	68
3.5 INTRODUÇÃO AO PLANO DE DISCAGEM	68
3.5.1 CONTEXTOS	69
3.5.2 EXTENSÕES	70
3.5.3 PRIORIDADES	71

3.5.4 APLICAÇÕES	71
3.5.5 CRIANDO UM AMBIENTE DE TESTES	71
3.5.6 CRIANDO UM PLANO DE DISCAGEM SIMPLES	74
MEU PRIMEIRO PLANO DE DISCAGEM	75
UM EXEMPLO MAIS ÚTIL	76
INTERLIGANDO CANAIS COM A APLICAÇÃO DIAL()	77
3.6 LAB. IMPLANTANDO UMA APLICAÇÃO SIMPLES	78
3.7 SOFISTICANDO UM POUCO MAIS.	78
3.8 EXEMPLO DE UMA URA SIMPLES	79
3.9 SUMÁRIO	79
3.10 QUESTIONÁRIO	81

CAPÍTULO 4 – VOZ SOBRE IP COM O ASTERISK.

4.1 OBJETIVOS	85
4.2 INTRODUÇÃO	85
4.3 BENEFÍCIOS DA VOZ SOBRE IP	85
4.3.1 PACKET TELEPHONY CALL CENTER	86
4.3.2 UNIFIED MESSAGING	86
4.3.3 CHAMADA BASEADA EM CARTÃO	86
4.4 ARQUITETURA DO ASTERISK E VOZ SOBRE IP	87
4.5 COMO ESCOLHER UM PROTOCOLO	89
4.5.1 SIP	89
4.5.2 IAX	89
4.5.3 MGCP	89
4.5.4 H323	89
4.6 CONCEITO DE PEERS, USERS E FRIENDS	90
4.7 CODECS E CONVERSÃO DE CODECS	90
4.8 SUMÁRIO	92
4.9 QUESTIONÁRIO	93

O PROTOCOLO IAX E O ASTERISK

5.1 OBJETIVOS DO CAPÍTULO	97
5.2 INTRODUÇÃO	97
5.2 TEORIA DE OPERAÇÃO	98
5.3 FORMATO DOS FRAMES	99
FRAME COMPLETO	99
MINI FRAME	100
5.4 USO DE BANDA PASSANTE	101
5.4.1 USO DE BANDA DO IAX	102
5.5 NOMENCLATURA DOS CANAIS	103
5.5.1 FORMATO DE UMA CONEXÃO DE SAÍDA.	103
5.5.2 EXEMPLOS DE CANAIS DE SAÍDA:	103
5.5.3 FORMATO DE UMA CONEXÃO DE ENTRADA	104

5.5.4 EXEMPLO DE CANAIS DE ENTRADA	104
5.6 CENÁRIOS DE USO	104
5.6.1 SERVIDOR IAX:	104
5.6.2 CLIENTE IAX	105
5.6.3 COMO FAZER PARA DISCAR PARA UM PROVEDOR	105
5.6.4 ABREVIANDO OS COMANDOS	105
5.6.5 COMO FAZER PARA RECEBER UMA LIGAÇÃO	106
5.6.6 TRUNK IAX	107
5.6.7 COMO CONFIGURAR UM TRUNK IAX	107
5.7 AUTENTICAÇÃO NO IAX	110
5.7.1 CONEXÕES DE ENTRADA	110
5.7.2 CONEXÕES DE SAÍDA	112
5.8 CONFIGURAÇÃO DO ARQUIVO IAX.CONF	114
5.8.1 CONFIGURAÇÃO DA SEÇÃO GERAL	114
5.8.2 CONFIGURAÇÃO DOS CLIENTES IAX	115
5.8.3 CAMPOS DO TIPO “USER”:	116
5.8.4 CONFIGURAÇÃO DE “PEERS” IAX	117
5.9 EXEMPLO: ARQUIVO DE CONFIGURAÇÃO IAX	117
5.10 COMANDOS DE CONSOLE	118
5.11 SUMÁRIO	118
5.12 QUESTIONÁRIO	120

O PROTOCOLO SIP E O ASTERISK **123**

6.1 OBJETIVOS	123
6.2 VISÃO GERAL	123
6.3 TEORIA DA OPERAÇÃO DO SIP	123
6.4 PROCESSO DE REGISTRO DO SIP	125
6.5 OPERAÇÃO DO SIP EM MODO PROXY.	126
6.6 OPERAÇÃO EM MODO DE REDIRECT.	126
6.7 SIP NO MODO ASTERISK	127
6.8 CENÁRIOS DE USO SIP	128
6.8.1 CONECTANDO A UM PROVEDOR SIP.	128
6.8.2 ASTERISK COMO UM SIP SERVER	130
6.9 NOMENCLATURA DOS CANAIS SIP	132
6.10 ARQUIVO DE CONFIGURAÇÃO SIP.CONF	133
6.10.1 CONFIGURAÇÃO DA SEÇÃO GERAL [GENERAL]	133
6.10.2 CONFIGURAÇÕES DO SIP – PEERS E CLIENTS	134
6.11 SIP NAT TRAVERSAL	136
6.11.1 FULL CONE (CONE COMPLETO)	136
6.11.2 RESTRICTED CONE (CONE RESTRITO)	137
6.11.3 PORT RESTRICTED CONE (CONE RESTRITO POR PORTA)	138
6.11.4 SIMÉTRICO	138
6.11.5 SINALIZAÇÃO SIP	139
6.11.6 FLUXO DE MÍDIA RTP	139
6.11.7 UPnP	141

6.11.8 CONSULTA EXTERNA	141
6.11.9 STUN	142
6.11.10 MÍDIA ORIENTADA À CONEXÃO	143
6.11.11 RTP-RELAY	144
6.12 QUESTIONÁRIO	145
VISÃO GERAL DO PLANO DE DISCAGEM	147
<hr/>	
7.1 OBJETIVOS DO CAPÍTULO	147
7.2 VISÃO GERAL DO PLANO DE DISCAGEM	147
7.3 DESCRIÇÃO DO ARQUIVO EXTENSIONS.CONF	147
7.3.1 [GENERAL]	147
7.3.2 SEÇÃO [GLOBALS]	148
7.4 CONTEXTOS E EXTENSÕES	149
7.4.1 INTRODUÇÃO À CONTEXTOS E EXTENSÕES	149
7.4.2 COMO OS CONTEXTOS SÃO USADOS?	151
7.4.3 EXTENSÕES	151
7.5 SWITCHES	152
7.5.1 ENCAMINHANDO PARA OUTRO ASTERISK	152
7.6 VARIÁVEIS E EXPRESSÕES	152
7.6.1 USANDO VARIÁVEIS NOS PLANOS DE DISCAGEM	152
7.6.5 VARIÁVEIS ESPECÍFICAS DE APLICAÇÕES	155
7.6.6 VARIÁVEIS ESPECÍFICAS PARA MACROS	155
7.6.7 VARIÁVEIS DE AMBIENTE	156
7.7 FUNÇÕES DE MANUSEIO DE “STRINGS”	156
7.7.1 COMPRIMENTO DA STRING	156
7.7.2 SUBSTRINGS	157
7.7.3 CONCATENAÇÃO DE STRINGS	157
7.8 INCLUSÃO DE CONTEXTOS	158
7.7.2 COMO O PLANO DE DISCAGEM ENCONTRA A EXTENSÃO	159
7.7.3 PROCESSO “ENCONTRA ENQUANTO VOCÊ DISCA”.	160
7.7.4 EXEMPLO	162
7.7.5 ORDEM DE BUSCA DOS PADRÕES DE EXTENSÃO	163
7.7.6 CONTROLANDO O ORDENAMENTO	164
7.8 DEFININDO EXTENSÕES	165
7.8.1 CONTEXTOS BASEADOS EM HORÁRIO	167
7.8.2 DISCANDO 0 PARA PEGAR A LINHA EXTERNA.	168
7.8.3 ROTEAMENTO PELO ORIGINADOR DA CHAMADA	169
7.8.4 EVITANDO O TELEMARKETING	170
7.8.5 TOCANDO VÁRIAS EXTENSÕES	170
7.8.6 MENU DE VOZ	170
7.9 MACROS	171
7.10 EXTENSÕES PADRÃO E PRIORIDADES	173
7.11 PADRÕES DE EXTENSÃO	174
7.12 A BASE DE DADOS DO ASTERISK	175
7.12.1 FAMÍLIAS	175
7.12.2 APLICAÇÕES	175

8.4.3 EXEMPLO DE USO DO ASTERISK DB.	175
7.13 SUMÁRIO	177
7.14 QUESTIONÁRIO	178

CONSTRUINDO O PLANO DE DISCAGEM **181**

8.1 OBJETIVOS	181
8.2 UM PLANO DE DISCAGEM NA PRÁTICA	182
8.3 PASSO 1 - CONFIGURANDO OS CANAIS.	182
8.3.1 TRONCOS ANALÓGICOS (ZAPATA.CONF)	182
8.3.2 CANAIS SIP (SIP.CONF)	183
8.4 PASSO 2 - CONFIGURANDO O PLANO DE DISCAGEM	184
8.4.1 DEFININDO OS RAMAIS	184
8.4.2 DEFININDO A SAÍDA PARA DDD	184
8.4.3 DEFININDO A SAÍDA PARA DDI	185
8.4.5 DEFININDO AS CLASSES DE RAMAL	185
8.5 RECEPÇÃO DAS CHAMADAS	185
8.5.1 MENUS: EXPEDIENTE E FORA DO EXPEDIENTE	185
8.5.2 MENUS: PRINCIPAL E VENDAS	186
8.6 VISÃO GERAL DAS APLICAÇÕES	186
8.6.1 O COMANDO DIAL()	187
8.6.2 O COMANDO BACKGROUND()	189
8.6.3 O COMANDO ANSWER()	191
8.6.4 O COMANDO GOTO()	191
8.7 QUESTIONÁRIO	193

CONFIGURANDO RECURSOS AVANÇADOS **195**

9.1 OBJETIVOS	195
9.2 SUPORTE AOS RECURSOS DE PABX	195
9.2.1 RECURSOS COM SUPORTE PARA TODO TIPO DE CANAL	195
9.2.2 RECURSOS COM SUPORTE EM TELEFONES SIP	196
9.2.3 PARA TELEFONES ANALÓGICOS (ZAP).	197
9.3 ESTACIONAMENTO DE CHAMADAS	198
9.3.1 DESCRIÇÃO:	198
9.3.2 LISTA DE TAREFAS PARA CONFIGURAÇÃO	198
9.4 CAPTURA DE CHAMADAS - CALL-PICKUP	199
9.4.1 DESCRIÇÃO	199
9.4.2 LISTA DE TAREFAS PARA HABILITAR A CAPTURA.	199
9.5 TRANSFERÊNCIA DE CHAMADAS - CALL TRANSFER	199
9.5.1 DESCRIÇÃO	199
9.5.2 LISTA DE TAREFAS PARA CONFIGURAR	200
9.5.3 O ARQUIVO FEATURES.CONF	200
9.6 CONFERÊNCIA – CALL CONFERENCE	200
9.6.1 FORMATO:	200
9.6.2 DESCRIÇÃO:	201

9.6.3 CÓDIGOS DE RETORNO:	202
9.6.4 DETALHE DAS OPÇÕES:	202
9.6.5 LISTA DE TAREFAS DE CONFIGURAÇÃO	202
9.5.6 EXEMPLOS	202
9.5.7 ARQUIVO DE CONFIGURAÇÃO DO MEETME()	203
9.6 MÚSICA EM ESPERA – MUSIC ON HOLD	204
9.6.1 USANDO O MPG123.	204
EDITE OS ARQUIVOS PARA HABILITAR MÚSICA EM ESPERA	205
ESPECIFICANDO A MÚSICA	205
PERSONALIZANDO A MÚSICA EM ESPERA	205
9.7 QUESTIONÁRIO	207

DAC DISTRIBUIÇÃO AUTOMÁTICA DE CHAMADAS. **209**

10.1 OBJETIVOS	209
10.2 INTRODUÇÃO	209
10.3 MEMBROS	210
10.4 ESTRATÉGIAS	210
10.5 MENU PARA O USUÁRIO	210
10.6 NOVOS RECURSOS	210
10.7 CONFIGURAÇÃO	211
10.7.1 LISTA DE TAREFAS	211
10.7.2. CRIAR A FILA DE ATENDIMENTO.	211
10.7.3 DEFINIR PARÂMETROS DOS AGENTES.	212
10.7.4 CRIAR OS AGENTES NO ARQUIVO	212
10.7.5 COLOCAR A FILA NO PLANO DE DISCAGEM.	213
10.7.6 CONFIGURAR A GRAVAÇÃO	213
10.7.7 APLICAÇÕES DE APOIO PARA AS FILAS.	214
10.8 FUNCIONAMENTO DAS FILAS	214
10.8.1 LOGIN DO AGENTE	214
10.9 QUESTIONÁRIO	216

O CORREIO DE VOZ **219**

11.1 OBJETIVOS	219
11.2 INTRODUÇÃO	219
11.3 LISTA DE TAREFAS PARA CONFIGURAÇÃO	219
11.3.1 CONFIGURANDO O ARQUIVO VOICEMAIL.CONF	219
11.3.2 CONFIGURANDO O ARQUIVO EXTENSIONS.CONF	220
11.3.3 USANDO A APLICAÇÃO VOICEMAILMAIN()	220
11.3.4 SINTAXE DO VOICEMAIL():	221
11.3.5 CÓDIGOS DE RETORNO	223
11.4 ARQUIVO DE CONFIGURAÇÃO DO VOICEMAIL.	223
11.4.1 CONFIGURAÇÕES DA SEÇÃO [GENERAL].	223
11.4.2 VARIÁVEIS PARA EMAILSUBJECT E EMAILBODY.	228
11.4.3. CONFIGURAÇÕES PARA AS SEÇÕES [CONTEXT]	228

11.5 INTERFACE WEB PARA O CORREIO DE VOZ.	229
11.6 SUMÁRIO	229
11.6 QUESTIONÁRIO	230
RESPOSTA DOS EXERCÍCIOS	233
<hr/>	
RESPOSTAS DO CAPÍTULO 1	233
RESPOSTAS DO CAPÍTULO 2	235
RESPOSTAS DO CAPÍTULO 3	237
RESPOSTAS DO CAPÍTULO 4	240
RESPOSTAS DO CAPÍTULO 5	242
RESPOSTAS DO CAPÍTULO 6	244
RESPOSTAS DO CAPÍTULO 7	246
RESPOSTAS DO CAPÍTULO 8	248
RESPOSTAS DO CAPÍTULO 9	250
RESPOSTAS DO CAPÍTULO 10	252
RESPOSTAS DO CAPÍTULO 11	254

Introdução ao Asterisk

Neste capítulo vamos aprender o que é o Asterisk, qual é sua arquitetura e como pode ser utilizado.

1.1 Objetivos do capítulo

- Entender o que é o Asterisk, como surgiu o projeto e sua relação com outros projetos como o Zapata Telephony e qual o papel da Digium no Asterisk.
- Entender a arquitetura básica do Asterisk e se familiarizar com conceitos como aplicações, canais e codecs.
- Descobrir diversos cenários onde o Asterisk poderia ser usado.
- Entender as opções de desenvolvimento de novos recursos usando o Asterisk Manager Interface e Asterisk Gateway interface.

1.2 O que é o Asterisk

O Asterisk é um software de PABX que usa o conceito de software livre (GPL), criado pela Digium Inc. e uma base de usuários em contínuo crescimento. A Digium investe em ambos, o desenvolvimento do código fonte do Asterisk e em hardware de telefonia de baixo custo que funciona com o Asterisk. O Asterisk roda em plataforma Linux e outras plataformas Unix com ou sem hardware conectando a rede pública de telefonia, PSTN (Public Service Telephony Network).

O Asterisk permite conectividade em tempo real entre as redes PSTN e redes Voip.

Com o Asterisk, você não apenas tem uma troca excepcional do seu PABX. O Asterisk é muito mais que um PABX padrão. Com o Asterisk em sua rede, você criar coisas novas em telefonia como:

- Conectar empregados trabalhando de casa para o PABX do escritório sobre conexões de banda larga.

- Conectar escritórios em vários estados sobre IP. Isto pode ser feito pela Internet ou por uma rede IP privada.
- Dar aos funcionários, correio de voz, integrado com a “web” e seu e-mail
- Construir aplicações de resposta automática por voz, que podem conectar você ao sistema de pedidos, por exemplo, ou ainda outras aplicações internas.
- Dar acesso ao PABX da companhia para usuários que viajam, conectando sobre VPN de um aeroporto ou hotel.
- E muito mais...

O Asterisk inclui muitos recursos que só eram encontrados em sistemas de mensagem unificada “topo de linha” como:

- Música em espera para clientes esperando nas filas, suportando streaming de media assim como música em MP3.
- Filas de chamada onde agentes de forma conjunta atendem as chamadas e monitoram a fila.
- Integração para sintetização da fala (text-to-speech).
- Registro detalhado de chamadas (call-detail-records) para integração com sistemas de tarifação.
- Integração com reconhecimento de voz (Tal como o software de código aberto para reconhecimento de voz).
- A habilidade de interfacear com linhas telefônicas normais, ISDN em acesso básico (2B+D) e primário (30B+D).

1.2.1 Qual o papel da Digium?

A digium é baseada em Huntsville, Alabama, A Digium é a criadora e desenvolvedora primária do Asterisk, o primeiro PABX de código aberto da indústria. Usado em conjunto com as placas de telefonia PCI, ele oferece uma abordagem estratégica com excelente relação custo/benefício para o transporte de voz e dados sobre arquiteturas TDM, comutadas e redes Ethernet.

A digium é hoje o principal patrocinador do Asterisk e um dos líderes na indústria do PABX em código aberto, sendo Mark Spencer o criador e principal mantenedor do Asterisk, ele é hoje admirado pelo grande trabalho que fez e pela responsabilidade que carrega.

1.2.3 O projeto Zapata

O projeto ZAPATA foi conduzido por Jim Dixon. Ele é o responsável pelo desenvolvimento do hardware da DIGIUM. É interessante ressaltar que o hardware também é aberto e pode ser produzido por qualquer empresa. Hoje a placa com 4 E1/T1s é produzida pela Digium e também pela Varion (www.govarion.com). A história do projeto zapata pode ser vista em :

<http://www.asteriskdocs.org/modules/tinycontent/index.php?id=10>)

Uma pequena tradução pode ser encontrada abaixo,

Por Jim Dixon

Há 20 ou 25 anos atrás, a AT&T começou a oferecer uma API permitindo aos usuários customizar a funcionalidade de seu sistema de correio de voz e auto-atendimento chamado Audix. O Audix rodava em plataforma Unix e custava como tudo em telefonia até o momento, milhares de dólares por porta com uma funcionalidade bastante limitada.

Em uma tentativa de tornar as coisas possíveis e atrativas (Especialmente para quem não tinha um PABX AT&T) alguns fabricantes vieram com uma placa que podia ser colocada em um PC que rodava DOS e respondia a uma única linha telefônica (FXO apenas). As placas não tinham uma qualidade tão boa quanto as atuais e muitas terminaram como secretárias eletrônicas igualmente ruins.

Novas placas de telefonia foram lançadas com preços muito salgados e as companhias continuaram gastando na faixa de milhares de dólares por porta. Afinal de contas, mesmo com as margens altas de muitos fabricantes, as placas de telefonia possuíam muita capacidade de processamento na forma de DSPs, processadores de sinais digitais. Se você observar ainda hoje um gateway de voz sobre ip, vai ver que boa parte do custo ainda está relacionada aos DSPs.

No entanto, o poder de processamento dos microcomputadores continuou crescendo. De forma a provar o conceito inicial comprei uma placa Mitel89000C "ISDN Express Development Card" e escrevi um driver para o FreeBSD. A placa ocupou bem pouco processamento de um Pentium III

600Mhz, provando que se não fosse a limitação do I/O (A placa gerenciava de forma ineficiente o I/O exigindo muitos wait-states) ela poderia atender de 50 à 75 canais. Como resultado do sucesso, eu sai e comprei o necessário para criar um novo desenho de cartão ISA que usasse o I/O de forma eficiente. Eu consegui dois T1s (48 canais) de dados transferidos sobre o barramento e o PC gerenciou isto sem problemas. Então eu tinha as placas e ofereci-as para venda (Umás 50 foram vendidas) e coloquei o desenho completo (incluindo arquivos de plotagem da placa) na web. .

Como o conceito era revolucionário e sabia que faria ondas na indústria, Eu decidi colocar um nome inspirado no revolucionário mexicano e dei o nome à organização de Emiliano Zapata e decidi chamar a placa de “tormenta”. Assim começou a telefonia ZAPATA. Escrevi um driver completo e coloquei na rede. A resposta que eu obtive foi quase sempre, “ótimo e você tem para Linux?”.

Pessoalmente eu nunca havia visto o linux rodar antes, mas fui rapidamente ao Fry's (Uma loja enorme de produtos eletrônicos, famosa nos EUA) e comprei uma cópia do Linux Red Hat 6.0. Eu dei uma olhada nos drivers e usei o Vídeo Spigot como base para traduzir o driver de BSD para Linux.

De qualquer forma minha experiência com Linux não era grande e comecei a ter problemas em desenvolver o módulo do kernel na forma de módulos carregáveis. De qualquer forma liberei-o na Net sabendo que algum guru no Linux iria rir dele e talvez me ajudar a reformatá-lo em “Linuquês” apropriado. Em 48 horas eu recebi um e-mail de um cara no Alabama (Mark Spencer), que se ofereceu para fazer exatamente isto. Note apenas que, ele disse que tinha algo que seria perfeito para a coisa toda (O Asterisk).

Neste momento o Asterisk era um conceito funcional, mas não tinha uma forma real de funcionar de forma prática e útil. O casamento do sistema de telefonia Zapata e o desenho da biblioteca de hardware/driver e interface permitiu à ele crescer para ser um PABX real que poderia falar com telefones reais, linhas e etc.

Além disso, Mark era brilhante em VOIP, redes, na parte interna do sistema etc., e tinha um grande interesse em telefones e telefonia, mas tinha experiência limitada em sistemas de telefonia e como eles funcionavam, particularmente na área de interfaces de hardware. Desde o início eu estava e sempre estive lá para ajudá-lo nestas áreas, ambos fornecendo informação e implementando código nos drivers e no switch (PABX). Nós e mais recentemente outros, fazemos um bom time trabalhando em um objetivo comum

de trazer o estado da arte em tecnologia de Telecom ao público por um custo realista.

Desde o cartão ISA, eu desenhei o “Tormenta 2 PCI Quad T1/E1, o qual o Mark vende como Digium T400P e E400P, e agora a Varion está vendendo como V400P (Ambos T1 e E1). Todos os arquivos de projeto (incluindo foto e arquivos de plotagem) estão disponíveis em [zapatatelephony.org](http://www.zapatatelephony.org) (<http://www.zapatatelephony.org>) para uso público. Mais desenhos de maior densidade estão à caminho,

“Como qualquer um pode ver, com o trabalho dedicado de Mark (um monte do meu e outras pessoas) nos drives da ZapTel e no software do Asterisk, as tecnologias vêm de um longo tempo e crescem e melhoram a cada dia”

1.4 Porque o Asterisk?

Eu me lembro do meu primeiro contato com o Asterisk, a primeira reação ao encontrarmos algo novo que compete com aquilo que conhecemos é rejeitar. Foi o que aconteceu, na primeira vez que vi o Asterisk ele concorria com uma solução que eu estava apresentando. De qualquer forma, eu sempre procuro levantar todas as informações sobre as alternativas aos projetos que faço e tento descobrir quais os pontos fortes e fracos de uma solução como o Asterisk. Posso dizer que após alguns dias eu fiquei pasmo, sabia que o Asterisk traria uma mudança profunda em todo o mercado de telecomunicações e voz sobre IP. O Asterisk é o Apache da telefonia. Deixe-me então dar várias razões para o Asterisk e algumas limitações que ainda existem quando da publicação deste livro.

1.4.1 Redução de custos extrema

Se você comparar um PABX convencional com o Asterisk talvez à diferença seja pequena, principalmente pelo custo do hardware e dos telefones IP. Entretanto, o Asterisk só pode ser comparado a um PABX digital estado da arte. Comparar uma central analógica de quatro troncos e 16 ramais com o Asterisk é no mínimo injusto.

Quando você adiciona recursos avançados como VoIP, URA e DAC, a diferença vai à mais de dez para um em custo fácil. Para dar exemplo, uma única porta de URA hoje com acesso à mainframe, que foi cotada recentemente para um cliente nosso custou US\$ 1700,00.

1.4.2 Ter controle do seu sistema de telefonia

Este é um dos benefícios mais citados, ao invés de esperar alguém configurar o seu PABX proprietário (alguns nem mesmo dão a senha para o cliente final), configure você mesmo. Total liberdade e interface padrão. No fim das contas é LINUX.

1.4.3 Ambiente de desenvolvimento fácil e rápido

O asterisk pode ser programado em C com as APIs nativas, ou em qualquer outra linguagem usando AGI.

1.4.4 Rico e abrangente em recursos

Como temos ressaltado desde o início, poucos são os recursos encontrados em equipamentos PABX vendidos no mercado que não possam ser encontrados ou criados no Asterisk. Já o reverso, para encontrar tudo que tem no Asterisk em um PABX convencional...

1.4.5 É possível prover conteúdo dinâmico por telefone.

Como o Asterisk é programado com C ou outras linguagens de domínio da maioria dos programadores, as possibilidades de prover conteúdo dinâmico por telefone são sem limite.

1.4.6 Plano de discagem flexível e poderoso

Mais uma vez o Asterisk se supera. Se pensarmos, a maioria das centrais, nem mesmo rota de menor custo possuem. Com o Asterisk este processo é simples e prático.

1.4.7 Roda no Linux e é código aberto

Uma das coisas mais fantásticas do Linux é a comunidade de software livre. Quando eu acesso o Wiki, ou os forums de software em código aberto eu percebo que a adoção de usuários é muito rápida, milhares de questões e relato de problemas são enviados todos os dias. O Asterisk é provavelmente um dos softwares que mais pessoas têm disponíveis para testes e avanços. Isto torna o código estável e permite a rápida resolução de problemas.

1.4.8 Limitações de acesso à rede pública no Brasil

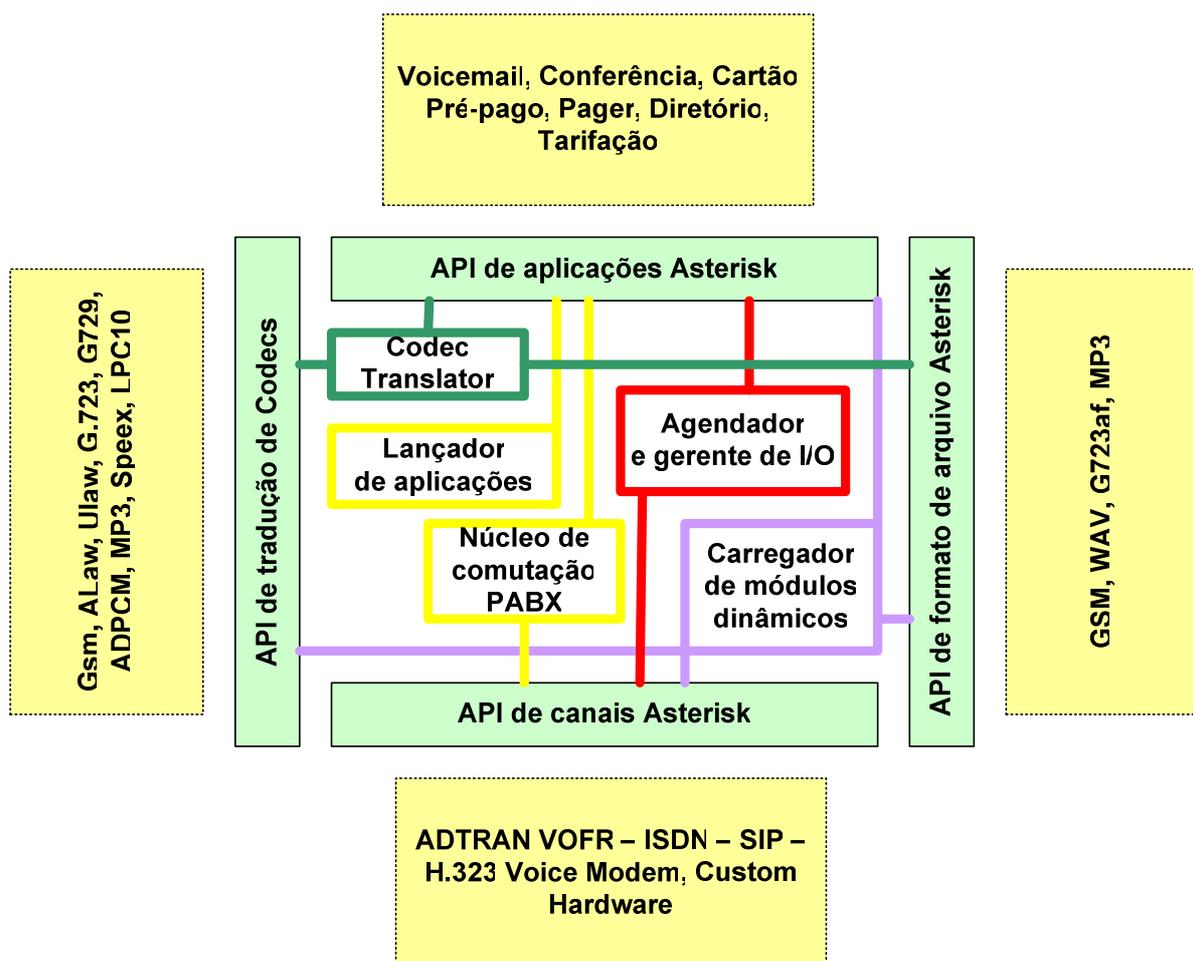
Ainda falta no Asterisk um driver para acesso à **R2 Brasil** com código aberto. Já existem algumas implementações no Brasil, mas o código por enquanto está fechado. Isto limita o acesso à rede pública. Felizmente em Santa Catarina onde tenho meu escritório tanto a GVT quanto a Brasil Telecom dispõe de sinalização ISDN. Em alguns lugares como São Paulo, é difícil conseguir um

ISDN e o mais comum são ainda circuitos E1 com sinalização R2. (A conexão à rede pública com FXO pode ser feita para linhas analógicas, esta restrição se aplica apenas à acessos digitais).

1.4.9 Limitações da arquitetura do Asterisk

O Asterisk usa a CPU do servidor para processar os canais de voz, ao invés de ter um DSP (processador de sinais digitais) dedicado a cada canal. Enquanto isto permitiu que o custo fosse reduzido para as placas E1/T1, o sistema é muito dependente da performance da CPU. Minha recomendação é preservar ao máximo a CPU do Asterisk, rodá-lo sempre em uma máquina dedicada e testar o dimensionamento antes de implantar. Na minha opinião, o Asterisk deve ser sempre implementado em uma VLAN específica para VoIP, qualquer tempestade de broadcasts causada por loops ou vírus pode comprometer o seu funcionamento devido ao uso de CPU das placas de rede quando este fenômeno acontece.

1.5 Arquitetura do Asterisk



A figura acima mostra a arquitetura básica do Asterisk. Vamos explicar abaixo os conceitos relacionados à figura acima como canais, codecs e aplicações.

1.5.1 Canais

Um canal é o equivalente à uma linha telefônica na forma de um circuito de voz digital. Ele geralmente consiste de ou um sinal analógico em um sistema POTS¹ ou alguma combinação de CODEC e protocolo de sinalização (GSM com SIP, Ulaw com IAX). No início as conexões de telefonia eram sempre analógicas e por isso mais suscetíveis à ruídos e eco. Mais recentemente, boa parte da telefonia passou para o sistema digital, onde o sinal analógico é codificado na forma digital usando normalmente PCM (Pulse Code Modulation). Isto permite que um canal de voz seja codificado em 64 Kilobits/segundo sem compactação.

Alguns dos hardwares que o Asterisk suporta:

- Zaptel – Wildcard T410P – Placa E1/T1 com quatro portas (PCI 3.3 volts apenas)
- Zaptel – Wildcard T405P – Placa E1/T1 com quatro portas (PCI 5 volts apenas)
- Zaptel – TDM400P – Placa com quatro portas para tel. analógicos e ADSI,
- Zaptel - TE110P – Placa com E1/T1 com uma porta, meio-comprimento.
- Quicknet, - as placas quicknet, tanto PhoneJack quanto LineJack podem ser usadas com o Asterisk
- ISDN4Linux – É um driver antigo para placas ISDN BRI, acesso básico. Placas neste padrão poderão ser usadas no Asterisk.
- ISDN CAPI – É a outra forma de suportar as placas ISDN BRI no Linux. Placas que suportam este padrão poderão ser usadas com o Asterisk.

¹ POTS – Plain Old Telephony System, sistema de telefonia convencional, baseado normalmente em linhas analógicas.

- Voicetronix: possui placas com maior densidade de canais FXS e FXO que as da Digium.

Canais que o Asterisk suporta:

- Agent: Um canal de agente DAC.
- Console: Cliente de console do Linux, driver para placas de som (OSS ou ALSA).
- H323: Um dos protocolos mais antigos de VoIP, usado em muitas implementações.
- IAX e IAX2: Inter-Asterisk Exchange protocol, o próprio protocolo do Asterisk.
- MGCP: Media Gateway Control Protocol, outro protocolo de VOIP.
- Modem: Usado para linhas ISDN e não modems.
- NBS: Usado para broadcast de som.
- Phone: Canal de telefonia do Linux.
- SIP: Session Initiation Protocol, o protocolo de VoIP mais comum.
- Skinny: Um driver para o protocolo dos telephones IP da Cisco.
- VOFR: voz sobre frame-relay da Adtran.
- VPB: Linhas telefônicas para placas da Voicetronix.
- ZAP: Para conectar telephones e linhas com placas da Digium. Também usado para TDMoE (TDM sobre Ethernet) e para o Asterisk zphfc (ISDN em modo NT).

Alguns drivers que podem ser instalados:

- Bluetooth: Permite o uso de dispositivos Bluetooth para mudar o roteamento.
- CAPI: canal ISDN CAPI

- mISDN: canal mISDN channel
- SCCP: Um driver alternativo para o Skinny.

1.5.2 Codecs and Conversões de CODEC

Obviamente é desejado colocar tantas chamadas quanto possíveis em uma rede de dados. Isto pode ser feito codificando em uma forma que use menos banda passante. Este é o papel do CODEC (COder/DECoder), alguns CODECs como o g.729 permitem codificar à 8 Kilobits por segundo, uma compressão de 8 para 1. Outros exemplos são ulaw, alaw, gsm, ilbc e g729.

O Asterisk suporta os seguintes CODECs:

- G.711 ulaw (usado nos EUA) – (64 Kbps).
- G.711 alaw (usado na Europa e no Brasil) – (64 Kbps).
- G.723.1 – Precisa de licenciamento (5.3-6 Kbps)
- G.726 - 32kbps no Asterisk 1.0.3, 16/24/32/40kbps no CVS HEAD.
- G.729 – Precisa de licença, a menos que esteja usando o modo pass-thru. Versão gratuita disponível para uso em países sem patentes ou para uso educacional. (8Kbps)
- GSM – (12-13 Kbps)
- iLBC – (15 Kbps)
- LPC10 - (2.5 Kbps)
- Speex - (2.15-44.2 Kbps)

1.5.3 Protocolos

Enviar dados de um telefone à outro seria fácil se os dados encontrassem seu próprio caminho para o outro telefone. Infelizmente isto não acontece, é preciso um protocolo de sinalização para estabelecer as conexões, determinar o ponto de destino, e também questões relacionadas à sinalização de telefonia como campanha, identificador da chamada, desconexão etc. Hoje é comum o uso do SIP (Session Initiated Protocol), muito embora outros protocolos também sejam expressivos no mercado como o H.323, o MGCP e recentemente

o IAX que é excepcional quando se trata de trunking e NAT (Network Address Translation). O asterisk suporta:

- SIP
- H323
- IAXv1 e v2
- MGCP
- SCCP (Cisco Skinny).

1.5.4 Aplicações

Para conectar as chamadas de entrada com as chamadas de saída ou outros usuários do asterisk são usadas diversas aplicações como o Dial, por exemplo. A maior parte das funcionalidades do Asterisk são criadas na forma de aplicações como o VoiceMail(), correio de voz, o Meetme(), conferência, entre outras.

1.6 Cenários de uso do Asterisk

Abaixo vamos mostrar alguns cenários de uso do Asterisk e como ele se encaixa no seu modelo atual de telefonia.

1.6.1 Visão Geral

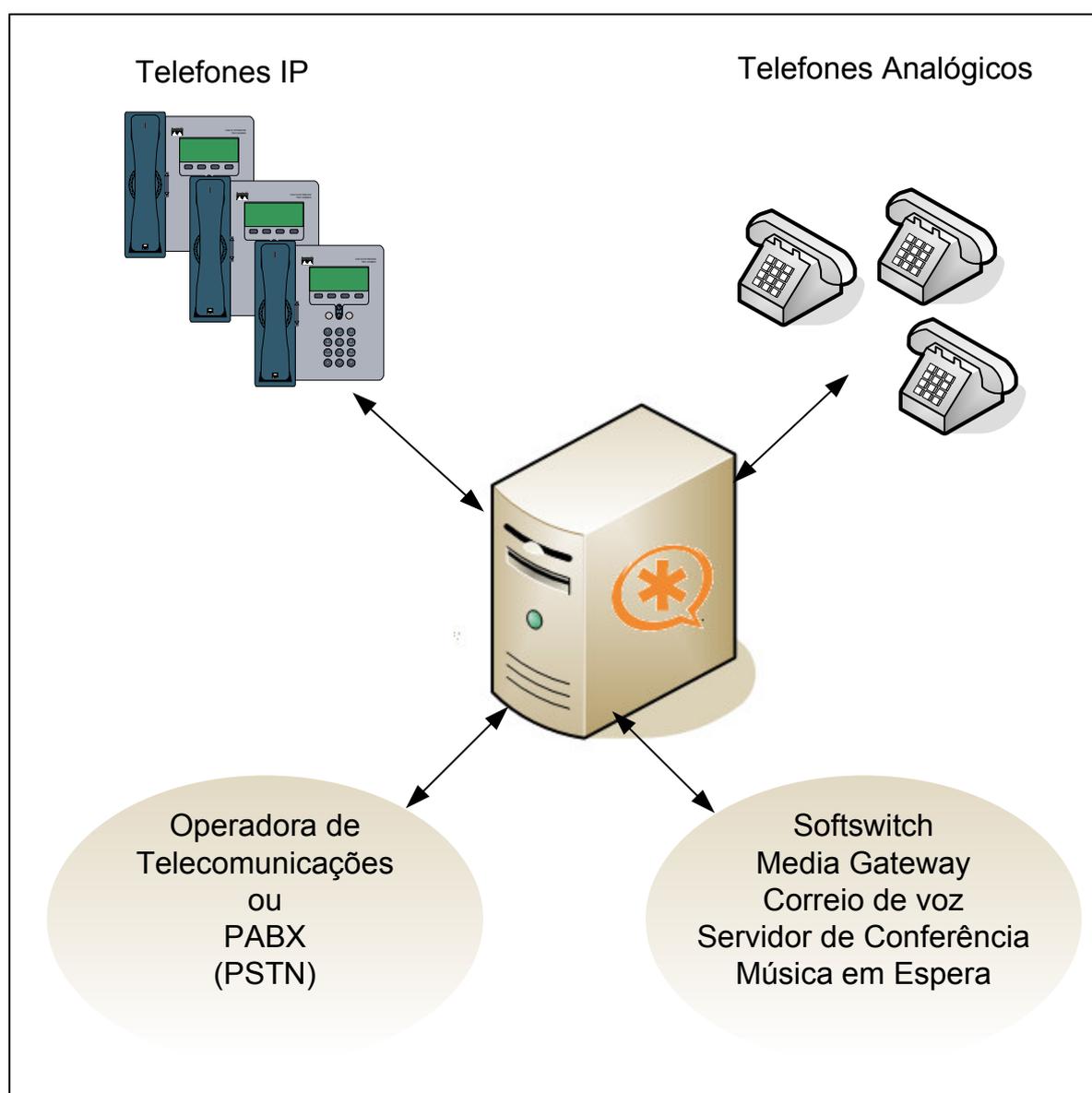


Figura 1 – Visão Geral

Dentro de uma visão geral, o Asterisk é um PABX híbrido que integra tecnologias como TDM² e telefonia IP com funcionalidade de unidade de resposta automática e distribuição automática de chamadas. “Que definição!!”,

² TDM – TDM - multiplexação por divisão de tempo, toda a telefonia convencional está baseada neste conceito, quando falarmos em TDM estaremos nos referindo a circuitos T1 e E1. E1 é mais comum no Brasil e Europa, T1 é mais usado nos EUA.

neste momento do livro é provável que você não esteja entendendo todos estes termos, mas ao longo dos capítulos, você estará cada vez mais familiarizado. Na figura acima podemos ver que o Asterisk pode se conectar a uma operadora de telecomunicações ou um PABX usando interfaces analógicas ou digitais. Pode se comportar como um servidor de conferência, correio de voz, unidade de resposta automática, distribuidor automático de chamadas e servidor de música em espera. Os telefones podem ser IP, analógicos ou ADSI que é um telefone analógico com display digital.

Vamos conceituar de uma forma um pouco mais detalhada:

Correio de voz – Permite que quando o usuário não atender ao telefone por estar ocupado ou ausente, receba um “prompt” solicitando que deixe uma mensagem na caixa postal. É semelhante à uma secretária eletrônica ou caixa de mensagens do celular. O Asterisk apresenta esta funcionalidade, sem custo adicional.

Sistema de mensagens unificadas – É um sistema onde todas as mensagens são direcionadas para um único lugar, por exemplo, a caixa de correio eletrônico do usuário. Neste caso as mensagens de e-mail, junto com as mensagens do correio de voz e fax seriam encaminhadas para a caixa postal do usuário. No Asterisk também dá para fazer.

Distribuidor automático de chamadas e fila de atendimento – Este é um dos conceitos menos óbvios da telefonia. Na primeira vez que eu vi isto, eu pensei, mas eu já tenho isto na minha central, ela distribui para vários ramais, o primeiro que atende para de tocar os outros, porque tanto “OOHH” para esta coisa de DAC (ACD em inglês, Automatic Call Distribution). A resposta é simples, em um DAC, as pessoas normalmente se autenticam em uma fila de atendimento para receber as chamadas, o distribuidor verifica se o usuário está com o telefone livre antes de passar a chamada. Se nenhum operador estiver livre ele segura a chamada na fila com aquela “musiquinha” e uma mensagem como “Você ligou para..... Sua ligação é muito importante.....” (Que nós adoramos!!). No primeiro atendente que é liberado, o DAC passa a ligação. DAC é fundamental em qualquer sistema de atendimento e qualquer Call Center receptivo. Há muito mais sobre DAC do que está escrito aqui, o sistema de roteamento pode ser muito sofisticado. DAC custa uma pequena fortuna na maioria das plataformas convencionais.

Servidor de música em espera – Parece uma idiotice isso, mas acredite ou não, na maioria das centrais telefônicas é preciso colocar um aparelho de CD ligado à um ou vários ramais, para que o usuário fique ouvindo a “musiquinha”.

Se me permitem, na era digital isto é o “fim da picada”. Asterisk, “MP3 neles!!”.

Discador automático – Isto é muito útil em telemarketing, pode se programar o sistema para discar automático e distribuir numa fila. Mais uma tecnologia que é vendida separadamente em outros PABX. No Asterisk você pode programar a discagem e existem diversos exemplos de discador disponíveis na Internet.

Sala de Conferência – Permite que vários usuários falem em conjunto. É implementado como sala de conferência, você escolhe um ramal para ser a sala de conferência e todos os que discarem para lá estão imediatamente conectados. Tem várias opções como senha, por exemplo.

Estas são algumas das funcionalidades atuais do Asterisk, novas aplicações estão surgindo à cada dia, com a contribuição de centenas de pessoas ao redor do mundo.

PABX – Softswitch no modelo convencional

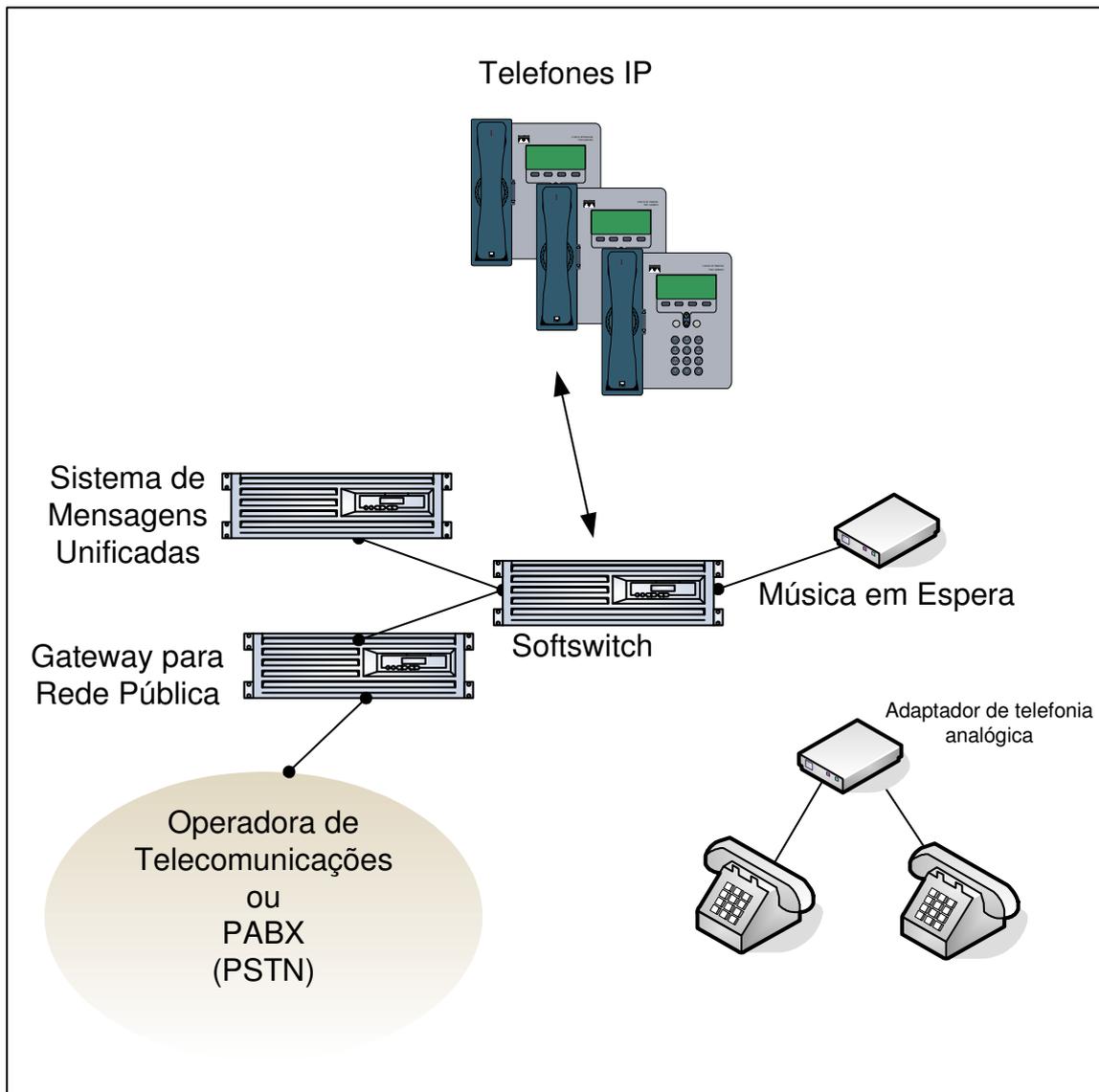


Figura 2 - PABX por software do tipo convencional (softswitch)

Já é comum nos dias de hoje o uso de softswitches, que são PCs que comutam circuitos de hardware na forma de interfaces padrão de telefonia. Entretanto a forma de comercialização destes equipamentos segue muitas vezes a lógica mostrada na figura 2, todos os componentes são separados e muitas vezes de diferentes fabricantes. Em muitos casos, mesmo a tarifação é feita por um servidor separado. Os custos da aquisição de cada um destes componentes é elevado e a integração muitas vezes difícil.

1.6.2 Telefonia do jeito Asterisk

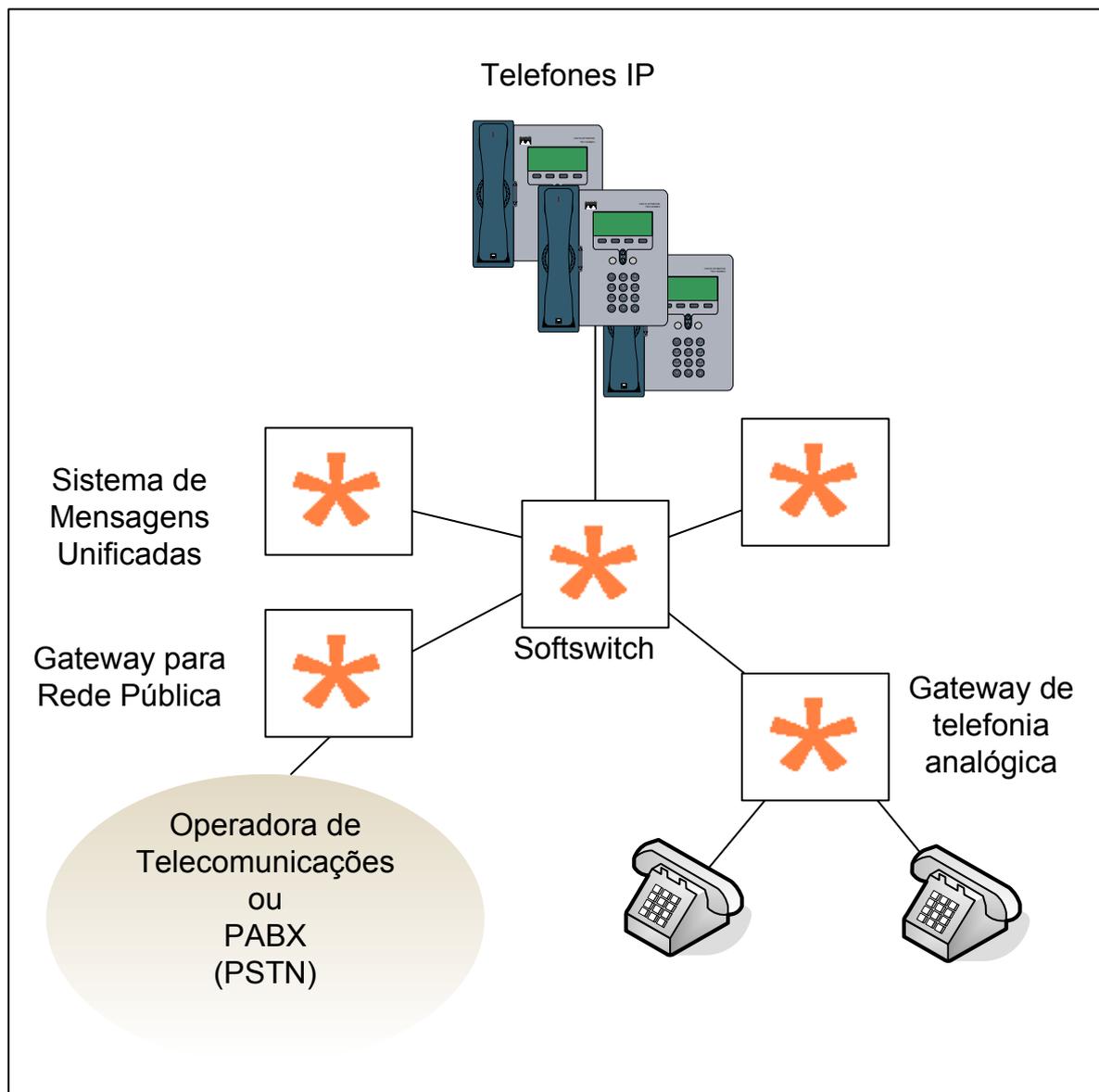


Figura 3 – Telefonia do jeito Asterisk

O Asterisk faz todas estas funções de forma integrada, o licenciamento é gratuito (GPL General Public License) e pode ser feito em um único ou em vários servidores de acordo com um dimensionamento apropriado. Incrível dizer isto, mas posso atestar que às vezes é mais fácil implementar o Asterisk do que até mesmo especificar um sistema de telefonia convencional, com todo o seu licenciamento por usuário, por linha, por sabe se lá o que.

Se me permitem a crítica neste livro, os grandes fabricantes além de caros criaram uma barreira enorme à adoção dos seus produtos, pois é preciso um especialista para descobrir o que está ou não incluído nas licenças e de que jeito se licencia. Esse é o um dos pontos individuais que mais contribuíram para que eu abraçasse o Asterisk.

1.6.3 O clássico PABX 1x1

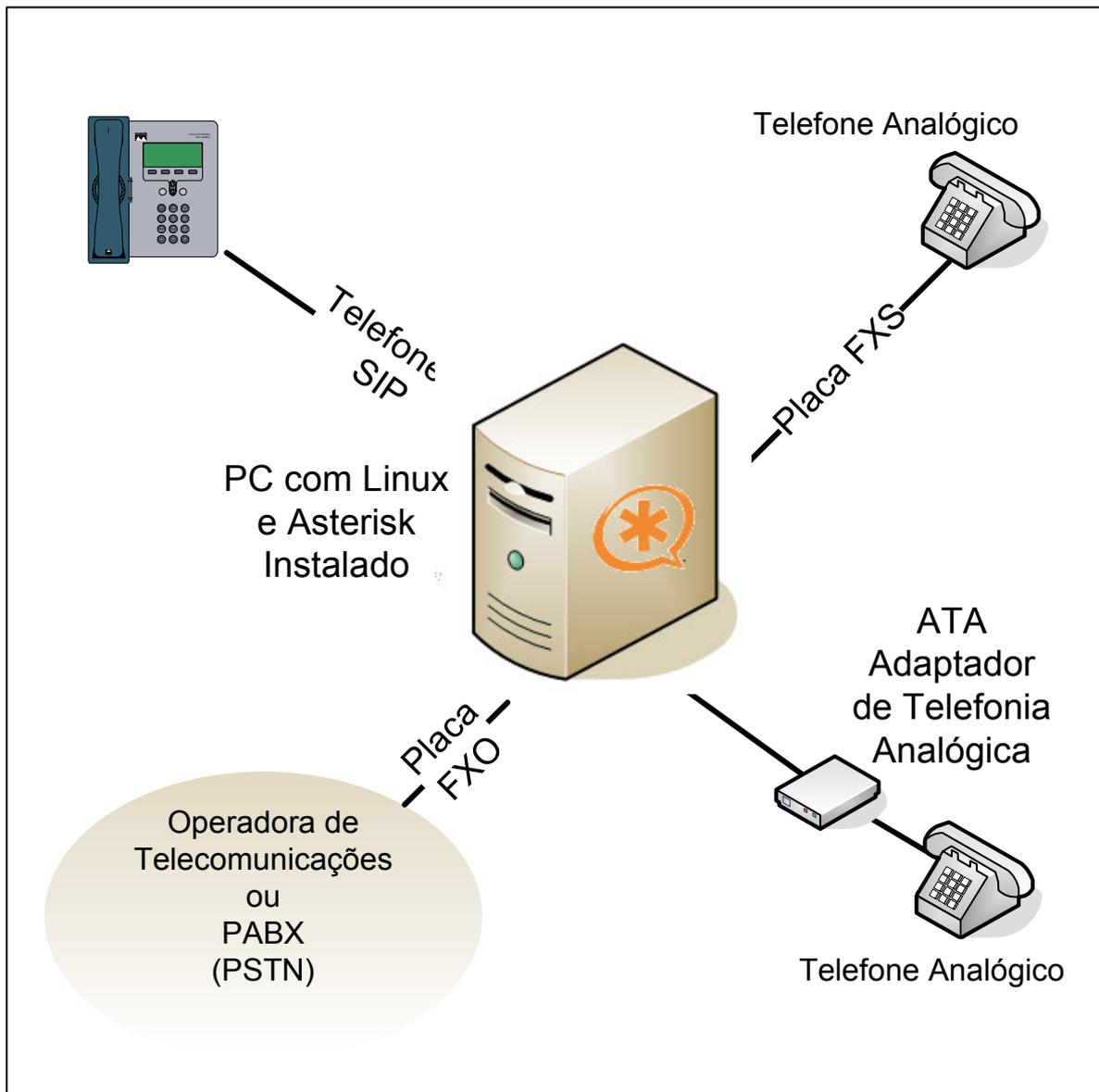


Figura 4 - O clássico PABX 1x1

Na figura 4, você vê um exemplo de um PABX de um tronco e uma linha. Este é um dos sistemas mais simples que você pode construir com o Asterisk. Apesar de ter pouca utilidade prática ele permite que se conceituem alguns pontos importantes. Em primeiro lugar o PABX 1x1 possui uma placa FXO (Foreign Exchange Office) para se ligar às operadoras ou a uma interface de ramal. Você pode adquirir uma placa desta da Digium sob o nome TDM400P. Outras duas possibilidades para uma interface FXO são um voice-modem com chipset Intel MD3200 (Cuidado, apenas alguns chipsets funcionam, teste antes de comprar, eu testei a Ambient MD3200 e funciona legal, se você não quiser correr risco, compre a placa da Digium).

1.6.4 Crescendo o seu PABX usando um banco de canais

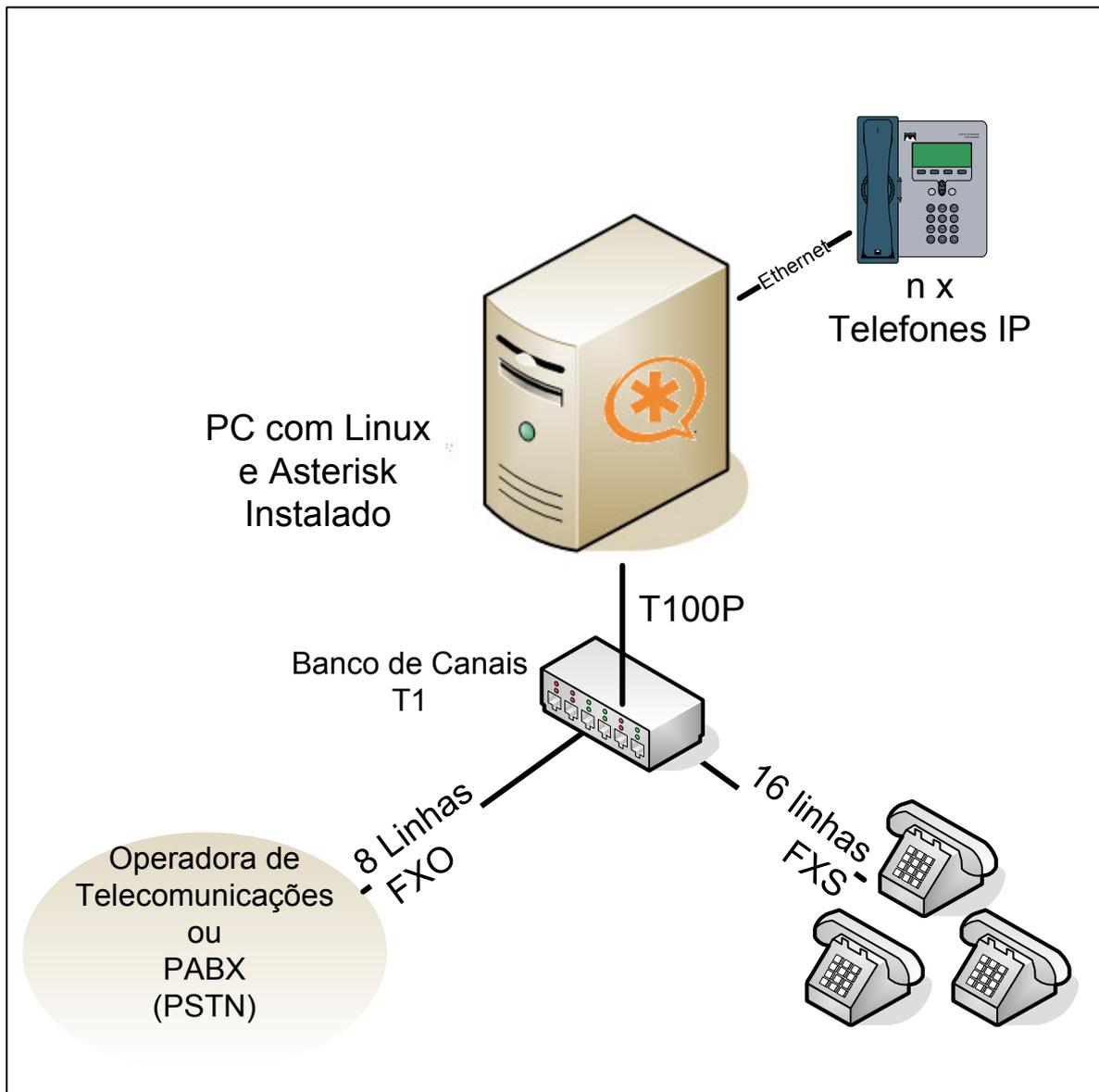


Figura 5 – PABX com banco de canais

Chega uma hora que é difícil continuar colocando placas no PC. A maioria das placas-mãe não permite muito mais do que 4 ou 5 slots PCI. Se você quiser atender oito troncos e 16 ramais, já fica difícil. Por exemplo, se você usar a TDM400P apenas quatro canais por placa são possíveis. Neste caso você pode usar um banco de canais. Um banco de canais é um multiplexador onde entra um E1 (30 canais) ou T1 (24 canais) e no banco de canais estes sinais são abertos em diversas interfaces analógicas FXS, FXO e mesmo E+M. A Adtran é uma das empresas que fabrica estes bancos de canais. Existem diversos fabricantes que fabricam bancos de canal GSM, o que permite que você ligue até 30 linhas de celular no seu Asterisk. Como sempre é bom testar ou consultar alguém que já tenha feito isto, você não vai investir milhares de dólares antes de ter certeza que funciona.

1.6.5 Interligação de filiais à matriz

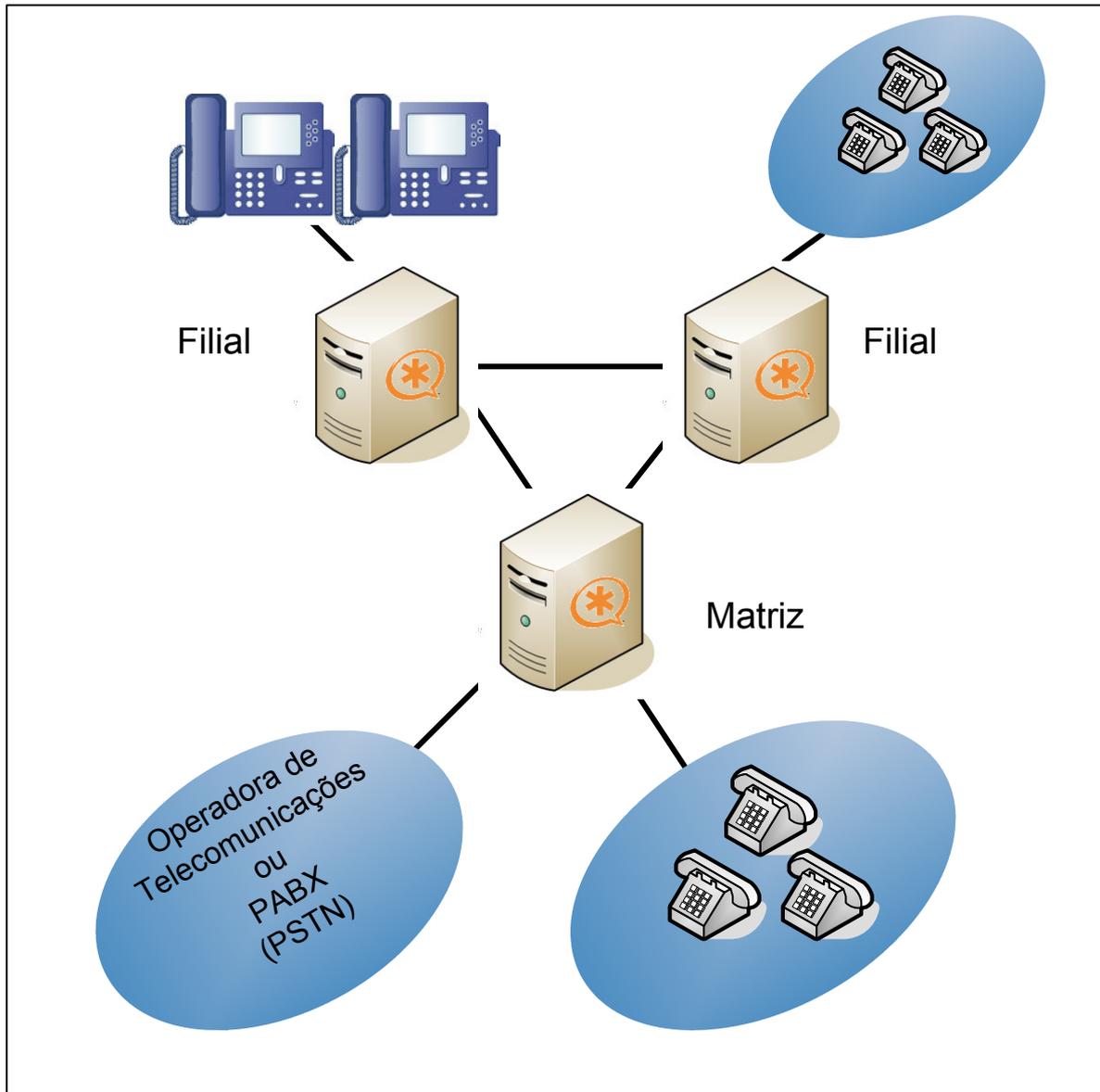


Figura 6 – Interligação de matriz e filiais

O Asterisk possui a funcionalidade de um gateway de mídia. Ele pode converter os sinais analógicos (FXS, FXO) ou digitais (ISDN) vindos da central telefônica, ou dos telefones do cliente em voz sobre IP e transmitir pela rede corporativa de dados. Com a convergência à redução do número de circuitos e um melhor aproveitamento dos recursos. Os projetos mais comuns são conhecidos como “Toll-Bypass” (Contornando a tarifação), pois eliminam os custos de operadora de longa distância nos telefonemas entre as filiais da empresa.

Media Gateway – Um gateway de mídia permite que suas ligações em telefonia analógica possam ser convertidas em Voz sobre IP, por exemplo, e transmitida pela rede de dados até outro escritório sem passar pela tarifação da rede pública. Este é o ponto número um da implementação de voz sobre IP,

reduzir a conta. Se você tem um Asterisk em cada filial, você pode interconectá-los usando IAX trunked, uma das melhores tecnologias de conexão de PABX por IP. Isto é o que me espanta, apesar de ser um software livre, este recurso de protocolo, em particular é superior ao que eu tenho encontrado em equipamentos pagos.

1.6.6 Unidade de resposta automática

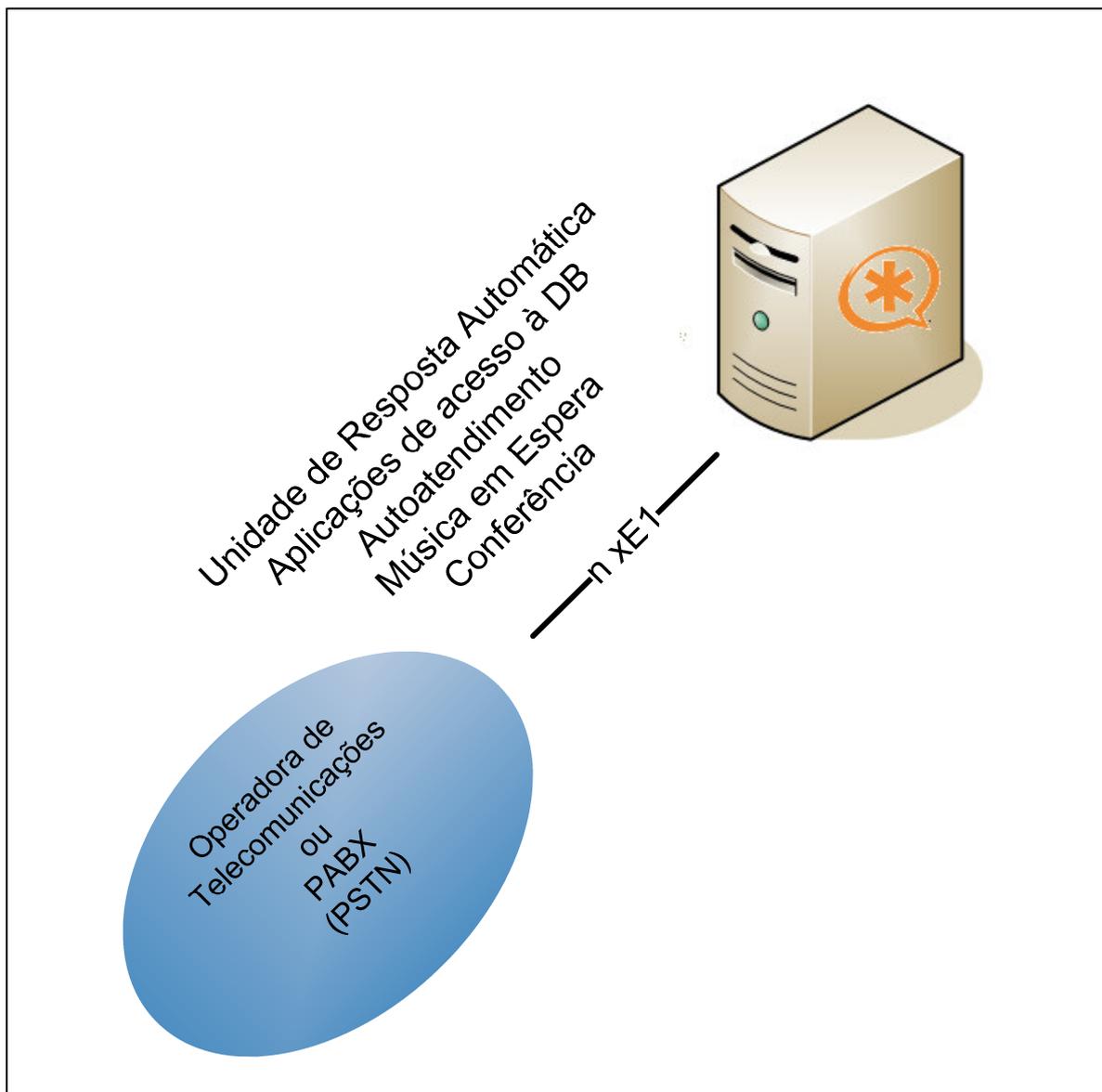


Figura 7 – Unidade de resposta automática

Unidade de resposta automática – Para mim este é um dos pontos altos do Asterisk, ele permite criar uma URA (Em inglês IVR, Interactive Voice Response) bastante personalizável. Isto permite que você, por exemplo, crie um sistema de consulta à estoque e preços para os vendedores, um sistema de atendimento à posição dos pedidos e inúmeras outras aplicações. É bom lembrar que ao contrário dos EUA e Europa, a quantidade de computadores por

Brasileiro é relativamente pequena e o telefone ainda é um dos meios de acesso mais universais no Brasil.

Usando recursos como o AGI (Asterisk Gateway Interface) que é muito semelhante ao CGI (Common Gateway Interface), as possibilidades de programação são muito amplas, a linguagem de programação pode ser escolhida pelo desenvolvedor. Acredito que serão cada vez mais comuns os portais de voz, para disseminar a informação por um conjunto ainda maior de pessoas.

1.7 Interface de gerenciamento do Asterisk.

A interface de gerenciamento do Asterisk permite ao programados se conectar ao Asterisk e emitir comandos ou ler eventos de PABX usando a interface sockets do TCP/IP. Integradores vão achar este recurso útil quando tentarem rastrear o status de um cliente dentro do Asterisk e direcionar o cliente baseado em uma regra personalizada, talvez até dinâmica.

Um protocolo de linha do tipo “chave:valor” é utilizado entre o cliente e o Asterisk. As linhas são terminadas com CRLF.

1.7.1 Comportamento do protocolo

O protocolo tem a seguinte semântica:

- Antes de você emitir quaisquer comandos você deve se logar usando a ação “Login”.
- Os pacotes podem ser transmitidos em qualquer direção à qualquer momento após a autenticação.
- A primeira linha do pacote deve ter uma chave “Action” quando enviado pelo cliente e “Event” quando enviado do Asterisk ao cliente.
- A ordem das linhas dentro de um pacote não é importante, então você pode usar um tipo de dado de dicionário não ordenado em sua linguagem de programação nativa para armazenar um único pacote.

1.7.2 Tipos de pacote

O tipo de pacote é determinado pela existência de uma das seguintes chaves:

Action: Um pacote enviado pelo cliente ao Asterisk, pedindo que uma ação em particular seja executada.

Response: A resposta enviada pelo Asterisk para a última ação enviada pelo cliente.

Event: Dados pertencentes à um evento gerado dentro do núcleo do Asterisk ou um módulo de extensão.

1.7.3 Autenticação

Contas de usuário são configurada em `/etc/asterisk/manager.conf`. Uma conta de usuário consiste de um conjunto de hosts que podem acessar a interface de gerenciamento, uma senha e uma lista de permissões, cada um pode ser ou “read”, “write”, ou “read/write”.

Exemplos de aplicações usando a interface de gerenciamento:

- Pode se criar uma aplicação que gera uma discagem a partir de uma página de Web.
- Pode se criar uma aplicação que monitora as ligações entrantes e jogam uma tela personalizada para a estação de trabalho que recebeu a ligação.

Detalhes de programação da interface de gerenciamento do Asterisk estão fora do escopo deste material.

1.8 Asterisk Gateway Interface – AGI.

AGI é a interface de gateway do Asterisk, muito similar ao CGI (Common gateway Interface). Uma interface para adicionar funcionalidade ao Asterisk com muitas linguagens de programação diferentes. Perl, PHP, C, Pascal, Bourne Shell, Java é só escolher.

- AGI pode controlar o plano de discagem, chamado em `extensions.conf`
- EAGI dá à aplicação a possibilidade de acessar e controlar o canal de som além da interação com o plano de discagem.
- DEADAGI permite o acesso ao canal morto após o hangup.

1.8.1 Usando o AGI

O AGI funciona, fazendo com que o programa se comunique com o Asterisk através do standard input (Em um programa normal, seria o teclado, no AGI é o Asterisk que envia estes dados) e do standard output (Em um programa normal seria a tela do computador, no AGI o programa envia comandos como se estivesse escrevendo na tela). Desta forma qualquer linguagem pode ser usada.

Com o AGI é possível programar o Asterisk como uma URA consultando bancos de dados e retornando informações usando text-to-speech (texto para fala).

O que é FastAGI?

O FAST AGO permite que um aplicativo possa ser executado sobre uma conexão TCP/IP usando a porta #4573 deste modo descarregando o Asterisk desta tarefa. O servidor JAVA do outro lado usa um servidor JAGIServer para executar as aplicações.

Exemplo:

```
exten=>5551212,1,Agi(agi://192.168.0.2)
```

1.9 Sumário

O Asterisk é um software com licenciamento GPL, que transforma um PC comum em uma poderosa central telefônica. Foi criado por Mark Spencer da Digium que comercializa o hardware de telefonia. O Hardware de telefonia também é aberto e foi desenvolvido por Jim Dixon no projeto Zapata Telephony.

A arquitetura do Asterisk, se compõe basicamente de:

- **CANAIS** que podem ser analógicos, digitais ou Voip.
- **PROTOCOLOS** de comunicação como o SIP, H323, MGCP e IAX que são responsáveis pela sinalização de telefonia.
- **CODECS** que fazem a codificação da voz de um formato para outro, permitindo que seja transmitida com compressão de até oito vezes (G729a).

- **APLICAÇÕES** que são responsáveis pela funcionalidade do PABX.

O Asterisk pode ser usado em inúmeras aplicações, desde um PABX para uma pequena empresa até sistemas de resposta automática de alta densidade.

1.10 Questionário

1. Marque as opções corretas. O Asterisk tem quatro componentes básicos de arquitetura

- CANAIS
- PROTOCOLOS
- AGENTES
- TELEFONES
- CODECS
- APLICAÇÕES

2. Se for necessário criar um PABX com 4 troncos e oito telefones, você pode usar um PC com Linux e três placas TDM400P uma com quatro canais FXO e duas com quatro canais FXS cada. A afirmação acima está:

- CORRETA
- INCORRETA

3. Um canal FXS gera tom de discagem, enquanto um canal FXO recebe o tom vindo da rede pública ou de um outro PABX. A afirmação acima está:

- CORRETA
- INCORRETA

4. Marque as opções corretas, O Asterisk permite os seguintes recursos:

- Unidade de Resposta Automática
- Distribuição automática de chamadas
- Telefones IP
- Telefones Analógicos
- Telefones digitais de qualquer fabricante.

5. Para tocar música em espera o Asterisk necessita de um CD Player ligado em um ramal FXO. A afirmação está:

- CORRETA
- INCORRETA

6. É responsável pelo atendimento automático de clientes, normalmente toca um “prompt” e espera que usuário selecione uma opção. Am alguns casos pode ser usada em conjunto com um banco de dados e conversão texto para fala.

Estamos falando de uma:

- URA
- IVR
- DAC
- Unified Messaging

7 – Marque as opções corretas, Um banco de canais é conectado ao Asterisk através de uma interface:

- E1
- T1
- FXO
- FXS

8 – Marque a opção correta. Um canal E1 suporta ___ canais de telefonia enquanto um T1 suporta ___ canais.

- 12, 24
- 30, 24
- 12,12
- 1,1

9 – Nas plataformas de telefonia convencional, normalmente URA, DAC e Correio de voz estão incluídos no PABX. Esta afirmação está:

- CORRETA
- INCORRETA

10 – Marque as opções corretas, É possível interligar usando o Asterisk várias filiais através de voz sobre IP reduzindo a despesa com ligações de longa distância. Em uma filial:

- O Asterisk pode ser a central telefônica para todos os usuários.
- O Asterisk pode integrar uma central telefônica existente
- Podem ser usados apenas telefones IP ligados à um Asterisk centralizado
- Redundância e confiabilidade não são importantes quando se ligam IP fones.

página deixada intencionalmente em branco

Baixando e instalando o Asterisk

Neste capítulo vamos abordar a instalação e execução do Asterisk.

2.1 Objetivos do capítulo

Ao final deste capítulo você deverá estar apto à:

- Dimensionar o hardware necessário para o Asterisk.
- Instalar o Linux com as bibliotecas necessárias para o Asterisk.
- Descarregar o Asterisk do CVS da Digium.
- Compilar o Asterisk no Linux.
- Executar e explicar as opções de execução do Asterisk.

2.2 Introdução

Este capítulo vai ajudá-lo a preparar seu sistema para a instalação do Asterisk. O Asterisk funciona em muitas plataformas e sistemas operacionais, mas nós escolhemos manter as coisas simples e ficar em uma única plataforma e distribuição do Linux. Vamos usar o Suse Linux 9.2 neste livro. As instruções abaixo podem funcionar com outra distribuição do Linux, mas isto não foi testado. O Asterisk é conhecido por funcionar na maioria das distribuições. Testamos no Suse 9.2 e no CentOS que usamos para distribuir uma versão LiveCD.

2.3 Hardware Mínimo

O Asterisk pode ser intensivo em processador, pois ele usa o próprio processador da máquina para fazer o processamento dos sinais digitais. Se você estiver construindo um sistema complexo com carga elevada é importante entender este conceito. Para construir seu primeiro PABX um processador compatível com Intel que seja melhor que um Pentium 300Mhz com 256 MB RAM é o suficiente. O Asterisk não requer muito espaço em disco, cerca de 100 MB compilados, mais código fonte, voice-mail, prompts customizados e todos requerem espaço.

Se você usar apenas VOIP, nenhum outro hardware é necessário. Pode se usar softfones como os da XTEN (X-Lite) e entroncar com operadoras gratuitas como o Free World Dialup <http://www.freeworlddialup.com/>. Um sistema com apenas VOIP permite que você avalie o Asterisk sem custos. Entretanto se você quiser explorar todo o poder do Asterisk você vai acabar querendo instalar uma das placas da Digium.

Nota: Muitas pessoas rodando o Asterisk requerem uma fonte de clock para fornecer a temporização. As placas da Digium têm esta capacidade por padrão. Para sistemas sem uma fonte de temporização, existe o ztdummy, ele usa a controladora USB como fonte de temporização. Algumas aplicações como o Meetme (Conferência) precisam desta temporização. Existem dois tipos de controladores USB, UHCI e OHCI, é necessário um UHCI para que o sistema funcione. Os sistemas com OHCI também funcionam, mas vão precisar do módulo zaprtc.

Dica: Você pode usar uma placa de fax/modem com chipset Intel 537 ou MD3200, ela se comporta como uma Digium X100P. Nós conseguimos algumas destas placas por um preço bastante acessível, bem mais baixo que importar da Digium.

2.3.1 Montando o seu sistema

O hardware necessário para o Asterisk não é muito complicado. Você não precisa de uma placa de vídeo muito sofisticada ou periféricos como portas seriais, a porta paralela ou portas USB podem ser completamente desabilitadas. Uma boa placa de rede é essencial. Se você estiver usando uma das placas da Digium, é bom verificar as instruções da sua placa-mãe para determinar se os Slots PCI suportam estas placas. Muitas placas-mãe compartilham interrupções em slots PCI. Conflitos de interrupção são uma fonte potencial de problemas de qualidade de áudio no Asterisk. Uma maneira de liberar IRQs é desabilitar na BIOS tudo que não for necessário.

2.3.2 Questões de compartilhamento de IRQ

Muitas placas de telefonia como a X100P podem gerar grandes quantidades de interrupções, atendê-las toma tempo. Os drivers podem não conseguir fazê-lo em tempo se outro dispositivo estiver processando a mesma IRQ compartilhada e a linha de IRQ não puder receber outra interrupção. Tende a funcionar melhor em sistemas multiprocessados. Em sistemas monoprocessados você pode ter muitas perdas de interrupção e clock desalinhado. Quaisquer das placas da Digium e outras placas de telefonia podem estar sujeitas ao mesmo problema. Como a entrega precisa de IRQs é uma necessidade primária em telefonia, você não deve compartilhar IRQs com nada. Nem sempre isto ocorre, mas você deve prestar atenção ao problema. Se você está usando um computador dedicado para o Asterisk, desabilite o máximo de dispositivos que você não vá usar.

A maioria das BIOS permite que você manualmente designe as IRQs. Vá até a BIOS e olhe na seção de IRQs. É bem possível que você consiga configurar as interrupções manualmente por slot.

Uma vez iniciado o computador, veja em `/proc/interrupts` as IRQs designadas.

```
#
cat /proc/interrupts
CPU0
0: 41353058 XT-PIC timer
1: 1988 XT-PIC keyboard
2: 0 XT-PIC cascade
3: 413437739 XT-PIC wctdm <-- TDM400
4: 5721494 XT-PIC eth0
7: 413453581 XT-PIC wcfxo <-- X100P
8: 1 XT-PIC rtc
9: 413445182 XT-PIC wcfxo <-- X100P
12: 0 XT-PIC PS/2 Mouse
14: 179578 XT-PIC ide0
15: 3 XT-PIC ide1
NMI: 0
ERR: 0
```

Acima você pode ver as três placas da Digium cada uma na sua IRQ. Se este for o caso, você pode ir em frente e instalar os drivers de hardware. Se não

for o caso, volte na BIOS até que as placas não estejam compartilhando as IRQs.

2.4 Escolhendo uma distribuição do Linux.

O Asterisk foi originalmente desenvolvido para rodar em Linux, embora possa ser usado no BSD e OS X. No entanto, as placas PSTN da Digium foram desenhadas para trabalhar com Linux i386. Se você for novo com Asterisk procure usar o Linux.

2.4.1 Requisitos do Linux

Várias distribuições foram usadas com sucesso como RedHat, Mandrake, Fedora, Debian, Slackware e Gentoo foram usadas com sucesso com o Linux. Na Voffice nós usamos o Suse versão 9 e tem funcionado bem. Se você descobrir que algo não funciona com seu sistema em particular reporte o erro em <http://www.digium.com/bugtracker.html>.

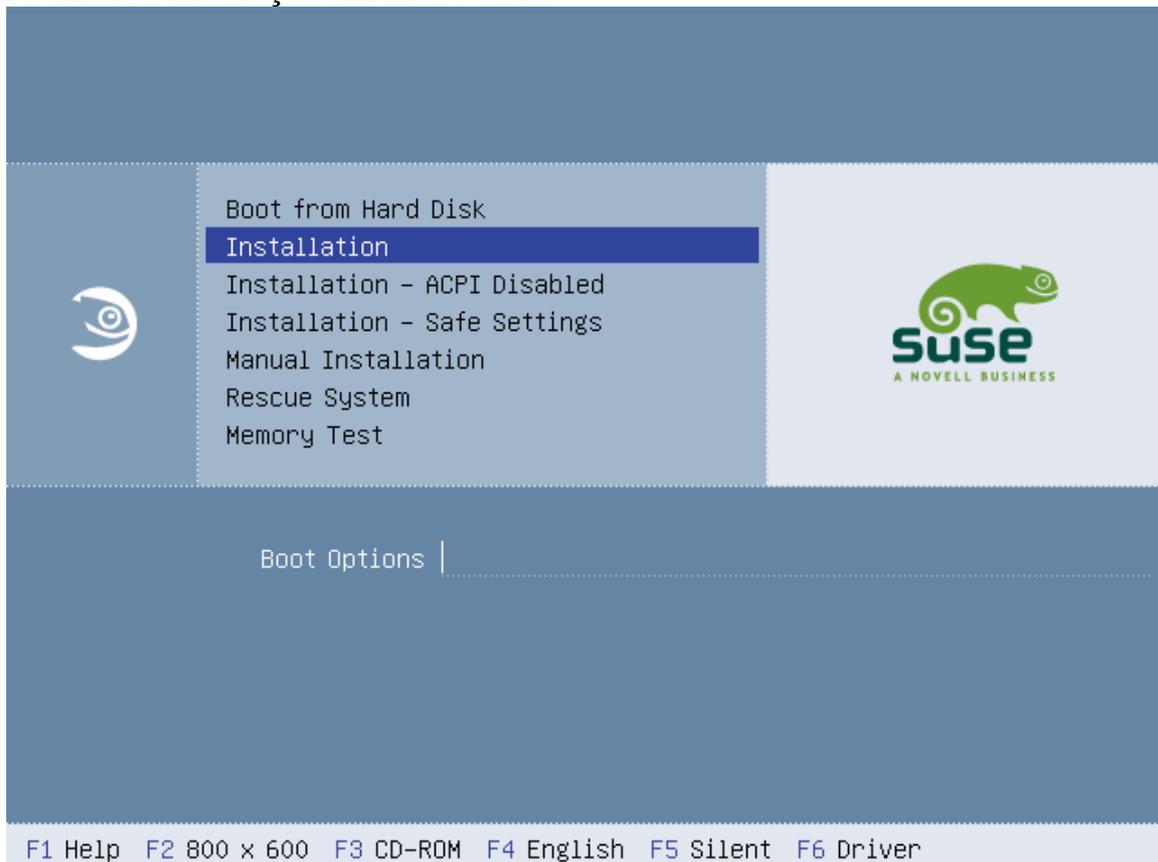
2.4.2 Pacotes necessários.

Antigamente existiam alguns pacotes que eram necessários para instalar o Asterisk como o readline e readline-devel que não são mais necessários. Não existe hardware especial tal como uma placa de som e o único pacote necessário é o próprio Asterisk. Se você estiver usando hardware da Digium ou ztdummy, você vai precisar do pacote zaptel. O pacote zaptel é necessário para que algumas aplicações sejam incluídas em tempo de compilação. Se você escolher compilar o Asterisk e não o zaptel, mas descobrir que esta faltando uma aplicação relacionada ao pacote zaptel (Como o Meetme()), você terá de compilar o zaptel e então recompilar o Asterisk para que a aplicação seja incluída. Para interfaces T1 e E1 o pacote *libpro* é necessário. *Bison* é necessário para compilar o Asterisk. Os pacotes de desenvolvimento ncurses e ncurses-development são necessários se você quiser construir novas ferramentas (Como o astman). As bibliotecas zlib e zlib-devel são necessárias agora para compilar. Isto se deve a adição do DUNDi (Distributed Universal Number Discovery) protocol.

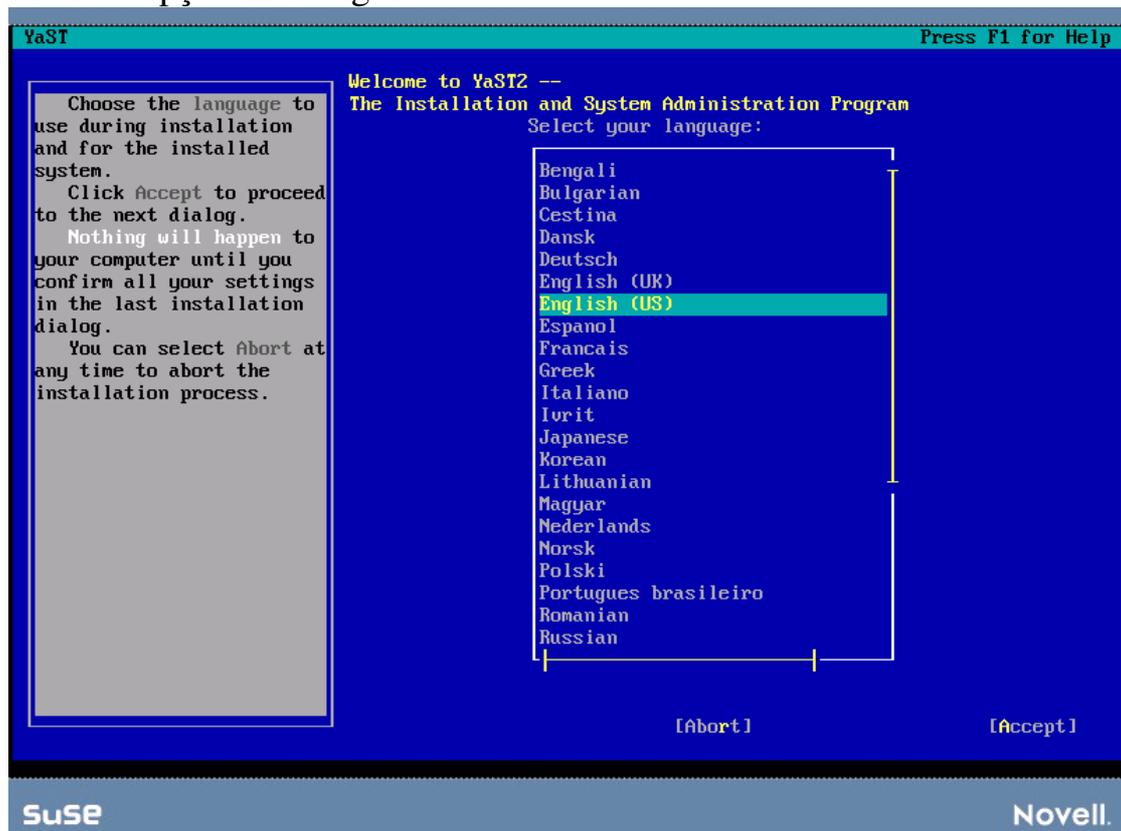
2.5 Instalando o Linux para atender ao Asterisk.

Para nossa instalação vamos usar o Suse Linux. É uma distribuição bastante usada e não devemos ter problemas durante as instalações.

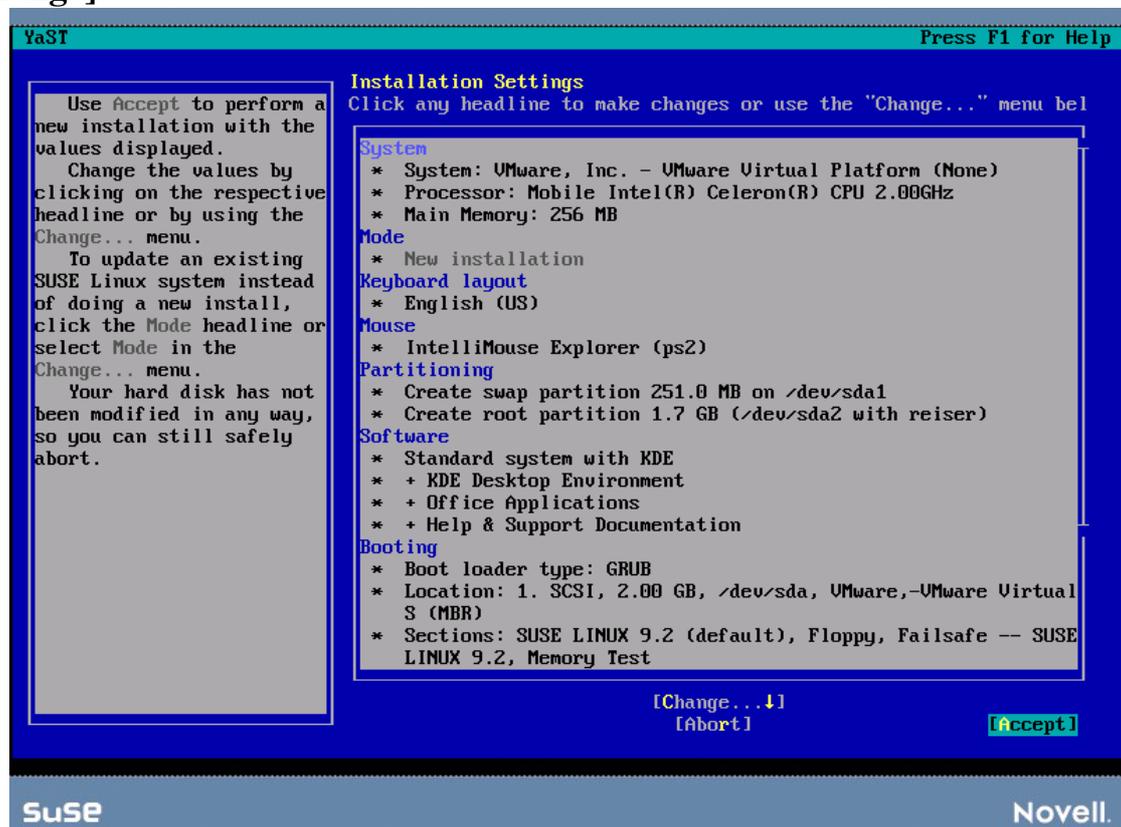
Passo 1: Coloque o CD e faça o boot usando o disco 1 do Suse 9.2. Pressione F2 e inicie a instalação em modo texto.

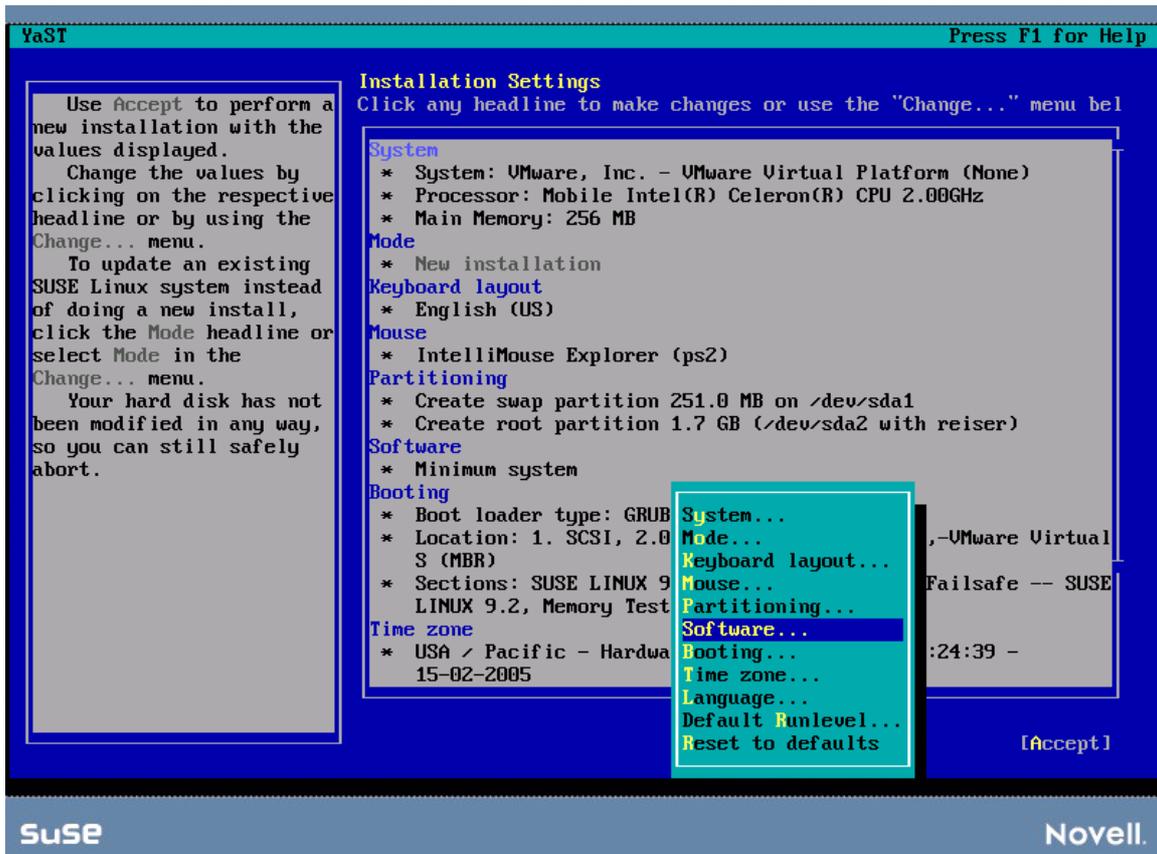
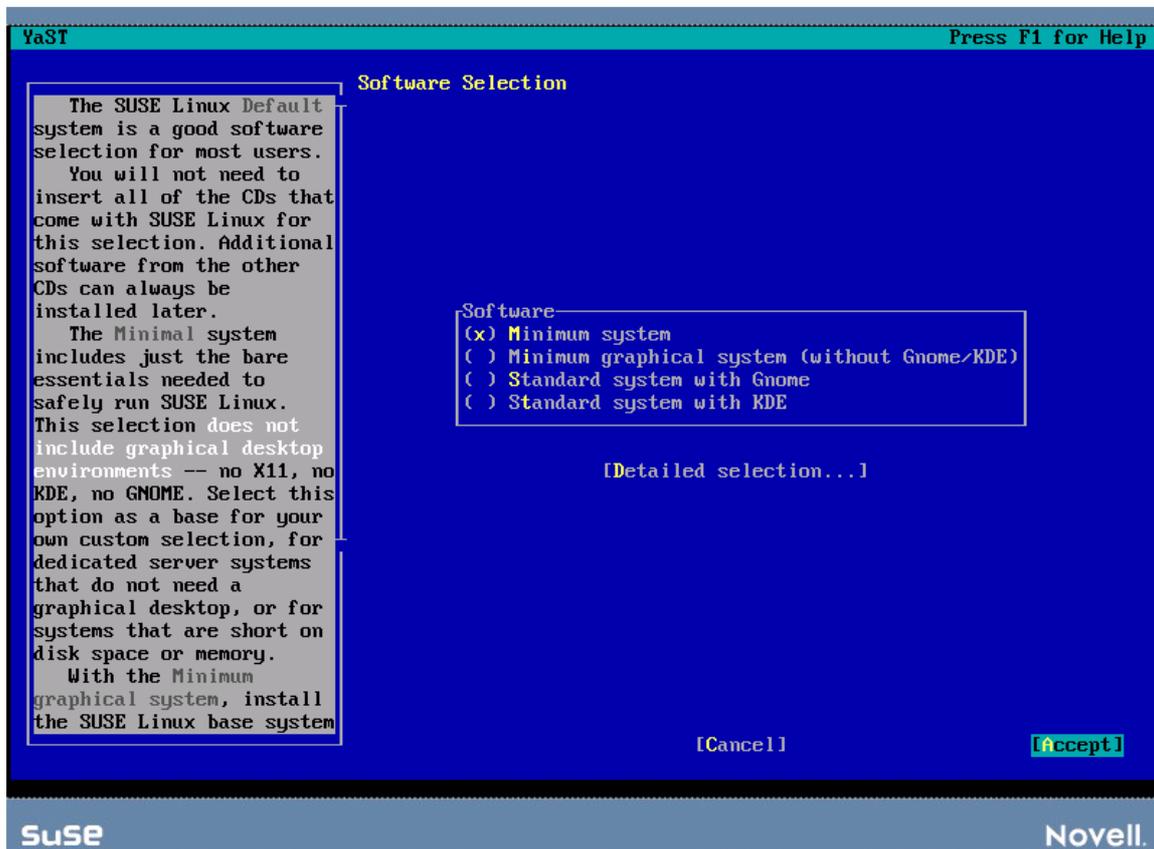


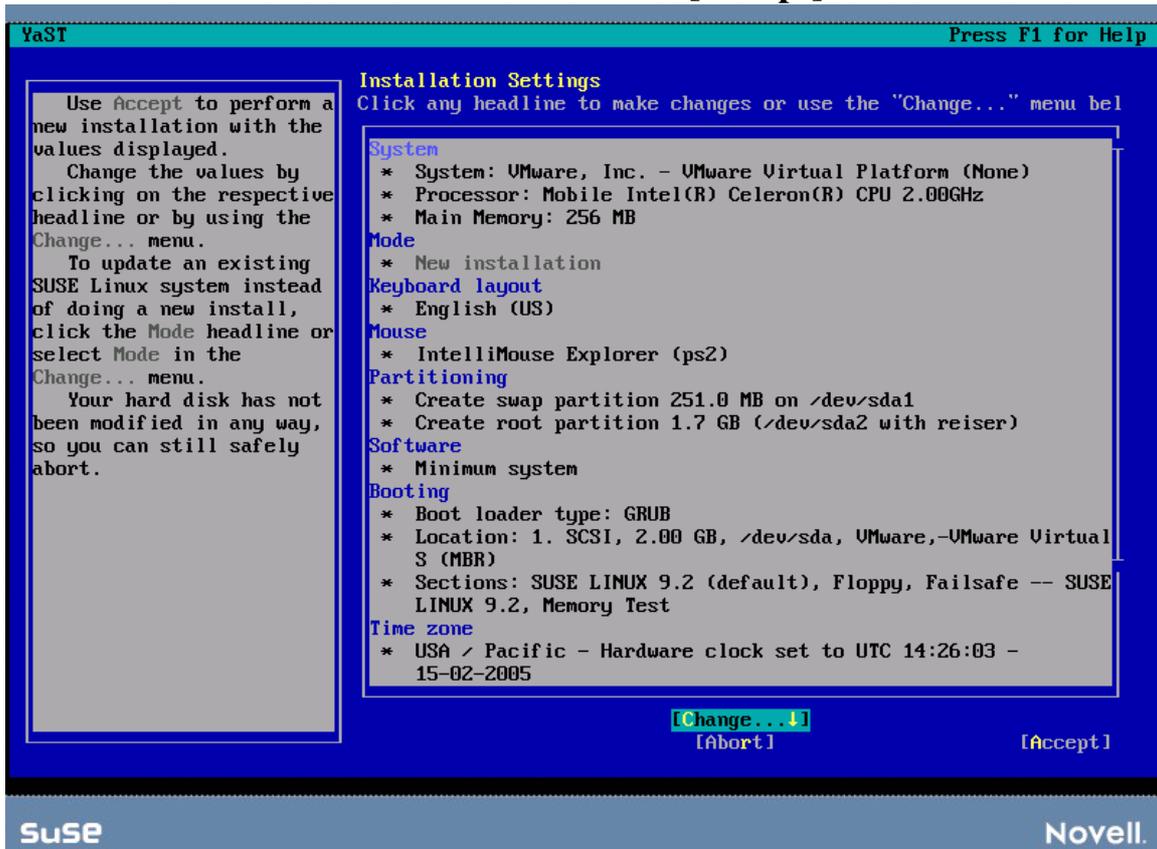
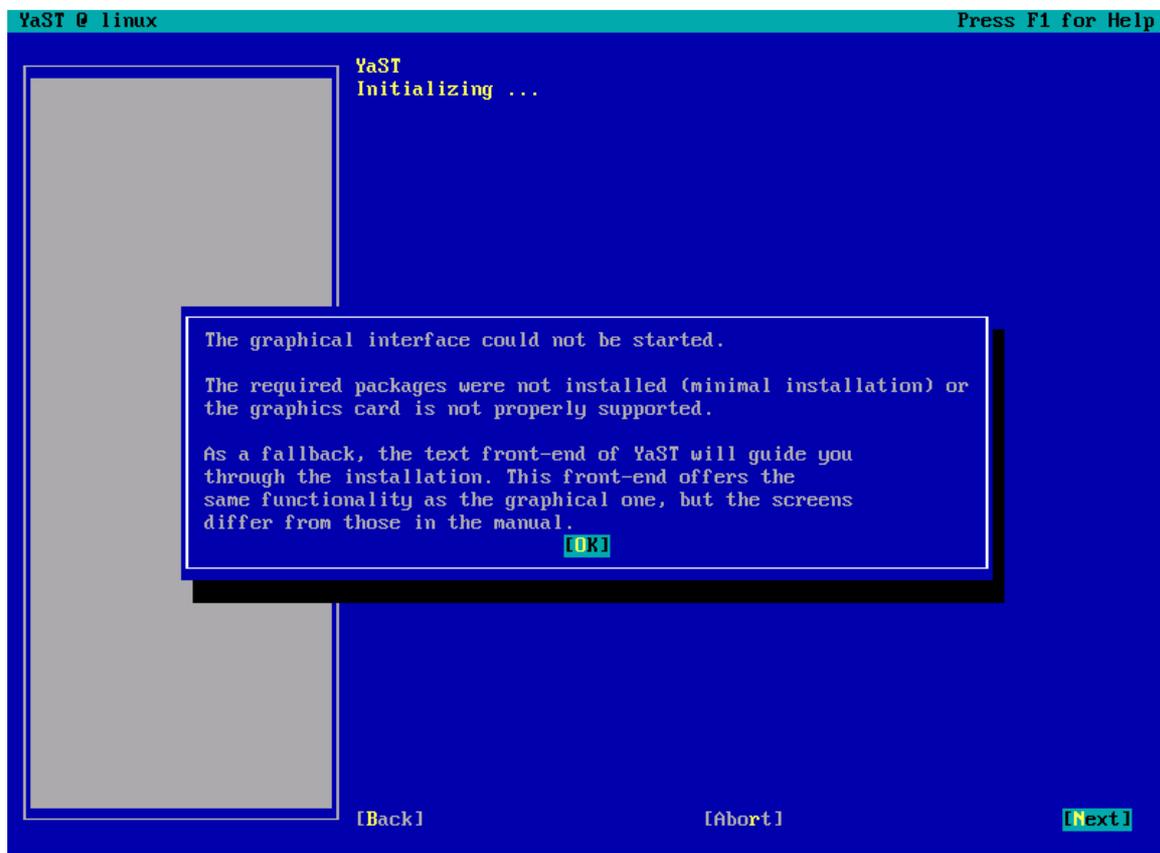
Passo 2: Escolha a linguagem, eu sempre escolho Inglês para servidores, considero a opção mais segura.



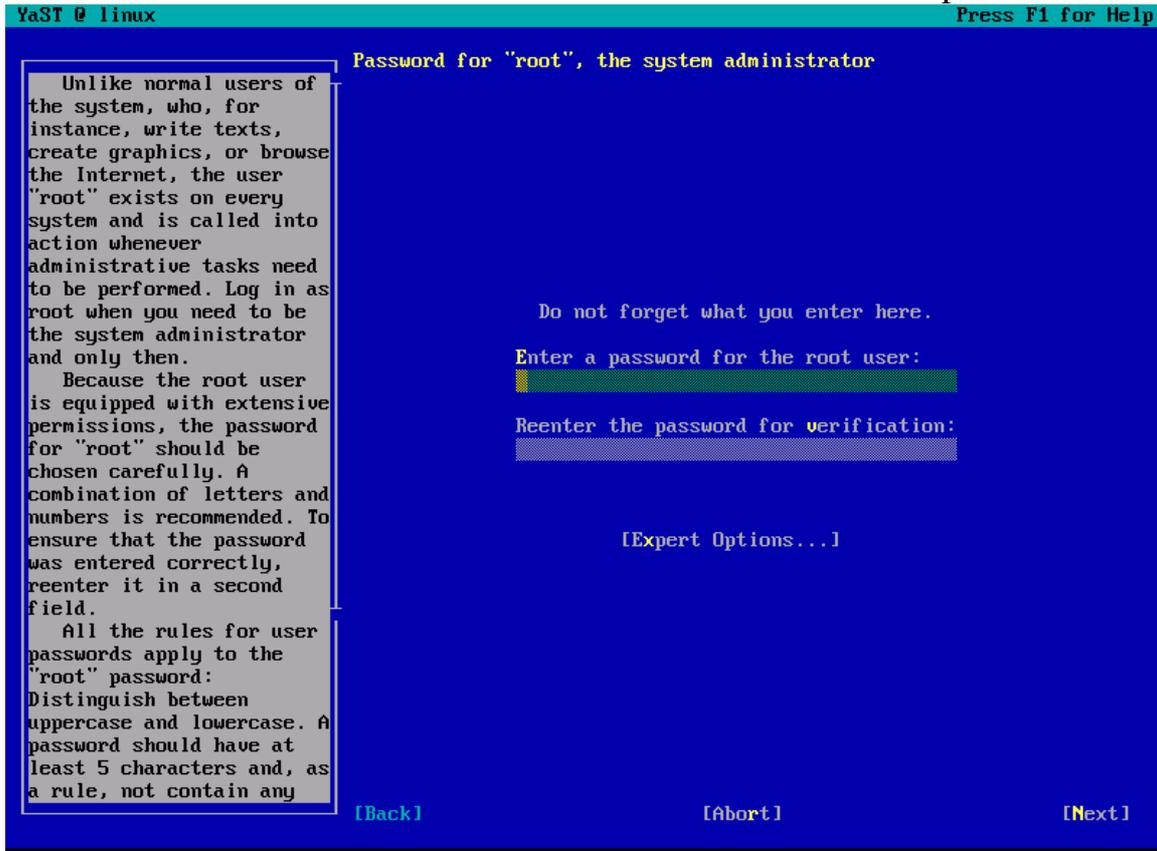
Passo 3: Você receberá a seguinte tela, usando [TAB] vá para a opção [change].



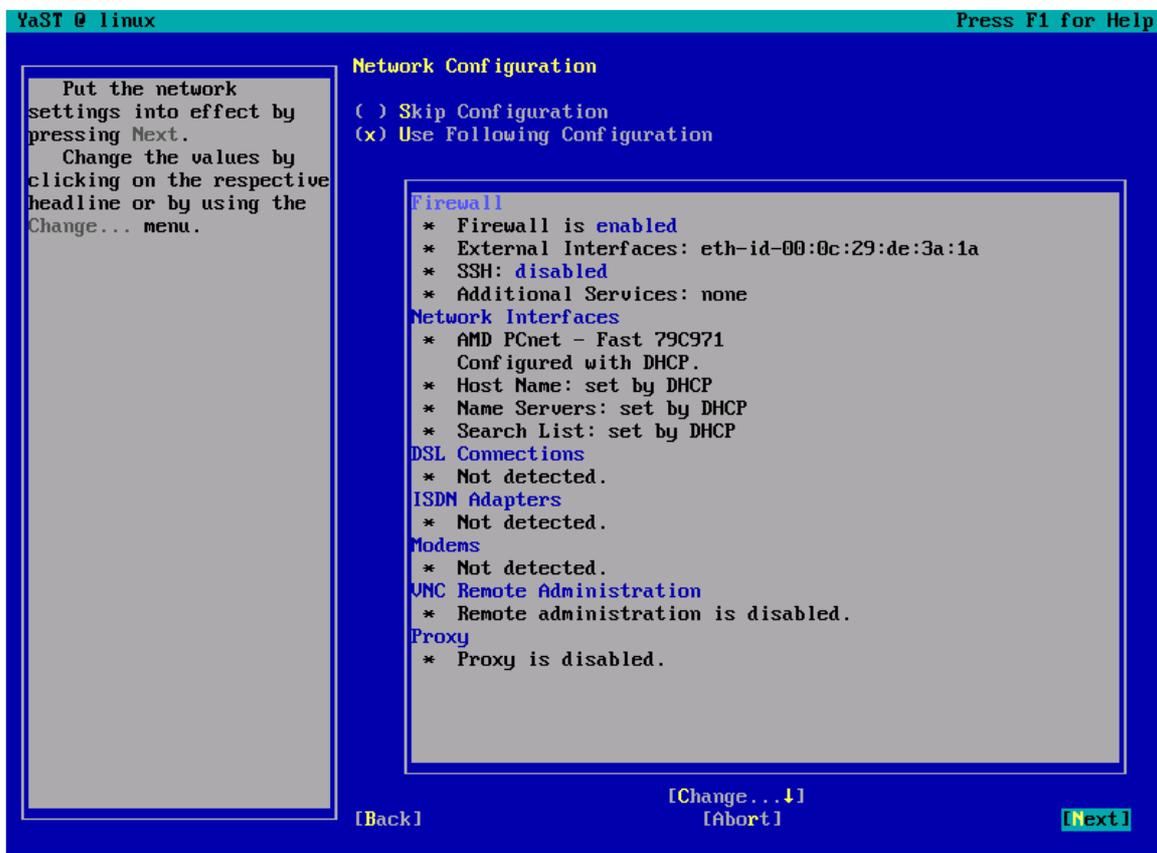
Passo 4: Escolha o item Software...**Passo 5: Escolha, sistema mínimo "Minimum System"**

Passo 6: Voltando a tela inicial escolha aceitar **[Accept]**.**Passo 7:** Confirme a mensagem que mostra que faremos uma instalação com a interface texto do Suse.

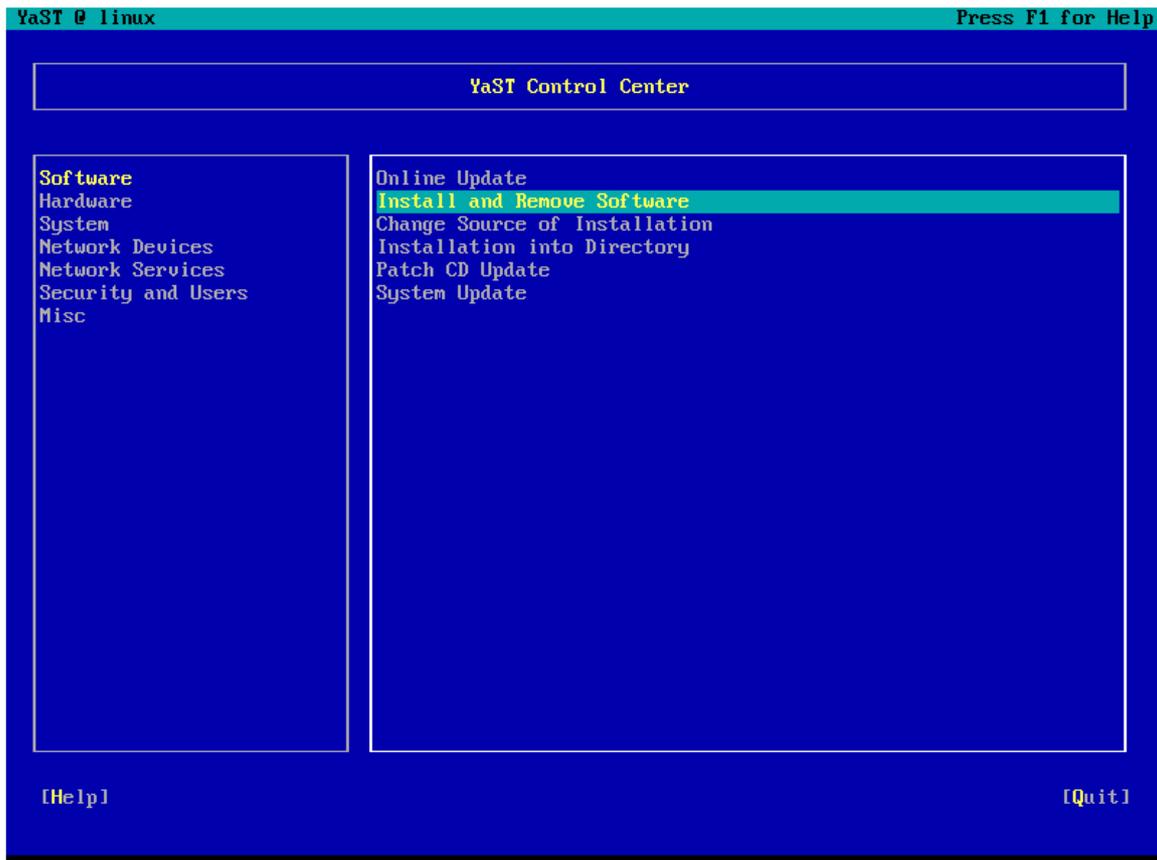
Passo 8: Entre com a senha do Root e entre com ela novamente para confirmar.



Passo 9: Use a configuração abaixo. Em sistemas em produção você vai precisar definir um endereço estático para o servidor Asterisk.



Passo 12: Selecionar pacotes adicionais. Para que você possa compilar o Asterisk é necessário que você selecione vários pacotes adicionais. Entre como root no sistema e carregue o YaST. Entre na opção Adicionar e Remover Software.



Passo 13: Entre na janela de instalação e instale os seguintes pacotes:

- Kernel Sources
- gcc - GNU C Compiler and Support Files
- cvs – Concurrent Versions System
- ncurses – New curses libraries
- ncurses-devel – Bibliotecas para desenvolvimento com ncurses
- bison – The GNU parser generator
- Termcap – Termcap library
- openssl – Secure Sockets and TLS Layer Security

- openssl-developer – Bibliotecas do openssl.
- zlib-devel

2.6 Obtendo e compilando o Asterisk

Agora que você já instalou o Linux e as bibliotecas necessárias, vamos partir para a instalação do Asterisk.

2.6.1 O que é CVS?

CVS é um repositório central que desenvolvedores usam para controlar o código fonte. Quando uma mudança é feita ela é enviada para o servidor CVS onde fica imediatamente disponível para download e compilação. Outro benefício de usar um CVS é que se algo estava funcionando até um ponto, mas uma mudança fez com que parasse de funcionar, a versão para qualquer tipo de arquivo em particular pode ser retornada a certo ponto. Isto é verdade para toda árvore também. Se você descobrir que algo estava funcionando até um ponto, mas a instalação da última versão do Asterisk fez com que o sistema parasse de funcionar, você pode voltar atrás para qualquer ponto no tempo. Veja a seção de como baixar os arquivos do CVS.

2.6.2 Drivers para as placas de telefonia

A digium assim como outros fabricantes fabrica placas de telefonia para serem usadas com o Asterisk. Vamos abordar neste caso a placa X100P, a TDM400P e a Wildcard E100P que serão provavelmente as placas mais usadas no Brasil.

2.6.2.1 Digium X100P

É uma das placas mais simples da Digium com uma porta FXO que pode ser ligada à rede pública ou à uma interface de ramal de um PABX.



2.6.2.2 TDM400P

A placa Wildcard TDM400P é uma placa analógica até quatro canais. Os canais podem ser FXO ou FXS dependendo dos módulos adquiridos.



2.6.2.3 TE110P

A placa E100P já uma placa para 30 canais digitais no padrão E1-ISDN. Com esta placa você pode se conectar de forma digital à sua central telefônica ou à rede pública.



Vamos mostrar nesta seção, como carregar os drivers de telefonia das placas analógicas e digitais da Digium conhecidas como zaptel (Zapata Telephony).

2.6.2.4 Obtendo os drivers

Para obter os drivers da Zaptel para uso com hardware digium, você tem de verificar a parte da zaptel no servidor CVS da Digium. Exemplo, baixando os drivers da Zaptel do CVS:

```
cd /usr/src/  
export CVSROOT=:pserver:anoncvs@cvs.digium.com:/usr/cvsroot  
cvs loginpassword is anoncvs  
cvs checkout -r v1-0 zaptel
```

Você será conectado ao servidor CVS onde ele vai descarregar todos os arquivos necessários para compilar os drivers da zaptel. Estes arquivos vão ser armazenados em /usr/src/zaptel.

2.6.2.5 Compilando os drivers

Você vai precisar compilar os módulos da Zaptel se você planeja usar o ztdummy ou qualquer hardware da Digium. Os seguintes comandos irão compilar e instalar módulos para quaisquer hardwares da Digium que você possa ter instalado no seu sistema.

Exemplo: Compilando os drivers da Zaptel.

```
cd /usr/src/zaptel/  
make clean  
make install
```

Se você usar qualquer distribuição que use o kernel 2.6, você precisa fazer um passo adicional antes de fazer o **make install**.

```
cd /usr/src/zaptel  
make clean  
make linux26  
make install
```

2.6.3 Compilando o ztdummy

O ztdummy é usado quando você não tem quaisquer hardwares da Digium para os recursos de temporização, mas precisa deles para usar os aplicativos “Música em Espera **MusicOnHold()**” e “Conferência **MeetMe()**”. O Driver ztdummy requer que você tenha uma controladora USB UHCI. Se você estiver usando uma controladora USB OHCI, você terá de usar o zaprtc. Você pode verificar se a sua placa-mãe tem uma controladora UHCI USB rodando o **lsmmod** da linha de comando.

```
# lsmod
Module Size Used by Not tainted
...
uhci-hcd 29725 0 [unused]
<-- usb-uhci
usbcore 105342 3uhci-hcd
```

A tela acima mostra os módulos USB carregados. Se você vir uma linha que lê usb-uhci. Isto mostra que o módulo UHCI está carregado e pronto para ser usado com o ztdummy.

2.6.3.1 Editando o Makefile

Para compilar o ztdummy você tem de editar o arquivo makefile localizado no seu diretório /usr/src/zaptel. Encontre a linha contendo:

```
MODULES=zaptel tor2 torisa wusb wcfxo wcfxs \ ztdynamic ztd-eth wct1xxp wct4xxp #
ztdummy
```

Retire o comentário do modulo ztdummy removendo o susenido (#). Salve o arquivo e faça a compilação normalmente. Uma vez que você tenha compilado com sucesso o ztdummy, você pode carregá-lo na memória usando o comando modprobe.

2.7 Instalando e configurando o hardware

A configuração das placas da Digium é feita no arquivo /etc/zaptel.conf. Os procedimentos para a carga destas placas estão descritos mais abaixo. As placas da Digium além de ter o papel de conexão a um PABX ou à rede pública também são usadas como fonte para a sincronização de tempo.

2.7.1 Passos necessários para instalação do hardware.

1. Instalar o hardware no PC.
2. Carregar os drivers de Kernel.
3. Configurar o arquivo zaptel.conf.
4. Executar o comando ztcfg para configurar o driver.

2.7.2 Instalar o Hardware no PC

Instale as placas no seu PC. Certifique-se que as placas de telefonia possuem um IRQ dedicado a interface. Desabilite todo o hardware desnecessário.

2.7.3 Carregar os drivers de kernel

Você deve carregar o modulo `zaptel` e um módulo correspondente a placa que você está instalando:

```
modprobe zaptel
```

Tabela dos drivers da Digium

Placa	Driver	Descrição
TE410P	wct4xxp	4xE1/T1-3.3V PCI
TE405P	wct4xxp	4xE1/T1-5V PCI
TDM400P	wcfxs	4 FXS/FXO
T100P	wct1xxp	1 T1
E100P	wct1xxp	1 E1
X100P	wcfxo	1 FXO

Modprobe é usado para carregar os drivers da `zaptel` na memória de forma que se possa ter acesso ao hardware do sistema. Nós sempre carregamos o driver `zaptel` na memória primeiro. Após ele carregamos os drivers específicos para o tipo de dispositivo que estamos carregando (FXS, FXO, `ztdummy`, etc.) Podemos carregar o módulo da `Zaptel` com o seguinte comando, por exemplo:

```
modprobe wcfxs
```

Se o modulo `zaptel` carregou com sucesso, você não deve ver qualquer saída do comando após teclar enter. Você pode verificar se foi carregado com sucesso rodando o comando `lsmod`.

```
#lsmod
Module Size Used by Not tainted
zaptel 175132 0 (unused)
```

Como podemos ver, o primeiro módulo listado é o `zaptel`. O módulo `zaptel` é usado para nossos módulos de canal, por isto esta como “unused” (não usado). Isto irá mudar uma vez que carreguemos as portas FXS ou FXO.

Caso não exista nenhuma placa de telefonia. Coloque no ar o `ztdummy` para prover o sincronismo. Sem isto, aplicativos como Conferência e Música em espera não vão funcionar.

```
modprobe ztdummy
```

Novamente, não se deve ver nenhuma saída do comando. A verificação pode ser feita usando o *lsmod*.

2.7.4 Configurando o arquivo zaptel.conf

De forma a configurar os parâmetros regionais e de sinalização para os canais de telefonia físicos o arquivo *zaptel.conf* precisa ser editado. Este arquivo contém muitas opções e parâmetros que não vão estar incluídos neste material.

Abra o arquivo *zaptel.conf* com seu editor favorito. Você vai notar um grande número de linhas comentadas. Você pode ler esta informação para seu conhecimento, mas por enquanto ignore a maior parte dela. As únicas duas linhas que não estão comentadas por default vêm próximas ao fim do arquivo. Os parâmetros *loadzone* e *defaultzone*. Por enquanto deixe as como default.

Nota: O parâmetro *loadzone* permite a você especificar os tons que podem ser gerados por aquela interface. Uma lista separada por vírgula lista os códigos de país com duas letras para aquele país ou região. A lista atualmente disponível de tons está definida em *zonedata.c* no diretório */usr/src/zaptel*.

Dentro do arquivo *zaptel.conf*, nós definimos o tipo de sinalização que o canal vai usar. Nós também definimos quais canais vão ser carregados. Esta é a informação que será usada para configurar os canais com o uso do comando *ztcfg* que discutiremos mais tarde neste capítulo. Quando você lidar com uma interface FX o hardware é descrito baseado no que ele conecta (FXO se conecta à rede pública (Office) enquanto FXS se conecta à um telefone (Station)).

Vamos assumir que estamos configurando o *zaptel.conf* para uso com um TDM400P que contém uma interface FXO e outra FXS.

```
#  
# Zaptel Configuration File  
#  
fxoks=1  
fxsks=4  
loadzone=us  
defaultzone=us
```

A TDM11B (Uma FXO e uma FXS na placa TDM400P) vem padrão com o módulo FXS conectado na primeira porta. O módulo FXO é conectado na quarta porta da placa. A linha *fxoks=1* então diz ao módulo *wcfxs* para usar

sinalização FXO na primeira porta da TDM400P. Da mesma forma `fxsks=4` especifica que a quarta porta vai usar sinalização FXS.

Após carregar o driver, você deve configurar os canais usando `ztcfg`.

O comando `ztcfg` é usado para configurar a sinalização usada para a interface física FX. `ztcfg` irá usar a configuração de sinalização em `zaptel.conf`. Para ver a saída do comando você deve usar `-vv` para colocar o programa em modo verbose.

```
#
/sbin/ztcfg -vv
Zaptel Configuration
=====
Channel map:
Channel 01: FXO Kewlstart (Default) (Slaves: 01)
Channel 04: FXS Kewlstart (Default) (Slaves: 04)
2 channels configured.
```

Se os canais carregarem com sucesso, você vai ver uma saída similar à acima. Um erro comum é inverter a sinalização entre os canais. Se isto acontecer você vai ver algo parecido como:

```
ZT_CHANCONFIG failed on channel 1: Invalid argument (22)
Did you forget that FXS interfaces are configured with FXO signalling
and that FXO interfaces use FXS signalling?
```

Após os canais terem sido configurados com sucesso, você está pronto para iniciar o uso do seu hardware com o Asterisk.

2.7.5 Resumindo, como carregar uma placa de telefonia.

```
modprobe zaptel
modprobe wcfxo ; ou a placa que você adquiriu
ztcfg
asterisk -vvvvvvvvvvvvvvvcg
```

2.8 Obtendo e compilando o Asterisk

De forma a obter o Asterisk, você deve retirá-lo do servidor CVS da Digium. Quando do fechamento deste livro estava na versão 1.0.5.

Obtendo a versão principal

```
cd /usr/src/
```

```
export CVSROOT=:pserver:anoncvs@cvs.digium.com:/usr/cvsroot
cvs login
password is anoncvs
cvs checkout -r v1-0 asterisk libpri asterisk-sounds asterisk-addons
```

Compilando

Se você já compilou o software antes, compilar o Asterisk vai parecer bem simples. Rode os seguintes comandos para compilar e instalar o Asterisk após você ter baixado ele do servidor CVS.

```
cd /usr/src/asterisk/
make clean
make
make install
make samples
```

2.9 Iniciando e parando o Asterisk

Antes de usar o Asterisk, você deve criar os arquivos de configuração. Muito embora a quantidade de configurações possíveis seja muito grande, apenas um pequeno conjunto é necessário de forma a iniciar o Asterisk com sucesso.

Com esta configuração mínima, já é possível iniciar o Asterisk com sucesso.

```
/usr/sbin/asterisk -vvvgc
```

Use o comando **stop now** para derrubar o Asterisk. Veja os comandos disponíveis na interface de linha de comando do Asterisk.

```
CLI>stop now
```

2.9.1 Parâmetros de linha de comando do Asterisk.

O processo de executar o Asterisk é bem simples. Se o Asterisk for rodado sem argumentos, ele é lançado como um daemon (Processo que espera conexões em uma porta TCP ou UDP).

```
/sbin/asterisk
```

Você pode acessar a console de um processo do Asterisk que já esteja em execução. Mais de uma console pode ser conectada ao Asterisk simultaneamente. Use:

```
/sbin/asterisk -r
```

2.9.2 Abaixo os parâmetros disponíveis

- **-h:** Help mostra as opções de parâmetros de linha de comando.
- **-C <configfile>:** Inicia o Asterisk com arquivo de configuração diferente do padrão `/etc/asterisk/asterisk.conf`
- **-f:** Foreground. Inicia o Asterisk, mas não coloca um processo em Background.
- **-c:** Habilita o modo de console. Inicia o Asterisk em Foreground (na frente, implica na opção `-f`), com uma console com interface de linha de comando.
- **-r:** Console remota.
- **-n:** Desabilita a cor na console.
- **-i:** Pede pelos códigos criptográficos de inicialização.
- **-p:** Roda como pseudo-realtime. Roda com prioridade de tempo real.
- **-q:** Modo silencioso suprime as mensagens.
- **-v:** Inclui mensagens detalhadas, (múltiplos `v`'s = mais verbose).
- **-d:** Habilita debug extra em todos os módulos
- **-g:** Faz com que o Asterisk descarregue o núcleo em caso de *segment violation*.
- **-x<cmd>:** Executa o comando `<cmd>` (válido apenas com `r`)

2.10 Iniciando o Asterisk em tempo de inicialização.

Sistemas operacionais diferentes têm métodos levemente diferentes de iniciar os programas em tempo de inicialização. O diretório `/usr/src/asterisk/contrib/init.d` contém scripts para alguns sistemas operacionais. Escolha um ou crie um que atenda sua aplicação.

Você pode escolher lançar `/sbin/asterisk` diretamente, ou fazer uso do shell script instalado em `/sbin/safe_asterisk` que executa o Asterisk e tenta re-executar no caso do Asterisk “dar pau”.

Vamos usar o script de inicialização do Asterisk criado por Martin Mielke para o SuSe. O script pode ser baixado em:

(http://www.leals.com/~mm/asterisk/asterisk_suse.sh)

Como instalar:

1. Entre como root
2. Salve o script como `/etc/init.d/asterisk`
3. Faça um link simbólico de `/etc/init.d/rc3.d` para o script para que o Asterisk inicie no boot do sistema.

```
cd /etc/init.d/rc3.d
ln -s ../asterisk S90asterisk
```

4. Faça um link simbólico de `/etc/init.d/rc0.d` para o script de forma que ele seja descarregado no *shutdown* do sistema.

```
cd /etc/init.d/rc0.d
ln -s ../asterisk K10asterisk
```

Como usar:

Uma vez instalado, se você desejar iniciar ou para o Asterisk manualmente, Entre como root e digite:

```
/etc/init.d/asterisk opção
```

Onde a *opção* pode ser:

start: Inicia o Asterisk

stop: Para o Asterisk

status: Verifica o status do Asterisk

restart: Reinicializa o Asterisk

2.11 Considerações sobre a instalação do Asterisk

2.11.1 Sistemas em produção

Se o Asterisk for instalado em um ambiente de produção, deve-se prestar atenção no projeto do sistema. O servidor deve ser otimizado de forma que as funções de telefonia tenham prioridade sobre os outros processos do sistema. Na maioria dos casos o Asterisk não deve rodar outros processos, principalmente se forem intensivos em CPU. Se forem necessários processos que utilizam muita CPU como bancos de dados, por exemplo, estes devem ser instalados eventualmente em um servidor separado.

De uma forma geral o Asterisk é um sistema sensível a variações de performance da máquina. Isto significa que em um sistema em produção o ideal é não usar interfaces gráficas como o KDE ou GNOME.

2.11.2 Considerações sobre a rede

Se você vai usar telefones IP, o que é muito provável é importante que você preste atenção a algumas questões sobre a rede. Os protocolos de voz sobre IP são muito bons e resistentes a perdas de pacotes, atrasos e variações de atrasos. Entretanto se você abusar, a qualidade de voz não será boa. Só é possível garantir a qualidade da voz utilizando QoS fim-a-fim, o que é inviável principalmente em telefonia sobre a Internet. Desta forma seguem algumas recomendações.

Implemente QoS fim-a-fim sempre que possível. Mesmo em switches de 100Mbps onde é raro ter um congestionamento, vale a pena, um vírus ou uma condição de rede inesperada pode por tudo a perder.

Seja conservador, use, por exemplo, uma conexão de Internet exclusiva para softfones e telefones IP. Na maioria das vezes os backbones têm folga no tráfego, mas a conexão de acesso é congestionada pelo próprio usuário com downloads, navegação, e-mail entre outros.

Evite hubs de 10 e 100 Mbps, as colisões nestes equipamentos, causam variação no atraso (*jitter*). Jitter é um dos piores inimigos da telefonia IP.

Oriente os usuários de telefonia sobre a Internet que não é possível garantir a qualidade. Manter as expectativas em um nível realista evita problemas futuros e comentários como “Se eu soubesse que era assim...”.

Quando usar uma rede IP privada com equipamentos que suportam QoS fim-a-fim, se a qualidade da voz estiver ruim, verifique imediatamente, existe algum problema na sua rede. Com QoS bem implementado a qualidade de voz é perfeita, “sem desculpas”.

2.12 Sumário

Neste capítulo você aprendeu que o requisito mínimo de hardware do Asterisk é um Pentium 300 Mhz com 256 MB RAM e pouco mais de 100 MB de espaço livre em disco para instalar o Asterisk. Aprendeu a instalar o Linux e baixar e compilar o Asterisk. Além disso, vimos também como instalar e configurar as placas da Digium analógicas e digitais.

2.11 Questionário

1. Qual a configuração mínima para o Asterisk.

2. As placas de telefonia para o Asterisk têm um processador próprio (DSP), não precisando assim de muita CPU do servidor.

- Correto
- Incorreto

3. Para que a telefonia IP funcione com perfeição é necessário que à rede possua QoS fim-a-fim.

- Correto
- Incorreto

4. É possível obter uma boa qualidade de voz em uma rede que não esteja congestionada com switches de 100 Mbps.

- Correto
- Incorreto

5. Liste abaixo as bibliotecas necessárias para compilar o Asterisk.

6. Se você não tem uma placa zaptel, você precisa de uma fonte de tempo. O driver ztdummy faz este papel aproveitando uma biblioteca USB. Isto é necessário, pois algumas aplicações como o _____ e o _____ precisam de uma referência de tempo.

7. O CVS é o sistema de controle de versões do Asterisk. Desta forma você só pode baixar a última versão.

- Correto
- Incorreto

8. Quando você faz uma instalação do Asterisk, o melhor é não instalar os pacotes gráficos como o KDE e GNOME, pois o Asterisk é sensível na questão de CPU e interfaces gráfica roubam muitos ciclos de CPU do servidor.

Correto

Incorreto

9. Os arquivos de configuração do Asterisk ficam em _____.

10. Para instalar os arquivos de configuração de exemplo você precisa executar o seguinte comando.

Configuração do Asterisk

Neste capítulo você vai aprender a configurar o Asterisk dentro de uma configuração de um PABX simples.

3.1 Objetivos do capítulo

Ao final deste capítulo você poderá:

- Entender e saber editar os arquivos de configuração do Asterisk.
- Instalar um hardware Digium no seu Linux.
- Entender a configuração básica do hardware Zaptel e de canais Zapata.
- Entender e configurar um plano de discagem simples.
- Configurar um PABX simples.

3.2 Introdução

Neste capítulo vamos aprender alguns conceitos que vão nos permitir configurar o Asterisk da forma mais básica. No final temos um exercício que poderá ser feito usando hardware de uma porta FXO ou usando uma conexão para o FreeWorldDialup ou IAXTEL. Nos capítulos seguintes vamos aumentar gradualmente a dificuldade. É muito importante que se entendam os conceitos apresentados neste capítulo para prosseguir aos capítulos seguintes.

3.3 Arquivos de configuração do Asterisk

O Asterisk é controlado através de arquivos de configuração localizados no diretório `/etc/asterisk`. O formato dos arquivos de configuração do Asterisk é semelhante aos arquivos “.ini” do Windows. O arquivo está em ASCII dividido em seções com o nome da seção em chaves ([]’s). Em seguida vêm os pares de Chave, Valor separado por um sinal de igual (=) ou por um sinal de igual seguido pelo sinal de maior que (=>). O ponto e vírgula é o caractere de comentário. O (=) e o (=>) podem ser usados de forma idêntica, linhas em branco são ignoradas. Arquivo de exemplo:

```

;
; A primeira linha sem ser comentário deve ser um título de sessão.
;
[sessao1]
chave = valor ; Designação de variável
[sessao2]
objeto => valor ; Declaração de objeto

```

O interpretador do Asterisk interpreta (=) e (=>) de forma idêntica. A sintaxe é apenas para tornar o código mais legível. Embora os arquivos compartilhem a mesma sintaxe, existem pelo menos três tipos distintos de gramática.

3.3.1 Grupo simples

O formato de grupo simples é o mais básico e usado por arquivos de configuração onde os objetos são declarados com todas as opções na mesma linha. Os arquivos `extensions.conf`, `meetme.conf` e `voicemail.conf` seguem este formato.

```

[sessao]
objeto1 => op1,op2,op3
objeto2 => op1b,op2b,op3b

```

Neste exemplo, o objeto1 é criado com opções op1, op2 e op3 enquanto o objeto 2 é criado com op1b, op2b e op3b.

Entidades individuais

A sintaxe de entidades individuais é usada por arquivos de configuração no qual objetos são declarados com muitas opções e onde estas opções raramente são compartilhadas com outros objetos. Neste formato uma seção é associada com cada objeto. Existe normalmente uma seção `[general]` para as configurações globais.

Exemplo:

```

[general]
globalop1=valorglobal1
globalop2=valorglobal2
[objeto1]
op1=valor1
op2=valor2
[objeto2]
op1=valor3
op2=valor4

```

Neste exemplo, a seção geral define duas variáveis globais. Em seguida dois objetos são criados [objeto1] e [objeto2].

3.3.2 Formato de objeto com herança de opções

Este formato é usado pelo phone.conf, mgcp.conf e zapata.conf e outras interfaces onde há muitas opções. Entretanto, a maioria das interfaces e objetos compartilha o mesmo valor para opções com outros. Nesta classe de arquivo de configuração, tipicamente existem uma ou mais seções que contém declarações de um ou mais canais ou objetos. As opções para o objeto são especificados acima da declaração do objeto e podem ser mudadas para a declaração de outro objeto. É um conceito difícil de entender, mas muito fácil de usar. Considere o exemplo abaixo:

```
[sessao]
op1 = bas
op2 = adv
objeto=>1
op1 = int
objeto => 2
```

As primeiras duas configuram o valor da opção op1 e op2 para “bas” e “adv” respectivamente. Quando o objeto 1 é instanciado, ele é criado com sua opção 1 sendo “bas” e sua opção 2 sendo “adv”. Após declara o objeto 1, mudamos o valor da opção 1 para “int”. E então criamos o objeto 2, agora o objeto 2 é criado com sua opção 1 sendo “int” e sua opção 2 permanecendo “adv”.

3.3.3 Objeto entidade complexa

O formato objeto entidade complexa é usado pelo iax.conf e sip.conf e outras interfaces nas quais existem numerosas entidades com muitas opções e que tipicamente não compartilham um grande volume de configurações comuns. Cada entidade recebe seu próprio contexto (As vezes existe um contexto reservado tal como [general] para as configurações globais. As opções então são especificadas na declaração de contexto. Considere:

```
[entidade1]
op1=valor1
op2=valor2
[entidade2]
op1=valor3
op2=valor4
```

A entidade [entidade1] tem valores valor1 e valor2 para opções op1 e op2 respectivamente. A entidade [entidade2] tem valores valor3 e calor4 para as opções op1 e op2.

3.4 Drivers de canal Asterisk

Nesta seção vamos mostrar como definir vários drivers de canal.

3.4.1 Zapata.conf

O arquivo de configuração zapata.conf contém parâmetros relacionados à canais TDM fornecidas pela interface zaptel. Os canais devem ser definidos neste arquivo antes que possam ser usados com o Asterisk. Além disso, um número de recursos relacionados à operação dos canais pelo Asterisk, pode ser definido aqui.

O arquivo zapata.conf consiste da várias instruções no formato de herança de opções, as opções se mantém até que sejam sobrescritas.

As seguintes palavras-chave estão disponíveis no zapata.conf

context: Define o contexto para aquele canal: Este é o contexto para entrada de chamadas pelo canal. Exemplo:

```
context=default
```

channel: Define o canal ou faixa de canais. Cada definição de canal irá herdar todas as opções colocadas a frente no arquivo. Canais podem ser especificados individualmente, separado por vírgulas ou como uma faixa separada por um hífen.

```
Channel=>1-15
Channel=>16
Channel=>17,18
```

group: Permite que um número de canais seja tratado como um para o propósito de discagem. Se você discar usando um grupo, o primeiro canal disponível será usado. Se forem telefones, ao ligar para o grupo todos receberão a campainha ao mesmo tempo. Com vírgulas você pode especificar que um canal pertence a mais de um grupo.

```
group=1  
group=3,5
```

switchtype: Configura o tipo de sinalização usado para a linha PRI. Os valores aceitáveis são:

```
5ess: Lucent 5ESS  
euroisdn: EuroISDN  
national: National ISDN  
dms100: Nortel DMS100  
4ess: AT&T 4ESS
```

```
switchtype = EuroISDN
```

Dica: Todas as implantações que fiz no Brasil, principalmente usando roteadores Cisco caíram em EuroISDN e Qsig, no momento ainda não há suporte à QSIG no Asterisk. As conexões à rede pública pelo menos em Santa Catarina também são EuroISDN.

pri_dialplan: Configura uma opção necessária para alguns *switches* (centrais e operadoras) que requerem que um plano de discagem seja passado. Esta opção é ignorada pela maioria dos equipamentos. Opções válidas são *private*, *national* e *international*

```
pri_dialplan = national
```

signaling: Configura o tipo de sinalização para os seguintes tipos de definição de canal. Estes parâmetros devem coincidir com os definidos no arquivo `/etc/zaptel.conf`. As escolhas corretas são baseadas no hardware disponível. O Asterisk não inicia se a definição estiver incorreta ou impossível de trabalhar. Se as instruções não batem com o arquivo `zaptel.conf`, ou se o dispositivo não está presente ou configura de forma apropriada. Vou separar em dois grupos, a sinalização RBS e Digital.

RBS – Simula o sinal analógico na linha digital. Ideal para uso com bancos de canal. Algumas centrais suportam. Pessoalmente nunca tive muito sucesso

com estas sinalizações na integração de roteadores, mas são fundamentais no uso de bancos de canal (channel banks).

- `fxo_gs`: FXO groundstart
- `fxs_gs`: FXS groundstart
- `fxo_ks`: FXO com sinalização Kewlstart que nada mais é que Loopstart com supervisão de desconexão
- `fxs_ks`: FXS com sinalização Kewlstart
- `fxo_ls`: FXO loopstart
- `fxs_ls`: FXS loopstart
- `em`: E&M usado para conexão entre centrais
- `em_w`: E&M pulsado
- `featd`: Feature Group D, Compatível com Adtran. Para uso com ATLAS e equipamentos similares da Adtran (versão DTMF)
- `featdmf`: Standard Feature Group D (versão MF).
- `featb`: Feature Group B

Uma das sinalizações deste tipo que falta é o R2 Brasil. Acredito que em breve deva estar disponível, há vários grupos tratando disto no momento.

Normalmente quando eu projeto uma interligação entre roteadores (podemos extrapolar este conceito para o Asterisk) e centrais telefônicas com a rede pública eu procuro usar a sinalização PRI. Normalmente a integração é simples e rápida. Se você tiver disponível PRI, não escolha outra. Se você não tiver disponível, veja o que é preciso para conseguir, algumas centrais precisam de placas específicas e licenciamento de software. O Asterisk se comporta exatamente com o um roteador no momento de se integrar à rede pública ou a outras centrais telefônicas.

pri_cpe: Use a sinalização PRI como CPE/Client/User/Slave. É usado para terminar uma linha PRI em canais do Asterisk. Esta é a sinalização mais simples. Se você pediu o circuito de uma rede pública, deve funcionar de imediato. Se você vai se conectar a uma outra central, cuidado, é comum que a central estejam configurados como CPE também, pois é o caso mais comum. Neste caso peça ao técnico responsável pela central para que ele configure a central telefônica à qual você vai se interligar como Máster (A nomenclatura muda de fabricante para fabricante, alguns se referem como Máster/Slave, outros como Host/User e outros como Network/Client, assegure-se de que você consiga falar a mesma língua do técnico da central).

pri_net: Usa sinalização PRI como Rede/Master/Network

Nota: A sinalização de telefonia analógica pode ser uma fonte de confusão. Canais FXS são sinalizados com sinalização FXO e canais FXO são sinalizados com FXS. O Asterisk fala com os dispositivos internos como se estivesse do lado oposto.

Andamento da chamada

Estes itens são usados para emular a sinalização existente em linhas digitais como um PRI, que traz informações sobre o progresso da chamada. Os canais analógicos em geral não passam estas informações.

busydetect: Tenta detectar um sinal padrão em linhas analógicas FXO, FXS e E+M. (Em linhas digitais T1 e E1 usando CAS (Channel Associated Signaling) sinalizações analógicas como E+M, immediate start e wink start).

callprogress: Ao habilitar call progress o Asterisk irá tentar monitorar o estado da chamada e detectar ocupado, campainha e linha ativa. Este recurso só funciona com tons de telefones americanos.

```
callprogress=no  
busydetect=yes
```

Opções para multilink PPP (Usado com PRI, requer que o outro lado suporte).

Estas opções são usadas para ajustar as opções de multi-link PPP em linhas PRI que suportem isto. Esta tecnologia permite agrupar vários canais de um circuito E1 ou T1 com sinalização PRI em um canal de dados de forma dinâmica, compartilhando assim o canal entre voz e dados. Isto pode ser usado em um provedor de forma a dar acesso não só a canais de voz, mas permita que usuários com modems possam se conectar ao servidor, veja ZapRAS (Servidor de Acesso Remoto). Note que o Multilink PPP apenas faz o papel de permitir o agrupamento de linhas (Normalmente quando um usuário usa uma linha ISDN com acesso BRI ele agrupa dois canais de dados de 64Kbps para transmitir à 128Kbps, por isto é importante o papel do MultilinkPPP).

minunused: O número mínimo de canais disponíveis. Se existirem menos canais disponíveis, o Asterisk não irá tentar alocar quaisquer canais à conexão de dados. Formato: Número Inteiro.

minidle: O número mínimo de canais livres para agrupar um link de dados. O Asterisk irá manter este número de canais abertos para dados, ao invés de pegá-los de volta para voz. Formato: inteiro.

idledial: O número à discar como o número livre. É tipicamente o telefone agrupador. do servidor de acesso remoto. Canais ociosos serão enviados para

esta extensão. Formato: Inteiro que não conflite com nenhuma outra extensão no plano de numeração e tenha sido definido como um `idleext`.

Idleext: A extensão usada como extensão ociosa. Recebe um valor na forma de `exten@context`. De uma forma geral a extensão será uma extensão para rodar a aplicação ZapRAS.

```
minunused => 3
minidle => 1
idleext => 5000@default
idledial => 5000
```

Parâmetros de temporização

Estes valores são usados apenas com linhas T1 (Não PRI). Como é muito incomum no Brasil este tipo de sinalização eu não vou perder o tempo do caro leitor com isto. Os parâmetros são:

`prewink`, `preflash`, `wink`, `rxwink`, `rxflash`, `flash`, `start`, `debounce`, `rxwink`, `prewink`.

Opções de identificador de chamadas (Caller ID)

Existem várias opções de identificação de chamada. Algumas opções podem ser desligadas. A maior parte está habilitada por default.

usecallerid: Habilita ou desabilita a transmissão do identificador de chamadas para os seguintes canais. (Sim/Não)

hidecallerid: Configura se vai ocultar o CallerID. (Sim/Não)

`calleridcallwaiting:` Configura se vai receber a identificação de chamadas durante uma indicação de espera de chamada.

callerid: Configura a string de callerID para um dado canal. Esta chave recebe uma string formatada apropriadamente contendo o nome e o telefone a ser suprido como CallerID. O originador pode ser configurado como *asreceived* em interfaces de tronco para passar o CallerID recebido à frente.

```
usecallerid => yes
hidecallerid => no
```

Importante: Apenas linhas como PRI podem transmitir a identificação de chamadas. As operadoras exigem que você configure seu CallerID de acordo com a numeração que você recebeu de 10 dígitos. Se você não passar o CallerID com a numeração correta sua chamada não é completada, muito embora você consiga receber chamadas.

```
callerid = "Flavio Eduardo Goncalves" <48 3025-8500>
```

Opções de recursos de chamada

Estas opções habilitam ou desabilitam recursos avançados em linhas FXS.

adsi: (Analog Display Services Interface). É um conjunto de padrões da indústria de telecom. Foi usado por algumas companhias telefônicas para oferecer serviços como compras de passagens. Pouco comum no Brasil, a opção habilita ou desabilita o suporte à ADSI.

cancallforward: Habilita ou não o siga-me de chamadas. Siga-me é habilitado com *72 e desativado com *73.

immediate: Quando o Asterisk está no modo *immediate*, ao invés de prover o tom de discagem, ele imediatamente pula para a extensão s. Este recurso pode ser usado para criar uma *hotline*.

threewaycalling: Configura se vai ser permitido conferência à três daquele canal

transfer: Habilita ou desabilita a transferência usando a tecla flash. Para usar esta opção, threewaycalling deve estar configurado para yes.

```
adsi = no
cancallforward = yes
immediate = no
threewaycalling = yes
transfer = yes
```

Opções de qualidade de áudio

Estas opções ajustam certos parâmetros do Asterisk que afetam a qualidade do áudio em canais zapata.

echocancel: Desabilita ou habilita cancelamento de eco. É recomendável que permaneça ligado. Aceita 'yes' (128 taps) , 'no' ou o número de *taps* que podem ser 16, 32, 64, 128 ou 256. Cada *tap* é uma amostra de um fluxo de dados. Em um T1 isto é 1/8000 de um segundo. De acordo com o número de *taps* isto é igual à 2,4,6,8,16 ou 32 ms de comprimento.

Explicação: Como o cancelamento de eco funciona?

A maioria dos algoritmos de cancelamento de eco opera gerando múltiplas cópias do sinal recebido, cada uma atrasada por um pequeno espaço de tempo. Este pequeno fluxo é conhecido como *tap*. O número de taps determina o tamanho do atraso do eco que pode ser cancelado. Estas cópias atrasadas são

então ajustadas e subtraídas do sinal original recebido. O truque é ajustar o sinal atrasado para exatamente o necessário de forma à remover o eco e nada mais. Os métodos usados em determinar o peso do tap ou fatores de escalonamento e o que distingue um do outro.

echocancelwhenbridged: Habilita ou desabilita o cancelamento de eco durante uma chamada, puramente TDM.. Em princípio, as chamadas puramente TDM não deveriam requerer cancelamento de eco, mas frequentemente o desempenho do áudio é melhorado. Formato: yes/no.

rxgain: Ajusta o ganho de recebimento. Isto pode ser usado para aumentar ou diminuir o volume de entrada e compensar diferenças de hardware. Formato: Percentual da capacidade -100% à 100%.

txgain: Ajusta a transmissão. Isto pode ser usado para levantar ou diminuir o volume de saída para compensar diferenças de hardware. Recebe o mesmo argumento do rxgain.

```
echocancel=yes  
echocancelwhenbridged=yes  
txgain=-10%  
rxgain=10%
```

Opções de registro de chamadas

Estas opções mudam a maneira em que as chamadas são gravadas no registro detalhado de chamadas (CDR – Call Detail Records).

amaflags: Configura as AMA flags afetando a categorização das entradas no registro de chamadas. Aceita estes valores:

- **billing:** Marca o registro para tarifar.
- **documentation:** Marca o registro para documentar.
- **omit:** Não registra os chamados.
- **default:** Configura a default do sistema.

accountcode: Configura o código da conta para as chamadas colocadas no canal. O código da conta pode ser qualquer string alfanumérica.

```
accountcode=financeiro  
amaflags=billing
```

Outras opções

Algumas opções que não se encaixam nas categorias prévias.

language: Liga a internacionalização e configure a linguagem. Este recurso irá configurar as mensagens do sistema para uma dada linguagem. Embora o recurso esteja preparado, Inglês é a única linguagem que foi completamente gravada para a instalação padrão do Asterisk.

mailbox: Este comando pode dar uma mensagem avisando o usuário de que há uma mensagem esperando no correio de voz. Esta mensagem pode vir por meio de um sinal audível, ou visual se o telefone suportar. Tem como argumento o número da caixa de correio de voz.

stripmsd: Retira o 'dígito mais significativo', o primeiro dígito ou dígitos de todas as chamadas de saída em um dado grupo de canais. Formato: inteiro com o número de dígitos a retirar. Esta opção não é mais usada, existe uma opção melhor no extensions.conf usando a aplicação StripMSD.

3.4.2 Nomenclatura dos canais ZAP

Os canais ZAP configurados no arquivo zapata.conf usam o seguinte formato:

```
Zap/[g]<identificador>[c][r<cadence>
```

<identificador>- Identificador numérico para o número de canal físico do canal selecionado.

[g] – Identificador do grupo ao invés do canal

[c] – Pede confirmação de resposta. Um número não é considerado respondido até que a parte chamada pressione #

[r] – Campanha personalizada.

[cadence] Um inteiro de um à quatro.

Exemplos:

```
zap/2 - Canal 2
```

```
zap/g1 - Primeiro canal disponível no grupo 1
```

3.4.3 Exemplo de arquivo completo

No exemplo abaixo as opções usadas estão em negrito. Este arquivo é o arquivo real de configuração da nossa placa E100P ligado a GVT com DDR. A sinalização é pri_cpe e estão disponíveis 15 canais. Quase todas as opções relacionadas a telefonia analógica (que normalmente usam channel banks) não

estão configuradas. É ainda incomum o uso de channel banks devido à dificuldade e custo de importação destes equipamentos.

```
echocancel=yes
echocancelwhenbridged=no
echotraining=yes
group=1
callgroup=1
pickupgroup=1
callerid=<4830258580>
amaflags=documentation
accountcode=escritorio
musiconhold=default
signalling = pri_cpe
group = 1
channel => 1-15
```

3.4.4 Configuração dos telefones IP SIP

Neste capítulo ainda vamos dar uma visão geral de como configurar os telefones SIP. O objetivo neste momento é que você possa configurar um PABX simples. Mais a frente você vai ter uma sessão inteira dedicada ao SIP e poderemos ver em detalhes a configuração.

O SIP é configurado no arquivo `/etc/asterisk/sip.conf` e contém parâmetros relacionados à configuração dos telefones e operadoras SIP. Os clientes devem estar configurados antes que possam fazer e receber chamadas.

O arquivo SIP é lido de cima para baixo. A primeira seção contém as opções globais [general]. Estas opções são: o endereço IP e número de porta ao qual o servidor está ligado.

As seções seguintes definem os parâmetros de clientes tais como o nome do usuário, senha, e endereço IP default para clientes não registrados. A primeira seção é a [general] e as seções seguintes são o nome do cliente entre chaves ([]'s) seguida das respectivas opções.

Configurações globais (Seção [general])

- **allow:** Permite que um determinado *codec* seja usado.
- **bindaddr:** Endereço IP onde o Asterisk irá esperar pelas conexões SIP. O comportamento padrão é esperar em todas as interfaces e endereços secundários.

- **context:** Configura o contexto padrão onde todos os clientes serão colocados, a menos que seja sobrescrito na definição da entidade.
- **disallow:** Proíbe um determinado *codec*.
- **port:** Porta que o Asterisk deve esperar por conexões de entrada SIP. O padrão é 5060.
- **tos:** Configura o campo TOS (tipo de serviço) usado para o SIP e RTP. Os valores aceitáveis são *lowdelay*, *throughput*, *reliability* e *mincost*. Um inteiro de 0-255 deve ser especificado.
- **maxexpirey:** Tempo máximo para registro em segundos.
- **defaultexpirey:** Tempo padrão para registro em segundos.
- **register:** Registra o Asterisk com outro host. O formato é um endereço SIP opcionalmente seguido por uma barra normal (/) e a extensão.

3.4.5 Arquivo exemplo do sip.conf seção geral [general]

```
[general]
port = 5060
bindaddr = 10.1.30.45
context = default
disallow = speex
disallow = ilbc
allow = ulaw
maxexpirey = 120
defaultexpirey = 80
register => john@freeworlddialup.com/775657
```

3.4.6 Opções para cada telefone

Após a seção geral, seguem as definições das entidades padrão SIP. É bom lembrar que nesta seção vamos apenas dar uma introdução ao arquivo sip.conf. Teremos uma seção específica para detalhar os outros parâmetros. As entradas são divididas em três categorias:

- **peer:** Entidade que o Asterisk envia chamadas (Provedor).
- **user:** Entidade que faz chamadas através do Asterisk.

- **friend:** Os dois ao mesmo tempo , o que faz sentido para os telefones

type: Configura a classe de conexão, opções são peer, user e friend.

host: Configura o endereço IP ou o nome do host. Pode se usar também a opção ‘dynamic’ onde se espera que o telefone se registre, é a opção mais comum.

username: Esta opção configura o nome do usuário que o Asterisk tenta conectar quando uma chamada é recebida. Usado por alguma razão o valor não é o mesmo do nome do usuário do cliente registrado.

secret: Um segredo compartilhado usado para autenticar os *peers* e *users* fazendo uma chamada.

3.4.7 Exemplo completo do SIP

```
[general]
port=5060
bindaddr=10.1.30.45
context=default

register => 5551010@freeworlddialup.com

[cisco]
type=friend
secret=mysecret
host=10.1.30.50
canreinvite=no
mailbox=8580
context=trusted

[xlite]
type=friend
secret=xlite
host=dynamic
defaultip=10.1.30.17
mailbox=8590
```

3.5 Introdução ao plano de discagem

O plano de discagem é o coração do Asterisk, na medida, que ele define como o Asterisk irá gerenciar as chamadas. Ele consiste de uma lista de instruções ou passos que o Asterisk deveria seguir. Essas instruções são

disparadas a partir dos dígitos recebidos de um canal ou aplicação. É fundamental para configurar o Asterisk, que se entenda o plano de discagem.

A maior parte do plano de discagem está contida no `extensions.conf` no diretório `/etc/asterisk`. O arquivo pode ser separado em quatro partes:

- Aplicações
- Contextos
- Extensões
- Prioridades

Neste capítulo vamos ver com criar um plano de discagem básico, mas que atende as nossas necessidades atuais. Mais a frente vocês terão a oportunidade de conhecer o plano de discagem mais a fundo.

Se você instalou os arquivos de exemplo, já existe o `extensions.conf`. No nosso caso será mais interessante começar do zero. Isto vai ajudar o aprendizado e poderemos abordar passo a passo cada parte do arquivo do plano de discagem.

3.5.1 Contextos

Os contextos têm um papel importante no Asterisk na organização e segurança do plano de discagem. Os contextos também definem o escopo e permitem separar diferentes partes do plano de discagem. Um ponto que chama atenção é que os contextos estão ligados diretamente aos canais. Cada canal existe dentro de um contexto. Quando uma ligação entra no Asterisk por um canal ela é processada dentro de um contexto.

Exemplificando, vamos supor que você tenha duas classes de ramais, aqueles que podem fazer ligações de longa distância e aqueles que não. Você pode criar dois contextos, `[gerente]` e `[funcionário]`. Dentro do contexto `[gerentes]` quando o dígito “0” é discado, ouve-se o tom de discagem da rede pública. Dentro do contexto `[funcionário]` quando o dígito “0” é discado é recebida, por exemplo, uma gravação “ligação não autorizada”.

Por outro lado, uma ligação é recebida dentro do contexto do canal. Com isto diferentes canais podem ser recebidos em diferentes telefones dependendo do contexto selecionado. Isto pode ser útil para ter uma recepção diferente para cada companhia compartilhando um mesmo servidor Asterisk.

Contextos também são usados para criar menus de voz que dão ao usuário uma lista de extensões para escolher pressionando as teclas de um telefone multifrequencial. Esta funcionalidade é normalmente conhecida como auto-atendente. Auto-atendimento será visto em capítulos posteriores.

Os contextos recebem o seu nome dentro de chaves ([]s). Por exemplo, se nós fossemos criar um contexto para a entrada de chamadas, poderíamos definir como:

```
[entrada]
```

Todas as instruções colocadas após a definição são partes do contexto. Para iniciar um novo contexto, simplesmente digite o novo contexto [novocontexto]. No início do arquivo extensions.conf existe um contexto chamado [globals]. O contexto globals é onde as variáveis são definidas e podem ser usadas por todo o plano de discagem.

3.5.2 Extensões

Dentro de cada contexto serão definidas diversas extensões. No Asterisk, uma extensão é uma string que vai disparar um evento. Veja o exemplo:

```
exten=>8580,1,Dial(SIP/8580,20)
exten=>8580,2,voicemail(u8580)
exten=>8580,101,voicemail(b8580)
```

A instrução “exten=>” descreve qual o próximo passo para a ligação. O “8580” é o conjunto de dígitos que foi recebido (número discado). O “1”, “2” e “101” são as prioridades que determinam a ordem de execução dos comandos. Neste exemplo, discando “8580” irá tocar o telefone IP registrado como “8580”, se não atender em 20 segundos será desviado para a prioridade 2 na caixa de correio de voz com a mensagem “não atende”. Se estiver ocupado é desviado para a prioridade 101, vai para o correio de voz com a mensagem, “ocupado”.

Extensões determinam o fluxo das chamadas. Embora as extensões possam ser usadas para especificar as extensões, elas podem ser usadas para mais do que isto no Asterisk. Uma extensão pode ser criada com a sintaxe definida abaixo:

```
exten=> número (nome), prioridade, aplicação
```

O comando “exten=>” é seguido por um número da extensão, uma vírgula, a prioridade, outra vírgula e finalmente a aplicação.

3.5.3 Prioridades

Prioridades são passos numerados na execução de cada extensão. Cada prioridade chama uma aplicação específica. Normalmente estes números de prioridade começam com 1 e aumentam de um a um em cada extensão. Os números de prioridade como você viu acima nem sempre são consecutivos. As prioridades são rodadas na ordem numérica.

3.5.4 Aplicações

As aplicações são partes fundamentais do Asterisk, elas tratam o canal de voz, tocando sons, aceitando dígitos ou desligando uma chamada. As aplicações são chamadas com opções que afetam a sua forma de funcionamento. Você pode usar *show applications* na interface de linha de comando do Asterisk. Na medida em que você construir seu primeiro plano de discagem você vai aprender a usar as aplicações de forma apropriada.

3.5.5 Criando um ambiente de testes

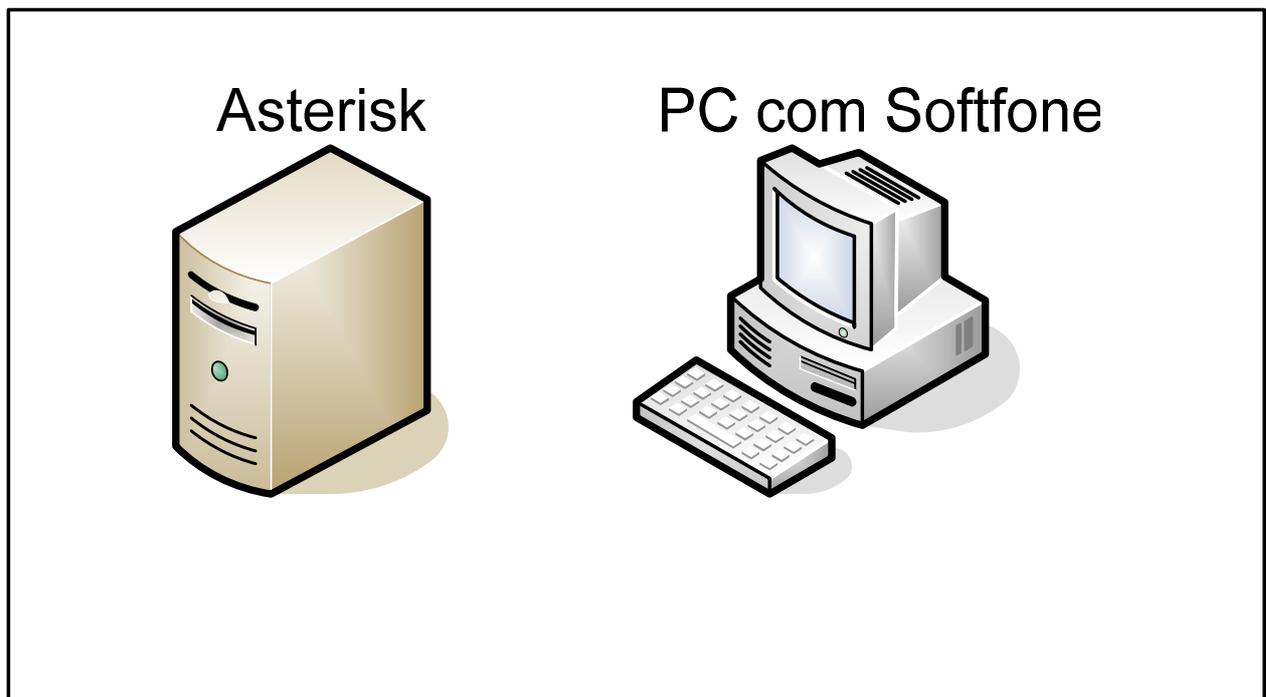


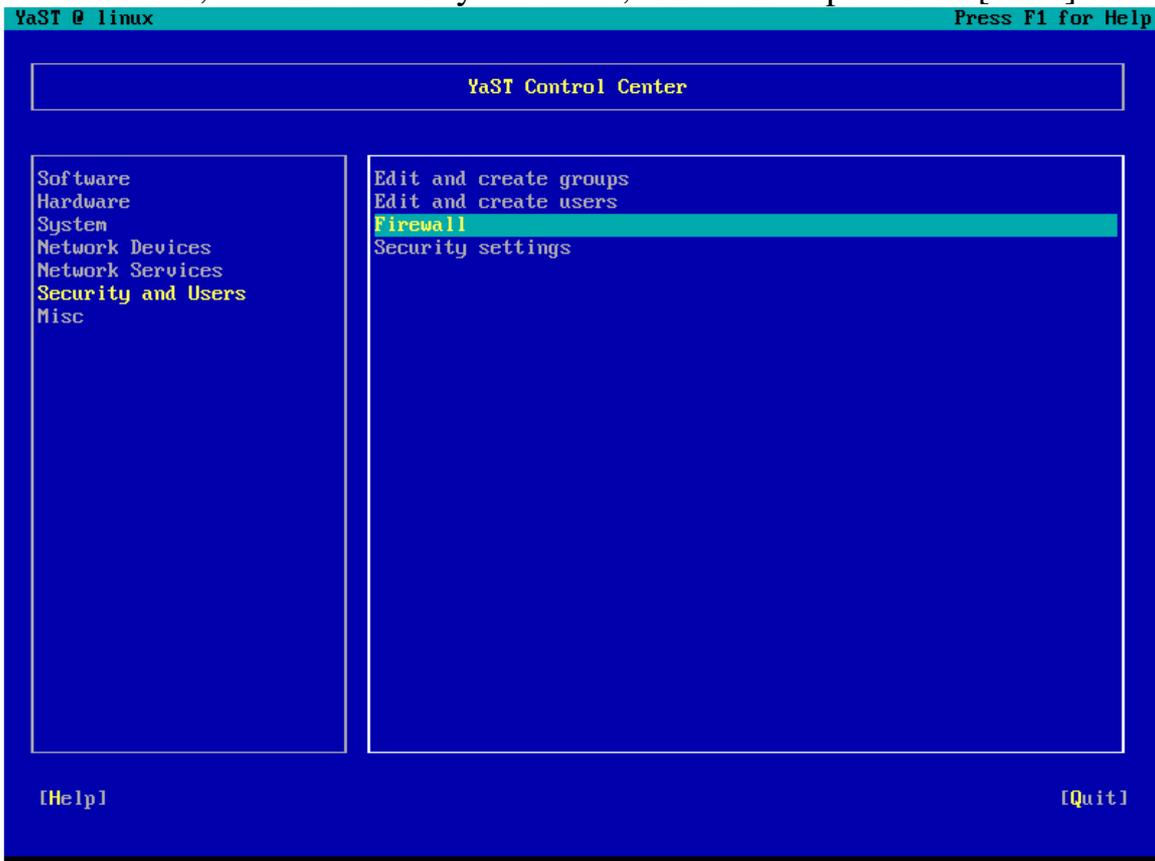
Figura 3.1 Laboratório de voz sobre IP

Para fazer o laboratório descrito abaixo você vai precisar de dois PCs, Pentium 300Mhz ou maior com 256 MB RAM pelo menos. Se você tiver um único PC pode usar o vmware (não é freeware) e você precisa de pelo menos 512 MB RAM. Vamos usar o softfone gratuito da XTEN o xlite. Você pode baixá-lo de www.xten.com.

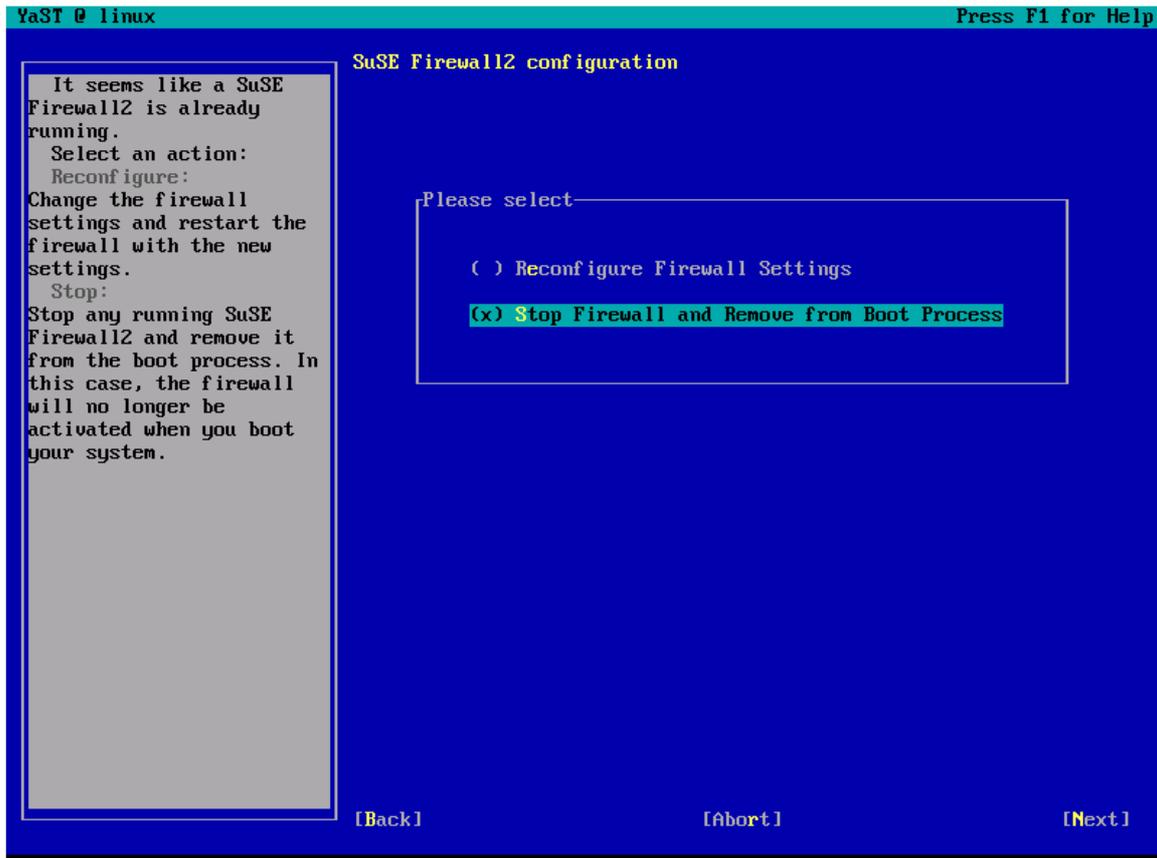
Passo 1: Desabilitar o Firewall do SuSe

De uma forma geral, quando você instala o SuSe 9.2, o serviço de firewall é automaticamente habilitado. Para que os protocolos de voz sobre IP possam operar, é preciso que o Firewall esteja desabilitado ou configurado para tal. Como o objetivo deste material não é abordar como configurar um Firewall, vamos simplesmente desabilitar o Firewall. Se você quiser configurar o Firewall, entretanto, as portas são (TCP e UDP 5060 e Faixa UDP 10000-20000).

Entre no YaST, escolha “security services”, “firewall” e pressione [enter].



Escolha a opção “Stop Firewall e Remove from Boot Process.”



Passo 2: Edite o arquivo sip.conf e adicione a configuração de um ramal.

```
[general]
port=5060
bindaddr=10.1.30.45 ; Coloque aqui o endereço IP do seu servidor
context=default
allow=all

[8000]
type=friend
secret=8000
host=dynamic
canreinvite=no
```

Passo 3: Configure o xlite de forma à acessar o Asterisk.

- a) Execute o programa de instalação
- b) Pressione “next” na primeira tela
- c) Aceite o contrato de licença
- d) Aceite as próximas telas na situação padrão até terminar a instalação. Em outras palavras NEXT->NEXT->FINISH.

e) Entre no menu no X-LITE pressionando o ícone mostrado abaixo:



f) Na tela seguinte escolha “system settings”.

g) Na próxima escolha SIP PROXY.

h) Escolha Default

i) Preencha os seguintes campos:

Display Name: 8000 Username: 8000 Authorization User: 8000 Password: 8000 Domain/Realm: Endereço IP do servidor SIP Proxy: Endereço IP do servidor

j) Feche o XLITE e abra de novo.

k) Confirme que o telefone foi registrado no Asterisk com o comando **sip show peers**.

3.5.6 Criando um plano de discagem simples

Agora estamos prontos para criar o `extensions.conf` da forma mais simples possível. Neste exemplo tudo que o Asterisk irá fazer é responder à uma chamada, tocar um som que diz “adeus” e desligar.

A extensão especial ”s”

Vamos começar vendo a extensão especial ‘s’ que significa início (start). De uma forma geral a chamada inicia no contexto ao qual o canal pertence. A extensão ‘s’ é imediatamente executada ao entrar em um canal. No nosso exemplo vamos criar um plano de discagem com a extensão ‘s’

As aplicações `Answer()`, `Hangup()` e `Playback()`

Se vamos responder a uma chamada, é melhor conhecermos as aplicações que vão fazer isto. A aplicação `answer()` é usada para responder à um canal que

está tocando. Ela faz a configuração inicial da chamada e pode fazer outras funções. Poucas aplicações não requerem que necessariamente se responda (`Answer()`) o canal antes de fazer qualquer outra coisa

A aplicação `playback()` é usada para tocar um arquivo de som previamente gravado sobre um canal. Quando a aplicação `Playback()` está sendo executada, qualquer dígito pressionado pelo usuário é simplesmente ignorado. O Asterisk vem com muitos sons pré-gravados que normalmente são encontrados em `/var/lib/asterisk/sounds`. Quando da publicação deste material, os sons em português já devem estar disponibilizados em www.asteriskbrasil.org. O formato é `Playback(nomedoarquivo)`, ele toca o arquivo com a extensão `.gsm` no diretório de sons padrão.

A aplicação `hangup()` faz exatamente o que seu nome diz. Ela desliga um canal ativo. Você deveria usá-la no fim do contexto uma vez que você queira desligar quem não precisa estar conectado no sistema.

Meu primeiro plano de discagem

Agora, nós estamos prontos para nosso primeiro exemplo de plano de discagem. Por favor, prestem atenção à maneira que cada prioridade chama uma aplicação. Note que neste exemplo temos apenas uma extensão criada com SIP. Estes exemplos assumem que você tem uma placa FXO. Vamos mostrar os exemplos e explicar passo à passo.

```
[entrada]
exten=>s,1,answer()
exten=>s,2,playback(goodbye)
exten=>s,3,hangup()
```

Uma chamada que entre pela FXO é enviada dentro do contexto `[entrada]` (Este contexto `[entrada]` deve estar configurado no arquivo `zapata.conf` para o canal FXO) e é enviada para a extensão 's'. Nós temos três prioridades no contexto, 1, 2 e 3. Cada prioridade chama um aplicativo. Vamos olhar de perto as prioridades:

A prioridade 1 chama a aplicação `answer()`, o Asterisk toma conta da linha e configura a chamada. Após responder a linha, o asterisk vai para a próxima prioridade.

Prioridade 2, o Asterisk chama a aplicação `playback()` para tocar o arquivo de som `goodbye.gsm` (por enquanto vamos usar as mensagens em inglês), esta mensagem dará uma mensagem de adeus ao usuário.

Por fim a prioridade 3 irá desconectar o usuário.

Um exemplo mais útil

Agora que já começamos simples, vamos incrementar aprendendo um pouco dos aplicativos `background()` e `goto()`. Estas duas aplicações irão permitir nos criar planos de discagem, com muito mais funcionalidade.

A chave para sistemas interativos baseados no Asterisk está na aplicação `Background()`. Ela permite que você grave um arquivo de som, mas quando o originador pressiona uma tecla isso interrompe a gravação e manda para a extensão correspondente aos dígitos discados.

Outra aplicação muito útil é o `goto()`. Como seu nome implica, ele pula de um contexto atual, extensão e prioridade para um contexto específico e prioridade. A aplicação `goto()` torna fácil a movimentação entre diferentes partes do plano de discagem. O formato do comando `goto()` precisa do contexto de destino e prioridade como argumentos.

```
exten=>extensão, prioridade,Goto(contexto,extensão, prioridade)
```

Formatos válidos do comando `Goto()` são :

```
Goto(contexto,extensão,prioridade)
Goto(extensão,prioridade)
Goto(prioridade)
```

Neste exemplo, vamos assumir que somos uma empresa de suporte, treinamento e vendas. Vamos criar um sistema interativo que permita ao usuário selecionar para que área da empresa ele queira ser redirecionado. Em primeiro lugar vamos usar o comando `Background()` para tocar a mensagem “disque 1 para suporte, 2 para treinamento e 3 para vendas”, neste momento ainda não vamos explorar o tratamento de dígitos inválidos. Em cada contexto vamos tocar uma gravação como “você foi redirecionado para o suporte (treinamento ou vendas)”.

```
[entrada]
exten=>s,1,Answer()
exten=>s,2,Background(saudação)
exten=>s,3,hangup()
exten=>1,1,playback(suporte)
exten=>1,2,goto(suporte,s,1)
exten=>2,1,playback(treinamento)
exten=>2,2,goto(treinamento,s,1)
exten=>3,1,playback(vendas,)
```

```
exten=>3,2,goto(vendas,s,1)
```

Vamos seguir este exemplo passo à passo. Quando alguém liga na interface FXO (Configurada para o contexto [entrada]), a ligação é passada para a extensão 's' dentro do contexto [entrada]. A extensão 's' atende a ligação e usando o comando Background() toca uma saudação e aguarda pela discagem de um dígito. Após discar o dígito '1' por exemplo, o sistema vai para a extensão '1' na prioridade '1' e toca uma mensagem no arquivo suporte.gsm (Algo como, "você ligou para o suporte técnico" atenderemos em alguns minutos). Em seguida temos o comando goto() que manda a ligação para o contexto (suporte) onde ela será tratada por uma fila de atendimento.

Interligando canais com a aplicação Dial()

Nós vamos adicionar ao nosso exemplo a aplicação Dial(). Ao invés de mandar a ligação para outro contexto, vamos atendê-la em um ramal específico.

```
[entrada]
exten=>s,1,Answer()
exten=>s,2,Background(saudação)
exten=>s,3,hangup()
exten=>1,1,playback(suporte)
exten=>1,2,Dial(SIP/8000)
exten=>2,1,playback(treinamento)
exten=>2,2,Dial(ZAP/1)
exten=>3,1,playback(vendas,)
exten=>3,2,Dial(IAX/8002)
```

Ao comparar com o exemplo anterior, apenas criamos um atalho. Ao invés de enviar para um outro contexto para o tratamento da chamada, enviamos diretamente à um canal SIP no ramal 8000 ou à um canal Zaptel (FXS - Analógico) ou ainda à um canal IAX no ramal 8002.

Neste ponto você já deve estar entendendo o uso de várias aplicações como o Answer(), Background(), Goto(), Hangup() e Playback() e o básico do comando Dial(). Isto é fundamental para o aprendizado daqui para frente. Se ainda não está claro volte e leia de novo, é fundamental que fique entendido o processo antes que se passe a frente.

Com um entendimento básico das extensões, prioridades e aplicações é simples criar um plano de numeração básico. Nos próximos capítulos vamos fazer um plano de discagem ainda mais poderoso.

3.6 Lab. Implantando uma aplicação simples

Nestes exercícios não vamos usar o `extensions.conf` de exemplo, vamos copiar o `extensions.conf` para `extensions.conf.bak` e criar um arquivo `extensions.conf` novo. Para renomear o arquivo `extensions.conf` atual use:

```
mv extensions.conf extensions.conf.bak
```

Edite o arquivo `extensions.conf` e adicione as seguintes linhas:

```
[default]
exten=>9000,1,Answer()
exten=>9000,2,Playback(demo-thanks)
exten=>9000,3,hangup
```

A partir do softfone disque 9000.

3.7 Sofisticando um pouco mais.

Vamos fazer algumas gravações para o exercício seguinte. Para fazer estas gravações vamos usar o aplicativo `Record()`, as gravações serão armazenadas no diretório `/var/lib/asterisk/sounds`. Adicione as seguintes linhas no arquivo `extensions.conf`.

```
[default]
exten=>9000,1,Answer()
exten=>9000,2,Playback(demo-thanks)
exten=>9000,3,hangup

; Use a extensão 9001 para gravar:
; “Você ligou para a XYZ, disque 1 para suporte, 2 para treinamento, 3 para vendas”.
exten=>9001,1,Wait(2)
exten=>9001,2,Record(menu:gsm)
exten=>9001,3,Wait(2)
exten=>9001,4,Playback(menu)
exten=>9001,5,Wait(2)
exten=>9001,6,Hangup

; Use a extensão 9002 para gravar “Você ligou para o suporte”
exten=>9002,1,Wait(2)
exten=>9002,2,Record(suporte:gsm)
exten=>9002,3,Wait(2)
exten=>9002,4,Playback(suporte)
exten=>9002,5,Wait(2)
exten=>9002,6,Hangup
```

```

; Use a extensão 9003 para gravar "Você ligou para o treinamento"
exten=>9003,1,Wait(2)
exten=>9003,2,Record(treinamento:gsm)
exten=>9003,3,Wait(2)
exten=>9003,4,Playback(treinamento)
exten=>9003,5,Wait(2)
exten=>9003,6,Hangup

; Use a extensão 9004 para gravar "Você ligou para vendas"
exten=>9004,1,Wait(2)
exten=>9004,2,Record(vendas:gsm)
exten=>9004,3,Wait(2)
exten=>9004,4,Playback(vendas)
exten=>9004,5,Wait(2)
exten=>9004,6,Hangup

```

3.8 Exemplo de uma URA simples

Vamos agora criar uma URA simples usando os comandos Background() e Goto(). Neste exemplo, você vai discar 8000 para ser direcionado para o menu da URA.

```

[default]
exten=>8000,1,Goto(entrada,s,1)

[entrada]
exten=>s,1,Answer()
exten=>s,2,Background(menu)
exten=>s,3,Wait(2)
exten=>s,4,Goto(s,2)

exten=>1,1,playback(suporte)
exten=>1,2,hangup()
exten=>2,1,playback(treinamento)
exten=>2,2,hangup()
exten=>3,1,playback(vendas,)
exten=>3,2,hangup()

```

3.9 Sumário

Neste capítulo você aprendeu que os arquivos de configuração ficam em /etc/asterisk. Para usar o Asterisk é preciso em primeiro lugar configurar os canais (Ex. sip.conf e zapata.conf). Existem basicamente três formatos o grupo simples, herança de opções entidade complexa.

O plano de discagem é criado no arquivo extensions.conf, nele são criados contextos, aplicações, extensões e prioridades. As aplicações que usamos forma Playback(), Background(), Dial(), Goto(), Hangup() e Answer().

Background() é um comando importante na criação de uma URA. Dial() é o principal comando do plano de discagem

3.10 Questionário

1. São exemplos de arquivos de configuração de canais Asterisk.

- zaptel.conf
- zapata.conf
- sip.conf
- iax.conf

2. É importante definir o contexto no arquivo de canais, pois quando uma ligação deste canal (sip, iax, zap) chegar ao Asterisk ele será tratado no arquivo extensions.conf neste contexto.

- Correto
- Incorreto

3. O parâmetro switchtype no arquivo zapata.conf define o tipo de PABX ao qual o Asterisk está ligado. Isto é válido para conexões no padrão E1 com sinalização ISDN PRI. Normalmente no Brasil e na Europa este padrão deve ser definido como **National**.

- Correto
- Incorreto

4. Apesar de ser considerado uma linha digital, o E1 pode ser configurado com sinalização associada ao canal (CAS) neste caso cada timeslot pode se comportar como um canal analógico FXS ou FXO, por exemplo. Isto é útil para a conexão à channel-banks (bancos de canal).

- Correto
- Incorreto

5. SIP Session Initiated Protocol é o protocolo da ITU usado para conexões de voz sobre IP. Ele é bastante antigo e vem sendo substituído recentemente pelo H.323.

- Correto
- Incorreto

6. Dado a configuração abaixo do arquivo sip.conf, na seção [general] está definido o endereço IP 10.1.30.45, onde o SIP estará esperando por conexões.

Se fosse necessário que todas as placas de rede da máquina esperassem por uma conexão SIP, `bindaddr` deveria estar configurado para: _____.____.____.____

```
[general]
port = 5060
bindaddr = 10.1.30.45
context = default
disallow = speex
disallow = ilbc
allow = ulaw
maxexpirey = 120
defaultexpirey = 80
```

7. No arquivo abaixo, os telefones 8000 e 8001 foram definidos com a opção `canreinvite=no`. Com isto, quando uma ligação é feita de um telefone para o outro, o Áudio vai diretamente de um telefone para outro sem passar pelo Asterisk.

```
[8000]
type=friend
secret=8000
host=dynamic
canreinvite=no

[8001]
type=friend
secret=8000
host=dynamic
canreinvite=no
```

- Correto
 Incorreto

8. A principal diferença entre o comando `Playback()` e o comando `Background()` é que o `Playback()` simplesmente toca uma mensagem e passa ao comando seguinte, enquanto o `Background` aguarda que você digite algo e desvia para algum lugar no plano de discagem baseado nos dígitos discados.

- Correto
 Incorreto

9. Quando uma ligação entra no Asterisk por uma interface de telefonia (FXO) sem identificação de chamada, esta ligação é desviada para a extensão especial:

- '0'
 '9'

- 's'
- 'i'

10. Os formatos válidos para o comando Goto() são:

- Goto(contexto, extensão, prioridade)
- Goto(prioridade, contexto, extensão)
- Goto(extensão, prioridade)
- Goto(prioridade)

Página deixada intencionalmente em branco

Capítulo 4 – Voz sobre IP com o Asterisk.

Neste capítulo vamos aprender conceitos básicos sobre voz sobre IP aplicados a realidade do Asterisk.

4.1 Objetivos

- Compreender os benefícios e aplicações da voz sobre IP.
- Compreender como o Asterisk trata voz sobre IP.
- Entender o conceito de canais TDM, SIP, IAX e H323.
- Compreender que protocolos são mais apropriados para que situações
- Entender os conceitos de peers, users e friends.
- Codecs e conversão de codecs.

4.2 Introdução

Neste capítulo vamos aprender alguns conceitos gerais sobre Voip. Isto é importante, pois daqui para frente boa parte da telefonia, principalmente à baseada no Asterisk será canalizada para “Voz sobre IP”. Por isso é importante que exista um entendimento básico desta tecnologia.

4.3 Benefícios da voz sobre IP

O benefício chave do VoIP é combinar redes de voz e dados para reduzir custos. Se você olhar estritamente em custo por minuto, a economia com VoIP pode não ser suficiente para justificar o investimento neste tipo de serviço. Em alguns países onde o custo de uma ligação telefônica pode chegar a um dólar por minuto, certamente é justificável. Em outros lugares onde os custos de telecomunicações estão caindo dia-a-dia, isto pode não ser suficiente. Entretanto existem outros benefícios associados com VoIP, como o uso de uma única infra-estrutura, adição, mudança e remoção de pontos são mais simples do que em telefonia tradicional, até porque o número do telefone passa a ser uma configuração da linha e não do telefone. Algumas pessoas têm dito que escolheram telefonia IP usando Asterisk pela liberdade de fazer as

configurações elas mesmas sem ter de depender de um serviço externo, o que é comum com centrais de telefonia tradicionais.

Outro ponto fundamental é o uso de aplicações de telefonia. Este sim, eu considero motivo número 1 da adoção de VoIP. Mobilidade, Unidade de Resposta Audível, Filas de Atendimento e “Integração Telefonia Computador” são recursos que em VoIP são econômicos e fáceis de implementar. O poder destas aplicações em um país, onde o telefone chega em locais onde a Internet ainda não chegou é de extrema importância.

Algumas aplicações que estão crescendo rapidamente em telefonia IP são:

4.3.1 Packet Telephony Call Center

Você pode reduzir drasticamente o custo de um Call-Center usando voz sobre IP. Além de o Asterisk possuir os principais recursos para Call-Centers ativos como discador, gravação e bilhetagem, o Asterisk possui uma grande gama de recursos para implementar Call-Centers receptivos e CRM, como fila de atendimento, gravação, unidade de resposta automática, sintetização e reconhecimento de voz e bilhetagem avançada. Até mesmo uma interface TAPI (Microsoft Telephony API) está disponível para o uso de Screen-Pop.

Outro ponto interessante é a possibilidade de integrar Call-Centers distantes permitindo um uso otimizado dos agentes de telefonia independente da localidade. É possível também usar agentes trabalhando em casa ligados ao mesmo Call-Center. Pense, você pode ter um Call-Center que fala 10 línguas, com pessoas em 10 locais diferentes do mundo atendendo para você.

4.3.2 Unified Messaging

O uso de e-mail e fax integrados à central telefônica é um dos excepcionais recursos do Asterisk. Estes recursos em outras plataformas custam caríssimos e são difíceis de implementar e manter. O licenciamento de algumas é tão confuso que para obter uma cotação se levam mais de 15 dias.

4.3.3 Chamada baseada em cartão

Um dos mercados que mais crescem é o de chamada com cartão. Toda operadora tem algum tipo de cartão pré-pago. No Brasil principalmente as operadoras de telefonia celular. Este tipo de serviço depende de regulamentação, mas pensando em VoIP, é possível estabelecer um número em cada cidade (0800) onde o usuário liga, coloca o número do seu cartão e fala com qualquer lugar do mundo com VoIP, sem ter de discar 0800. Como eu

disse isto depende da ANATEL e talvez não se popularize no Brasil como se popularizou nos EUA.

4.4 Arquitetura do Asterisk e Voz sobre IP

Como pode ser visto na figura abaixo, as tecnologias e protocolos de voz sobre IP são tratados como canais do Asterisk. O Asterisk pode usar simultaneamente protocolos do tipo TDM como o ISDN e interfaces analógicas junto com canais VoIP nos padrões SIP, H323, MGCP, IAX e SCCP.

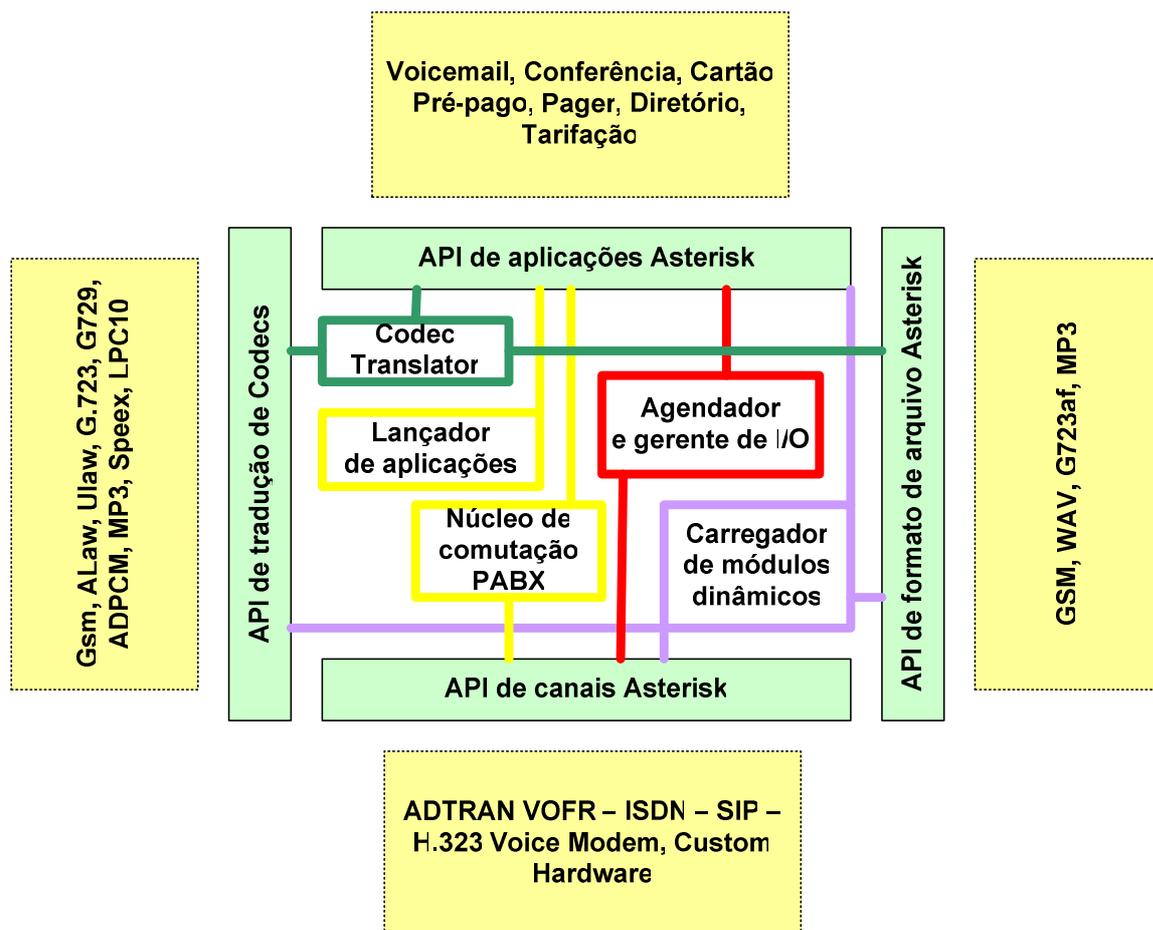


Figura 1 Arquitetura do Asterisk.

O ponto fundamental da arquitetura do Asterisk é que ele funciona como um gateway de mídia entre todos estes protocolos e não apenas como um proxy de sinalização. Com isto um canal pode estar configurado em IAX2 com codec GSM e se comunicar com outro com SIP e Codec G.711.

Nos capítulos seguintes vamos explicar um pouco das características de cada um destes protocolos, ver quando eles são mais recomendados. Em capítulos a frente vamos dar uma ênfase maior aos dois mais utilizados que são o SIP Session Initiated Protocol e o IAX Inter Asterisk Exchange, como a maioria dos gateways e provedores no Brasil ainda usam H.323 vamos falar um pouco sobre a implementação deste protocolo.

Presentation	G.729/G.711
Session	H.323/H.323Gateway/SIP/SDP
Transport	RTP/UDP/RSVP
Network	IP/LLQ/IP Precedence
Link	MLPPP/Frame Relay/ATM AAL1/LLQ
Physical	*****

Figura 4-2 Distribuição dos protocolos de VoIP dentro do modelo OSI.

Como se pode ver na figura acima, voz sobre IP é composto de diversos protocolos envolvendo várias camadas do modelo OSI. De qualquer forma VoIP é na verdade uma aplicação que funciona sobre as redes IP atuais. Estaremos aqui tratando principalmente das camadas de transporte, sessão, apresentação e aplicação.

Na camada de transporte, a maior parte destes protocolos usa o RTP/RTCP sendo o primeiro um protocolo de mídia e o segundo um protocolo de controle. A exceção é o IAX que implementa um transporte de mídia próprio. Todos eles usam o UDP para transportar a voz.

Na camada de sessão entram os protocolos de voz sobre ip propriamente ditos, H323, SIP, MGCP, IAX e SCCP.

Na camada de sessão os CODECs definem o formato da apresentação da voz com suas diferentes variações de compressão.

4.5 Como escolher um protocolo

4.5.1 SIP

Padrão aberto descrito pela IETF, largamente implementado, as principais operadoras VoIP estão usando SIP. É o protocolo padrão de fato para telefonia IP no momento. Ponto forte, padrão da IETF, adoção do mercado. Pontos fracos, problemas no uso do NAT, uso da banda com RTP é alto.

4.5.2 IAX

Protocolo proprietário do Asterisk. Eficiente em banda passante e principalmente pode passar facilmente por firewalls com NAT. Se quiser usar SIP com NAT na Internet pode se usar o SER (Sip Express Router) em conjunto o Asterisk. Pontos fortes, eficiência em banda passante, segurança e facilidade com NAT. Ponto fraco, proprietário.

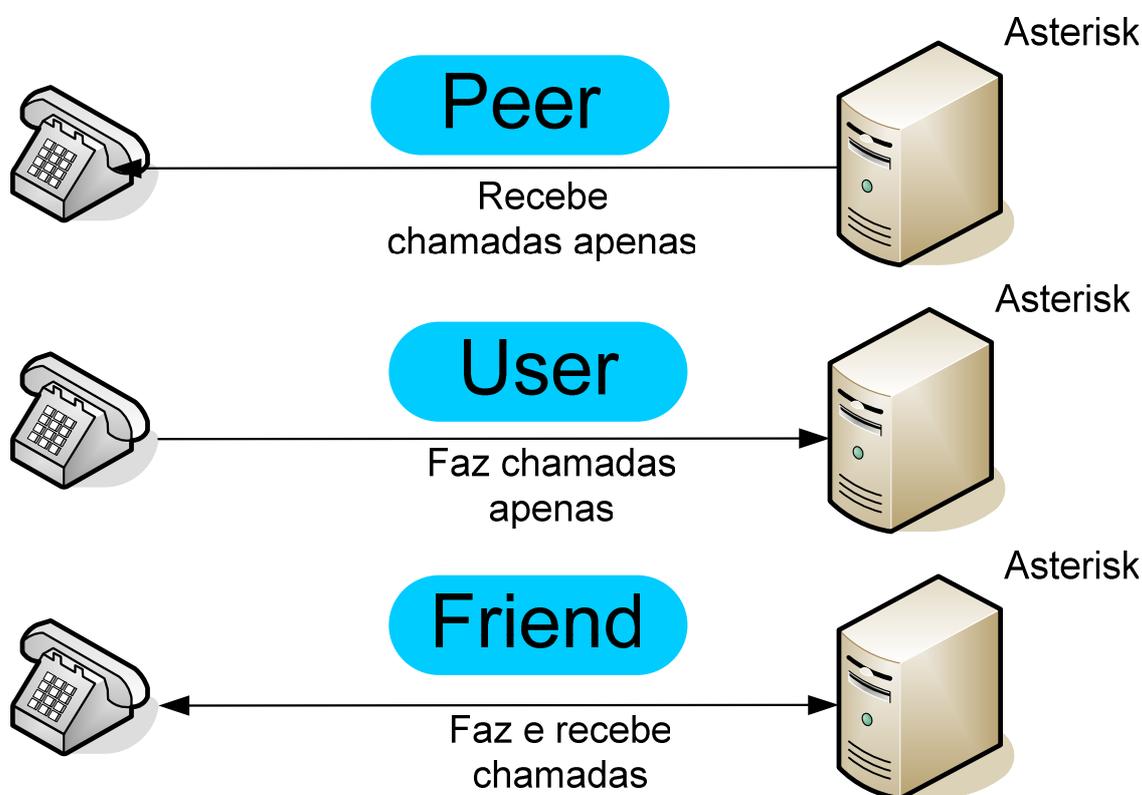
4.5.3 MGCP

É um protocolo para ser usado em conjunto com o H323, SIP e IAX. Sua grande vantagem é a escalabilidade. Toda a inteligência é implementada no Call Agent ao invés dos gateways. Simplifica muito a configuração. Pontos fortes: Gerenciamento centralizado, pontos fracos, o protocolo é pouco adotado ainda.

4.5.4 H323

Largamente usado em voz sobre ip. Essencial na conectividade com projetos mais antigos usando roteadores Cisco ou gateways de voz. H323 ainda é padrão para fornecedores de PBX e roteadores, muito embora eles comecem a adotar o SIP. Excelente para videoconferência. Pontos fortes, larga adoção do mercado, padronização pela ITU. Pontos fracos: complexo, pouco adotado em telefonia IP.

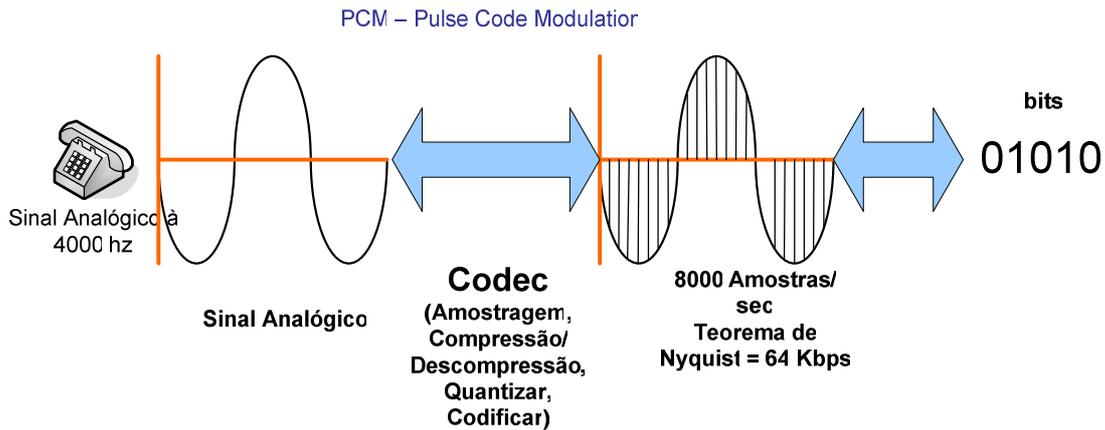
4.6 Conceito de Peers, Users e Friends



Existem três tipos de clientes SIP e IAX. O primeiro é o user. Usuários podem fazer chamadas através de um servidor Asterisk, mas não podem receber chamadas do servidor. Isto é útil em uma situação onde você pode prover alguns serviços telefônicos ao cliente, mas nunca deve poder chamar o telefone, tal como um provedor de longa distância. O segundo é o peer. Um peer é um cliente para o qual você pode passar as chamadas, mas que você nunca vai receber chamadas dele. Isto pode ser útil para ter um telefone que só recebe chamadas, ou passar chamadas à um servidor Asterisk de uso especial como um voice mail.

Normalmente, o servidor ou dispositivo irá precisar ser ambos um “user” e um “peer” ao mesmo tempo, neste caso, você o definiria como um “friend”, que é um atalho para “user” e “peer”. Um “friend” pode ambos enviar e receber chamadas de um servidor. Um telefone provavelmente cairá nesta categoria, assim como um servidor remoto que precise acesso à suas extensões.

4.7 Codecs e conversão de Codecs



Codecs são usados para converter um sinal analógico de voz em uma versão codificada digitalmente. Codecs variam na qualidade do som, banda passante necessária e requisitos computacionais. Cada serviço, programa, fone ou gateway, tipicamente, suporta vários codecs diferentes e quando vão falar um com outro negociam que codec que vão usar. Alguns codecs como o G.729 necessitam de pagamento de royalties para o seu uso.

O Asterisk suporta os seguintes Codecs:

- **GIPS:** 13.3 Kbps e acima
- **GSM:** 13 Kbps, 20ms de 'tamanho do frame'
- **iLBC:** 15Kbps, 20ms de 'tamanho do frame', 13.3 Kbps com 30ms de 'tamanho do frame'.
- **ITU G.711:** 64 Kbps, também conhecido como alaw/ulaw.
- **ITU G.722:** 48/56/64 Kbps.
- **ITU G.723.1:** 5.3/6.3 Kbps, 30ms de 'tamanho do frame'.
- **ITU G.726:** 16/24/32/40 Kbps.
- **ITU G.728:** 16 Kbps
- **ITU G.729:** 8 Kbps, 10ms de 'tamanho do frame'.
- **Speex** - 2.15 to 44.2 Kbps
- **LPC10** - 2.5 Kbps
- **DoD CELP** - 4.8 Kbps

Os Codecs podem ser traduzidos de um para outro, muito embora existam casos onde isto não funciona muito bem. É possível também usar o modo pass-thru onde o Asterisk, não se encarrega de traduzir os fluxos de mídia que seguem diretamente de um telefone para outro. Nesta opção não é possível, gravar, ou usar o voice-mail.

4.8 Sumário

Neste capítulo você aprendeu que o Asterisk trata a voz como um canal, assim como um canal TDM. Aprendeu como tudo se encaixa dentro do modelo OSI. Aprendeu a comparar os protocolos H.323 o mais velho, IAX2 o mais eficiente, mas proprietário e ainda pouco adotado e SIP o mais aceito, mas com problemas sérios com NAT. Entendeu como diferenciar users (client), peers(server) e friends(ambos). Entendeu o conceito de codecs e tradução de codecs.

4.9 Questionário

1. Cite pelo menos quatro benefícios do uso de Voz sobre IP

2. Convergência é a unificação das redes de voz, vídeo e dados em uma única rede e seu principal benefício é a redução com os custos de manutenção de redes separadas.

- Correto
- Incorreto

3. O Asterisk não pode usar simultaneamente recursos de PSTN (Rede pública de telefonia e de voz sobre IP, pois os codecs não são compatíveis).

- Correto
- Incorreto

4. A Arquitetura do Asterisk é de um SIP proxy com possibilidade outros protocolos.

- Correto
- Incorreto

5. Dentro do modelo OSI, os protocolos SIP, H.323 e IAX2 estão na camada de:

- Apresentação
- Aplicação
- Física
- Sessão
- Enlace

6. SIP é hoje o protocolo mais aberto (IETF) sendo implementado pela maioria dos fabricantes.

- Correto
- Incorreto

7. O H.323 é um protocolo sem expressão, pouco usado foi abandonado pelo mercado em favor do SIP.

- Correto
- Incorreto

8. O IAX2 é um protocolo proprietário da Digium, apesar da pouca adoção por fabricantes de telefone o IAX é excelente nas questões de:

- Uso de banda
- Uso de vídeo
- Passagem por redes que possuem NAT
- Padronizado por órgãos como a IETF e ITU.

9. “Users” podem receber chamadas

- Correto
- Incorreto

10. Sobre codecs assinale o que é verdadeiro

- O G711 é o equivalente ao PCM (Pulse Code Modulation) e usa 64 Kbps de banda.
- O G.729 é gratuito por isto é o mais utilizado, usa apenas 8 Kbps de banda.
- GSM vêm crescendo, pois ocupa 12 Kbps de banda e não precisa de licença.
- G711 u-law é comum nos EUA enquanto a-law é comum na Europa e no Brasil.
- G.729 é leve e ocupa pouca CPU na sua codificação.

Página deixada intencionalmente em branco

O Protocolo IAX e o Asterisk

Neste capítulo vamos aprender um pouco sobre o protocolo IAX, seus pontos fortes e pontos fracos. Detalhes como interfaces trunked e integração entre dois Asterisks também serão vistos.

5.1 Objetivos do Capítulo

Ao final deste capítulo, você deve ser capaz de:

- Identificar os pontos fortes e fracos do protocolo IAX.
- Demonstrar em que cenários de uso o IAX pode ser usado.
- Explicar o uso do modo trunked e suas vantagens.
- Calcular a quantidade de banda passante usada pelo IAX.
- Configurar o arquivo `iax.conf` para telefones e provedores.
- Descrever os principais comandos de console do IAX.

5.2 Introdução

Todas as referências ao IAX neste documento correspondem a versão 2, normalmente chamado de IAX2. O IAX2 substitui o IAX e como tal vamos nos referir a ele como apenas IAX daqui em diante.

O Inter-Asterisk eXchange Protocol fornece controle e transmissão de voz sobre redes IP. O IAX pode ser usado com qualquer tipo de mídia como voz e vídeo, mas foi pensado primariamente para chamadas em voz. Os objetivos do projeto do IAX derivaram da experiência com os protocolos de voz sobre ip como o SIP (Session Initiated Protocol) e o MGCP (Media Gateway Control Protocol) para controle e o RTP para o fluxo-multimídia (streaming media) e são:

- Minimizar o uso de banda passante para o tráfego de ambos, media e controle com ênfase específica em chamadas de voz individuais.

- Prover transparência à NAT (Network Address Translation).
- Ter a possibilidade de transmitir informações sobre o plano de discagem.
- Suportar a implantação eficiente de recursos de paging e intercomunicação.

5.2 Teoria de operação

IAX é um protocolo de mídia e sinalização “peer-to-peer”. Isso significa que os dispositivos mantêm conexões associadas com as operações de protocolo. Com respeito ao componente de sinalização do IAX, ele é mais parecido com o SIP do que com o MGCP, que é um protocolo de controle do tipo mestre-escravo.

A abordagem do projeto básico do IAX multiplexa a sinalização e múltiplos fluxos sobre uma única associação UDP entre dois hosts de Internet. Nesta faceta do projeto, ele se torna dois protocolos, o primeiro é o protocolo de sinalização das sessões, o segundo o protocolo para transportar os fluxos de mídia. Esta abordagem difere da arquitetura geral dos protocolos baseados na IETF com dois protocolos separados para, sinalização (MGCP e SIP) e fluxo de mídia (RTP/RTCP). Como o IAX usa o mesmo protocolo para sinalização e mídia em uma mesma porta UDP, ele não sofre dos problemas de atravessar dispositivos que fazem NAT (Network Address Translation), como, por exemplo, roteadores ADSL.(característica fundamental para operadoras de telefonia IP). O IAX usa a porta UDP 4569 para comunicar todos os pacotes. O IAX então usa um número de chamada de 15 bits para multiplexar os fluxos sobre uma única associação UDP.

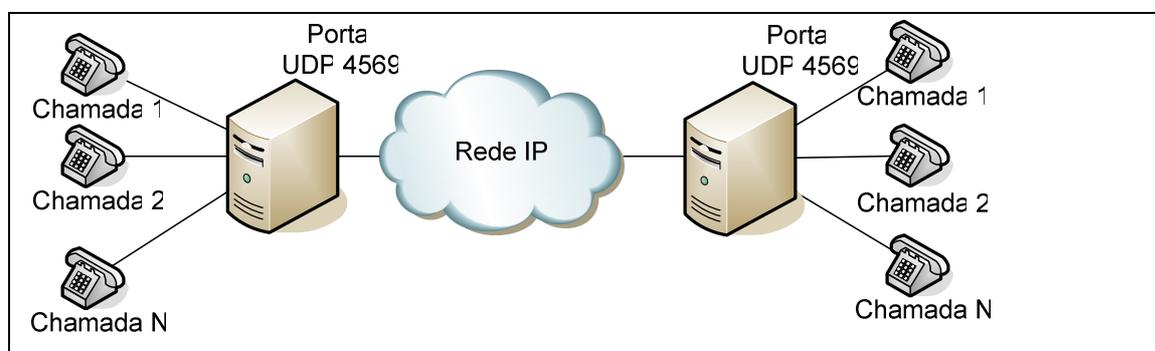


Figura 5.1: Múltiplas chamadas sobre uma única associação de porta UDP.

O valor de 0 é um número especial de chamada reservado em cada host. Quando tentando fazer uma chamada, o número de chamada do destino ainda

Mini Frame

O miniframe é usado para enviar o áudio ou vídeo (mídia) com um mínimo de sobrecarga de protocolo. O formato do miniframe segue abaixo.

										1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	2	3	3		
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	
F	Número originador da chamada																Timestamp															
Dados																																

Figure 5-3 Formato binário de um mini-frame

O timestamp do Mini Frame é truncado. O cliente geralmente mantém o timestamp completo de 32 bits. Quando enviando mini frames, os 16 bits de ordem mais baixa são enviados no campo timestamp. Quando o timestamp de 16 bits dá a volta (estoura) um frame completo é enviado para permitir que o outro lado sincronize.

Campo	Descrição
F	Setado para 1 indica que é um frame completo
Source Call Number	Número de chamada originador do lado de transmissão do frame
R	Setado para um indica que o frame está sendo retransmitido e o valor de 0 para a transmissão inicial.
Destination Call Number	Número de chamada de destino do lado receptor do frame.
Timestamp	Timestamp completo 32-Bits
OSeqno	Número de seqüência do fluxo de saída
ISeqno	Número de seqüência do fluxo de entrada
Frame Type	Tipo de Frame
C	Formato do valor da subclasse
Subclasse	Subclasse

Tabela 5.1 Descrição dos campos do Frame Completo

Campo	Descrição
F	Setado para 0 indica que é um frame incompleto
Source Call Number	Número de chamada originador do lado de transmissão do frame completo
Timestamp	Timestamp 16-Bits
Dados	Dados

Tabela 5.2 Descrição dos campos do Mini Frame

Uma descrição completa do protocolo IAX pode ser encontrada em <http://www.cornfed.com/iax.pdf>

5.4 Uso de banda passante

O uso de banda passante em voz sobre IP é afetado por uma série de fatores. Desde o CODEC, até questões como compressão de cabeçalhos e

O IAX permite o uso do modo Trunked. Neste caso, quando mais de uma ligação é feita, o overhead dos cabeçalhos IP é diminuído, encaminhando múltiplos pacotes de voz de diferentes ligações em um único pacote. Com isto a necessidade de banda é reduzida. Abaixo uma tabela da banda passante utilizada com SIP e IAX comparando com o modo trunked. Existem diversas calculadoras de banda na Internet para facilitar o trabalho do dimensionamento. Uma que pode ser acessado de forma gratuita é: <http://www.packetizer.com/voip/diagnostics/bandcalc.html>

Considere a banda ocupada pelos CODECs da tabela abaixo:

Protocolo	Kbps
lpc10	2.5
g.723.1	6.1
g.729a	8
iLBC	9
GSM	13
g.726	32
g.711	64

O uso de banda é influenciado por outras variáveis como o número de ligações simultâneas e pela compressão de cabeçalho RTP. Compressão de cabeçalho é uma tecnologia definida pela IETF que pode ser implementada em canais do tipo PPP, HDLC e Frame-Relay. Compressão de cabeçalho RTP é uma tecnologia ponto a ponto e não pode ser implementada sobre Ethernet. Por isto, o Asterisk sempre enviará os dados como RTP sem compressão. Um roteador com QoS habilitado pode fazer a compressão de cabeçalhos para os links de WAN, fazendo com que este tráfego tenha um uso de banda maior na

LAN que na WAN. No caso do IAX ele não usa o RTP e passa o protocolo de mídia junto com o protocolo de sinalização.

Considere também o tamanho dos cabeçalhos (bytes).

Protocolo	octetos	bits	+ip/udp
iax2	4	32	32
iax2(t)	10	80	38
cRTP	2 ou 4	16 ou 32	2 ou 4 bytes
RTP	12	96	40

A quantidade de pacotes por segundo = Codec bit rate / voice payload size.

5.4.1 Uso de banda do IAX

Protocolo IAX2 em modo normal (Kbps)

Protocolo	1	2	4	8	16	32
lpc10	11.9	23.8	47.5	95.0	190.0	380.0
g.723.1	13.9	27.8	55.6	111.2	222.4	444.8
g.729a	17.4	34.8	69.5	139.0	278.0	556.0
iLBC	18.4	36.8	73.5	147.0	294.0	588.0
GSM	22.4	44.8	89.5	179.0	358.0	716.0
g.726	41.4	82.8	165.5	331.0	662.0	1324.0
g.711	73.4	146.8	293.5	587.0	1174.0	2348.0

Tab. 5.3 Uso de banda por chamadas agregadas (IAX normal)

Protocolo IAX2 em modo trunked (Kbps)

Protocolo	1	2	4	8	16	32
lpc10	14.2	20.7	29.7	47.7	83.7	155.7
g.723.1	17.8	27.9	44.1	76.5	141.3	270.9
g.729a	19.7	31.7	51.7	91.7	171.7	331.7
iLBC	20.7	33.7	55.7	99.7	187.7	363.7
GSM	24.7	41.7	71.7	131.7	251.7	491.7
g.726	43.7	79.7	147.7	283.7	555.7	1099.7
g.711	75.7	143.7	275.7	539.7	1067.7	2123.7

Tab. 5.4 Uso de banda por chamadas agregadas (IAX trunked)

Protocolo cRTP (RTP compactado)³

Protocolo	1	2	4	8	16	32
g.723.1(t)	6.4	12.8	25.6	51.2	102.4	208.4

Tab. 5.5 Uso de banda usando compressed RTP (para efeito de comparação)

³ O cálculo feito (g.723.1 “t” de teórico), mostra que a banda utilizada é menor. Houve um erro de avaliação do cabeçalho no estudo que já foi retirado da Internet.

Protocolo RTP (sem compactação)

Protocolo	1	2	4	8	16	32
lpc10	13.4	26.9	53.8	107.5	215.0	430.0
g.723.1(t)	16.0	32.0	64.0	128.0	256.0	512.0
g.723.1	17.0	34.1	68.2	136.3	272.6	545.2
g.729a	18.9	37.9	75.8	151.5	303.0	606.0
iLBC	19.9	39.9	79.8	159.5	319.0	638.0
GSM	23.9	47.9	95.8	191.5	383.0	766.0
g.726	42.9	85.9	171.8	343.5	687.0	1374.0
g.711	74.9	149.9	299.8	599.5	1199.0	2398.0

Tab. 5.6 Uso de banda usando RTP sem compactação.

Pode-se notar que o RTP compactado é muito eficiente em termos de banda e que mesmo o IAX trunked precisa de muitos canais para chegar à mesma eficiência. Um detalhe é que cRTP só está disponível em linhas dedicadas ou frame-relay, não é possível fazer cRTP (pelo menos não até o momento) em redes Ethernet ou pela Internet, isto traz uma grande vantagem ao IAX trunked nestas topologias.

5.5 Nomenclatura dos canais**5.5.1 Formato de uma conexão de saída.**

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

<user>	Identificação do usuário no ponto remoto, ou nome do cliente configurado no iax.conf (opcional)
<secret>	Senha (opcional). Como alternativa pode ser o nome do arquivo de uma chave RSA sem a extensão (.key, ou .pub), e dentro de [chaves] como por ex. [onomedoarquivo]
<peer>	Nome do servidor ao qual se conectar
<portno>	Número da porta para a conexão no servidor(opcional).
<exten>	Extensão no servidor Asterisk remoto(opcional) <context>: Contexto para usar no servidor asterisk remoto (opcional)
<options>	A única opção disponível é 'a' que significa pedido para autoresponder.

5.5.2 Exemplos de canais de saída:

IAX/mark:asdf@myserver/6275@default	chama "myserver" usando "mark" como nome do usuário e adsf como senha e pede a extensão 6275 no contexto default.
IAX/iaxphone/s/a	chama o "iaxphone" pedindo resposta imediata

IAX/john:[johnrsa]@somewhere.com	chama somewhere.com, usando john como nome do usuário e uma chave RSA para autenticação. O formato de um nome de canal IAX usado para uma conexão de entrada é simplesmente:
----------------------------------	--

5.5.3 Formato de uma conexão de entrada

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

<username>	O nome do usuário, se conhecido
<host>	O host para se conectar
<callno>	O número de chamada local

5.5.4 Exemplo de canais de entrada

IAX[mark@192.168.0.1]/14	Chama o número 14 do usuário “mark” no ip 192.168.0.1
IAX[192.168.10.1]/13	Chama o número 13 do ip 192.168.10.1

5.6 Cenários de uso

5.6.1 Servidor IAX:

O Asterisk suporta telefones IP baseados em IAX como o SNOM e o AIXy e também softfones como o Firefly e Gnophone. Para configurar um telefone IAX basta configurar o arquivo iax.conf e o próprio telefone.

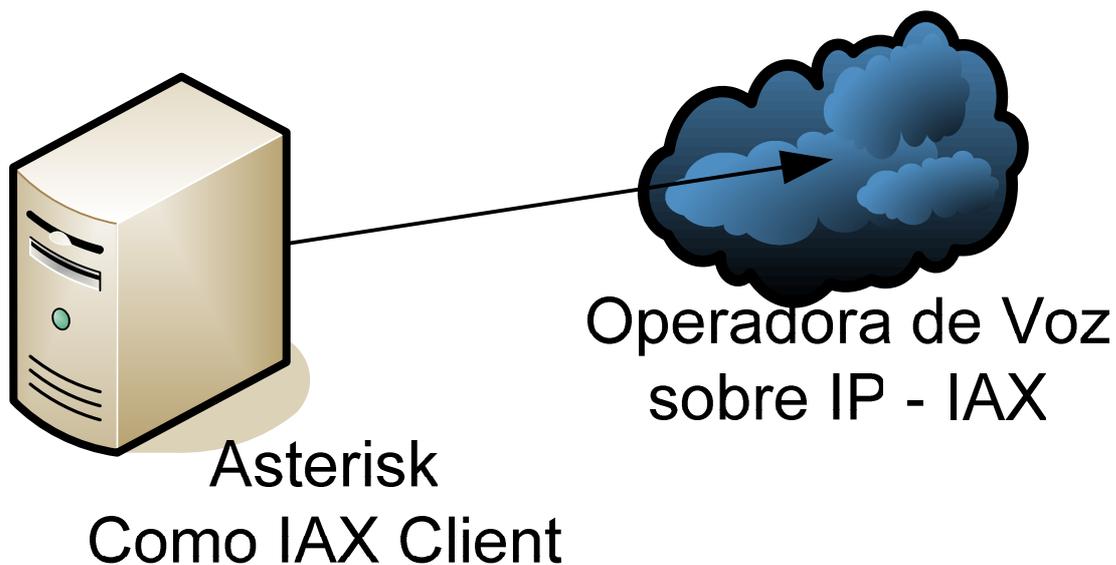
Exemplo de configuração de um telefone IAX:

```
[8580]
type=friend
context=default
auth=md5
secret=senha
notransfer=0
caller id=8580
host=dynamic
allow=gsm
```

5.6.2 Cliente IAX

Neste cenário é possível se conectar a um provedor de telefonia IAX como o Free World Dialup, www.freeworlddialup.com. Na verdade, é possível se conectar ao FWD usando SIP e IAX, por default a conexão é padrão SIP, mas IAX pode ser ativado. Existe também o IAXTEL. Uma lista de provedores IP no mundo pode ser encontrada em:

<http://www.voip-info.org/wiki-VOIP+Service+Providers>.



5.6.3 Como fazer para discar para um provedor

Para discar é preciso apenas usar o comando Dial(), usando a nomenclatura de canal apropriada.

Exemplo: Ligando para o ramal de serviço 612 no fwd.

Edite o arquivo extensions.conf e inclua a seguinte linha no seu contexto default.

```
exten=>612, 1,Dial(IAX2/621538:senha@iax2.fwdnet.net/612,20,r)
```

Nada mais é necessário se o que você quer é simplesmente discar para um número dentro do provedor.

5.6.4 Abreviando os comandos

Eventualmente, você não vai querer todas estas senhas no plano de discagem. Você pode criar entradas no arquivo `iax.conf` que vão simplificar o processo de configuração.

No arquivo `iax.conf` use:

```
[621538]
type=peer
secret=senha
host=iax2.fwdnet.net
```

Agora no arquivo `extensions.conf` você pode usar a forma abreviada.

```
exten=>612, 1,Dial(IAX2/621538/612,20,r)
```

5.6.5 Como fazer para receber uma ligação

Para receber uma ligação é necessário se registrar no provedor para que ele saiba em que endereço nosso servidor se encontra. Além disso, é preciso criar uma entrada do tipo “user” para que possa ser feita a recepção da chamada.

Na seção `[general]` do arquivo `iax.conf` use:

```
register=>621538:senha@iax2.fwdnet.net
```

A ligação será recebida na extensão ‘s’ do contexto default.

Na seção `[iaxfwd]` do arquivo `iax.conf` use (está no exemplo do `iax.conf` é só retirar o comentário). Isto é necessário para que ocorra a autenticação. Através do uso de chave pública você pode ter certeza de que a ligação está sendo recebida do “freeworlddialup”. Se alguém tentar usar este mesmo caminho as chaves de autenticação não vão bater.

```
[iaxfwd]
type=user
context=default
auth=rsa
inkeys=freeworlddialup
```

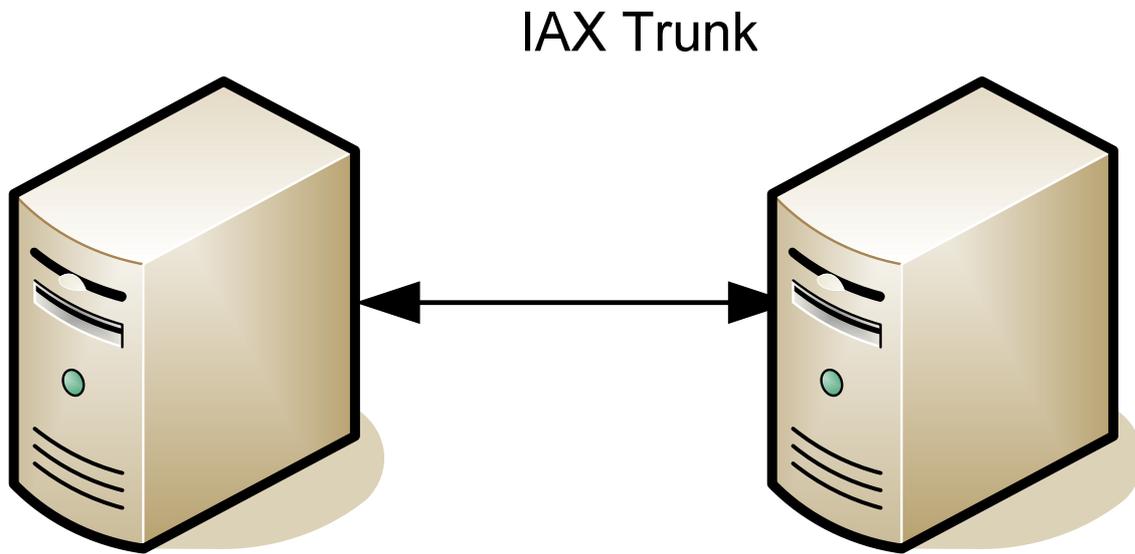
Use o seguinte comando para ver se o registro ocorreu com sucesso.

```
CLI>iax2 show registry
```

Dentro do arquivo `extensions.conf` é preciso receber esta ligação e tratá-la. A extensão 's' no contexto `[default]` tem de estar configurada.

5.6.6 Trunk IAX

Dois servidores Asterisk podem ser ligados usando o protocolo IAX2.



Interligação entre dois servidores
Asterisk usando IAX

5.6.7 Como configurar um trunk IAX

Para configurar um trunk IAX, vamos usar o conceito de mestre-escravo para tornar mais fácil o entendimento.

Os seguintes passos têm de ser feitos para que o servidor escravo possa fazer e receber ligações do servidor mestre. Para simplificar o exemplo colocamos todas as ligações no contexto `[default]`. Mais tarde neste material você vai aprender a colocar as chamadas em contextos mais apropriados.

1. Configurar o arquivo iax.conf do servidor escravo para:
 - 1.1. Se registrar no mestre.
 - 1.2. Receber chamadas do mestre.
2. Configurar o arquivo iax.conf do servidor mestre para:
 - 2.1. Receber chamadas do escravo.
 - 2.2. Se registrar no escravo.
3. Configurar o arquivo extensions.conf do escravo:
 - 3.1. Para discar para o mestre.
4. Configurar o arquivo extensions.conf do mestre
 - 4.1. Para discar para o escravo.

Vamos supor para o exemplo abaixo que **todos os ramais no servidor mestre começam com 41xx e todos os ramais do escravo começam com 42xx.**

Passo 1 - iax.conf do servidor escravo

1.1 Registrar no mestre

No arquivo iax.conf

```
register=>mestre:senha@mestre.net.br
```

1.2 Receber chamadas do mestre

No arquivo iax.conf

```
[mestre]
type=user
auth=plaintext
context-default
secret=senha
host=dynamic
callerid='master'
trunk=yes
notransfer=yes
```

```
[mestre]
type=user
auth=plaintext
context-default
secret=senha
callerid='master'
trunk=yes
```

```
nottransfer=yes
```

```
[4100]  
type=friend  
auth=plaintext  
context-default  
secret=senha  
callerid='master'
```

```
[4101]  
..... ; Definição dos outros canais IAX (Ramais)
```

Passo 2. Configurar o arquivo iax.conf do mestre

2.1 Receber e fazer chamados do escravo e o registro

```
[escravo]  
type=user  
auth=plaintext  
context-default  
secret=senha  
host=dynamic  
callerid='escravo'  
trunk=yes  
nottransfer=yes
```

```
[escravo]  
type=user  
auth=plaintext  
context-default  
secret=senha  
callerid='escravo'  
trunk=yes  
nottransfer=yes
```

```
[4200]  
type=friend  
auth=plaintext  
context-default  
secret=senha  
callerid='master'
```

```
[4201]  
..... ; Definição dos outros canais IAX (Ramais)
```

2.2 Se registrar no escravo

```
register=>escravo:senha@escravo.net.br
```

Passo 3 – Configurar o arquivo extensions.conf no servidor escravo.

```
[default]
exten => _41XX,Dial(IAX2/master/${EXTEN})

exten=>_42XX,Dial(IAX2/${EXTEN})
```

Passo 4 – Configurar o arquivo extensions.conf no servidor mestre.

```
[default]
exten => _42XX,Dial(IAX2/master/${EXTEN})

exten=>_41XX, Dial(IAX2/${EXTEN})
```

5.7 Autenticação no IAX

Vamos analisar a autenticação do IAX do ponto de vista prático e aprender a escolher as melhores opções dependendo das necessidades de segurança da configuração.

5.7.1 Conexões de entrada

Quando o Asterisk recebe uma conexão de entrada, a informação da chamada inicial pode incluir um nome de usuário (Do campo **username**) ou não. Além disso, a conexão de entrada tem um endereço IP que o Asterisk usa para a autenticação também.

Se o nome do usuário é fornecido, o Asterisk faz o seguinte:

1. Pesquisa o iax.conf para uma entrada “type-user” (ou “type=friend”) com a seção nome ([username]); se não encontra, recusa a conexão.
2. Se a entrada encontrada tem configurações do tipo “deny/allow” (negar/permitir), compara o endereço IP do originador da chamada. Se a conexão não for permitida, recusa a conexão.
3. Faz a checagem da senha (secret) (plaintext, md5 ou rsa); se falha, recusa a conexão.

4. Aceita a conexão e envia para o originador para o contexto especifica na configuração *context=* da entrada no arquivo *iax.conf*.

Se um nome de usuário não é fornecido, o Asterisk faz o seguinte:

1. Pesquisa por um “*type=user*” (ou *type=friend*) no arquivo *iax.conf* sem um segredo especificado e também nas restrições do tipo “*deny/allow*”. Se uma entrada é encontrada, aceita a conexão e usa o nome da entrada do *iax.conf* como o nome do usuário se conectando.
2. Pesquisa por um “*type=user*” (ou *type=friend*) no *iax.conf* com um segredo ou chave RSA especificado e também verifica restrições “*deny/allow*” . Se uma entrada é encontrada, tenta autenticar o chamador usando o segredo especificado ou a chave, e se ele passa, aceita a conexão e usa o nome encontrado no *iax.conf* como nome do usuário.

Suponha que seu arquivo *iax.conf* tenha as seguintes entradas:

```
[guest]
type=user
context=guest

[iaxtel]
type=user
context=incoming
auth=rsa
inkeys=iaxtel

[iax-gateway]
type=friend
allow=192.168.0.1
context=incoming
host=192.168.0.1

[iax-friend]
type=user
secret=this_is_secret
auth=md5
context=incoming
```

Se uma chamada de entrada tem um nome de usuário especificado de:

- *guest*

- iaxtel
- iax-gateway
- iax-friend

Então o Asterisk irá tentar autenticar a conexão usando apenas a entrada correspondente em `iax.conf`. Se qualquer outro nome de usuário for especificado, a conexão será recusada.

Se nenhum usuário tiver sido especificado, o Asterisk irá autenticar a conexão como usuário `guest` (convidado). Entretanto se você não tiver um usuário `guest` no seu arquivo `iax.conf`, o usuário que está se conectando, pode especificar o segredo associado com qualquer outro `iax.conf` que não tenha um endereço IP restrito. Em outras palavras, se você não tem uma entrada `guest` no seu `iax.conf`, você tem várias entradas com senhas que podem ser adivinhadas e que vão permitir que o usuário se conecte no sistema.

Usando chaves RSA como seus segredos é uma forma de evitar este problema sem criar um usuário `guest`. Entradas RSA não são “adivinháveis”, outro método é usar restrições de IP em tantas entradas quanto possíveis.

5.7.2 Conexões de saída

Conexões de saída podem pegar suas informações de autenticação de:

- A descrição do canal IAX2 passado pelo comando `Dial()`.
- Uma entrada “`type=peer` ou `friend`” no arquivo `iax.conf`.
- Uma combinação dos dois.

Suponha que seu arquivo iax.conf tenha as seguintes entradas:

```
[iaxtel-outbound]
type=peer
username=iaxtel_username
secret=iaxtel_secret
host=iaxtel.com

[iax-gateway]
type=friend
allow=192.168.0.1
context=incoming
host=192.168.0.1
```

Então o comando:

```
Dial(IAX2/iaxtel-outbound/1234)
```

Conectaria ao host iaxtel.com, usando o usuário e senha especificados no arquivo iax.conf para autenticação.

Já o comando:

```
Dial(IAX2/user2:secret2@iaxtel-outbound/1234)
```

Também irá se conectar com o host iaxtel.com, mas irão especificar user2:secret2 como nome e senha respectivamente e isto sobrescreve as entradas no iax.conf.

```
Dial(IAX2/iaxtel.com/1234)
```

Novamente a conexão seria feita ao host iaxtel.com, mas nenhum nome de usuário foi especificado. Como nenhuma das entradas no arquivo iax.conf bate com a descrição do canal iaxtel.com a conexão é recusada.

```
Dial(IAX2/iax-gateway/5678)
```

Se conectaria ao host 192.168.0.1, sem especificar nenhum nome, e se o host pedir um segredo, nenhum segredo será fornecido. Presumivelmente este tipo de entrada seria usada para conexões entre hosts com alto grau de confiança.

5.8 Configuração do arquivo iax.conf

Como todo o resto no Asterisk, a configuração do IAX é feita no arquivo `/etc/asterisk/iax.conf`.

O arquivo de configuração do IAX é uma coleção de seções, cada qual representa uma entidade dentro do escopo do IAX (A exceção da seção geral).

A primeira seção é tipicamente a seção geral. Nesta área, um número de parâmetros que afetam todo o sistema pode ser configurado. Especificamente os codecs default, portas, endereços, comportamento do jitter, bits de TOS e registros.

5.8.1 Configuração da seção geral

A primeira linha da seção geral é sempre:

[general]

Seguindo a primeira linha está um número de outras possibilidades:

<code>port = <portnum></code>	Isto configure a porta em que o IAX vai se ligar. A porta default é 5036. É recomendado manter este valor.
<code>bindaddr = <ipaddr></code>	Isto permite ligar o IAX à um endereço IP específico ao invés de ligar o * à todos os endereços.
<code>bandwidth = [low medium high]</code>	A seleção de banda passante inicializa a seleção de codecs a valores apropriados para uma dada banda. Escolhendo "High" habilita todos os codecs e é recomendado para conexões de 10Mbps para cima. Escolhendo "médium" elimina os codecs U-law A-law deixando apenas os codecs com 32Kbps ou menos (Com MP3 como um caso especial). Isto pode ser usado em conexões de banda larga se desejado. O "low" elimina o ADPCM e MP3, deixando apenas o G723, GSM e LPC10.
<code>allow/disallow=[gsm lpc10 g723.1 adpcm ulaw alaw mp3 linear all]</code>	O "allow" e "disallow" permite fazer uma sintonia fina na seleção de codecs além da banda passante inicial em uma base codec por codec. A configuração recomendada é selecionar "low" e depois desabilitar LPC10. O LPC10 tem a voz robotizada.
<code>jitterbuffer = [yes no]</code> <code>dropcount = <dropamount></code> <code>maxjitterbuffer = <max></code>	Estes parâmetros controlam a operação do buffer de jitter (Buffer de variação no atraso dos pacotes). O buffer de jitter deve sempre estar habilitado a

maxexcessbuffer = <max>	menos que você saiba que todas as suas conexões estão na LAN. A quantidade de drops é o número máximo de pacotes de voz que você vai permitir que sejam descartados. Valores úteis são 3-10. O “maxjitterbuffer” é a quantidade máxima de buffer de jitter que pode ser usado. O “maxexcessbuffer” é a quantidade máxima de excesso no buffer de jitter que é permitido antes que o buffer de jitter seja lentamente comprimido para eliminar a latência.
accountcode = <code> amaflags = [default omit billing documentation]:	Estes parâmetros afetam a geração do detalhamento de chamadas. O primeiro configura o código de contabilização para registros recebidos com o IAX. O código de contabilização pode sobrescrever a base por usuário para chamadas entrantes. O “amaflags” controla como o registro é etiquetado (“omit” faz com que nenhum registro seja escrito. “billing” e “documentation” etiquetam os registros como registros para cobrança e documentação respectivamente e “default” seleciona o default do sistema).
tos=[lowdelay throughput reliability min cost none]	O IAX pode opcionalmente configurar o bit TOS (Type of Service) do cabeçalho IP. Isto ajuda a performance no roteamento. O valor recomendado é “lowdelay”, que muitos roteadores vão reconhecer e dar prioridade melhorando a qualidade da voz.
register=><name>[:<secret>]@<host>[:port]:	Várias entradas como esta podem ser colocadas na seção geral. O registro permite ao Asterisk notificar um servidor Asterisk remoto (com um endereço fixo) qual seu endereço atual. Para que o registro funcione, o Asterisk remoto vai precisar ter uma entrada com o mesmo nome to tipo “dynamic peer” (e segredo (secret) se fornecido). O nome é um campo obrigatório e é o nome do “peer” remoto a quem nós desejamos nos identificar. Entretanto se em chaves([]’s) então é interpretado como o nome de uma chave RSA a ser usada. Neste caso o Asterisk deve ter a chave privada e o servidor remoto terá a correspondente chave pública (/var/lib/asterisk/keys/<name>.key). O campo “host” é obrigatório e é o nome do host ou endereço IP do servidor Asterisk remoto. A especificação da porta é opcional.

5.8.2 Configuração dos clientes IAX

[identifier]	A seção inicia com um identificador em chaves ([]’s). O identificador deve ser uma string alfanumérica.
--------------	---

<code>type=[user peer friend]</code>	Esta linha diz ao asterisk como interpretar esta entidade. Usuários são dispositivos que se conectam à nós, enquanto “peers” são pessoas às quais nos conectamos, e “friend” é um atalho para criar um “user” e um “peer” com informações idênticas.
--------------------------------------	--

5.8.3 Campos do tipo “User”:

<code>context = <context></code>	Uma ou mais linhas de contexto podem ser especificadas pelo usuário, deste modo dando ao usuário a possibilidade de colocar as chamadas em dados contextos. Contextos são usados pelo Asterisk para dividir o plano de discagem em unidades lógicas, cada uma com a habilidade de ter números interpretados de forma diferente, ter seu próprio modelo de segurança, manuseio auxiliar da comutação e incluir outros contextos. A maioria dos usuários usa o contexto “default”.
<code>permit = <ipaddr>/<netmask>deny = <ipaddr>/<netmask></code>	<p>As regras de permitir e negar (permit e deny) podem ser aplicadas à usuários, permitindo a eles se conectar de determinados endereços IP e não de outros. As regras são interpretadas na seqüência e são todas avaliadas em um dado endereço IP, com o resultado final sendo a decisão (Diferente das listas de controle de acesso Cisco e da maior parte dos Firewalls). Por exemplo:</p> <pre>permit=0.0.0.0/0.0.0.0 deny=192.168.0.0/255.255.255.0</pre> <p>Irá negar qualquer um em 192.168.0.0 com mascara de 24 bits (classe C)</p> <p>e:</p> <pre>deny=192.168.0.0/255.255.255.0 permit=0.0.0.0/0.0.0.0</pre> <p>Não irá negar qualquer um já que a regra final permite todos, deste modo sobrepondo a negação. Se nenhuma regra estiver listada, qualquer um pode se conectar de qualquer lugar.</p>
<code>callerid = <callerid></code>	Você pode sobrescrever o identificador de chamada passado pelo usuário para você (Se ele escolher enviar) de maneira

	que ele sempre esteja correto do ponto de vista do seu servidor.
auth = [md5 plaintext rsa]	Você pode selecionar que métodos de autenticação são permitidos. Múltiplos métodos podem ser especificados, separado por vírgulas. Se md5 ou texto simples (plaintext) é selecionado um segredo (secret) deve ser fornecido. Se autenticação RSA for especificada, então uma ou mais nomes de chaves deve ser especificado com "inkeys". Se nenhum segredo e nenhum método de autenticação for especificado, então nenhuma autenticação é necessária.
secret = <secret>	A linha "secret" especifica o segredo compartilhado para os métodos md5 ou autenticação em texto simples. Não sugerimos a autenticação em texto simples exceto para debugging
inkeys = key1[:key2...]	A linha "inkeys" especifica que chaves nós podemos usar para autenticar um "peer" remoto. Os arquivos de chave ficam em /var/lib/asterisk/keys/<name>.pub e são chaves públicas.

5.8.4 Configuração de "peers" IAX

allow=[gsm pc10 g723.1 adpcm ulaw alaw mp3 slin all] disallow=[gsm pc10 g723.1 adpcm ulaw alaw mp3 slin all]	A linha "inkeys" especifica que chaves nós podemos usar para autenticar um "peer" remoto. Os arquivos de chave ficam em /var/lib/asterisk/keys/<name>.pub e são chaves públicas.
host=[<ipaddr> dynamic]	A linha host é o nome do host ou endereço IP do host remoto, ou pode ser a palavra "dynamic" que significa que o host irá se registrar conosco.
defaultip=<ipaddr>	Se o host usa registro dinâmico, o Asterisk ainda pode ter o endereço IP default para usar quando registro não foi feito ou expirou.

5.9 Exemplo: Arquivo de configuração IAX

```
[general]
port=5036

bandwidth=high
disallow=all
;allow=gsm

tos=lowdelay
```

```
[guest]
type=user
context=from-iaxtel

[nufone]
type=friend
secret=somedumbpassword
context=NANPA
host=switch-1.nufone.net
disallow=all
allow=gsm

[sjc1]
type=friend
secret=password123
auth=plaintext
host=64.162.134.251
context=intern
allow=all
```

5.10 Comandos de console

iax2 debug	habilita IAX debugging
iax2 no debug	Desabilita IAX debugging
iax2 set jitter	Seta o buffer de jitter IAX
iax2 show cachê	Mostra o plano de discagem do IAX na memória
iax2 show channels	Mostra os canais ativos do IAX
iax2 show peers	Mostra os pares definidos do IAX
iax2 show registry	Mostra o status de registro do IAX
iax2 show stats	Mostra estatísticas do IAX
iax2 show users	Mostra os usuários IAX definidos
iax2 trunk debug	Pede um debug do trunk IAX trunk
init keys	Inicializa as chaves RSA
show keys	Mostra as informações de chave RSA

5.11 Sumário

Neste capítulo você aprendeu a diferenciar os pontos fortes e fracos do IAX. Foi possível demonstrar os cenários de uso do Asterisk como client usando o Firefly e entroncamento de dois PABX com IAX trunked. Foi demonstrado que o modo trunked economiza banda enviando diversas ligações em um mesmo pacote evitando a criação de novos pacotes e cabeçalhos. Foi possível ver que a quantidade de banda usada depende do codec, bem como das configurações. Aprendeu a configurar o arquivo iax.conf para se conectar a

telefones e provedores e pode usar alguns dos comandos de console do Asterisk relacionados ao Asterisk.

5.12 Questionário

1. Podemos citar como principais benefícios do IAX a economia de banda e facilidade de passar por Firewalls com NAT.

- Correto
- Incorreto

2. No protocolo IAX os canais de sinalização e mídia passam separados. Esta afirmação é:

- Correta
- Incorreta

3. O IAX emprega os seguintes tipos de frames

- Frame Completo
- Frame Incompleto
- Mini-Frame
- Trunked Frame

4. A banda passante usada pelo protocolo IAX é a soma da carga de voz (payload) mais os cabeçalhos (Marque todas as que se aplicam)

- IP
- UDP
- IAX
- RTP
- cRTP

5. Comparando o protocolo IAX e o protocolo cRTP (compressed RTP) podemos afirmar que em uma rede baseada na transmissão pela Internet como o ADSL (Marque uma opção)

- O IAX2 é sempre a melhor opção.
- O cRTP não pode ser implantado neste tipo de circuito.
- O cRTP ocupa menos banda por isto é a melhor opção.
- A partir de 16 linhas o IAX2 passa a ser a melhor opção.

6. Quando o IAX é usado no modo trunk, apenas um cabeçalho é usado para transmitir múltiplas ligações. A afirmação acima está:

- Correta
- Incorreta

7. O protocolo IAX2 é o mais comum para conectar provedores de telefonia IP, pois passa fácil pelo NAT. A afirmação acima está

- Correta
- Incorreta

8. Em um canal IAX como o abaixo, a opção <secret> pode ser tanto uma senha como uma _____.

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

9. O contexto é adicionado para cada cliente IAX, isto permite que diferentes clientes possuam diferentes contextos. Pode-se pensar em contexto como uma classe de ramal onde o cliente será colocado. A afirmação está

- Correta
- Incorreta

10. O comando IAX2 show registry mostra informações sobre:

- Os usuários registrados
- Os provedores ao qual o Asterisk se conectou.

Página deixada em branco intencionalmente

O protocolo SIP e o Asterisk

6.1 Objetivos

Ao final deste capítulo você deverá ser capaz de:

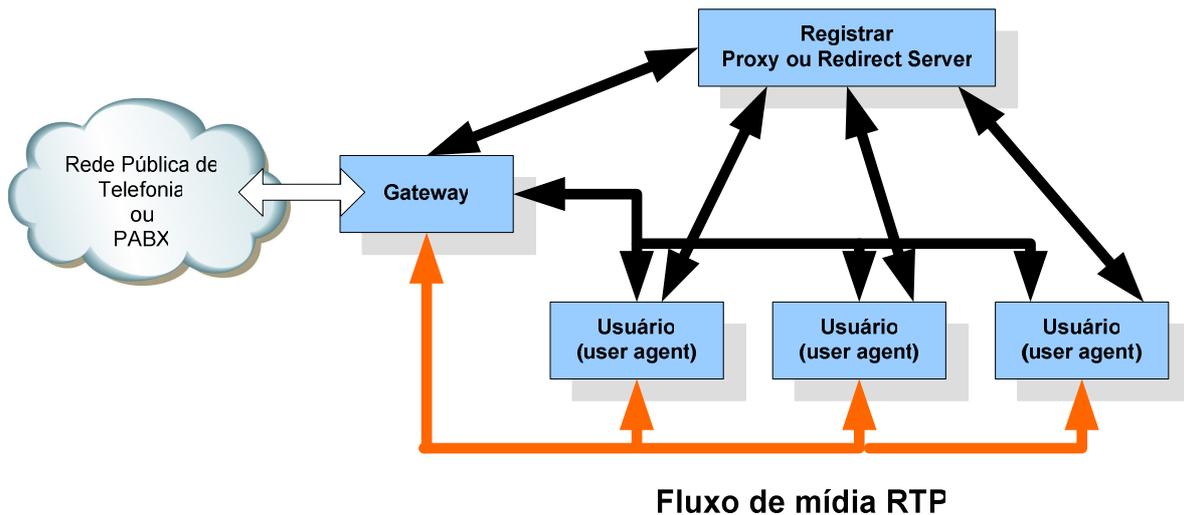
- Entender a teoria de operação do SIP
- Entender os pontos fortes e fracos do SIP
- Descrever os cenários de uso do SIP
- Mostrar como passar por problemas com NAT
- Configurar um telefone SIP
- Configurar um provedor de telefonia IP SIP

6.2 Visão geral

O SIP (Session Initiated Protocol) é um protocolo baseado em texto, similar ao HTTP e SMTP, desenhado para iniciar, manter e terminar sessões de comunicação interativa entre usuários. Tais sessões incluem, voz, vídeo, chat, jogos interativos e realidade virtual. Foi definido pela IETF e vem se tornando o padrão de fato em telefonia IP.

6.3 Teoria da Operação do SIP

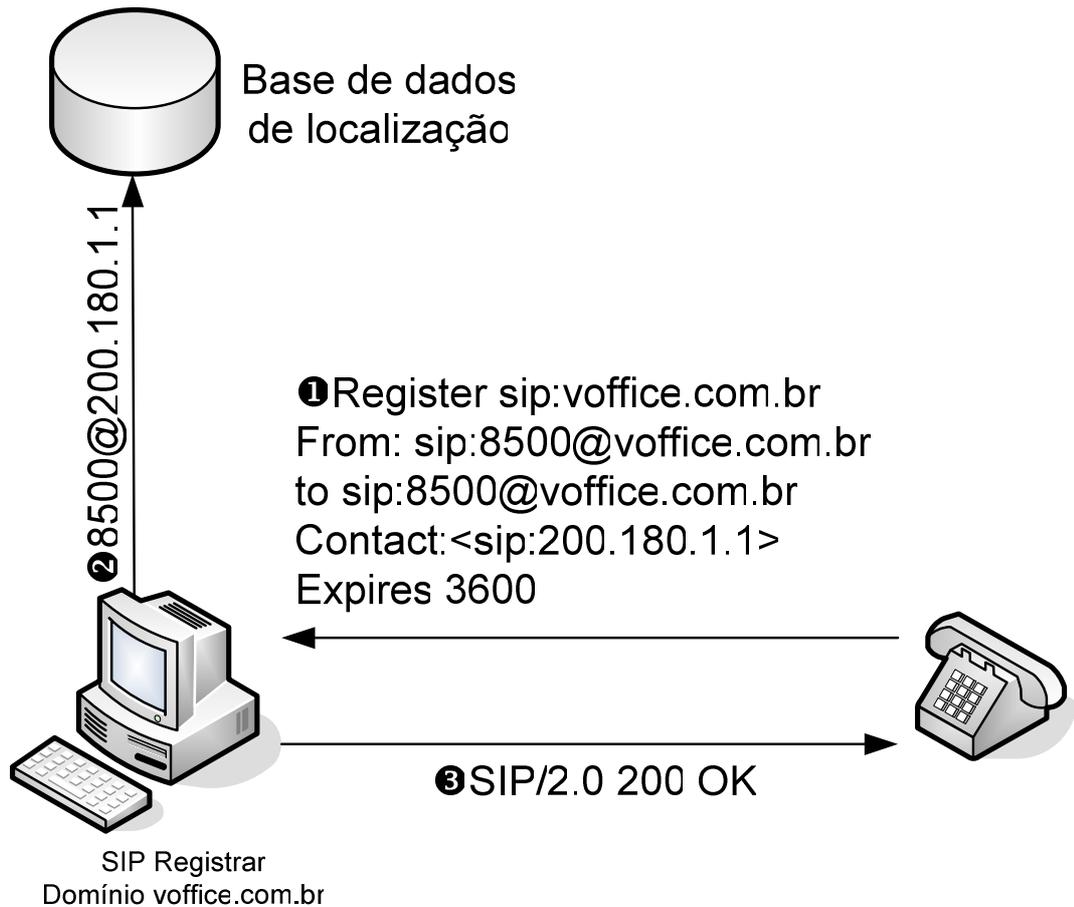
O SIP é um protocolo de sinalização de voz sobre IP que possui os seguintes componentes:



- **UAC (user agent client)** – cliente ou terminal que inicia a sinalização SIP.
- **UAS (user agent server)** – servidor que responde a sinalização SIP de um UAC.
- **UA (user agent)** – terminal de rede SIP (telefones SIP, ou gateway para outras redes), contém UAC e UAS.
- **Servidor Proxy** – Recebe pedidos de conexão de um UA e transfere ele para outro servidor proxy se a estação em particular não está em sua administração.
- **Servidor de Redirecionamento** – Recebe pedidos de conexão e envia-os de volta ao emissor incluindo os dados de destino ao invés de enviá-los diretamente à parte chamada.
- **Servidor de localização** – recebe pedidos de registro de um UA e atualiza a base de dados de terminais com eles.

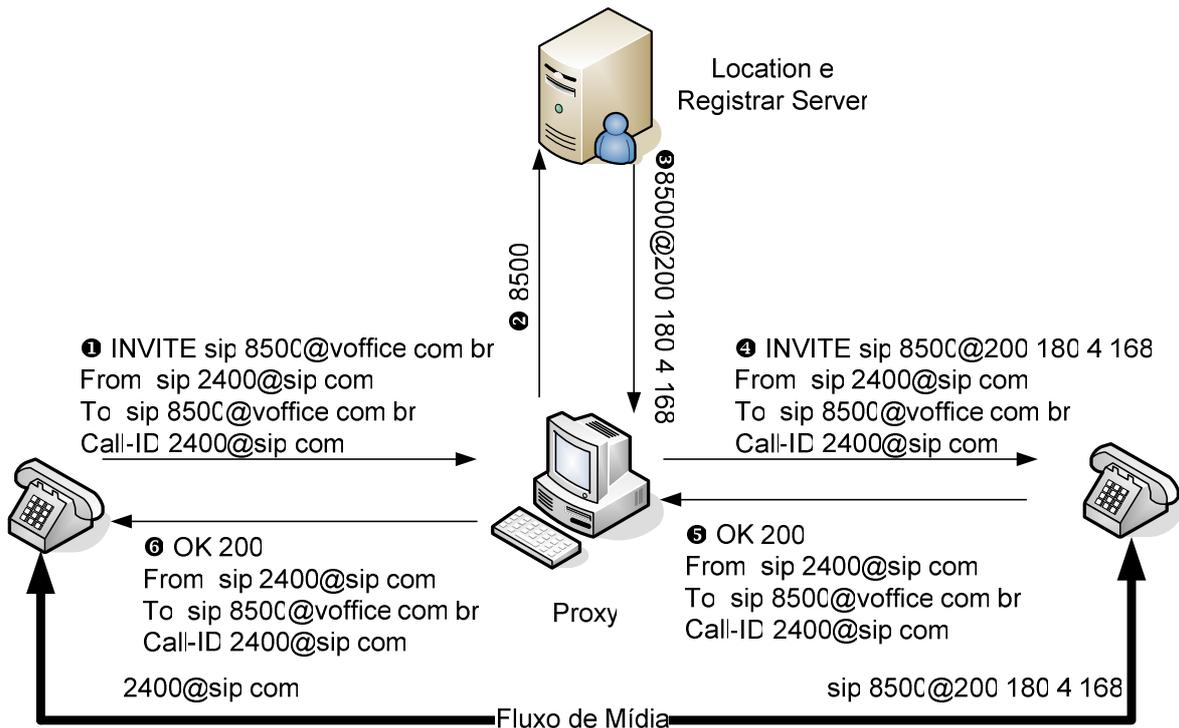
Todas as seções do servidor (Proxy, Redirect e Location) estão tipicamente disponíveis em uma única máquina física chamada proxy server, que é responsável pela manutenção da base de dados de clientes, estabelecimento de conexões, manutenção e término e redirecionamento de chamadas.

6.4 Processo de Registro do SIP

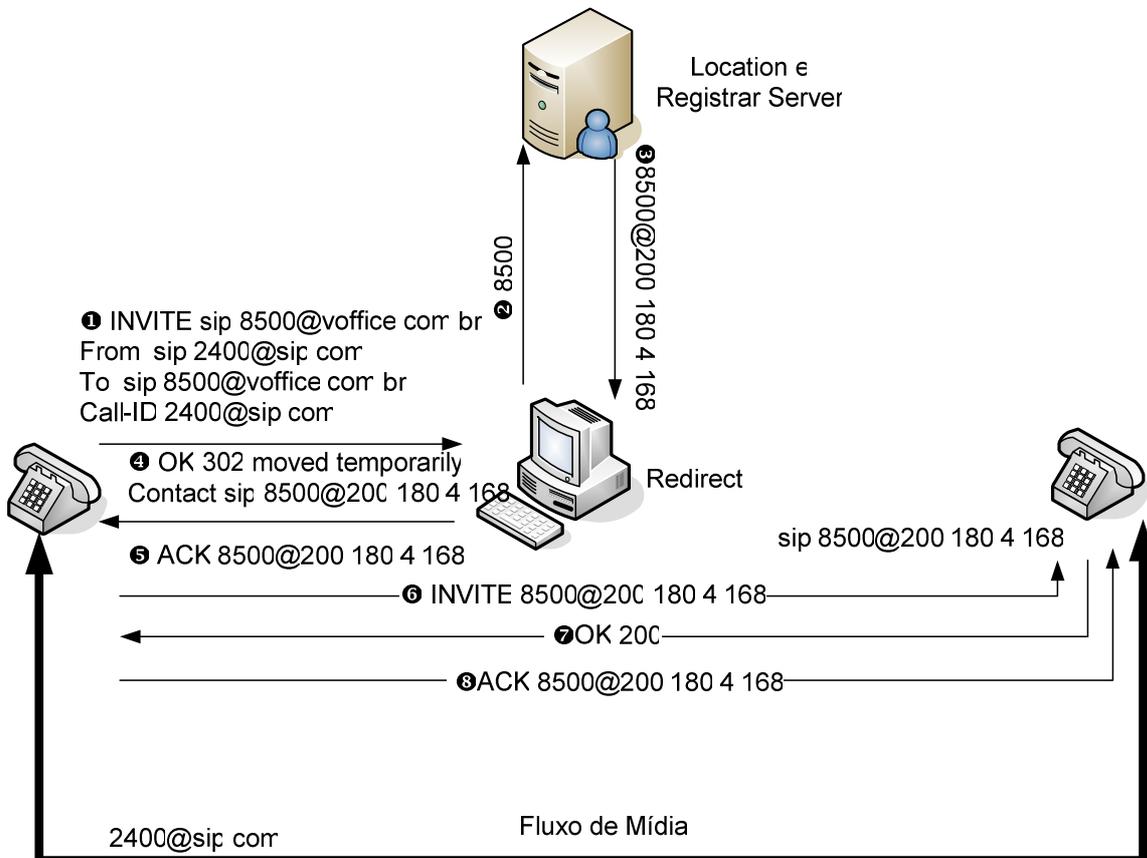


Antes que um telefone possa receber chamadas, ele precisa se registrar em uma base de localização. É neste local que o nome será associado ao endereço IP onde o telefone se encontra. No nosso caso usamos como nome o ramal 8500. Poderia ser também um endereço no formato sip:flavio@voffice.com.br.

6.5 Operação do SIP em modo proxy.



6.6 Operação em modo de redirect.



6.7 SIP no modo Asterisk

É importante ressaltar que o Asterisk não é nem um SIP Proxy nem um SIP Redirector. O Asterisk é um Media Gateway. Ele poderia ser mais bem descrito como um “back-to-back user agent”. Em outras palavras ele conecta dois canais SIP como se fossem canais de um PBX. É possível usar em conjunto com o Asterisk um SIP Proxy como o SIP Express Router <http://www.iptel.org/ser/>.

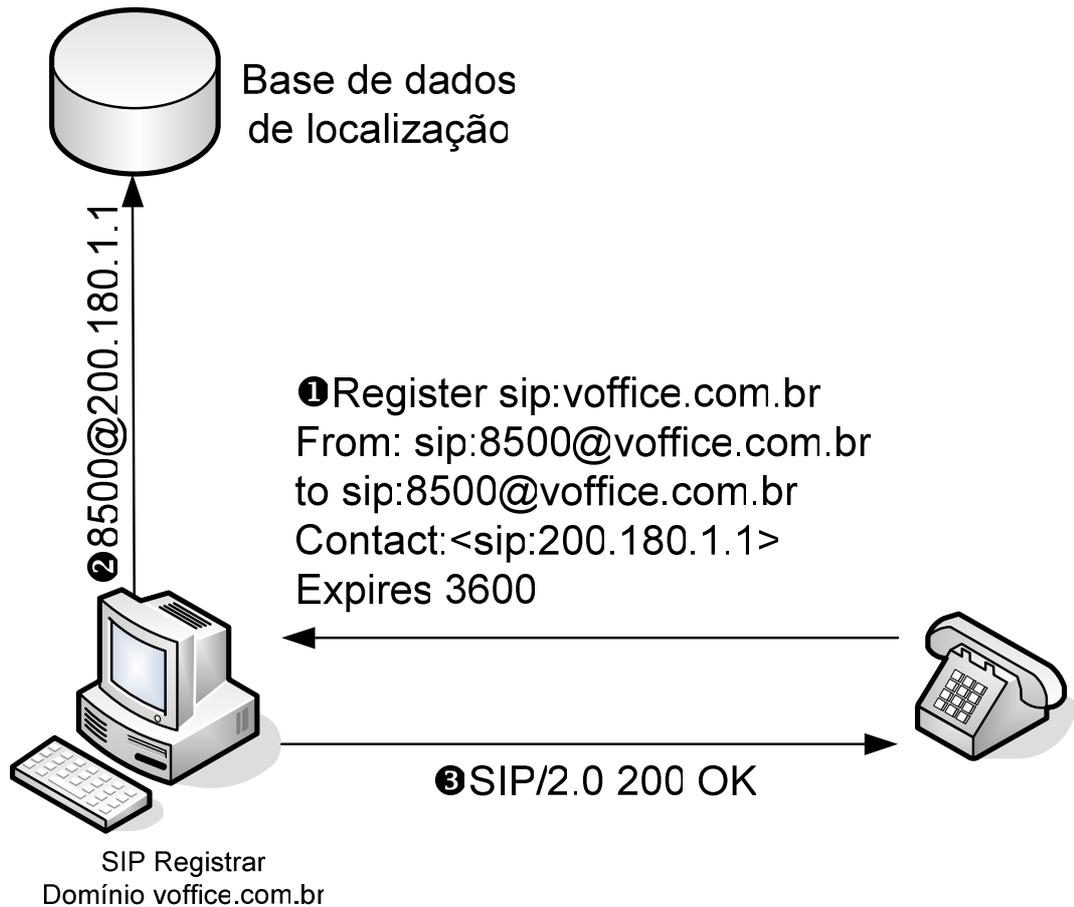
As mensagens básicas enviadas em um ambiente SIP são:

- INVITE – pedido de estabelecimento de conexão.
- ACK – reconhecimento do INVITE pelo receptor final da mensagem.
- BYE – término da conexão.
- CANCEL – término de uma conexão não estabelecida.
- REGISTER – registro do UA no SIP proxy.
- OPTIONS – pedido de opções do servidor

Respostas a mensagens do SIP são em formato texto como no protocolo http. Aqui estão as respostas mais importantes.

- 1XX – mensagens de informação (100–tentando, 180–campainha, 183–progresso).
- 2XX – pedido completado com sucesso (200 – OK).
- 3XX – encaminhamento de chamada, o pedido deve ser direcionado para outro lugar. (302 – temporariamente movido, 305 – use proxy).
- 4XX – erro (403 – Proibido).
- 5XX – erro de servidor (500 – Erro interno do servidor, 501 – Não implementado).
- 6XX – falha global (606 – Não aceitável).

6.4 Processo de Registro do SIP



Antes que um telefone possa receber chamadas, ele precisa se registrar em uma base de localização. É neste local que o nome será associado ao endereço IP onde o telefone se encontra. No nosso caso usamos como nome o ramal 8500. Poderia ser também um endereço no formato sip:flavio@voffice.com.br.

6.8.1.2 - Passo 1: Registrar o provedor (sip.conf)

```
register=>621538:password@fwd.fwdnet.net/4100
```

6.8.1.3 - Passo 2: Criar uma entrada do tipo [peer] para o provedor para simplificar a discagem (sip.conf).

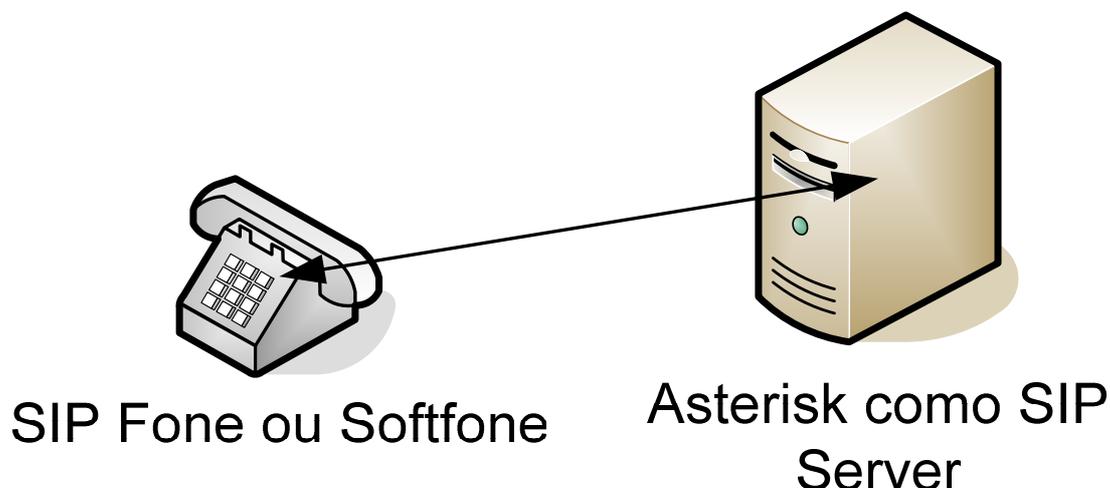
```
[provedorsip]
type=peer
context=default
secret=password
username=621538
host=fwd.pulver.com
fromuser=621538
fromdomain=fwd.pulver.com
nat=yes
```

6.8.1.4 - Passo 3: Criar uma rota de saída no plano de discagem.

Neste exemplo, vamos escolher o dígito 010 como rota de saída para o FWD. Para discar para o 610000, você deve discar 010610000. (Como se fosse uma nova operadora, “disque ‘10’ para o fwd”).

```
[fwd-out]
exten=>_010.,1,SetCIDNum(621538)
exten=>_010.,2,SetCIDName(Flavio Goncalves)
exten=>_010.,3,Dial(SIP/${EXTEN:3}@provedorsip)
exten=>_010.,4,Playback(invalid)
exten=>_010.,5,Hangup
```

6.8.2 Asterisk como um SIP server



SIP Server: Isto significa que clientes SIP (telefones, softones) registram para o servidor Asterisk e configuram sessões SIP com o servidor, chamadas e respostas a chamadas. O Asterisk não é um SIP Server com todos os recursos. Se você vai ter milhares de telefones SIP, você deveria usar o SER (SIP Express Router) e re-encaminhar as chamadas para o Asterisk para voicemail e acesso à rede pública.

Como configurar?

- Passo 1 – Configurar uma entrada no arquivo sip.conf para cada telefone.
- Passo 2 – Configurar o telefone.
- Passo 3 – Configurar a extensão no plano de discagem.

6.8.2.1 Passo 1 – Configurando o sip.conf

Telefone Grandstream:

```
[4101]
type=friend
context=from-sip
username=grandstream
callerid=Flávio E. Goncalves<8550>
host=dynamic
nat=no
canreinvite=yes
```

```
dtmfmode=info
mailbox=1234@default
disallow=all
allow=ulaw
allow=g729
```

Softphone XLITE da XTEN www.xten.com

```
[4102]
type=friend
username=xlite
callerid="Flavio E Goncalves"<8550>
host=dynamic
nat=yes
canreinvite=no
disallow=all
allow=gsm
allow=ulaw
```

Telefone da Cisco

```
[4103]
type=friend
username=cisco
secret=blah
nat=yes
host=dynamic
canreinvite=no
disallow=all
allow=ulaw
allow=g729
```

6.8.2.2 Passo 2 – Siga as instruções do seu telefone favorito para configurá-lo.

Lembre-se de que nome do usuário, senha, contexto e codec são os parâmetros mais importantes.

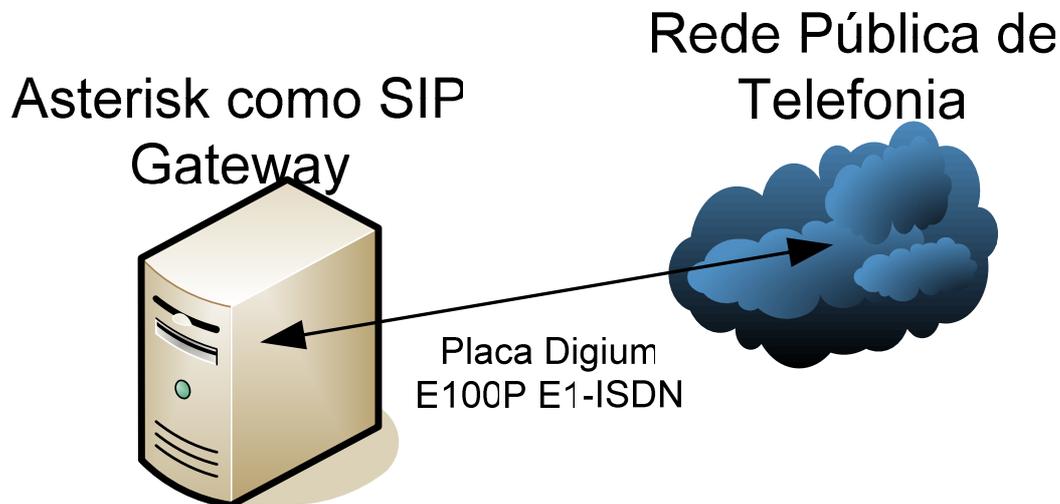
6.8.2.3 Passo 3 – Definir a extensão no arquivo sip.conf

Após você definir as contas de clients sip no sip.conf, você está apto à se logar no servidor Asterisk dos clientes e fazer e receber chamadas. Você precisa configurar as extensões em extensions.conf.

Exemplo:

```
exten=> _41XX,1, Dial(SIP/${EXTEN},10,t)
```

Se alguém chamar uma extensão iniciando em 41 mais dois dígitos, o cliente sip logado será discado de forma a receber a chamada.



SIP Gateway: O Asterisk atua como um gateway de mídia entre SIP, IAX, MGCP, H.323 e conexões a rede pública de telefonia. Como um exemplo, um servidor Asterisk pode ser conectado à uma rede ISDN para dar conectividade dos usuários com a rede pública.

6.9 Nomenclatura dos canais SIP

O formato do nome de um canal SIP usado para uma conexão de saída é:

O parâmetro identificador pode ser feito de três partes.

```
SIP/[exten@]peer[:portno]
```

peer: O nome do peer ao qual se conectar. Isto pode ser um dos abaixo:

- Um “peer” ou “friend” definido em sip.conf.
- Um endereço IP.
- Um nome de domínio. Para nomes de domínio o Asterisk irá primeiro olhar no registro DNS SRV para aquele domínio.

portno: A porta UDP a ser usada. Se omitido o Asterisk irá usar 5060.

exten: Se definido, então o asterisk irá requisitar ao “peer” que se conecte à extensão “exten”

Exemplos:

```
exten=>s,1,Dial(SIP/iphone)
exten=>s,1,Dial(SIP/info@voffice.com.br)
exten=>s,1,Dial(SIP/192.168.1.8:5060,20)
exten=>s,1,Dial(SIP/8500@sip.com:9876)
```

6.10 Arquivo de configuração sip.conf

Cada cliente do sip é identificado por um bloco de texto que se parece como o exemplo abaixo:

```
[xxx]
Type=xxx
Parametrô1=valor
Parâmetro2=valor
```

Onde xxx é o nome do usuário associado com o cliente SIP, ou é um nome arbitrário usado por outros arquivos de configuração para se referir, a este dispositivo SIP. Tipicamente se um telefone SIP tem um número de extensão de 123, então sua entrada correspondente neste arquivo irá iniciar com [123]. Note que você ainda tem de habilitar a extensão 123 no seu plano de discagem para alcançar este telefone.

A outra maneira em que as chamadas SIP de entrada batem com as seções [xxx] deste arquivo é examinar o endereço IP do pedido que está chegando e olhar para a seção peer [xxx] que tenha um valor de host correspondente. Se host=dynamic, então nenhuma correspondência será encontrada até que o cliente SIP esteja registrado.

6.10.1 Configuração da seção geral [general]

A seção geral do sip.conf inclui as seguintes variáveis:

allow = <codec>	Permite codecs na ordem de preferência (use DISALLOW=ALL primeiro, antes de permitir outros codecs)
disallow =all	Desabilita todos os codecs (configuração global)
autocretepeer =yes/no:	Se configurado, qualquer um estará apto a se logar como um peer (Sem checagem de credenciais, útil para operar com o SER).
bindaddr = 0.0.0.0	Endereço IP onde o serviços está instalado

	(Listen)
canreinvite = updateyesno	(configuração global)
context =default	Contexto default para a entradas de chamadas em extensions.conf.
defaultexpirey =120	Tempo padrão do registro de entrada e a saída.
externip =200.180.4.110	Endereço IP que será colocado em mensagens SIP, se o * estiver atrás de um dispositivo NAT.
localnet = 192.168.1.0/255.255.255.0	endereço local e máscara de rede
fromdomain = <domain>	Configura o “from” padrão como domínio nas mensagens SIP quando atuando como um cliente SIP
maxexpirey =3600	Tempo máximo de registro que iremos permitir.
nat =yesno	Configuração global
notifymime-type=text/plain	Permite sobrescrever o mime-type (mime = multipurpose internet mail extensions) no campo MWI NOTIFY usado nas mensagens on-line do voice-mail.
pedantic= yesno	Habilita a checagem, lenta, dos identificadores de chamada, cabeçalhos SIP com múltiplas linhas e cabeçalhos codificados em URI (uniform resource identifier)
port =<portno>	Porta TCP a qual se ligar
register => <username>@<sip client/peer id in sip.conf>/<extension>	Registra com um provedor SIP
srvlookup = yesno	Habilita DNS SRV lookups em chamadas
tos =<valor> lowdelay throughput reliability mincost none (valores numéricos também são aceitos, como tos=184.	Seta o campo tipo de serviço nos pacotes SIP. Quando usar o Asterisk em ambientes com Diffserv, tos=0xB8 seta os bits do Diffserv para EF (Expedited Forwarding).
videosupport =yesno useragent	Liga o suporte para vídeo no SIP Permite que o cabeçalho SIP “User-Agent” possa ser customizado.
trustpid =yesno	Se é possível confiar na identificação da parte remota
realm =meu realm	Muda a forma de autenticação do Asterisk(default) para sua própria. Requer Asterisk v1.x

6.10.2 Configurações do SIP – peers e clients

Estas variáveis podem ser configuradas para cada definição de peer SIP:

accountcode	Usuários podem estar associados à um código de contabilização.
amaflags=default omit bil ling documentation	Categorização de registros CDR (CDR – Registro detalhado de chamadas usado na bilhetagem).
callgroup	
canreinvite	Se o cliente é capaz de suportar SIP reinvides
context	Contexto no plano de discagem para as chamadas de saída deste cliente.

defaultip	Endereço Ip default deste cliente, se especificado host="dynamic". Usado se o cliente não registrou em qualquer outro endereço IP
dtmfmode	Como o cliente gerencia a sinalização DTMF
fromuser	Especifica o usuário para colocar no "from" ao invés do callerid (sobrescreve o callerid) quando colocando chamadas para outro peer (outro SIP proxy).
fromdomain=<domain>	Configura o domínio default "from:" nas mensagens SIP quando colocando chamadas para um "peer".
host=	Como encontrar o cliente: IP ou nome do host. Se você quiser que o telefone se registre, use a palavra dynamic ao invés do IP do host.
incominglimit e outgoinglimit	Limita o número de chamadas ativas simultâneas para um cliente SIP.
insecure	Não verifica o ip do host e a porta para o peer (não usuário).
language	O código de linguagem definido em indications.conf – Define a linguagem para os "prompts" e sinais locais de telefonia.
mailbox	Extensão da caixa postal de e-mail (Para indicações de espera de mensagens)
md5secret	Hash MD5 de "<usuário>:asterisk:<secret>" (pode ser usado ao invés de secret).
nat	Esta variável muda o comportamento do Asterisk para cliente atrás de um firewall. Isto não resolve o problemas se o Asterisk estiver atrás de um Firewall e o cliente fora.
permit,deny,mask	Endereço IP e restrição de rede
pickupgroup	Grupo que pode capturar as chamadas de colegas usando *8 e a aplicação pickup() na extensão *8
port	Porta SIP do cliente
qualify=yesno	Verifica se o cliente está alcançável ou não
restrictid=yesno	Para ter o callerid restrito -> Enviando como ANI; use isto para ocultar o callerid
rtptimeout	Termina as chamadas se não houver atividade RTP por x segundos, quando não estiver em espera (hold)
rtpholdtimeout	Termina a chamada se não houver atividade RTP quando em espera (hold) (deve ser maior que o rtptimeout)
type=peer user friend	Relacionamento com o cliente (provedor de saída ou full client)
secret	Se o Asterisk estiver agindo como um servidor SIP, então este cliente SIP deve logar com esta senha (Um segredo compartilhado). Se o Asterisk estiver agindo como SIP Client para um servidor SIP remoto que requer autenticação do tipo SIP INVITE, então este campo é usado para autenticar os SIP INVITES que o Asterisk envia para o servidor remoto
username	Nome do usuário usado no SIP INVITE
allow-disallow	Permitir ou negar codecs
musiconhold	Configura as classe de música em espera em chamadas de um telefone SIP.

Notas:

- O Asterisk não suporta chamadas SIP sobre TCP, apenas sobre UDP.
- O Asterisk usa a entrada de um fluxo RTP como fonte de sincronização de tempo para enviar o seu fluxo de saída. Se o fluxo de entrada é interrompido devido a supressão de silêncio então a música em espera terá cortes. Em resumo, **você não pode usar supressão de silêncio em telefones SIP.**

6.11 SIP NAT Traversal

A tradução de endereços IP (NAT) tem sido usada pela maioria dos provedores de serviço e empresas como uma maneira de contornar os problemas da falta de endereçamento IP. Normalmente as empresas recebem um pequeno bloco de endereços IP que varia normalmente de 1 a 256 endereços “válidos”. Já os usuários domésticos recebem um endereço válido dinâmico nos seus roteadores e usam endereços inválidos atrás destes roteadores.

O NAT resolve este problema mapeando os endereços internos para endereços públicos externos. Um endereço IP:Porta interno é mapeado para um endereço IP:Porta externo. Com este mapeamento o roteador sabe como encaminhar de volta um pacote vindo da rede externa. Este mapeamento é válido por um tempo pré-determinado, após o qual na ausência de tráfego é descartado.

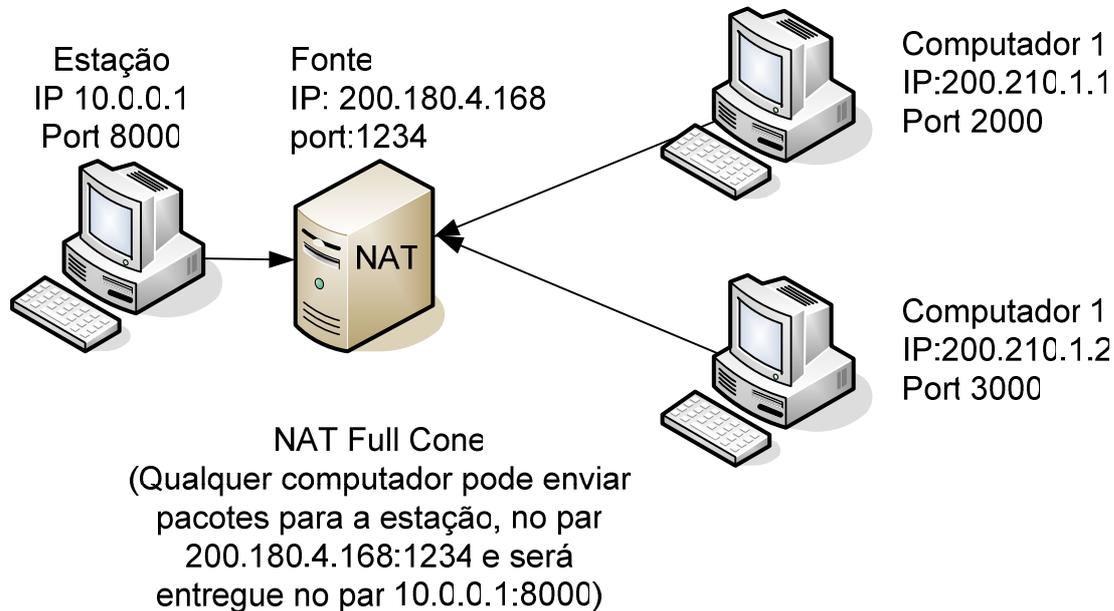
Existem quatro tipos de NAT. Como definidos

- Full Cone
- Restricted Cone
- Port Restricted Cone
- Symmetric

Para um dado endereço interno, os três primeiros tipos de NAT mantêm um mapeamento do seu endereço interno que é independente do endereço de destino sendo visto. O quarto tipo de NAT irá alocar um novo mapeamento para cada endereço de destino independente. A menos que haja uma tabela de mapeamento estático. O mapeamento que abre quando o primeiro pacote é enviado de um cliente através do NAT pode ser válido apenas por uma certa quantidade de tempo, (Tipicamente alguns minutos), a menos que os pacotes continuem, a ser enviados e recebidos em uma porta IP.

6.11.1 Full Cone (Cone Completo)

No caso do “Full Cone”, o mapeamento é bem estabelecido e qualquer um da Internet pública que queira alcançar um cliente atrás do NAT, precisa apenas saber o esquema de mapeamento de forma a mandar pacotes para ele.



Por exemplo:

Um computador atrás de um NAT com IP 10.0.0.1 enviando e recebendo na porta 8000 é mapeado para a porta externa IP no NAT de 200.180.4.168:1234. Qualquer um na Internet pode enviar pacotes para este endereço e porta IP e estes pacotes serão passados para o cliente na máquina esperando em 10.0.0.1:8000.

É o caso de Firewalls sem controle de sessão. Normalmente implementado através de filtros de pacotes e é o tipo mais inseguro de Firewall e cada vez menos comum nos dias de hoje.

6.11.2 Restricted Cone (Cone Restrito)

No caso de cone restrito, o par IP/Porta externo só é aberto uma vez que o computador interno envie dados para o endereço de destino IP específico. Por exemplo:

No caso onde o cliente envia um pacote para um computador externo, o NAT mapeia o cliente 10.0.0.1:8000 para o 200.180.4.168:1234. Assim o computador externo pode enviar pacotes de volta. Entretanto o NAT irá bloquear pacotes vindos de outros computadores externos.

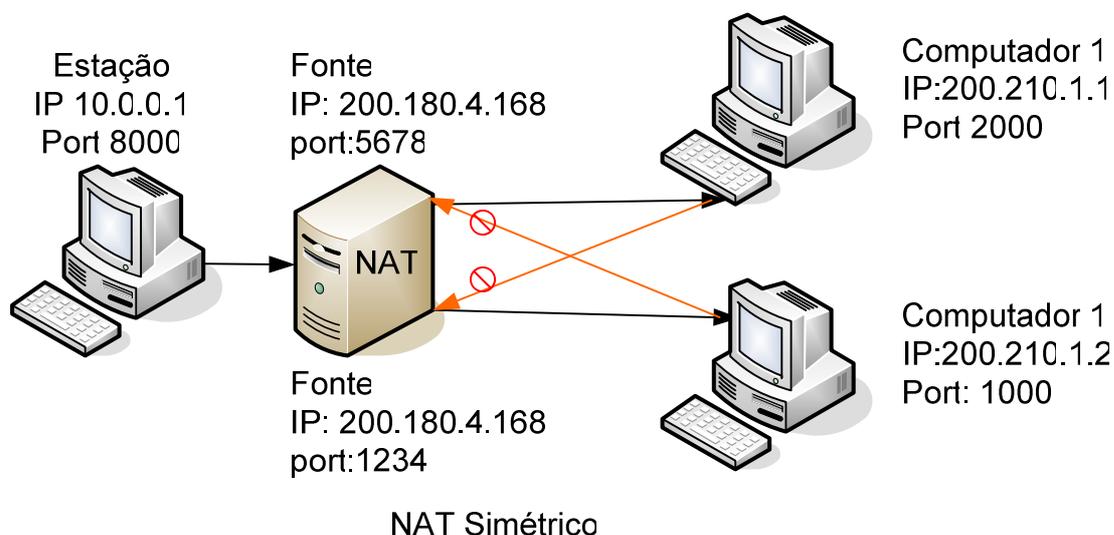
Note que neste caso o Firewall tem controle sobre a sessão, esperando pacotes pertencentes à uma sessão, mas uma vez aberto, aquele computador pode iniciar qualquer sessão independente da porta (200.210.1.1:3000 200.210.1.1:3001...).

6.11.3 Port Restricted Cone (Cone restrito por porta)

Um NAT do tipo “port restricted” é quase idêntico ao “Restricted Cone”, mas neste caso o NAT irá bloquear todos os pacotes a menos que o cliente tenha previamente enviado um pacote para o IP e porta que está enviando para o NAT. Desta forma, se o cliente enviar para um computador externo para a porta 1010, o NAT apenas irá permitir pacotes de volta se eles vierem de 200.180.4.168 na porta 1010.

Neste caso o Firewall tem um controle maior da sessão, só permitindo que pacotes pertencendo àquela sessão possam retornar, ao final da sessão, se o computador de destino resolver enviar pacotes de uma porta diferente (200.210.1.1:10000) estes não serão aceitos.

6.11.4 Simétrico



O último tipo de NAT é o simétrico, ele é diferente dos três primeiros. Um mapeamento específico do IP:Porta para um NAT público IP:Porta é dependente do endereço de destino para o qual o pacote é enviado. Então por exemplo, se um cliente envia de 10.0.0.1:8000 para o computador B, ele pode ser mapeado como 200.180.4.168:1234. Se enviar da mesma porta para um endereço IP diferente, ele é mapeado de forma diferente 200.180.4.168:5678. O computador A e B podem responder apenas para este mapeamento. Se qualquer um tentar enviar para a outra porta mapeada, estes pacotes são descartados

como no caso do Cone Restrito. O par externo IP:porta é aberto apenas quando o computador interno envia dados para um destino específico.

6.11.5 Sinalização SIP

Existem duas partes de uma chamada baseada em SIP. A primeira é a sinalização, que é um protocolo de mensagens para estabelecer uma chamada. O segundo é realmente o fluxo de mídia. Os pacotes de RTP viajam diretamente entre os dispositivos finais.

A sinalização SIP pode atravessar o NAT de uma forma bastante direta, desde que exista um proxy, a um salto de distância do NAT, que receba as mensagens SIP do cliente (Através do NAT) e então retorne as mensagens para o mesmo lugar. O Proxy precisa retornar os pacotes de SIP para a mesma porta de onde ele recebeu os pacotes (Não a porta SIP 5060). O SIP tem etiquetas (tags) que dizem ao proxy para fazer isto – A etiqueta recebido diz ao proxy para retornar um pacote para um IP específico e a etiqueta “rport” guarda a porta para onde retornar. A maioria dos proxys ainda não implementa a etiqueta “rport”, e alguns clientes não processarão as mensagens SIP corretamente.

Se estas etiquetas estiverem presentes, pelo menos em princípio o mecanismo existe para atravessar o NAT. Outro modo simples de atravessar o NAT é usar TCP para a sinalização SIP entre o cliente e o proxy. Desde que a conexão TCP é aberta através do NAT diretamente do cliente para o proxy. A sinalização irá proceder sem bloqueio. Novamente, muitos proxys ainda não implementaram ainda a opção TCP e trabalham apenas usando UDP. Note que a sinalização SIP deveria estar apta a atravessar quaisquer dos quatro tipos de NAT se o proxy retornar as mensagens de SIP da mesma porta fonte que ele recebeu a mensagem inicial. A mensagem inicial SIP, enviada para o proxy IP:Port, abre o mapeamento do NAT, e o proxy retorna os pacotes do NAT para o mesmo IP:Porta. Isto é permitido em qualquer cenário de NAT.

Registrar um cliente que está atrás de um NAT requer ou um **Registrar** que possa salvar o IP:Porta na informação de registro baseado na porta e IP que ele vê como fonte da mensagem SIP ou um cliente que saiba do seu endereço mapeado externamente e porta e possa inseri-lo na informação de contato como IP:Porta de forma a receber as mensagens SIP. É preciso tomar cuidado em usar um intervalo de registro menor que o “keepalive” para o mapeamento de NAT.

6.11.6 Fluxo de mídia RTP

O RTP para atravessar um NAT não tem uma solução tão fácil como à sinalização SIP. No caso do RTP, o corpo da mensagem SIP contém

informações sobre os pontos finais, necessárias a permitir a comunicação de um com o outro. Esta informação é contida na mensagem SDP. Os dispositivos preenchem esta informação de acordo com o que eles sabem sobre si mesmos.

Um cliente situado atrás de um NAT conhece apenas sua porta interna IP:Porta e é isto que ele coloca no corpo SDP da mensagem SIP. Quando o ponto de destino final quer enviar pacotes para o ponto originador, ele irá usar a informação SDP recebida contendo o endereço IP interno do originador e os pacotes nunca vão chegar lá. Aqui vai um exemplo de um “trace” de uma mensagem INVITE de um cliente SIP atrás de um NAT como recebida pelo gateway. Existe um proxy no meio do caminho.

```
001 INVITE sip:12125551212@211.123.66.222 SIP/2.0
002 Via: SIP/2.0/UDP 211.123.66.223:5060;branch=a71b6d57-507c77f2
003 Via: SIP/2.0/UDP 10.0.0.1:5060;received=202.123.211.25;rport=12345
004 From: <sip:2125551000@211.123.66.223>;tag=108bcd14
005 To: sip: 12125551212@211.123.66.222
006 Contact: sip: 2125551000@10.0.0.1
007 Call-ID: 4c88fd1e-62bb-4abf-b620-a75659435b76@10.3.19.6
008 CSeq: 703141 INVITE
009 Content-Length: 138
010 Content-Type: application/sdp
011 User-Agent: HearMe SoftPHONE
012
013 v=0
014 o=deltathree 0 0 IN IP4 10.0.0.1
015 s=deltathree
016 c=IN IP4 10.0.0.1
017 t=0 0
018 m=audio 8000 RTP/AVP 4
019 a=ptime:90
020 a=x-ssrc:00aea3c0
```

No trace acima, o Endereço IP na linha 003 do cabeçalho SIP é o endereço IP onde o cliente pensa que está (10.0.0.1). Mas o Proxy sabe que o endereço IP que ele realmente recebeu o pacote. Então ele adiciona as etiquetas “received” e “rport” com o endereço IP e a porta após o mapeamento do NAT. Estas etiquetas permitem ao proxy encaminhar as mensagens SIP de volta ao cliente via NAT.

Mas a informação que é usada de forma a passar os dados de voz (A conexão RTP) é mantida mais embaixo na mensagem nas linhas 014 e 016. O cliente espera receber na porta 8000 (m=) no IP 10.0.0.1 (c=), que é porta que ele vê à si próprio, e como existe um segundo ponto final irá retornar os pacotes. O resultado é que uma vez que a chamada esteja estabelecida (A sinalização SIP passa) o áudio não é recebido.

Se o cliente estiver atrás de um dos três primeiros tipos de NAT, então a solução de atravessar o NAT é simples. O cliente deve descobrir como seu IP:Porta aparece para o mundo e então deve colocar esta informação na mensagem SDP ao invés da informação do seu IP:Porta interno. Existem dois

métodos para um cliente de determinar o endereço publicamente mapeado para o IP:Porta. O primeiro é perguntar ao NAT, o segundo é perguntar a alguém fora do NAT na rede pública.

6.11.7 UPnP

Um cliente pode perguntar para o NAT como ele está mapeado para um par IP:Porta através de um protocolo chamado Universal Plug and Play. Esta é uma solução que está sendo promovida pela Microsoft (Entre outros). O cliente pergunta ao NAT via UPnP que mapeamento ele deve usar se ele quer receber na porta x. O NAT responde com o par IP:Porta que alguém na rede pública deveria usar para alcançar o cliente naquela porta.

Muitos fabricantes de dispositivos NAT já incluíram UPnP em seus produtos. Um problema é que o UPnP não vai funcionar no caso de NATs cascadeados.

Por exemplo:

Digamos que um ISP possua um bloco de endereços IP, mas não o suficiente para atender sua base de usuários. O ISP usaria NAT para dar endereço IP a seus clientes. Um dos seus clientes pode precisar de muitos endereços IP também de forma que ele deveria configurar seu próprio NAT. Se o cliente estiver rodando em um dos computadores locais usar o UPnP para determinar seu par IP:Porta ele receberá o mapeamento mais interno do NAT (O endereço interno do provedor). Existem também questões de segurança ainda não endereçadas com o UPnP. Além disso, existe uma grande base instalada que não usa o UPnP.

6.11.8 Consulta Externa

Na ausência de um mecanismo para se comunicar com o dispositivo NAT, o próximo melhor meio para o cliente determinar seu par IP:Porta externo é perguntar ao servidor situado na Internet Pública como ele vê seu endereço. Neste cenário existe um servidor que fica esperando estes pacotes (vamos chamar uma Probe NAT) . Quando ele recebe um pacote ele retorna uma mensagem da mesma porta para a fonte do pacote recebido contendo o par IP:porta que ele vê no cabeçalho do pacote enviado. Em todos os casos (Todos os 4 casos de NAT), o cliente irá receber um pacote de retorno. O cliente então vai determinar:

1. Se ele está atrás de um NAT (O IP:Porta contido é diferente do par IP:Porta que ele pensa que está)

2. Qual par IP:Porta público ele deveria usar para colocar na mensagem SDP de forma que o ponto final alcance-o

Por exemplo, se o cliente quer ser alcançado em 10.0.0.1:8000, ele irá primeiro enviar uma consulta à probe NAT pela porta 8000. A probe NAT irá realmente receber uma consulta do pacote 200.180.4.168:1234 e assim irá responder para o par IP:porta com o pacote contendo 200.180.4.168:1234. O cliente então colocar isto no seu SDP “m=AUDIO 1234” e “c=200.180.4.168”, o cliente continua escutando na porta 10.0.0.1:8000.

Isto irá funcionar nas seguintes situações:

1. O cliente deve enviar e receber o RTP na mesma porta.
2. O cliente deve enviar a mensagem SIP logo depois de enviar a consulta para a probe NAT. Se existir um longo atraso o NAT pode ter um timeout.
3. No caso de Restricted Cone e Port Restricted Cone, o cliente deve enviar o pacotes para o ponto final antes que o NAT permita pacotes do ponto final para o cliente. Isto não vai funcionar no caso de NAT simétrico, pois o endereço da probe NAT é diferente daquele do ponto final e deste modo o mapeamento da probe NAT irá ver é diferente daquele que o ponto final usa para enviar pacotes até o cliente naquele par IP:Porta.

6.11.9 STUN

Simple Traversal de UDP através de NAT (Travessia simples do UDP sobre o NAT). é um protocolo para configurar o tipo de probe NAT como foi descrito. Ele realmente faz um pouco mais que apenas retornar o par IP:Porta público. ele pode também determinar o tipo de NAT que você está atrás. Clientes que usam o protocolo STUN já existem como o XTEN por exemplo. Os pedidos de STUN especificam os seguintes parâmetros:

RESPONSE-ADDRESS - O servidor STUN irá enviar sua resposta para o par IP:Porta especificado no atributo **RESPONSE-ADDRESS**. Se este campo não estiver presente, então o servidor envia sua resposta no par IP:Porta de onde ele recebeu o pedido. Se ambas as “flags” **Change IP** e **Change Port** não estiverem setadas, o STUN responde do par IP:Porta que o pacote inicial foi enviado. Se o **Change IP** estiver setado, o servidor responde de um IP diferente e se o **Change Port** estiver setado então ele responde de uma porta diferente.

A resposta do STUN contém as seguintes informações:

MAPPED-ADDRESS – O par IP:Porta do cliente como visto no primeiro servidor STUN fora do NAT à receber o pedido.

CHANGED-ADDRESS – O Endereço IP que deveria ser a fonte da resposta retornada se o pedido foi feito com o “flag” **Change IP** setado.

SOURCE-ADDRESS – O Par IP:Porta de onde a resposta STUN foi enviada.

Usando uma combinação de diferentes pedidos ao servidor STUN, um cliente pode determinar se ele está na Internet aberta ou se está atrás de um Firewall que bloqueia o UDP ou se ele está atrás de um NAT e que tipo.

6.11.10 Mídia orientada à conexão

A solução acima funciona bem (Servidor STUN) para os três primeiros tipos de NAT. O quarto caso (NAT simétrico) não irá permitir este esquema, pois ele tem diferentes mapeamentos dependendo do endereço IP alvo. Desta forma o mapeamento que o NAT designa entre o cliente e a probe NAT é diferente daquele entre o cliente e o gateway. No caso de NAT simétrico o cliente deverá enviar o RTP para e receber o RTP de volta do mesmo endereço IP. Qualquer conexão RTP entre um ponto final fora do NAT e um dentro do NAT deve ser estabelecido ponto à ponto e assim (Mesmo se uma conexão SIP já foi estabelecida) o ponto final fora do NAT deve esperar até ele receber um pacote de um cliente antes que ele possa saber para onde responder. Isto é conhecido como “**Mídia orientada a conexão**”.

Se for desejado que se falem ambos clientes que estão atrás de NATs e clientes na Internet aberta, então ele deve saber se pode confiar na mensagem SDP que ele recebe na mensagem SIP, e quando ele precisa esperar receber um pacote diretamente antes que o cliente abra um canal de volta para a par IP:porta fonte daquele pacote. Uma proposta para informar o ponto final para esperar um pacotes de entrada e adicionar uma linha na mensagem SDP (Vindo do cliente atrás do NAT).

a=direction:active

Quando o dispositivo lê esta linha, ele entende que o cliente iniciando irá ativamente estabelecer o par IP:Porta par a qual o dispositivo deve retornar o RTP, e que o par IP:Porta encontrado na mensagem SDP deve ser ignorado. A maioria dos clientes SIP não suportam a diretiva “a=”. Até eles suportarem deve existir algum tipo de tradutor no meio do fluxo SIP.

6.11.11 RTP-Relay

Se um dispositivo suporta mídia orientada a conexão, então o problema de atravessar um NAT simétrico está resolvido. Dois cenários ainda são problemáticos.

1. Se o ponto final não suporta a diretiva `a=direction:active tag`.
2. Se ambos os pontos finais estão atrás de NATs simétricos.

Em qualquer um dos casos, uma solução é ter um Relay de RTP no meio do fluxo RTP entre os pontos finais. O Relay RTP age como um segundo ponto final para o qual os dispositivos reais tentam se comunicar um com o outro. Tipicamente, existiria um servidor no meio do fluxo SIP (Aqui chamado de NAT Proxy) que iria manipular o SDP de forma a instruir os pontos finais à enviar o RTP para o Relay ao invés de diretamente de um para o outro. O Relay estabelecerá seu próprio mapeamento de uma sessão, guardando o par IP:porta de cada ponto final para onde ele deveria enviar os pacotes RTP. O seguinte é um típico fluxo de chamada que pode ser instanciado entre um agente usuário atrás de um NAT simétrico e um gateway de voz na Internet.

6.12 Questionário

1. O SIP é um protocolo do tipo texto similar ao _____ e _____.

- IAX
- HTTP
- H323
- SMTP

2. O SIP pode ter sessões do tipo: (marque todos que se aplicam)

- Voz
- Correio Eletrônico
- Vídeo
- Chat
- Jogos

3. Podemos citar como componentes do SIP o: (marque todos que se aplicam)

- User Agent
- Media gateway
- PSTN Server
- Proxy Server
- Registrar Server

4. Antes que um telefone possa receber chamados, ele precisa se _____.

5. O SIP pode operar em modo PROXY e modo REDIRECT, a diferença entre eles é que no caso do PROXY a sinalização sempre passa pelo computador intermediário (SIP Proxy) enquanto no modo REDIRECT os clientes sinalizam diretamente.

- Correto
- Incorreto

6. No modo PROXY o fluxo de mídia e a sinalização passam pelo “SIP proxy” e não diretamente de um cliente para o outro.

- Correto
- Incorreto

7. O Asterisk atua como um SIP Proxy.

- Correto
- Incorreto

8. A opção canreinvite=yes/no é de importância fundamental, pois vai definir se o fluxo de mídia vai passar pelo Asterisk ou não. A afirmação está:

- Correta
- Incorreta

9. O Asterisk suporta sem problemas supressão de silêncio em canais SIP. A afirmação está:

- Correta
- Incorreta

10. O tipo mais difícil de NAT para transpor é o:

- Full Cone
- Restricted Cone
- Port Restricted Cone
- Symmetric

Visão geral do plano de discagem

Neste capítulo vamos aprender a configurar o Asterisk naquilo que é conhecido como plano de discagem ou plano de numeração. O plano de numeração controla todo o funcionamento do Asterisk como PABX.

7.1 Objetivos do capítulo

Ao final deste capítulo você deve estar apto a criar um plano de discagem usando os seguintes recursos:

- Contextos, extensões e prioridades.
- Switches.
- Variáveis.
- Processamento das extensões.
- Inclusão de contexto
- Macros.
- Extensões padrão e prioridades.

7.2 Visão geral do plano de discagem

O plano de discagem é a peça mais importante na configuração do Asterisk e ele é configurado no arquivo `extensions.conf`. Ele controla como todas as chamadas de entrada e saída são encaminhadas e manuseadas.

É no `extensions.conf` que você controla o comportamento de todas as conexões através do seu PABX.

7.3 Descrição do arquivo `extensions.conf`

7.3.1 [general]

No topo do arquivo `extensions.conf`, você configura algumas configurações gerais na seção com cabeçalho `[general]`. Aqui estão algumas opções relacionadas ao plano de discagem:

- **static:** Neste estágio, esta opção afeta apenas a operação do comando `save dialplan`. o valor default é não, mas o arquivo exemplo do `extensions.conf` instalado com o Asterisk explicitamente seta `static=yes`.
- **writeprotect:** Se `writeprotect=no` e `static=yes`, então você pode salvar o plano de discagem atual com o comando `save dialplan`. Definições nas variáveis globais na categoria `[globals]` permanecem sem modificação. O valor default é “no”.

Cuidado:

“`save dialplan`” irá sobrescrever o seu arquivo `extensions.conf` com um novo gerado pelo plano de numeração atual. Uma copia de seu arquivo `extensions.conf` velho não será guardado. Todos os comentários serão perdidos. O arquivo exemplo do Asterisk vem com esta configuração perigosa, `writeprotect=no`, `static=yes`

7.3.2 Seção `[globals]`

Em seguida na seção `[globals]`, você pode definir as variáveis globais (ou constantes) e seus valores iniciais.

Na verdade as variáveis globais do Asterisk são normalmente usadas como constantes e não variáveis. Elas são usadas para simplificar mudanças futuras na configuração do PABX. O valor atual das variáveis globais pode ser mudado usando o comando `SetGlobalVar`. O valor da variável global pode ser referenciado, usando a sintaxe: `${nomedavariável}`. As variáveis globais não são sensíveis a maiúscula e minúscula.

Que extensões devem tocar quando uma chamada chegar

```
INCOMING=>Zap/3&Zap/4
```

Quanto tempo deve tocar antes de chegar ao voicemail

```
RINGTIME=>3
```

Que arquivo de áudio deve ser tocado como anuncio do voicemail.

```
VMANNOUNCE=>mysounds/my-vm-annouce
```

Definir os canais as quais nossas extensões estão ligadas

```
KITCHEN=>Zap/3
STUDY =>Zap/4
HALL =>Zap/5
```

Quando nós queremos fazer uma chamada externa que linhas devemos usar

Criando estas definições, por si só não tem efeito algum. O Asterisk não sabe por si só o que fazer com estas variáveis. É de sua responsabilidade saber o que fazer com elas no seu plano de discagem.

Neste exemplo, as variáveis globais, foram todas escritas com nomes em maiúscula. Nomes de variáveis globais não diferenciam maiúsculas de minúsculas. Elas são colocadas em maiúsculas, por conveniência, para diferenciar das variáveis de canal que normalmente têm maiúsculas e minúsculas.

7.4 Contextos e Extensões

Após as seções [general] e [globals], o restante do arquivo extensions.conf é usado na definição do plano de discagem. O plano de discagem consiste de uma coleção de contextos. Cada contexto consiste de uma coleção de extensões.

7.4.1 Introdução à contextos e extensões

Um plano de discagem consiste de uma coleção de contextos. Estas definições de contexto são a parte mais importante do arquivo extensions.conf e a parte mais importante da configuração do Asterisk.

Um contexto é apenas uma coleção de extensões. Exemplo de diagrama de contextos e extensões.

Context "default"

Extensão	Descrição
8580	Flavio
8581	Daniel
8582	Juliano

8583	Verificar o correio
8585	Sala de conferência
0	Telefonista

Neste exemplo, que foi dado o nome de “default”, as primeiras três extensões serão associadas a telefones. A quarta extensão (8583) será associada com a aplicação de correio de voz. A quinta extensão (8585) será associado à uma sala de conferência. Finalmente o 0 será associado com o operador.

Aqui está outro exemplo de um contexto:

Context "mainmenu":

Extensão	Descrição
s	Bem vindo a mensagens e instruções
1	Vendas
2	Suporte
3	Contabilidade
9	Diretório
#	Desligar

Neste exemplo de contexto, com o nome de “mainmenu” tem apenas extensões de um dígito. A extensão “s” é a extensão de início, onde o usuário inicia. Esta extensão irá tocar uma mensagem do tipo “Obrigado por discar para nossa empresa”, pressione 1,vendas, 2, suporte, 3, contabilidade, 9 para o diretório da empresa ou # para desligar. Cada opção do menu é de fato uma extensão e poderia ou discar uma extensão real ou fazer algo como enviar quem discou para outro menu.

Os contextos podem ser usados para implementar um número importante de recursos incluindo:

- Segurança: Permitir ligações de longa distância de certos telefones apenas.
- Roteamento: Rotear chamadas baseadas em uma extensão.
- Auto-atendente: Receber quem discou e pedir para entrar as extensões.
- Menus multicamada: Menus para vendas, suporte, etc.
- Autenticação: Pedir por senha para certas extensões.

- Callback: Reduzir as tarifas para chamadas de longa distância.
- Privacidade: Colocar em lista negra, pessoas que você não quer receber ligações.
- PBX Multihost: Sim, você pode ter hosts virtuais no seu PABX.
- Daytime/Nighttime: Você pode variar o comportamento após o horário.
- Macros: Criar scripts para funções normalmente usadas.

7.4.2 Como os contextos são usados?

Quando o Asterisk recebe uma chamada, de entrada ou de saída, esta chamada pertence a um contexto. Qual contexto a chamada pertence depende de que canal a chamada veio. Quando você configura os canais que você tem no se PABX, uma das coisas que você faz é definir em que contexto uma chamada naquele canal vai ser colocada, usando uma definição como:

```
context=incoming
```

Então a primeira forma em que os contextos são usado é fazer com que o Asterisk se comporte de forma diferente dependendo de onde a chamada esta vindo. Com certeza você vai ter pelo menos um contexto definido. Neste contexto você vai definir o que fazer com a chamada, se vai tocar uma das extensões, entrar no correio de voz ou tocar um anuncio de voz. Se você quer que o Asterisk trate as conexões de suas extensões internas de forma diferente, por exemplo, “poder fazer ddi”, você pode definir que diferentes canais entrem em diferentes contextos.

7.4.3 Extensões

Uma extensão pode ser uma de dois tipos, literal ou padrão.

Uma extensão literal pode ser um número, como o 123, e ele pode também conter símbolos padrão como * e # que aparecem em telefones normais. De forma que 12#89* é uma extensão válida. Alguns teclados de telefone multi-freqüencial tem teclas especiais A, B,C,D e extensões pode ser definidas com estas letras também. De fato, o nome da extensão pode conter qualquer letra ou número bem como alguns caracteres especiais. Note que muitos telefones VOIP conseguem discar números de extensão que podem ser uma string arbitrária, tais como **Escritório** no Asterisk. ,

As extensões são sensíveis à minúsculas ou maiúsculas. Bem, sim e não. Elas são sensíveis à minúsculas ou maiúsculas no sentido em que quando o Asterisk está tentando bater a extensão que o usuário discou contra as extensões que estão no contexto, a extensão deve bater inclusive considerando maiúsculas e minúsculas. Então se o usuário discar a extensão “ESCRITORIO” usando o seu telefone VOIP, o Asterisk não irá executar os comandos que você definiu para a extensão “Escritório”. Por outro lado o Asterisk não permite que você defina extensões diferentes com o mesmo nome diferindo apenas por maiúscula e minúscula.

7.5 Switches

7.5.1 Encaminhando para outro Asterisk

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

Especifica o encaminhamento para outro servidor. O usuário e chave precisam ser definidos no arquivo `iax.conf` de servidor que é chamado. O contexto é o contexto no servidor de destino.

7.6 Variáveis e expressões

Existe suporte para usar variáveis usando a construção `${NOMEDAVARIAVEL}`. Você pode usar expressões com o construção `$(EXPRESSION)`, onde as expressões podem ser “regular expressions” (São usadas para busca em strings), comparação, adição, subtração e muito mais.

7.6.1 Usando variáveis nos planos de discagem

O Asterisk pode fazer uso de variáveis globais ou específicas por canal como argumentos para os comandos. Variáveis são referenciadas no plano de discagem (`extensions.conf`) usando a sintaxe.

<code>\${varname}</code>

Onde “varname” é o nome da variável. Um nome de variável pode ser uma string alfanumérica iniciando com uma letra. Os nomes de variáveis definidos pelo usuário não são sensíveis à maiúsculas e minúsculas. `${VarName}` e `${varname}` se referem a mesma variável. Entretanto as variáveis definidas pelo Asterisk diferenciam maiúsculas e minúsculas. `${EXTEN}` funciona, mas `${exten}` não.

Existem três tipos de variáveis:

- Variáveis globais
- Variáveis de canal
- Variáveis de ambiente

7.6.2 Variáveis globais podem ser configuradas na categoria [globals] do arquivo extensions.conf ou usando o comando SetGlobalVar. Uma vez definido, eles podem ser referenciados por qualquer canal à qualquer hora.

7.6.3 Variáveis de canal são configuradas usando o comando SetVar. Cada canal recebe seu próprio espaço de variáveis, de forma que não há chance de colisões entre diferentes chamadas, e a variável é automaticamente destruída quando o canal é desligado.

7.6.4 Variáveis de ambiente fornecem um meio de acessar variáveis de ambiente Unix de dentro do Asterisk. Existe uma lista mais abaixo nesta página.

Se você definir uma variável de canal com o mesmo nome de uma variável global (lembre-se variáveis definidas pelo usuário, não se importam com maiúsculas e minúsculas), referências ao nome da variável irão retornar o valor da variável de canal.

Variáveis de canal predefinidas existem algumas variáveis de canal configuradas pelo Asterisk que você pode se referir como definições do plano de discagem. Variáveis definidas pelo Asterisk diferenciam caixas alta e baixa.

- `${ACCOUNTCODE}`: Código de contabilização
- `${ANSWEREDTIME}`: Horário quando a chamada foi atendida
- `${CALLERID}`: O identificador da chamada (nome e número)
- `${CALLERIDNAME}`: O nome do id. da chamada
- `${CALLERIDNUM}`: O número do id. da chamada
- `${CALLINGPRES}`: Variável de apresentação PRI Call ID para chamadas de entrada
- `${CHANNEL}`: Nome do canal atual

- `${CONTEXT}`: Nome do contexto atual
- `${DATETIME}`: Data e hora no formato YYYY-MM-DD_HH:MM:SS.
- `${DIALEDPEERNAME}`: Nome do quem foi chamado
`${DIALEDPEERNUMBER}`: Número de quem foi chamado.
- `${DIALEDTIME}`: Hora em que o número foi discado.
- `${DIALSTATUS}`: Status da chamada.
- `${DNID}`: Identificador do número discado.
- `${EPOCH}`: Época no estilo do Unix (Número de segundos desde 1970).
- `${EXTEN}`: A extensão atual.
- `${HANGUPCAUSE}`: O último código de término de chamada em um canal Zap conectado à uma interface PRI.
- `${INVALID_EXTEN}`: A extensão pedida quando redirecionada para a extensão *i* (inválida).
- `${LANGUAGE}`: A linguagem atual.
- `${MEETMESECS}`: O número de segundos que um usuário participou de uma conferência em uma sala de “MeetMe”.
- `${PRIORITY}`: A prioridade atual.
- `${RDNIS}`: O atual redirecionamento [DNIS](#), Caller ID que redirecionou a chamada.
- `${SIPDOMAIN}`: Domínio de destino SIP de uma chamada de entrada.
- `${SIP_CODEC}`: Usada para SetVar o codec SIP para uma chamada. `${SIPCALLID}`: O cabeçalho SIP dialog Call-ID.
- `${SIPUSERAGENT}`: O cabeçalho SIP user agent.

- `{TIMESTAMP}`: Date e hora no formato: YYYYMMDD-HHMMSS.
- `{TXTCIDNAME}`: Resultado da aplicação `TXTCIDName`.
- `{UNIQUEID}`: Identificador único da chamada atual.

7.6.5 Variáveis específicas de aplicações

Algumas aplicações usam entradas de dados extras e fornecem a saída na forma de variáveis de canal.

- **ChanIsAvail** retorna `{AVAILCHAN}`: O primeiro canal disponível.
- **Dial** recebe input de `{VXML_URL}`: Envia uma url XML para um Cisco 7960.
- **Dial** recebe entrada de `{ALERT_INFO}`: Configura a cadência de campanha por telefones Cisco.
- **Dial** retorna `{CAUSECODE}`: Se uma discagem falhou, esta é a mensagem de erro.
- **Dial** retorna `{DIALSTATUS}`: Texto com o código de status da última tentativa de discagem.
- **EnumLookup** retorna `{ENUM}`: O resultado da busca.
- **MeetMe** recebe input de `{MEETME_AGI_BACKGROUND}`: Um script de AGI para rodar.
- **MeetMe** retorna `{MEETMESECS}`: O número de segundos que o usuário esteve na conferência.
- **Hangup** lê a `{PRI_CAUSE}` variável para configurar os códigos de retorno PRI.
- **TXTLookup** retorna `{TXTCIDNAME}`: O resultado de um DNS lookup.

7.6.6 Variáveis específicas para Macros

Quando em um contexto de macro, algumas variáveis adicionais de canal estão disponíveis.

- `#{ARG1}`: O primeiro argumento passado pela macro.
- `#{ARG2}`: O segundo argumento passado pela macro e assim por diante.
- `#{MACRO_CONTEXT}`: O Contexto da extensão que disparou a macro.
- `#{MACRO_EXTEN}`: A extensão que disparou a macro.
- `#{MACRO_OFFSET}`: Configurado por uma macro para influenciar a prioridade de execução ao sair da macro .
- `#{MACRO_PRIORITY}`: A prioridade na extensão onde esta macro foi disparada.

7.6.7 Variáveis de ambiente

Você pode acessar variáveis do ambiente Unix da seguinte forma:

- `#{ENV(nomedavariável)}`.
- `#{ENV(ASTERISK_PROMPT)}`: O prompt atual da linha de comando CLI .
- `#{ENV(RECORDED_FILE)}`: O nome do arquivo gravado pela última vez com o comando [Record](#).

7.7 Funções de manuseio de “strings”

7.7.1 Comprimento da String

`#{LEN(string)}` retorna o comprimento da string `nomedavariável`.

```
Exemplo:  
exten=>100,1,SetVar(Fruta=pera)  
exten=>100,2,NoOp(#{LEN(Fruta)})  
exten=>100,3,NoOp(#{LEN(#{Fruta})})
```

O primeiro NoOp deveria mostrar um valor de 5 (O Comprimento da string “fruta”). A segunda operação NoOp deveria mostrar o valor de 4 (O comprimento da string “pera”).

Esta é uma boa maneira de verificar par uma string vazia ou nula.

7.7.2 Substrings

```
${string:offset:length}
```

offset: ponto de início

length: comprimento

Retorna uma substring da string, iniciando na posição definida por “offset” e retornando o comprimento de caracteres definido em “length”.

Se o offset é negativo, isto é pego da direita para esquerda a partir do fim da string.

Se o comprimento é omitido, ou é negativo, então todo o resto da string a partir do ponto de início (offset) é retornado.

Exemplos:

```
${123456789:1}-retorna a string 23456789
```

```
${123456789:-4}-retorna a string 6789
```

```
${123456789:0:3}-retorna a string 123
```

```
${123456789:2:3}-retorna a string 345
```

```
${123456789:-4:3}-retorna a string 678
```

Exemplos de uso:

```
exten=>_NXX.,1,SetVar(areacode=${EXTEN:0:3})
```

Pega os primeiros três dígitos da variável `${EXTEN}`

```
exten=>_516XXXXXXXX,1,Dial(${EXTEN:3})
```

Pega todos excetos os primeiros três dígitos da `${EXTEN}`

```
exten=>100,1,SetVar(whichVowel=4) exten=>100,2,SetVar(foo=AEIOU:${whichVowel}:1)
```

Seta `${foo}` para uma única letra U

7.7.3 Concatenação de Strings

Para concatenar duas strings, simplesmente escreva-as juntas.

```

${foo}${bar}
555${Onumero}
${PrefixoLongaDistancia}555${ONumero}

```

7.8 Inclusão de contextos

Um contexto de extensões pode incluir o conteúdo de outro. Por exemplo, considere os seguintes contextos.

Contexto default

Extensão	Descrição
101	Mark Spencer
102	Will Meadows
0	Telefonista

Contexto local

Extensão	Descrição
_9NXXXXXX	Chamadas locais
include=>	"default"

Contexto longa distância

Extensão	Descrição
_91NXXNXXXXXX	Chamadas de longa distância
include=>	"local"

Aqui nós definimos três extensões:

O contexto default permite discar três extensões: Mark, Wil e a telefonista. O contexto local tem um padrão de extensões para permitir a discagem de números de sete dígitos (chamadas locais). O contexto longa distância tem um padrão de extensões para permitir uma discagem de longa distância, e ele também inclui o contexto “local”, deste modo permitindo ao usuário fazer chamadas locais e também discar as extensões de Mark, Wil ou a telefonista.

Usando contextos de extensão você pode cuidadosamente controlar quem tem acesso aos serviços de discagem

Se mais de um padrão corresponder ao número discado, o Asterisk pode não usar o que você está esperando. Veja a seção, correspondente a ordem de busca das extensões do asterisk.

Quando o Asterisk recebe uma chamada de entrada em um canal, o Asterisk olha no contexto definido pelo canal por comandos dizendo ao Asterisk o que ele deveria fazer. O contexto define diferentes conjuntos de comandos dependendo de que extensão o usuário discou. Por exemplo, o contexto pode prover um conjunto de comandos para definir o que fazer se o usuário discou “123” e outro conjunto de comandos se o usuário discou “9” e outro se o usuário discou qualquer número iniciando com “555”.

Para alguns tipos de conexões, tais como chamadas de entrada de um telefone externo, o usuário não discou uma extensão. Neste caso o asterisk se comporta como se o usuário tivesse discado uma extensão especial chamada “s” (“start”). O Asterisk irá olhar para um número de extensão s na definição do contexto daquele canal para instruções sobre o que fazer para tratar a chamada.

Vamos dizer, por exemplo, que você tenha um canal. “ZAP/1” que é uma conexão à um aparelho telefônico no seu prédio. E vamos dizer que no arquivo de configuração para os canais ZAP (zapata.conf) você tenha definido context-john para o canal ZAP/1. Então, quando você usar o telefone para discar um número, o Asterisk vai procurar um contexto com o nome anna no arquivo extensions.conf para descobrir o que ele deveria fazer. Você começa a definição de um contexto no arquivo extensions.conf colocando o nome do contexto em chaves ([]'s) em uma linha como por exemplo:

```
[anna]
```

Para cada contexto, você deve definir uma ou mais extensões. O Asterisk as usa para comparar contra o número discado. Para cada extensão, você diz ao Asterisk o que fazer listando um conjunto de comandos.

7.7.2 Como o plano de discagem encontra a extensão

A discagem “encontra enquanto disca” ocorre em duas situações:

- Menus da unidade de resposta audível. Algo como disque 1 para..., 2 para
- Aparelhos telefônicos padrão conectados por interfaces FXS. Esta é uma situação onde você levanta o telefone, recebe o tom de discagem e começa a discar.

O que é o “encontra enquanto disca”?

Considere uma linha telefônica tradicional conectada na companhia telefônica. Quando você levanta o gancho e começa a discar, cada dígito é enviado e processado imediatamente, (“encontra enquanto disca”). Em contraste, considere um telefone celular, você digita o número de telefone e ao final pressiona o botão “envia” e o telefone envia tudo de uma vez.

Uma vez que a chamada tenha completado, ambos os telefones funcionam da mesma forma. Se por exemplo, você disca para uma companhia e é recebido pelo sistema de correio de voz, quaisquer dígitos que você disque, seja usando um telefone celular ou um comum são enviados à medida que você disca.

Para o Asterisk, o estilo “todo de uma vez” de discar é mais fácil de processar: O Asterisk só verifica se o número discado bate com qualquer padrão de extensão no contexto atual do plano de discagem.

O estilo “encontra enquanto disca” é um pouco mais difícil porque não está claro quando você vai terminar a discagem. De um lado você não quer que o Asterisk fique esperando; você quer que ele processe o número o mais rápido possível, por outro lado o Asterisk tem de esperar se algum dígito adicional irá afetar como ele gerencia sua chamada.

Entender como o “encontra enquanto você disca” funciona, vai ajudar a decidir como alocar seus números de extensão e como definir os padrões de extensão usados.

7.7.3 Processo “encontra enquanto você disca”.

Quando o Asterisk está tendo de processar o número discado, a primeira coisa que o Asterisk considera é:

Quando você teclar outro dígito, existirão padrões de extensão que poderiam bater com o número? O Asterisk considera o número que você discou até o momento e pesquisa os padrões de extensão definidos naquele contexto. Se a resposta à questão é sim, então o Asterisk vai esperar para ver se você vai teclar outro dígito, mesmo se existir um número de telefone completo que corresponda ao que você já digitou.

Por exemplo, vamos dizer que você tenha:

```
exten=>123,1,DoSomething(...)
exten=>_XXX.,1,DoSomethingElse(...)
```

Se você digitou 123, o Asterisk irá esperar para ver se você vai digitar outro dígito, porque não está claro que você quer a extensão 123. Por exemplo, a extensão 123999, poderia corresponder ao segundo padrão definido.

O Asterisk irá verificar todos os padrões de extensão definidos para o contexto atual. Se existir pelo menos um padrão possível, que possa ocorrer caso você tecele um novo dígito, o Asterisk irá esperar.

Quanto tempo o Asterisk irá esperar? Vai depender de dois fatores. Primeiro, se é uma nova chamada de entrada ou uma conexão estabelecida.

Novas chamadas de entrada são aquelas em que o Asterisk não iniciou o processamento de quaisquer comandos do plano de discagem. Para canais Zap, isto significa que alguém levantou o gancho e começou a discar e o Asterisk está processando cada dígito na medida em que ele é discado.

Para novas chamadas, o segundo fator que afeta quanto tempo o Asterisk espera por mais dígitos antes de desistir é o tipo de canal de onde a chamada esta vindo. Cada tipo de canal determina seu próprio período de timeout. Um canal Zap tem um tempo fixo de 3 segundos e isto não pode ser mudado a não ser modificando o código fonte e re-compilando o módulo Zap.

Conexões estabelecidas são aquelas onde o Asterisk já fez a conexão inicial, foi para a extensão (mesmo que a extensão s), iniciou o processamento de comandos e ficou sem comandos para processar e está esperando agora que algo seja discado. Tipicamente, um dos comandos que o Asterisk poderia ter executado é o comando Background para tocar uma mensagem gravada, apresentando um menu de URA e o Asterisk está esperando que seja discada a escolha do menu. Para conexões estabelecidas, o fator afetando o período de timeout é o valor DigitTimeout. O valor padrão de digittimeout é de 5 segundos, mas isto pode ser mudado usando o comando de mesmo nome.

O que o Asterisk faz em seguida

Se o asterisk esperou, mas não recebeu nenhum dígito no período de timeout, ou o Asterisk não encontrou quaisquer padrões de extensão que, se você discasse mais dígitos, poderia bater com seu número. E então o Asterisk vai considerar a questão: o número bate com qualquer extensão no contexto atual?

Sim: Pula para a primeira extensão que bater. Se mais de uma extensão bater, o asterisk pula para a primeira que ele encontrar. O Asterisk não

considera os padrões de extensão na ordem em que você os define, de forma que a extensão encontrada pode não ser a que você deseja.

Não: Se o número que você discou é inválido no contexto atual, então a ação que o asterisk tomou depende em que situações ele está.

1. Se esta é uma conexão de entrada nova, então o Asterisk irá retornar um sinal apropriado de “número inválido” para quem discou. Se for um canal Zap o usuário receberá um tom de discagem inválida, se for um telefone IP poderá eventualmente receber uma mensagem de discagem inválida. Nesta situação, nenhum comando do plano de discagem foi executado e não existe controle sobre o comportamento do Asterisk em como ele gerencia um número inválido. O Asterisk não pula para a extensão *i* nesta situação.

2. Se por outro lado, este é uma conexão estabelecida, o Asterisk pulou para uma extensão, iniciou a execução de comandos (ex. tocando uma mensagem de um menu da ura) e ficou sem comandos, esperando que o usuário disque alguma coisa. Neste caso a discagem de um número inválido faz com que o Asterisk pule para a extensão *i*. Se a extensão *i* não estiver definida neste contexto, o asterisk irá desligar.

7.7.4 Exemplo

Uma empresa quer que suas chamadas telefônicas sejam respondidas com uma mensagem de voz dando boas vindas à quem ligou e convidando-o a escolher uma extensão. A empresa tem seis extensões que são 1,2,21,22,31,32. Então o contexto criado para as chamadas entrantes.

```
[incoming]
exten=>s,1,Background(bem-vindo-a-empresa)
exten=>1,1,Dial(Zap/1)
exten=>2,1,Dial(Zap/2)
exten=>21,1,Dial(Zap/3)
exten=>22,1,Dial(Zap/4)
exten=>31,1,Dial(Zap/5)
exten=>32,1,Dial(Zap/6)
```

Quando você chamar a empresa, o asterisk irá tocar o arquivo bem-vindo-a-empresa.gsm. Após isto, fica sem comandos para executar e espera que você disque algo. Isto é o que ele vai fazer se você discar várias opções.

Número discado	Ação do Asterisk
1	Imediatamente faz a discagem Dial(Zap/1)
2	Espera pelo timeout e então faz Dial(Zap/2)
21	Imediatamente faz o Dial (Zap/3)

22	Imediatamente faz o Dial (Zap/4)
3	Espera pelo timeout então desliga
31	Imediatamente disca para Dial(Zap/5)
32	Imediatamente faz o Dial(Zap/6)
4	Imediatamente desliga

Note que quando alguém tenta discar a extensão 2, ele não é conectado imediatamente. O Asterisk espera para ver se quem discou vai teclar mais dígitos para determinar qual extensão é desejada 2,21 ou 22. Como todo mundo gostaria de ser atendido imediatamente, o ideal é não usar um plano de discagem ambíguo.

Se a companhia não quiser perder chamadas se for discado um número inválido, o ideal é definir uma extensão i (invalida) para este contexto e t (timeout).

7.7.5 Ordem de busca dos padrões de extensão

Cada contexto definido no plano de discagem dirá ao Asterisk como processar os números de telefone naquele contexto. Como você pode usar padrões para definir extensões, mais de um padrão de extensão pode bater com um dado número de telefone. O Asterisk não bate os números contra os padrões na ordem que você os define, os padrões são ordenados primeiro.

Problema exemplo:

Vamos dizer que para o contexto exemplo, você queira que números iniciando com 918 saiam por uma linha analógica conectada em Zap/1, e todos os outros números saiam pela Zap/2. Então você escreve algo como:

```
[exemplo]
exten=>_918.,1,Dial(Zap/1/${EXTEN})
exten=>_.,1,Dial(Zap/2/${EXTEN})
exten=>h,1,Hangup
```

Só que não funciona. Você descobre que não importando o número que você disca, todos os números são enviados via Zap/2. Isto ocorre porque o Asterisk ordena as extensões e pega a primeira que corresponda. Para ver a ordem pela qual o Asterisk ordena as extensões, digite o comando `show dialplan exemplo` na console. Você vai descobrir que a ordem é:

```
_
_918
h
```

Note que esta é uma ordem diferente da qual você definiu no seu arquivo `extensions.conf`. A linha com “`_.`” é agora a primeira, incluindo aqueles que começam com 918. Note também que a extensão `h` não pode ser alcançada, porque ela também corresponde ao `_.`

7.7.6 Controlando o ordenamento

Então como se faz para o Asterisk bater os padrões de extensão na ordem que você quer. Usando a palavra chave *include* para incluir outro contexto de padrões de extensões dentro do contexto atual.

```
[exemplo]
include=>exemplo-sub
exten=>h,1,Hangup
exten=>_918.,1,Dial(Zap/1/${EXTEN})

[exemplo-sub]
exten=>_.,1,Dial(Zap/2/${EXTEN})
```

O Asterisk (quando no contexto “exemplo”) irá processar os números discados na seguinte ordem.

```
_918
h
_
```

Note o que o Asterisk fez:

As entradas “`exten=>`” dentro do contexto exemplo são ordenadas e testadas primeiras. O conteúdo de cada contexto incluído é ordenado e testado em seguida. Os contextos incluídos são testados na ordem das linhas incluídos em `extensions.conf`.

Reloading

Se você quiser recarregar o plano de discagem após mudanças, sem recarregar toda a configuração do Asterisk. Use o comando de console `extensions reload`.

Um arquivo grande ou vários arquivos pequenos?

Com a instrução `#include <nome do arquivo>` no `extensions.conf`, outros arquivos são incluídos. Desta forma você pode configurar o sistema onde o

extensions.conf é o arquivo principal, users.conf contém os usuários locais, services.conf contém vários serviços como conferência. Desta forma o plano de discagem pode ser mais fácil de manter, dependendo do tamanho do seu setup. Não confunda o #include <nome do arquivo> com a instrução include <context>. O #include funciona em todos os arquivos de configuração do Asterisk.

7.8 Definindo extensões

Ao contrário de um PABX tradicional, onde as extensões estão associadas com telefones, interfaces, menus e assim por diante, no Asterisk uma extensão é definida como uma lista de comandos a executar. Os comandos são geralmente executados na ordem definida pela sua etiqueta de prioridade, por outro lado, alguns comandos como o “Dial” e o “Gotoif” tem a capacidade de redirecionar para outro lugar baseado em alguma condição.

Quando uma extensão é discada, o comando etiquetado com prioridade de 1 é executado, seguido pelo comando com prioridade 2 e assim por diante. Isto ocorre até a chamada ser desligada, ou um comando retornar com código de resultado 1 (indicando falha) ou até nenhum comando de maior prioridade existir ou a chamada ser roteada para uma nova extensão.

Em cada contexto você pode definir uma ou mais extensões. Para cada extensão, você define um conjunto de comandos. Para cada comando é dada uma linha separada no seguinte formato:

```
exten=>extension,priority,command(parameters)
```

extension é o nome da extensão, ou o nome literal da extensão ou um padrão. Você repete exatamente o mesmo nome para cada comando para a extensão

Priority é um número interior. Chamá-lo prioridade não é totalmente correto, é apenas o número do comando, usado para numerar os comandos relacionados à uma extensão. O Asterisk irá começar a execução na prioridade 1, se não existir uma linha com prioridade 1, então a extensão não irá bater nenhum número discado. Após executar o comando de prioridade 1, o Asterisk irá executar o comando definido com prioridade 2. Se não existir comando definido com prioridade 2, então o asterisk termina o processamento para esta extensão, mesmo se existir outro comando com prioridade 3 ou mais. Entretanto, alguns comandos, quando executados, fazem o Asterisk pular para uma nova prioridade ao invés de ir para a próxima, dependendo da pró

command é o nome do comando (também chamado aplicação) para se executar.

parameters Os parâmetros dependem do comando. Alguns comandos não têm parâmetros.

Exemplo:

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

Esta é a definição de uma extensão com o nome “123”. Quando uma chamada é feita para a extensão 123, o Asterisk irá responder a chamada ele mesmo, tocar um arquivo de áudio chamado tt-weasels, dar ao usuário a oportunidade de deixar uma mensagem de voz para a caixa 44 e então desligar.

Note que o Asterisk não se importa com a ordem em que você coloca as linhas no arquivo extensions.conf. Você pode misturar as linhas em uma ordem diferente como no seguinte exemplo, e isto não diferencia porque o Asterisk usa a prioridade de cada linha para determinar a ordem de execução.

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

Outras opções para definir extensões incluem a opção normalmente chamada de lógica ex-namorada. A lógica irá bater a extensão discada vindo de fora ou de dentro, baseado na identificação de chamada (callerid) da pessoa que está chamando. Por exemplo:

```
exten=>123/100,1,Answer()
exten=>123/100,2,Playback(tt-weasels)
exten=>123/100,3,Voicemail(123)
exten=>123/100,4,Hangup()
```

Isto irá bater na extensão 123 e executar as seguintes opções somente se o identificador de chamada do usuário é 100. Isto também pode ser feito com um padrão ao invés de um literal.

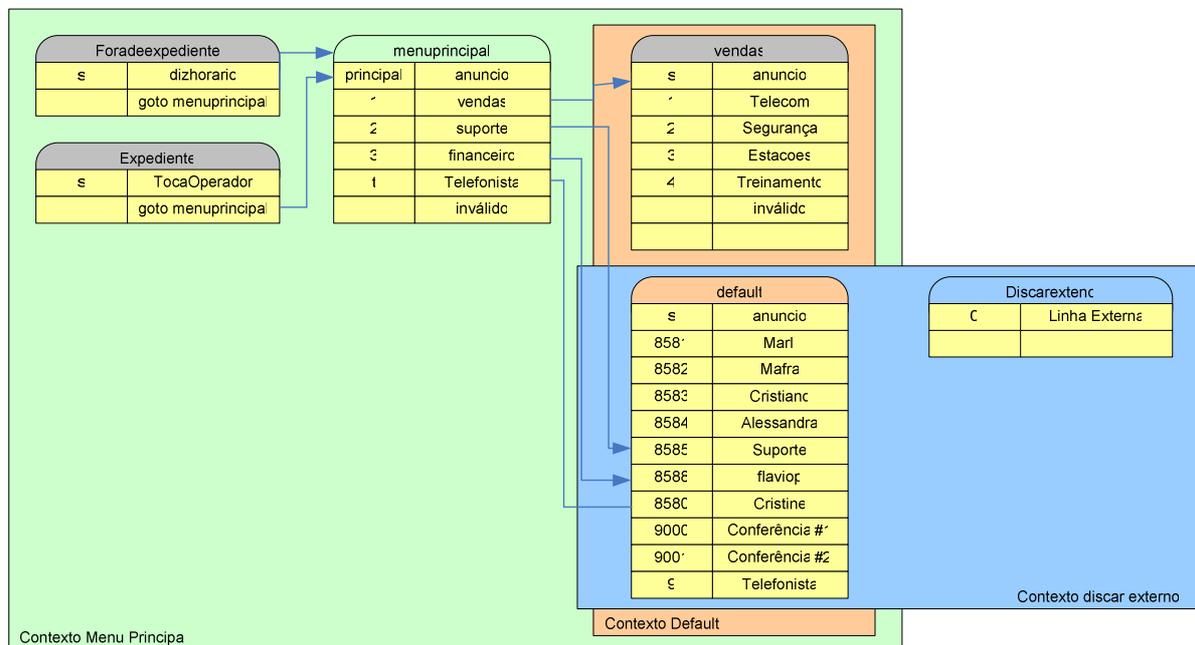
```
exten=>1234/_256NXXXXXX,1,Answer()
```

e assim por diante.

Isto irá bater no 1234 se o CallerID é algo começando com 256. Isto é muito útil para evitar que usuários locais liguem para o seu 0800 e você tenha de pagar por isto.

Um ponto importante a ser entendido é que cada canal VoIP (SIP, H323) ou PSTN (Zap, ISDN) é configurado dentro de um contexto. Portanto, cada canal pertence à um único contexto. Que extensões ou serviços este canal vai poder usar depender das instruções contidas no plano de discagem para aquele contexto.

7.8.1 Contextos baseados em horário



Na figura acima temos um plano de discagem feito de vários contextos. Quando uma chamada está entrando, ela vem por um canal. Este canal pertence à um contexto. No caso acima o canal pertence ao contexto “menuprincipal”. Dentro do contexto menuprincipal, temos dois contextos que foram incluídos “foradoexpediente” e “expediente”. Estes dois contextos foram incluídos usando o comando:

```
include => context|<times>|<weekdays>|<mdays>|<months>
```

Por exemplo:

```
; Este é o horário de expediente
include=>expediente|08:00-18:00|mon-fri|*|*
```

```
; Este é o horário fora de expediente
include=>foradeexpediente|18:00-23:59|*|*|*
include=>foradeexpediente|00:00-08:00|*|*|*
include=>foradeexpediente|*|sat-sun|*|*|*
```

De forma que um dos dois vai ter a extensão s (Start) eles vão tocar a chamada apropriada e transferir para o menu principal. Após o anúncio do menu principal o sistema irá esperar pela discagem de 1 dígito, podendo transferir para vendas ou para qualquer ramal no contexto default. Note que não é possível transferir para uma linha externa, pois o contexto discar externo, não está incluído no contexto menu principal.

Por outro lado se analisarmos a saída de chamadas, vamos imaginar que um canal tenha sido colocado no contexto [discarexterno] (Muito semelhante as classes de ramais nos PABX convencionais), ele pode discar um número externo, e também discar qualquer ramal no contexto default que também foi incluído no contexto [discarexterno]. É importante entender que o que define o contexto de um canal não é a entrada “exten=>” no arquivo extensions.conf, o que define o contexto de um canal é a configuração do canal (sip.conf ou zapata.conf dependendo do tipo de canal). Por isto um canal que tenha sido definido como contexto default pode discar apenas os outros ramais e um que tenha sido definido como [discarexterno], pode discar externamente e também para todos os ramais, pois o contexto default foi incluído no contexto discarexterno.

7.8.2 Discando 0 para pegar a linha externa.

Sempre é bom manter em um PABX novo a mesma experiência que em um PABX anterior. Quando você disca 0 no seu PABX atual, ele dá o tom da linha externa. Quando você disca 0 no Asterisk, ele fica mudo esperando você discar todo o resto. Para resolver esta “encrenquinha” tem um comando muito útil, a instrução ignorepat.

```
[directdial]
ignorepat => 0
exten => 0,1,Dial(Zap/g2/)
exten => 0,2,Congestion

[international]
ignorepat => 0
exten => _00.,1,Dial(Zap/g2/${EXTEN:1})
exten => _00.,2,Congestion
include => longdistance
```

```
[longdistance]
ignorepat =>0
exten => _00XXXXXXXXXX.,1,Dial(Zap/g2/${EXTEN:1})
exten => _00XXXXXXXXXX.,2,Congestion
include => local
```

A linha ignorepat instrui os canais do Asterisk à não tirar o tom de discagem enquanto aquele padrão está sendo discado, de forma que mesmo após discar 0, eles ainda recebem o tom de discagem.

Exemplos de discagem:

O tipo de extensão mais comum é aquela para discar para outra interface. Chamando outra interface é feito com a aplicação Dial. Neste exemplo usamos o comando Dial na sua forma mais simples.

```
exten => 8590,1,Dial(Zap/1,20)
exten => 8590,2,Voicemail(u8590)
exten => 8590,102,Voicemail(b8590)
```

Neste exemplo, uma das poucas exceções para a execução de uma extensão estando fora de ordem. Quando esta extensão é discada, a primeira coisa que o Asterisk tenta discar é a interface Zap/1, por um máximo de 20 segundos. Se a interface está ocupada, ele irá pular para a prioridade n+101 se tal prioridade existir nesta extensão. Neste caso, nós temos a prioridade (102), a qual envia o originador para a caixa postal de correio de voz 8590 com uma mensagem de ocupado (busy). Se simplesmente ninguém atender, então a execução continuará no passo seguinte, onde o originador será colocado no correio de voz com a mensagem de “ninguém atende” (unavailable).

7.8.3 Roteamento pelo originador da chamada

Neste exemplo, já conhecido como extensão anti-ex namorada, mostra como o Asterisk pode rotear baseado não somente no número de quem foi chamado, mas também no número de quem discou.

```
exten => 8590/482518888,1,Congestion
exten => 8590,1,Dial(Zap/1,20)
exten => 8590,2,Voicemail(u8590)
exten => 8590,102,Voicemail(b8590)
```

Neste exemplo, feito com base no exemplo anterior, foi adicionada uma regra especial, onde se o originador é 482518888 (roteamento pelo callerid é indicado colocando um ./ e o identificador da chamada) ele recebe imediatamente o tom de ocupado. Outros originadores são atendidos

normalmente. Um exemplo mais comum de roteamento pelo identificador da chamada é:

7.8.4 Evitando o telemarketing

```
exten => 100/1,Zapateller  
exten => 100,1,Wait(0)  
exten => 100,2,Dial(Zap/1)
```

Neste exemplo, se uma chamada é recebida sem identificador de chamada, então é executada a aplicação Zapateller (Ela toca um tom familiar especial, aquele que você ouve quando chama um número que não está em serviço, frequentemente fazendo com que discadores automáticos ‘normalmente usados em telemarketing’ desconectem).

Se o identificador de chamados CallerID for fornecido, então o aplicativo .Wait. é executado por 0 segundos (em outras palavras não faz nada). Em qualquer caso, o canal Zap/1 toca então indefinidamente (Sem timeout)

7.8.5 Tocando várias extensões

Frequentemente é desejado que uma dada extensão, em primeiro lugar toque um telefone, e então se não existir resposta, toque outro telefone (ou conjunto de telefones). Considere este operador

Considere este exemplo:

```
exten => 9,1,Dial(Zap/1,15)  
exten => 9,2,Dial(Zap/1&Zap/2&Zap/3,15)  
exten => 9,3,Playback(companymailbox)  
exten => 9,4,Voicemail(100)  
exten => 9,5,Hangup
```

Neste exemplo, quando alguém discar 9 para a telefonista, primeiro é tentado o canal Zap/1 que é o telefone da recepcionista. (Se o canal estiver ocupado ou não responder após 15 segundos, tentamos tocar um grupo de telefones, incluindo o da telefonista) por mais 15 segundos. Então isto irá tocar uma mensagem anunciando que ninguém está disponível, e, por favor, deixe uma mensagem na caixa postal da empresa. Finalmente quem chamou será desconectado.

7.8.6 Menu de voz

Um menu de voz é tipicamente implementado dentro do seu próprio contexto.

```
[vendas]
exten => s,1,Background(bemvindo-vendas)
exten => 1,1,Goto(default,100,1)
exten => 2,1,Goto(default,101,1)

[menuprincipal]
exten => s,1,Background(bemvindo-menuprincipal)
exten => 1,1,Goto(vendas,s,1)
exten => 2,1,Dial,Zap/2
exten => 9,1,Directory(default)
exten => 0,1,Dial,Zap/3
```

Um anúncio normalmente é tocado na extensão s, ao entrar no menu. Então a aplicação Background toca uma mensagem, enquanto espera que o usuário disque uma extensão. O exemplo acima apresenta dois menus, um chamado menu principal e outro chamado vendas. Quando alguém entra no contexto menu principal ele irá ouvir um anúncio (Como pressione 1 vendas, 2 suporte, 9 diretório e 0 telefonista). Ao discar 1, o originador é transferido para vendas, discando 2 irá tocar Zap/2, 0 irá tocar o Zap/3 e 9 irá apresentar o diretório da empresa.

O Asterisk pode fazer uso de variáveis globais e de canal para argumentos de determinadas aplicações. Variáveis são expressas no plano de discagem usando `${teste}` onde `.teste.` é o nome da variável. Uma variável pode ser qualquer string alfanumérica começando com uma letra, mas existem algumas variáveis com significados especiais com já visto anteriormente.

Considere o seguinte exemplo:

```
[globals]
Flavio => Zap/1
Daniel => Zap/2&SIP/pingtel
Ana => Zap/3
Cristiano => Zap/4

[mainmenu]
exten => 1,1,Dial(${Daniel}&${Flavio})
exten => 2,1,Dial(${Ana}&${Cristiano})
exten => 3,1,Dial(${Ana}&${Flavio})
```

Organizando o plano de discagem desta forma torna fácil a mudança de interfaces físicas para qualquer usuário em particular. Isto muda todas as referências no plano de discagem instantaneamente.

7.9 Macros

Uma macro é um conjunto de comandos que vai ser executado em seqüência. As macros são usadas primariamente para processar as extensões, de forma que não seja necessário no plano de discagem digitar várias linhas de comando para cada extensão.

Implementação da Macro.

Descrição:

Macro(macroname,arg1,arg2...)

Executa uma macro usando o contexto macro-<macroname>, pulando para a extensão “s”daquele contexto e executando cada passo, e então retornando ao final.

Note que você não pode usar qualquer outra extensão diferente de s para construir a macro, de forma que o controle é retornada ao contexto da chamada quando termino o processamento da macro.

A extensão que foi chamada, contexto, e prioridade são armazenadas em `#{MACRO_EXTEN}`, `#{MACRO_CONTEXT}` e `#{MACRO_PRIORITY}` respectivamente. Os argumentos se tornam `#{ARG1}` e `#{ARG2}` etc no contexto da macro. Se você usar o comando Goto para fora do contexto da Macro, a macro irá terminar e o controle será retornado ao local indicado pelo Goto (Ver comandos e aplicações). Se você colocar `#{MACRO_OFFSET}` no término, a macro tentará continuar na prioridade `MACRO-OFFSET+N+1`, se tal passo existir e N+1 caso contrário.

Códigos de retorno: A Macro retorna 1 se qualquer passo na macro retornar -1, senão retorna 0.

Exemplo:

```
[macro-stdexten]
;Macro extensão padrão
;
;#{ARG1}-Telefone à tocar (ring)
exten=>s,1,Dial(#{ARG1},20) exten=>s,2,Voicemail(u#{MACRO_EXTEN})
exten=>s,3,Goto(default,s,1) ;Se for pressionado #, retorna ao início
exten=>s,102,Voicemail(b#{MACRO_EXTEN}) ;Se ocupado, envia um voicemail
;com o anúncio de busy
exten=>s,103,Goto(default,s,1) ;Se for pressionado #, retorna ao início
```

Para chamar essa macro você usaria:

```
exten => 1234,2,Macro(stdexten,sip/7960mark)
```

Veremos diversas macros em exemplos do arquivo extensions.conf.

7.10 Extensões padrão e prioridades

O Asterisk usa alguns nomes de extensão para propósitos especiais.

- i : Invalid (Inválido).
- s :Start (Inicio).
- h: Hangup (Desligou).
- t : Timeout (Estourou o tempo).
- T: AbsoluteTimeout (Estouro de tempo absoluto).
- o: Operator (Telefonista).
- a: Chamado quando o usuário pressiona “*” durante a mensagem inicial do correio de voz.
- fax: usado para detecção de fax em canais ZAP.
- talk: usado em conjunto com a aplicação [BackgroundDetect](#).

Descrições:

s: Start. Usado para planos de discagem que entram em um contexto sem outras informações como o identificador de chamada. Mesmo se você conhecesse o identificador de chamada, você ainda tem um lugar para começar.

t: Timeout. Usado quando chamadas ficaram inativas após um “prompt” ter sido tocado. Também usado para desligar uma linha que esteja ociosa.

T: AbsoluteTimeout. Usado para chamadas que tenham sido desligadas devido ao AbsoluteTimeout() ter sido alcançado. É útil, por exemplo, para tocar uma notificação com Playback().

h: Hangup. Usado para limpar uma chamada. Pode ser usado para tocar uma mensagem de adeus antes de desligar.

i: Invalid. Usado quando discando para uma extensão desconhecida no contexto ou entrada desconhecida em um menu da URA (Unidade de Resposta Audível).

Os usos destas extensões podem alterar o conteúdo dos seus registros de bilhetagem (CDR – Call Detail Record). Em particular o campo **dst** não irá conter mais o número discado! Para contornar o problema você pode olhar dentro do Dial() com a opção “g” (“vai no contexto”) e considerar o uso do ResetCDR(w) e/ou NoCdr().

hint: mapeia uma extensão para um canal (usado para a lâmpada de ocupado). No Asterisk, um canal pode estar ocupado ou tocando, mas uma extensão é apenas uma string de números que ativam uma ou mais aplicações.

o: Extensão do operador, usado para a saída pressionando 0 no voicemail.

7.11 Padrões de extensão

Quando você define as extensões dentro de um contexto, você pode não somente usar números literais, nem nomes alfanuméricos, mas também pode definir extensões que correspondam a conjuntos de números discados usando padrões. Um nome de extensão é um padrão se ele inicia com um caractere sublinhado “_”. Os seguintes caracteres têm um significado especial:

X	corresponde à qualquer dígito de 0-9
Z	corresponde à qualquer dígito de 1-9
N	corresponde à qualquer dígito de 2-9
[1237-9]	corresponde à qualquer dígito ou letra entre chaves (neste exemplo, 1,2,3,7,8,9)
.	coringa, corresponde à um ou mais caracteres

Considere o contexto “routing”

Extensão	Descrição
_61XX	Escritório São Paulo
_63XX	Escritório Rio de Janeiro
_62XX	Escritório Salvador
_7[1-3]XX	Escritório Recife
_7[04-9]XX	Escritório Curitiba

Este contexto, dado o nome “routing”, envia chamadas para vários servidores de acordo com sua extensão. Esta organização decidiu que todas as

extensões de telefone terão quatro dígitos. Se um usuário discar uma extensão começando com 61 ou 63, isto será enviado ao escritório de São Paulo; 62 para o escritório do Rio, 71,72 e 73 para Recife e 70, 74, 75, 76, 77, 78, 79 vai para o escritório de Curitiba.

Mais exemplos de padrões

<code>_NXXXXXX</code>	Corresponde à um número de telefone de 7 dígitos
<code>_021XXXXXXXXXX</code>	Corresponde à uma saída pela Embratel

7.12 A base de dados do Asterisk

A base de dados do Asterisk usa a versão 1 do Berkley DB que é parecido com o registry do Windows. Este banco de dados pode ser usado pelo Asterisk para armazenar dados temporários, e configurações. Um exemplo de uso é a transferência com consulta, onde se o telefone está ocupado ele guarda a extensão em uma base de dados e fica re-tentando até conseguir.

Os dados são agrupados em famílias e identificados com uma chave que é única dentro da família.

7.12.1 Famílias

cidname: usado pelo comando LookupCIDName

blacklist: usado pelo comando LookupBlacklist

7.12.2 Aplicações

Existem operações que você pode usar dentro do plano de discagem em `extensions.conf`:

- **DBdel:** Deleta uma chave da base de dados.
- **DBdeltree:** Deleta uma família ou árvore da base de dados.
- **DBget:** Recupera um valor da base de dados.
- **DBput:** Armazena um valor na base de dados.

8.4.3 Exemplo de uso do Asterisk DB.

Siga-me, Siga-me quando ocupado.

Se você usar este exemplo, um telefone pode discar *21*<número> para Siga-me imediato ou *61* para siga-me se ocupado, e #21# ou #61# para cancelar a configuração.

Abreviações usadas:

CFIM – família base de encaminhamento.

CFBS – família base de encaminhamento quando ocupado.

```
[macro-stdexten]
;${ARG1}-Extensão
;${ARG2}-Dispositivo à tocar

exten=s,1,DBget(temp=CFIM/${ARG1}) ;Pega a chave CFIM, se
; não existir vai p/102
exten=s,2,Dial(Local/${temp}@pbx/n);Transfere
exten=s,3,Dial(${ARG2},20)
exten=s,4,DBget(temp=CFBS/${ARG1}) ;Pega a chave CFBS, se não
; existir vai p/105
exten=s,5,Dial(Local/${temp}@pbx/n) ;Encaminha quando ocupado ou
; não disponível.

exten=s,102,Goto(s,3)
exten=s,105,Busy
```

Notas:

- Esta macro lê as variáveis armazenadas na base de dados do Asterisk.
- A Macro executa o Dial(Local/<number>@pbx) se um número de redirecionamento for encontrado.
- Se o comando DBget não encontrar uma chave, ele vai para a prioridade 102.

```
[apps]
;Siga-me
exten=>_*21*X.,1,DBput(CFIM/${CALLERIDNUM}=${EXTEN:4})
exten=>_*21*X.,2,Hangup
exten=>#21#,1,DBdel(CFIM/${CALLERIDNUM})
exten=>#21#,2,Hangup

;Siga-me se ocupado ou não disponível
exten=>_*61*X.,1,DBput(CFBS/${CALLERIDNUM}=${EXTEN:4})
exten=>_*61*X.,2,Hangup
```

```
exten=>#61#,1,DBdel(CFBS/${CALLERIDNUM})  
exten=>#61#,2,Hangup
```

```
[pbx]  
exten=7000,1,Macro(stdexten,7000,MGCP/aaln/1@<myphone>)  
exten=7001,1,Macro(stdexten,7001,SIP/oej)
```

Não esqueça de incluir a extensão apps no seu contexto.

7.13 Sumário

Neste capítulo você aprendeu que o plano de discagem é a peça principal de configuração do Asterisk. Ele é formado de contextos extensões e prioridades. Você aprendeu a interligar dois aparelhos de PABX usando o comando Switch. Aprendeu a usar variáveis do tipo global de canal e de ambiente. Entendi como funciona o processamento das extensões e o sistema de inclusão de contextos. Aprendeu a criar macros para simplificar o trabalho de configurar as extensões. E por fim entendeu como funcionam e para o que servem as extensões padrões e as prioridades.

7.14 Questionário

1. Na configuração da seção [general] o valor padrão da opção writeprotect=no. Sendo assim se você emitir o comando save dialplan na linha de comando CLI>. (Marque todas que se aplicam).

- O Asterisk irá sobrescrever o extensions.conf com a configuração atual.
- Todos os comentários serão perdidos.
- Será feito um backup de extensions.conf em extensions.bak.
- A opção static=yes deve estar configurada para poder salvar o plano de discagem.

2. Normalmente as variáveis globais vão estar escritas em maiúsculas enquanto as variáveis de canal estarão começando com maiúscula e restante em minúsculas. Isto não é obrigatório, mas facilita a leitura.

- Verdadeiro
- Falso

3. E extensão 's' é usada como extensão de início dentro de um contexto, ela é normalmente usada nos seguintes casos.

- Na entrada de uma chamada sem CallerID
- Como entrada de um menu chamado pelo comando Background().
- Na entrada de uma chamada com CallerID
- Como entrada de um contexto que foi direcionado pelo Goto().

4. Cite quatro situações onde contextos poderiam ser utilizados:

5. Switches são usados para direcionar para outro PABX. A afirmação acima está:

- Correta
- Incorreta

6. Para usar uma variável no plano de discagem no Asterisk você deve usar o seguinte formato:

- `$(varname)`
- `{varname}`
- `$(varname)`
- `${varname}`

7. As variáveis disponíveis no Asterisk são: (Escolha 3)

- Constantes
- Variáveis públicas
- Variáveis de ambiente
- Variáveis globais
- Variáveis privadas
- Variáveis de canal

8. Para obter o comprimento de uma string você pode usar a função _____.

9. Para concatenar strings basta colocar uma do lado da outra como no exemplo abaixo. A afirmação está:

```
`${foo}``${bar}`  
555`${Onumero}`  
`${PrefixoLongaDistancia}555`${ONumero}`
```

- Correta
- Incorreta

10. Uma macro pode ser usada para automatizar uma série de operações em seqüência para uma extensão específica. O primeiro argumento passado pela chamada da macro é o:

- `${ARG1}`
- `${ENV1}`
- `${V1}`
- `${X}`

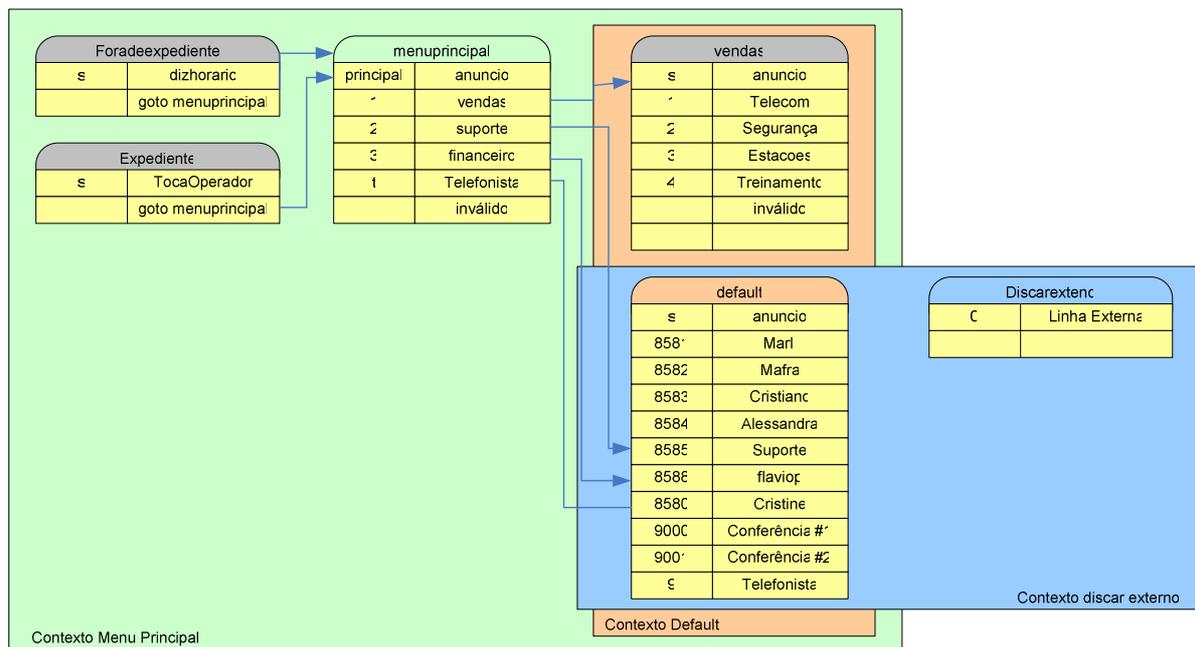
Página deixada intencionalmente em branco

Construindo o plano de discagem

8.1 Objetivos

- Um plano de discagem na prática.
 - Configurando os canais.
 - Configurando o arquivo `extensions.conf`.
 - Configurando a recepção das chamadas.
 - Configurando mensagem por horário.
- As aplicações mais usadas.
 - `Dial()`.
 - `Answer()`.
 - `Goto()`.
 - `Background()`.

8.2 Um plano de discagem na prática



Vamos construir um plano de discagem na prática para uma pequena central telefônica. A central possui as seguintes características e requisitos:

- 4 troncos analógicos.
- 16 ramais IP baseados em SIP.
- 3 classes de ramal.
 - restrito (só interno, 0800 e free).
 - ddd (ligações nacionais).
 - ddi (ligações internacionais).
- Mensagem quando fora do expediente
- Auto-atendimento

8.3 Passo 1 - Configurando os canais.

8.3.1 Troncos analógicos (zapata.conf)

Em primeiro lugar vamos configurar os troncos analógicos. Estes troncos são interfaces do tipo fxo. Você pode usar uma placa Digium do tipo T400P ou

quatro placas modem Motorola ou Intel (Aquela em que de cada 5 que se compra uma da certo).

Vamos assumir neste caso que o drivers do hardware drivers da placa de telefonia (zaptel.conf) já estão configurados e carregados. No arquivo zapata.conf você configura os canais zap correspondentes aos troncos da central.

```
signalling=fxo_ks
language=en
context=entrada
group=1
channel => 1-4
```

8.3.2 Canais SIP (sip.conf)

Vamos escolher a faixa de numeração do número 2000 ao número 2100. Dois codecs vão ser utilizados, GSM e G711 ulaw. O primeiro para telefones ligados pela Internet ou em linhas de baixa velocidade, o segundo para os telefones locais. Vamos convencionar que ramais dos números 2000 até o 2039 são restritos, dos ramais 2040 até o 2059 podem fazer DDD e do número 2060 em diante podem fazer DDI.

```
[general]
disallow=all
allow=gsm
allow=ulaw
port = 5060
bindaddr = 0.0.0.0
context = restrito

[2000]
type=friend
username=20
secret=senha
host=dynamic
mailbox=20
context=restrito
canreinvite=yes

[2040]
type=friend
username=20
secret=senha
host=dynamic
mailbox=20
context=ddd
```

```
canreinvite=yes  
dtmfmode=rfc2833
```

```
[2060]  
type=friend  
username=20  
secret=senha  
host=dynamic  
mailbox=20  
context=ddd  
canreinvite=yes  
dtmfmode=rfc2833
```

8.4 Passo 2 - Configurando o plano de discagem

Agora vamos configurar o plano de discagem para atender as especificações da central. Todas as configurações desta seção são feitas no arquivo `extensions.conf`.

8.4.1 Definindo os ramais

```
[ramais]  
exten=>_20XX,1,Dial(SIP/${EXTEN},20,r)  
exten=>_20XX,2,Hangup()
```

8.4.2 Definindo a saída para DDD

```
[ddd]  
  
; Discagem para DDD para número de 7 dígitos (BrT)  
exten=>_00XXXXXXXXXX,1,Dial(Zap/g1/${EXTEN:1},20,Tt)  
  
; Discagem para DDD para número de 8 dígitos  
exten=>_00XXXXXXXXXX,1,Dial(Zap/g1/${EXTEN:1},20,Tt)
```

8.4.3 Definindo a saída para DDI

```
[ddi]
; Discagem para DDI
exten=>_000XXXX.,1,Dial(Zap/g1/${EXTEN:1},20,Tt)
```

8.4.5 Definindo as classes de ramal

```
[restrito]
include=>ramais
```

```
[ddd]
include=>ramais
include=>ddd
```

```
[ddi]
include=>ramais
include=>ddd
include=>ddi
```

8.5 Recepção das chamadas

Para receber as chamadas, vamos usar dois contextos um para ligações durante o expediente e outra para ligações fora do expediente. A ligação vai ser atendida por um auto-atendedor e transferida diretamente para um ramal ou para um outro menu.

8.5.1 Menus: Expediente e fora do expediente

Nos menus abaixo, o sistema irá tocar uma mensagem avisando ao originador da chamada que a empresa se encontra fora do expediente, mas permitindo que o usuário disque um ramal (eventualmente alguém está fazendo hora-extra).

```
[entrada]
include=>expediente|08:00-18:00|mon-fri|*|*
```

```
; Este é o horário fora de expediente
include=>foradeexpediente|18:00-23:59|*|*|*
include=>foradeexpediente|00:00-08:00|*|*|*
include=>foradeexpediente|*|sat-sun|*|*|*
```

```
[expediente]
exten=>s,1,Goto(menuprincipal,s,1)
```

```
[fora do expediente]
```

```

exten=>s,1,Background(foradoexpediente)
exten=>s,2,hangup()
exten=>i,1,hangup()
exten=>t,1,hangup()
include=>ramais

```

8.5.2 Menus: Principal e Vendas

Durante o expediente o usuário é recebido pelo menu com autoatendimento. A mensagem será algo como: *“Bem vindo, você ligou para a NomedaEmpresa, Disque 1 para vendas, 2 para suporte e três para treinamento ou o ramal desejado”*.

```

[globals]
TELEFONISTA=SIP/2060
VENDAS=SIP/2035
SUPORTE=SIP/2004
TREINAMENTO=SIP/2036

[menuprincipal]
exten=> s,1,Background(bemvindo-menuprincipal)
exten=>1,1,Goto(vendas,s,1)
exten=>2,1,Goto(suporte,s,1)
exten=>3,1,Goto(treinamento,s,1)
exten=>i,1,Playback(Invalido)
exten=>i,2,hangup()
exten=>t,1,Dial(${TELEFONISTA},20,Tt)
include=>ramais

[vendas]
exten=>s,1,Dial(${VENDAS},20,Tt)

[suporte]
exten=>s,1,Dial(${SUPORTE},20,Tt)

[treinamento]
exten=>s,1,Dial(${TREINAMENTO},20,Tt)

```

Com isto a maior parte da funcionalidade do seu plano de discagem está pronta e agora você pode começar a testar e melhorar sua configuração. Na próxima seção vamos ver mais detalhes sobre as aplicações mais comuns.

8.6 Visão geral das aplicações

Nesta seção vamos ver com mais detalhes os seguintes comandos:

- Dial()
- Background()
- Answer()
- Goto()
- Directory()

8.6.1 O Comando Dial()

Tenta estabelecer uma nova conexão de saída em um canal e então conecta ao canal de entrada existente.

Sintaxe:

```
;Discando um canal  
Dial(tipo/identificador,timeout,opções, URL)
```

```
Disca múltiplos canais  
Dial(tipo1/identificar1&tipo2/identificar2/tipon/identificadorn,timeout,opções, URL)
```

Tenta discar para os canais especificados, (Cada um especificado pelo tipo e identificador) simultaneamente. O primeiro canal que responde “vence”, e então todos os outros canais são desligados.

O canal originador que dispara este comando Dial é então “Respondido”, se necessário os dois canais são conectados juntos (“bridged”) fazendo com que a conversação ocorra entre os dois. Quando o canal que disparou o comando Dial desliga o comando Dial termina.

Parâmetros:

Tipo: especifica o tipo de canal. Dever ser um dos canais registrados como “Zap”, “SIP”, “IAX2” e assim por diante.

Identificador: especifica o número de telefone à ser discado neste canal. O formato de “número de telefone” depende do canal e pode conter parâmetros adicionais (Ex. um toque de campanha personalizado) específico para canal em questão. O comando Dial simplesmente passa o identificador para o módulo do canal processar da maneira mais apropriada.

Se você deseja especificar mais de um canal para que o comando Dial tente – lembre-se que ele irá discar todos simultaneamente – Separe-os com o símbolo &.

O parâmetro **timeout** é opcional. Se não for especificado, o comando Dial irá esperar indefinidamente, saindo apenas quando o canal originador desligar; ou todos os canais discados retornarem uma condição de ocupado ou erro. Senão ele especifica o máximo de tempo que o comando Dial irá esperar para que um canal responda.

O parâmetro **opções**, que é opcional é uma string contendo zero ou mais das seguintes “flags” e parâmetros:

- t: permite ao usuário chamado transferir a chamada.
- T: permite ao usuário chamador transferir a chamada.
- r: Gera um tom de campainha para o chamador, não passando nenhum áudio de canal chamado até alguém responder. Use com cuidado e não coloque isto por default em todas as instruções dial, pois você está ocultando do usuário à informação de progresso da chamada.
- m: Provê música em espera para o originador até que o usuário chamado responda. Isto é mutuamente exclusivo com a opção r.
- M(x): Executa a macro(x) na conexão da chamada.
- h: Permite ao usuário chamado desligar discando *.
- H: Permite ao usuário originador desligar discando *.
- C: Reseta o CDR (Registros de Bilhetagem). Isto é como usar o comando NoCDR.
- P(x): Usa o **PrivacyManager**, usando o X como banco de dados. (x é opcional).
- g: Quando a pessoa chamada desliga, sai para executar mais comandos no contexto atual.
- A(x): Toca um anuncio (x.gsm) para a pessoa chamada.

- S(n): Desliga a chamada n segundos após a pessoa chamada atender.
- D(dígitos): Após a parte chamada responder, envia dígitos como um fluxo de DTMF, então conecta a chamada ao canal originador.
- L(x[:y][:z]): Limita a chamada à ‘x’ ms, aviando quando restam ‘y’ ms, repetidos à cada ‘z’ ms). Apenas ‘x’ é necessário.
 - As seguintes variáveis especiais são opcionais para limitar as chamadas .
 - LIMIT_PLAYAUDIO_CALLER – yes/no (default yes) – Toca o som para o originador.
 - LIMIT_PLAYAUDIO_CALLEE – yes/no (default yes) – Toca o som para o que foi chamado.
 - LIMIT_TIMEOUT_FILE – Arquivo para tocar quando o tempo estoura.
 - LIMIT_CONNECT_FILE – Arquivo à tocar quando a chamada inicia.
 - LIMIT_WARNING_FILE – Arquivo para tocar como aviso se ‘y’ está definido. Se LIMIT_WARNING_FILE não estiver definido, então uma macro de som especial para falar automaticamente quanto tempo resta (“Você tem [XX] minutos [YY] segundos”).
- f: Força o callerid a ser com configurado como uma extensão da linha que esta fazendo ou redirecionando a chamada. Por exemplo, algumas operadoras não permitem callerids de outras extensões que aquelas que estão designadas para você.

O parâmetro opcional URL irá também ser enviado à parte chamada quando da conexão com sucesso. Se a tecnologia de canal suportar o envio de URLs desta forma.

8.6.2 O comando Background()

Toca um arquivo de áudio, mas retorna o controle imediatamente.

Sintaxe:

Background(nome do arquivo)

Começa tocando um dado arquivo de áudio, mas imediatamente retorna, permitindo ao arquivo de áudio tocar no fundo enquanto os próximos comandos (se quaisquer) executem. (Comparando com o comando Playback, que também toca um arquivo de áudio, mas que espera até o áudio ter terminado antes de retornar o controle)

Se você fizer uma segunda chamada ao aplicativo Background, enquanto o primeiro ainda estiver tocando, o segundo arquivo será enfileirado para tocar após o término do primeiro.

Se o Asterisk ficar sem comandos para executar para uma dada extensão, mas a conexão ainda estiver ativa (O usuário ainda não desligou), o Asterisk irá terminar de tocar o áudio em background antes de ativar o temporizador ResponseTimeout.

Um uso do comando **Background()** é tocar uma mensagem de menu (Pressione 1 para vendas, 2 para suporte,...”), mas não ter nenhum comando adicional. Deste modo o Asterisk, não tendo comandos para executar, espera por quaisquer dígitos DTMF que o usuário discar enquanto ele continua tocando o arquivo de áudio. Se o usuário discar um número de extensão que é válido no contexto atual, o arquivo de áudio será interrompido e o Asterisk irá pular para a extensão discada. Se o usuário não discar uma extensão, então o Asterisk irá terminar de tocar o arquivo de áudio no fundo e então silenciosamente esperar pelo tempo em segundos definido pela variável **ResponseTimeout** até que o usuário disque algo. Se estourar o tempo o controle vai para a extensão t (se definida neste contexto, senão desliga).

Se o usuário discar uma extensão que é inválida no contexto atual, o controle pula para a extensão i (Se definida neste contexto, senão desliga).

Exemplo:

```
[incoming]
exten=>s,1,Answer
exten=>s,2,ResponseTimeout(5)
exten=>s,3,Background(meumenu)
exten=>s,4,Background(escolhaalgo)
exten=>t,1,Goto(s,2)
exten=>i,1,Playback(pbx-invalid)
exten=>i,2,Goto(s,2)
exten=>1,1,DoSomething
exten=>2,1,DoSomethingElse
exten=>3,1,Dial(...)
```

Neste exemplo, uma chamada vinda de um contexto de entrada é imediatamente respondida pelo Asterisk (passo s,1). Nós configuramos o tempo de resposta em 5 segundos (passo s,2). Então nós começamos a tocar o áudio do arquivo “meumenu.gsm” (passo s,3). Então nós enfileiramos o áudio do arquivo “escolhaalgo.gsm” para tocar uma vez que o primeiro áudio tenha terminado. (passo s,4). Como não existe passo s,5, o Asterisk continua tocando o meumenu.gsm e escolhaalgo.gsm até ter terminado e então silenciosamente espera mais cinco segundos por uma resposta. Se a qualquer momento, com o áudio tocando ou não, o usuário disca uma extensão então o Asterisk para de tocar o áudio e pula para a extensão. Se ele disca “1”, “2” ou “3”, então o Asterisk irá pular para a extensão apropriada.

Se o usuário disca qualquer outro número, o Asterisk pula para a extensão “i” e toca um áudio do arquivo “pbx-invalid.gsm” então volta para o início. Se o usuário não digitar nenhuma extensão no fim de 5 segundos de silêncio, então o Asterisk irá pular para a extensão “t”, e de lá seguem o **Goto** para voltar ao início novamente.

Note que se você usar o comando **Background** em uma macro ele vai sempre chamar uma extensão fora do contexto de quem chamou.

8.6.3 O comando Answer()

Descrição:

Responde um canal se ele estiver tocando

Sintaxe:

```
Answer()
```

Se o canal estiver tocando, responde ele, senão não faz nada.

Códigos de retorno:

Retorna 0 a menos que ele tente responder ao canal e falhe.

8.6.4 O comando goto()

Descrição:

Pula para uma prioridade em particular, extensão ou contexto.

Sintaxe:

```
Goto([[contextl]extension]priority)
```

Configura a prioridade para um valor específico opcionalmente configurando a extensão opcionalmente o contexto também. A extensão BYEXTENSION é especial, pois usa a extensão atual, deste modo permitindo que você vá para um contexto diferente sem especificar uma extensão específica.

Variantes válidas do comando:

```
Goto(context,extension,priority)
```

```
Goto(extension,priority)
```

```
Goto(priority)
```

Aviso: Inserindo um espaço após as vírgulas separando os parâmetros irá resultar em resultados inesperados.

Exemplo:

```
Goto(confexisting,1,1) irá procurar por uma extensão “ 1”, com um espaço na frente.
```

Códigos de retorno:

Sempre retorna 0, mesmo se um dado contexto, extensão, ou a prioridade seja inválido.

Exemplo:

```
exten=>1,1,Goto(submenu,s,1)  
exten=>600,4,Goto(s,6)
```

8.7 Questionário

1. Para incluir um contexto que depende do horário, você pode usar:

`include=> context|<times>|<weekdays>|<mdays>|<months>`

O comando abaixo:

`include=>expedientel08:00-18:00|mon-fri|*|*`

- Executa as extensões de segunda à sexta das 08:00 às 18:00.
- Executam as opções todos os dias em todos os meses.
- O comando é inválido.

2. Quando o usuário disca “0” para pegar a linha o Asterisk automaticamente corta o áudio. Isto é ruim, pois o usuário está acostumado a discar o “0” e ouvir o tom externo de discagem. Para criar este comportamento que o usuário está acostumado, pode-se usar comando _____.

3. Os comandos:

```
exten => 8590/482518888,1,Congestion
exten => 8590,1,Dial(Zap/1,20)
exten => 8590,2,Voicemail(u8590)
exten => 8590,102,Voicemail(b8590)
```

Faz com que um usuário que ligou para a extensão 8590:

Receba um sinal de ocupado se o CallerID=482518888

Receba um sinal de ocupado independente do número discado

Vá para o canal ZAP/1

Vá para o VoiceMail() se o canal ZAP/1 estiver ocupado ou não atender, exceto no caso onde o CallerID for 482518888

4. Para concatenar várias extensões basta separá-las com o sinal ____.

5. Um menu de voz normalmente é criado com o comando inicial _____.

6. Você pode incluir arquivos dentro dos seus arquivos de configuração com o comando _____.

7. O Asterisk permite que se use no plano de discagem uma base de dados baseada em:

- Oracle
- MySQL
- Berkley DB
- PostgreSQL

8. Quando você usa o comando Dial(tipo1/identificar1&tipo2/identificar2) com vários identificadores, o Asterisk disca para cada um na seqüência e espera 20 segundos ou o tempo de timeout antes de passar para o outro número. A afirmação é:

- Falsa
- Verdadeira

9. No comando Background a música de fundo tem de ser tocada inteiramente antes que o usuário possa digitar algo. A afirmação é:

- Falsa
- Verdadeira

10. Os formatos válidos par o comando Goto() são:

- Goto (context,extension)
- Goto(context,extension,priority)
- Goto(extension,priority)
- Goto(priority)

Configurando recursos avançados

Neste capítulo veremos alguns recursos avançados do Asterisk PBX

9.1 Objetivos

Ao final deste capítulo você deve estar apto a configurar os seguintes recursos.

- Gerenciamento de chamadas.
 - Call Parking (Estacionamento de chamadas).
 - Call Pickup (Captura de chamadas).
 - Call Transfer (Transferência).
 - Call Conference (Conferência simples).
- Sala de conferência.
- Música em espera.

9.2 Suporte aos recursos de PABX

Nesta seção vamos ver uma série de recursos que foram introduzidos na versão 1.0 do Asterisk para o tratamento de chamadas. O arquivo `features.conf` controla a maior parte destes recursos.

É importante entender que alguns recursos são implementados pelo telefone e outros pelo Asterisk. Outro ponto importante é o caso de telefones IP baseados em SIP com a configuração `canreinvite=yes`. Neste caso a comunicação da mídia ocorre entre os dois telefones, se o usuário pressionar #, quem vai interpretar é o outro telefone e não necessariamente o Asterisk.

9.2.1 Recursos com suporte para todo tipo de canal

- **Música em espera:** Veja a aplicação `MusicOnHold()`.

- **Estacionamento de chamadas:** Suportado na instalação padrão.
- **Captura de chamadas:** Suportado na instalação padrão (*8).
- **Gravação de chamadas:** Usando a aplicação Monitor().
- **Conferência:** Usando a aplicação MeetMe().

9.2.2 Recursos com suporte em telefones SIP

- **Chamada em espera:** Normalmente implementado pelo telefone.
- **Transferência às cegas:** Implementado no * opcional no telefone.
- **Chamada em espera com consulta:** Implementado no telefone
- **Siga-me incondicional:** *
- **Siga-me se não atender:** Criado no plano de discagem.
- **Siga-me se estiver ocupado:** Criado no plano de discagem.
- **Transferência assistida:** *
- **Chamada a três:** Normalmente implementado pelo telefone.
- **Filtragem das chamadas:** Criado no plano de discagem.
- **Rediscagem automática:** É possível usando AGI.
- **Rediscagem manual:** Telefone
- **Não perturbe:** *
- **Mensagem em espera:** *, mas deve ser suportado no telefone.

- **Indicação de chamada em espera: ***, deve ser suportado no fone.

9.2.3 Para telefones analógicos (Zap).

- **Captura de chamadas:** Suportada na instalação padrão.
- **Chamada em espera:** Normalmente implementado no telefone.
- **Transferência às cegas: ***
- **Chamada em espera com consulta:** Implementado no telefone.
- **Siga-me incondicional: ***
- **Siga-me se não atender:** Criado no plano de discagem.
- **Siga-me se estiver ocupado:** Criado no plano de discagem.
- **Transferência assistida: ***
- **Chamada a três:** Normalmente implementado pelo telefone.
- **Filtragem das chamadas:** Criado no plano de discagem.
- **Rediscagem automática:** É possível usando AGI.
- **Rediscagem manual: ***
- **Não perturbe: ***
- **Mensagem em espera: ***, mas deve ser suportado no telefone.
- **Indicação de chamada em espera: ***, deve ser suportado no fone.
- **Encontre-me: ***

9.3 Estacionamento de chamadas

9.3.1 Descrição:

É usado para estacionar uma chamada. Isto auxilia quando você está atendendo em um telefone fora da sua sala. Estaciona a chamada em uma determinada extensão e quando se move de volta a sua sala digita a extensão onde a chamada está estacionada.

Por default a extensão 700 é usada para estacionar uma chamada. No meio de uma conversa, pressione # para iniciar uma transferência, então disque 700. O Asterisk irá agora anunciar a extensão do estacionamento, mais provavelmente 701 ou 702. Desligue, e o originador da chamada será colocado em espera na extensão anunciada. Vá para um telefone diferente e disque 701 (a extensão anunciada) e então você pode continuar conversando. Se o originador ficar estacionada por mais tempo que o limite então o Asterisk irá tocar a extensão discada originalmente.

9.3.2 Lista de tarefas para configuração

1. Colocar chamadas no estacionamento (Obrigatório).
2. Configurar o arquivo features.conf (Opcional).

9.3.2.1 Habilitando chamadas em estacionamento: (obrigatório)

Inclua no arquivo extensions.conf a linhas

```
include=>parkedcalls
```

9.3.2.2 Configure o arquivo extensions.conf (opcional)

```
parkext=>700      ;Qual extensão discar para estacionar
parkpos=>701-720; ;Que extensões estacionar chamadas
context=>parkedcalls ;Que contexto as extensões estacionadas
                ;estão,precisa a instrução INCLUDE no          ; arquivo
extensions.conf.
parkingtime=>45   ;Número de segundos que uma chamada
                ;pode ser estacionada (padrão 45              ; segundos).
                ;Após o tempo expirar, a chamada volta      ;
para o ramal original.
```

Notas:

- A extensão de estacionamento não será mostrada no comando show dialplan.

- É preciso reiniciar o asterisk após modificar o arquivo features.conf. Um reload simples não funciona.
- É preciso poder transferir para poder estacionar uma chamada, então verifique as opções t e T do comando Dial().
- As transferências baseadas no Asterisk funcionam somente se o Asterisk estiver no meio do caminho (canreinvite=no no sip.conf por exemplo).

9.4 Captura de chamadas - Call-pickup

9.4.1 Descrição

A captura permite que você puxe um chamado de um colega no mesmo grupo de chamadas. Isto evita que você tenha de levantar para atender um telefone do seu vizinho que não para de tocar.

Discando *8 você pode capturar uma chamada no seu grupo de chamadas. Este recurso pode ser alterado via arquivo features.conf.

9.4.2 Lista de tarefas para habilitar a captura.

1. Configure o grupo à que suas extensões pertencem dentro de um dos seguintes arquivos sip.conf, iax.conf, zapata.conf (obrigatório).
2. Configure o arquivo features.conf

<code>pickupexten=*8;</code> Configura a extensão de captura. O padrão é *8

9.5 Transferência de chamadas - Call Transfer

9.5.1 Descrição

A transferência de chamadas pode ser feita de suas formas. A transferência às cegas (blind transfer) e a transferência assistida.

Na transferência as cegas você usa os dígitos que estiverem definidos no parâmetro **blindxfer** no arquivo features.conf. Por exemplo, se o **blindxfer** for configurado como “#1”, você disca este número e o número de destino e desliga o telefone.

O **atxfer** permite a transferência assistida ou supervisionada. Funciona assim:

Durante uma conversa, você disca a seqüência de transferência assistida #2, O asterisk fala “transfer” e então dá a você um tom de discagem, enquanto coloca o originador em música de espera. Você disca para quem vai transferir e fala com ele para introduzir a chamada, então desliga, a transferência está completa. Se a pessoa transferida não quiser atender, ela simplesmente desliga o telefone e você recebe a chamada.

9.5.2 Lista de tarefas para configurar

1. Se o telefone for padrão SIP assegure-se que a opção `canreinvite=no` está habilitada (obrigatório).
2. Edite o arquivo `features.conf` para configurar os dígitos para transferência de chamadas (opcional).

9.5.3 O arquivo `features.conf`

```
transferdigittimeout=>3      ;Segundos a esperar entre dígitos
                             ; quando transferindo uma chamada.

xfersound=beep              ;Para indicar uma transferência assistida
                             ;completada.

xferfailsound=beeperr;Para indicar uma transferência que falhou

[featuremap]
blindxfer=>#1                ;Transferência desassistida
disconnect=>*0                ;Desconectar
automon=>*1                   ;Gravação em um toque
atxfer=>*2                    ;Transferência assistida
```

9.6 Conferência – Call Conference

A conferência no Asterisk é feita usando o aplicativo `MeetMe()`. O `MeetMe()` é uma ponte de videoconferência, muito simples de usar com a vantagem de funcionar com qualquer tipo de canal. Alguns telefones padrão SIP, por exemplo, têm seus próprios mecanismos de conferência que também podem ser usados, mas o método padrão para conferência é o `MeetMe()`. Vamos dar uma olhada mais a fundo no aplicativo.

9.6.1 Formato:

```
MeetMe(Confno[loptions][lpassword])
```

9.6.2 Descrição:

Coloca o usuário em uma sala de conferência específica. Se o número da conferência for omitido, será pedido um ao usuário.

A string da opção pode conter ou não um ou mais dos seguintes caracteres.

- 'm'—Configura o modo ouvir apenas.
- 't'—Configura o modo apenas fala.
- 'i'— Anuncia a entrada e saída de usuário.
- 'p'—Permite ao usuário sair da conferência pressionando #.
- 'X'—Permite ao usuário sair da conferência discando uma extensão válida de um dígito.

A extensão é executada em $\${MEETME_EXIT_CONTEXT}$ ou no contexto atual se a variável não tiver sido definida.

- 'v'—modo vídeo.
- 'q'—modo quieto (não toca os sons de entrada e saída).
- 'd'—adiciona dinamicamente a conferência.
- 'D'—dinamicamente adiciona a conferência, pedindo pelo pin (personal identification number). Quando ouvir o pedido do pin, se o usuário não quiser um pin designado para a conferência ele deve discar '#'.
• 'M'—Habilita música em espera quando a conferência tem um único usuário.
- 'e'—Seleciona uma conferência vazia.
- 'E'—Seleciona uma conferência vazia sem pin.

- 'b'—Roda o script AGI especificado em `#{MEETME_AGI_BACKGROUND}` o padrão é `conf-background.agi` (para canais zap apenas, não funciona com canais não zap na mesma conferência).
- 'a'—Configura o modo de administração.
- 'A'—Configura o modo marcado.
- 's' —Apresenta um menu (usuário ou administrador) quando '*' é recebido (envia para o menu).
- 'x'—Fecha a conferência e desliga em todos os outros quando o último usuário marcado sai.
- 'w'—Espera até o usuário marcado entrar na conferência antes de permitir que alguém fale. Todos os outros usuários conectados vão ouvir música em espera até o usuário marcado entrar.

9.6.3 Códigos de retorno:

A aplicação sempre retorna -1.

9.6.4 Detalhe das opções:

Nem a opção 's' nem a opção 'p' funcionam se usados junto com 'X'.

- Usuário (sem a opção 'a' sendo configurada): Ao pressionar * toca o menu de voz "Por favor, pressione 1 para emudecer e voltar".
- Admin (com a opção a sendo configurada): Ao pressionar * toca o menu de voz "Por favor, pressione 1 para emudecer/desemudecer, 2 para travar/destravar esta conferência".

9.6.5 Lista de tarefas de configuração

1. Escolha as extensões para a sala de conferência (obrigatório)
2. Inclua o comando MeetMe() no arquivo `extensions.conf` (obrigatório)
3. Edite o arquivo `meetme.conf` para configurar as senhas (opcional)

9.5.6 Exemplos

Exemplo 1: Sala de conferência simples

1. Coloque no arquivo extensions.conf a sala de conferência 101.

```
exten=>500,1,MeetMe(101|123456)
```

2. Defina no arquivo meetme.conf a senha (pin) 123456 na extensão 500

Exemplo 2 – Sala de conferência autenticada

1 – Defina a sala de conferência no arquivo extensions.conf

```
exten=> 18,1,Answer
exten=> 18,2,Wait(1)
exten=> 18,1,Answer
exten=> 18,2,Wait(1)
exten=> 18,3,Authenticate(5678)
exten=>18,4,MeetMe(18|p)
exten=>18,5,Playback(vm-goodbye)
exten=>18,6,Hangup
```

2 – Defina no arquivo meetme.conf as senhas para cada extensão.

Nota importante:

A aplicação MeetMe precisa de um temporizador para funcionar. Existem diferentes maneiras de fazer um temporizador funcionar, mas não vai funcionar por default se você não tiver um hardware Digium Zaptel instalado. Neste momento, apenas dispositivos zaptel podem ser usados. Se você não tem um dispositivo Zaptel use o driver ztdummy para temporização.

9.5.7 Arquivo de configuração do MeetMe()

Este arquivo é usado para configurar o modulo de conferência do MeetMe.

Exemplo:

```
;
;Arquivo de configuração para as salas de conferência do Meetme()
;
[rooms]
;Formato: conf=>confno,pincode
;
conf=>9000
conf=>9001,123456
```

Atualmente não é necessário nem reload, nem restart para que o Asterisk note as mudanças no arquivo meetme.conf.

A aplicação Meetme conta com dois aplicativos de apoio:

```
MeetMeCount(confno[|var])
```

Toca o número de usuários em uma conferência MeetMe. Se a variável for especificada, a mensagem não será tocada e a variável receberá o valor da contagem de usuários.

```
MeetMeAdmin(confno,command,user):
```

Roda o comando admin para a conferência especificada em confno.

- 'K' Tira todos os usuários da sala de conferência.
- 'k' Tira um usuário da sala de conferência.
- 'L' Trava a conferência.
- 'l' Destrava a conferência.
- 'M' Emudece a conferência.
- 'm' Desemudecer a conferência

Aplicações avançada. Se você desejar, é possível ainda usar a aplicação MeetMe2 que possui uma interface gráfica para o controle da conferência, ou ainda o app_conference que não precisa de um temporizador e é mais eficiente em termos de tradução de codecs.

9.6 Música em espera – Music on hold

Existem diversas formas de usar a música em espera. As opções podem ser mpg321 com madplay, Ou baixar o mpg123 que é a opção mais comum.

9.6.1 Usando o mpg123.

Antes que você possa usar a música em espera, você precisa baixar e compilar o pacote mpg123. Certifique-se que você não tenha instalado o mpg321.

Baixe uma cópia do mpg123 em <http://www.mpg123.de/cgi-bin/siteexplorer.cgi?/mpg123/>

Digite os seguintes comandos:

```
tar -zxvf mpg123-<version>.tar.gz
cd mpg123
make linux
make install
ln -s /usr/local/bin/mpg123 /usr/bin/mpg123
```

Edite os arquivos para habilitar música em espera

No arquivo `/etc/asterisk/zapata.conf`, adicione a linha:

```
[channels]
musiconhold=default
```

No arquivo `/etc/asterisk/musiconhold.conf`, tire o comentário da linha:

```
default=>mp3:/var/lib/asterisk/mohmp3
```

Você deve reiniciar o asterisk após alterar o arquivo `musiconhold.conf`

Especificando a música

O arquivo exemplo da música em espera (`/etc/asterisk/musiconhold.conf`) irá conter:

```
[classes]
;default => quietmp3:/var/lib/asterisk/mohmp3
;loud => mp3:/var/lib/asterisk/mohmp3
;random => quietmp3:/var/lib/asterisk/mohmp3,-z
```

Retire o comentário da linha “`default=>`”

Então os arquivos MP3 no diretório `/var/lib/asterisk/mohmp3` serão tocados para music on hold. Se existirem múltiplos arquivos no diretório, eles serão tocados sequencialmente.

Personalizando a música em espera

Se você quiser personalizar a música em espera, crie um subdiretório no caminho `/var/lib/asterisk/mohmp3`. Por exemplo “`peessoal`”, em seguida adicione a seguinte linha no seu arquivo `musiconhold.conf`:

```
peessoal=>quietmp3:/var/lib/asterisk/mohmp3/peessoal
```

Coloque seus arquivos mp3 no diretório pessoal e os arquivos serão tocados sequencialmente. No plano de discagem, você deve configurar a música em espera como no exemplo abaixo:

```
Exten=>100,1,SetMusicOnHold(pessoal)
Exten=>100,2,Dial(Zap/2)
```

Nota: formato dos arquivos MP3

É importante que o formato da música MP3 não seja em bitrate variável. Se for o caso use uma ferramenta de edição MP3 para converter o arquivo para bitrate constante. Como você vai usar um telephone 128 bits por segundo é suficiente. Outro ponto importante é remover as identificações ID3, isto pode ser feito com uma ferramenta em Perl disponível em <http://fibiger.org/mp3tag.html>

Exemplos:

Configurando o arquivo extensions.conf para testar MusicOnHold

```
[local]
exten => 6601,1,WaitMusicOnHold(30)
```

9.7 Questionário

1. Com relação à Call Parking assinale as afirmativas verdadeiras.

- Por Default a extensão 800 é usada para Call Parking
- Quando for para o outro telefone para disque 700 para recuperar a chamada
- Por Default a extensão 700 é usada para Call parking
- Digite a extensão anunciada para recuperar a chamada

2. Para que o Call Pickup funcione é preciso que as extensões estejam no mesmo _____. No caso de extensões Zap isto é configurado em _____.

3. No caso de transferência de chamadas existem as transferências _____, onde o ramal de destino não é consultado antes e a transferência _____ onde é possível verificar se o usuário está na extensão.

4. Para fazer uma transferência assistida você usa o ___ enquanto para fazer uma transferência às cegas (blind) você usa ____.

- #1, #2
- #2, #1
- #3, #1
- #4, #2

5. Para fazer conferência no Asterisk é necessário usar o aplicativo _____.

6. Se for necessário administrar uma conferência, você pode usar o aplicativo _____ e tirar um usuário da sala.

- MeetMe()
- MeetMeConsole()
- MeetMeAdministrator()
- MeetmeAdmin()

7. A música em espera pode ser feita pelos seguintes aplicativos.

- mpg123
- mpg321
- mp3player

Página deixada em branco intencionalmente

DAC Distribuição automática de chamadas.

10.1 Objetivos

Ao final deste capítulo você deve ser capaz de:

- Compreender a utilidade do DAC.
- Compreender o seu funcionamento básico.
- Instalar e configurar o DAC

10.2 Introdução

Filas de atendimento permitem que as chamadas de entrada no PABX possam ser tratadas de forma mais eficiente. De uma forma geral os seguintes passos são executados em uma fila de distribuição de chamadas.

1. As chamadas entrada são colocadas na fila.
2. Membros da fila atendem (extensões que autenticaram como agentes).
3. Uma estratégia de gerenciamento e distribuição da fila é usada.
4. É tocada música enquanto o usuário espera na fila.
5. São feitos anúncios para membros e para quem está na fila.

A principal aplicação para filas é o atendimento dentro de uma empresa ou Call Center. Usando filas você evita de perder chamadas quando seus agentes estão ocupados e melhora a produtividade na medida que só distribui para àqueles que estão prontos para atender.

Outro ponto chave das filas de atendimento e distribuição automática de chamadas são as estatísticas que podem ser geradas como TME (tempo médio de espera), TMA (tempo médio de atendimento), chamadas abandonadas, que permitem que você possa dimensionar cientificamente seu atendimento.

Filas são definidas no arquivo “**queues.conf**”. Agentes são pessoas que respondem às chamadas que foram colocadas em uma fila específica. Um agente se autentica indicando que está pronto para receber uma chamada,

agentes são definidos no arquivo **agents.conf**. O Asterisk transfere uma chamada recebida para a fila, que é então transferida para um agente disponível.

10.3 Membros

Membros são aqueles canais que estão ativos e respondendo à fila. Membros podem ser canais diretos (ex. telefones conectados ao Asterisk). Você pode definir também membros como indivíduos que se autenticam de qualquer conexão para receber uma chamada. Agentes são definidos no arquivo **agents.conf**. O login dos agentes é feito através da aplicação **agentlogin**.

10.4 Estratégias

Chamadas são distribuídas entre os membros de uma fila de acordo com uma entre várias estratégias.

- **ringall**: Toca todos os canais disponíveis até que um atenda.
- **roundrobin**: Distribui igualmente as chamadas pelas interfaces.
- **leastrecent**: Distribui para a interface que menos recebeu chamadas.
- **fewestcalls**: Toca aquela com menos chamadas completadas.
- **random**: Toca uma interface aleatória
- **rrmemory**: Roundrobin com memória, se lembra onde deixou a última chamada.

10.5 Menu para o usuário

Você pode definir um menu para o usuário, enquanto ele estiver esperando. Para este menu, você pode usar extensões de um dígito. Para habilitar esta opção, defina o contexto para o menu na configuração da fila.

10.6 Novos recursos

Queue() agora tem opções para **penalidade (penalty)**, **tempo para se arrumar (wrapuptime)** e **prioridade (priority)**. Estes recursos foram adicionados ao sistema de filas do Asterisk.

Prioridade funciona assim:

Os novos recursos permitem que as filas atuem como filas de prioridade, além de modo atual FIFO (primeiro que entra é o primeiro que sai). Isto dá a habilidade de enfileirar uma chamada não no fim da fila, mas em qualquer lugar dela de acordo com a prioridade da chamada.

Agora você pode ter uma única fila atendendo as chamadas ordenadas do mais importante para o menos importante. A prioridade de uma chamada entrando na fila é determinada por uma variável especial, `QUEUE_PRIO`. Valores mais altos da variável significam maior prioridade. Se a variável não receber um valor, todas as chamadas têm a mesma prioridade, por default (FIFO).

Exemplo:

Clientes importantes

```
exten=>111,1,Playback(welcome)
exten=>111,2,SetVar(QUEUE_PRIO=10)
exten=>111,3,Queue(support)
```

Clientes menos importantes

```
exten=>112,1,Playback(welcome)
exten=>112,2,SetVar(QUEUE_PRIO=5)
exten=>112,3,Queue(support)
```

10.7 Configuração

10.7.1 Lista de Tarefas

1. Criar a fila de atendimento (obrigatório).
2. Configurar os agentes (opcional).
3. Criar os agentes de atendimento (opcional).
4. Colocar a fila no plano de discagem (obrigatório).
5. Configurar a gravação dos agentes (opcional).
6. Verificar o funcionamento com `show agents` e `show queues`.

10.7.2. Criar a fila de atendimento.

queues.conf

```
[telemarketing]
music = default
;announce = queue-markq
;context = qoutcon
timeout = 2
retry = 2
maxlen = 0
member => Agent/300
member => Agent/301
[auditoria]
music = default
;announce = queue-markq
;context = qoutcon
timeout = 15
retry = 5
maxlen = 0
member => Agent/600
member => Agent/601
```

10.7.3 Definir parâmetros dos agentes.**agents.conf**

```
[agents]
;Define autologoff se apropriado. Isto é quanto tempo o telefone vai tocar sem
;resposta antes que o agente seja automaticamente desconectado da fila (em
;segundos).Note que se este valor for maior que o valor de timeout da fila,os ;agentes
nã serão removidos.
autologoff=15
; Define ackcall para pedir uma confirmação por '#' quando um
; agente se loga sobre agentcallpark. Default é "yes".
;ackcall=yes
; Define o tempo de wrapuptime. Esta é a quantidade mínima de tempo que
; o agente tem após desconectar antes que possa receber uma nova ligação.
;É o tempo que o agente tem por exemplo para dar andamento ao pedido
; do cliente. (Em milisegundos).
;wrapuptime=5000
;Define a música em espera para os agents.
;musiconhold => music_class
musiconhold => default
;Um beep opcional para tocar para os agentes com conexão
;permanente
;custom_beep=beep
;Grupos para agents (pode mudar no meio do arquivo apenas);
group=1
```

10.7.4 Criar os agentes no arquivo

agents.conf

```
;agent => agentid,agentpassword,name
[agents]
agent => 300,300,Test Rep - 300
agent => 301,301,Test Rep . 301
agent => 600,600,Test Ver - 600
agent => 601,601,Test Ver . 601
```

10.7.5 Colocar a fila no plano de discagem.**extensions.conf**

```
; Fila telemarketing.

exten=>_0800XXXXXXXX,1,Answer
exten=>_0800XXXXXXXX,2,SetMusicOnHold(default)
exten=>_0800XXXXXXXX,3,DigitTimeout,5
exten=>_0800XXXXXXXX,4,ResponseTimeout,10
exten=>_0800XXXXXXXX,5,Background(welcome)
exten=>_0800XXXXXXXX,6,Queue(telemarketing)

; Transfere para a fila auditoria
exten => 8000,1,Queue,(Auditoria) ; Conecta à auditoria
exten => 8000,2,Playback(demo-echotest) ; Nenhum auditor está disponível
enten => 8000,3,Goto(8000,1) ; Verifica o auditor novamente

; Login dos agents para as fila telemarketing (0800) e auditoria (8000)
exten => 9000,1,Wait,1 ; levemente pausada, não necessária
exten => 9000,2,AgentLogin()
; Irá receber 'login inválido' entre com o número do agente.
```

10.7.6 Configurar a gravação**queues.conf**

```
;Habilita a gravação de chamadas endereçadas aos agentes.
recordagentcalls=yes

;Formato usado para gravar as chamadas (wav, gsm, wav49). Default é ;"wav".
recordformat=gsm

;Insere no campo usuário do CDR o nome da gravação criada
;Default, desligado
;createlink=yes

;Texto à ser adicionado à gravação, permite formar um link url.
;urlprefix=http://host.domain/calls/
;O diretório opcional para salvar as conversações
;O padrão é: /var/spool/asterisk/monitor
savecallsin=/var/calls
```

10.7.7 Aplicações de apoio para as filas.

- **AddQueueMember**: Dinamicamente adiciona um membro na fila.
- **RemoveQueueMember**: Dinamicamente remove um membro da fila.
- **AgentLogin**: Login do agente na fila de chamadas.
- **AgentCallbackLogin**: Outro tipo de login.

Linha de comando na interface de linha de comando

- **show agents**: Mostra os agentes.
- **show queues**: Lista todos os agentes.
- **show queue <name>**: Mostra dados de uma fila específica.

10.8 Funcionamento das filas

10.8.1 Login do Agente

A aplicação de AgentLogin é usada para logar os agents. Através do uso de filas, você pode enviar chamadas para atendentes.

Sintaxe:

```
Agentlogin([agentnumber])
```

Nota: Se o número agente não é passado, uma mensagem de login inválido será tocada e pedirá o número do agente e senha.

Pode ser usado também o aplicativo AgentCallbackLogin(), neste caso o agente irá colocar o telefone no gancho e quando alguém estiver na fila o telefone do agente será chamado, ao invés de ficar o tempo todo aberto como o AgentLogin().

Exemplo: Discagem de saída passada aos agentes, baseado em filas.

Parte 1: Login do agente

Um agente da fila telemarketing tira o telefone do gancho e disca #9000, ele ouve a mensagem de login inválido (e é pedido o número e senha). A fila auditoria segue o mesmo processo. A fila no qual ele termina depende da separação em agentes.conf.

Parte 2: Fila

Uma vez na fila, o agente irá ouvir a MOH (música em espera, se definido), ou nada. Quando uma conexão ocorrer, a chamada é conectada à um agente na fila telemarketing. O agente irá ouvir um beep (A MOH irá parar se estiver tocando), e será conectada àquela chamada.

Parte 3: Término da chamada

Quando um agente termina com a chamada ele pode:

- Pressionar * para desconectar e permanecer na fila telemarketing.
- Desligar o telefone, o agente será desconectado da fila telemarketing.
- Pressionar #8000 e a chamada será transferida para a fila auditoria.

10.9 Questionário

1. Cite quatro estratégias de roteamento do sistema de fila de atendimento.

2. É possível gravar a conversação dos agentes usando _____ no arquivo agents.conf.

3. Para logar um agente usa-se o comando Agentlogin([agentnumber]). Quando o agente termina a chamada ele pode pressionar:

- * para desconectar e permanecer na fila
- Desligar o telefone e ser desconectado da fila
- Pressionar #8000 e a chamada será transferida para a fila de auditoria
- Pressionar # para desligar.

4. As tarefas obrigatórias para configurar uma fila de atendimento são:

- Criar a fila
- Criar os agentes
- Configurar os agentes
- Configurar a gravação
- Colocar no plano de discagem

5. Qual a diferença entre os aplicativos AgentLogin() e AgentCallBackLogin().

6. Quando em uma fila de atendimento, você pode definir um determinado número de opções que o usuário pode discar. Isto é feito incluindo um _____ na fila.

- Agente
- Menu
- Contexto
- Aplicativo

7. As aplicações de apoio `AddQueueMember()`, `AgentLogin()` , `AgentCallBackLogin` e `RemoveQueueMember()` devem ser incluídas no:

- Plano de discagem
- Interface de linha de comando
- Arquivo `queues.conf`
- Arquivo `agents.conf`

8. É possível gravar os agentes, mas para isto é preciso de um gravador externo.

- Verdadeiro
- Falso

9. “Wrapuptime” é o tempo que o agente precisa após o término de uma chamada para se preparar para a próxima ou completar processos em relação a chamada atendida

- Verdadeiro
- Falso

10. Uma chamada pode ser priorizada dependendo do CallerID dentro de uma mesma fila. A afirmativa está:

- Correta
- Incorreta

O Correio de voz

11.1 Objetivos

Ao final deste capítulo você deve ser capaz de:

- Compreender para o que serve o Correio de voz.
- Compreender o funcionamento básico.
- Instalar e configurar o Correio de voz.
- Enviar o correio de voz para o e-mail.

11.2 Introdução

O recurso de correio de voz permite que uma chamada ocupada ou não atendida seja enviada para uma secretária eletrônica. Uma mensagem de aviso com áudio anexado é enviada para o receptor da mensagem. É um dos recursos padrão do Asterisk. É interessante como sistemas que em uma arquitetura de comutação de circuitos eram até então muito custosos agora com telefonia IP se tornam padrão.

11.3 Lista de tarefas para configuração

Para configurar o correio de voz você deve seguir os seguintes passos:

1. Configure o arquivo `voicemail.conf` para criar os usuários e senhas do voicemail (obrigatório).
2. Configure o arquivo `extensions.conf` para que uma chamada ocupada ou não atendida seja direcionada para o utilitário `VoiceMail()`. (obrigatório).
3. Configure uma extensão para que o usuário possa ouvir seu correio de voz e administrá-lo.

11.3.1 Configurando o arquivo `voicemail.conf`

```
[general]
```

```
cidinternalcontexts = house-admin, house-toll, house-local, house-intercom-only

[default]
1234=>1234,SomeUser,email@address.com,pager@address.com,saycid=yesldialout
=fromvmlcallback=fromvmlreview=yesloperator=yes
```

Veremos mais opções a frente.

11.3.2 Configurando o arquivo extensions.conf

Você pode usar uma macro para criar as extensões.

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

[macro-online]
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(online,${ PHONE1 })
exten=>6602,1,Macro(online,${ PHONE2 })
```

11.3.3 Usando a aplicação VoiceMailMain()

```
exten=>9000,1,VoiceMailMain()
```

Quando você configurar esta linha no arquivo extensions.conf, o VoiceMailMain() vai levá-lo para um menu de opções onde você poderá executar uma série de opções:

Menu

- 0 Opções da caixa postal
 - 1 Grava sua mensagem de não disponível
 - 2 Grava sua mensagem de ocupado
 - 3 Grava seu nome
 - 4 Grava sua mensagem temporária
 - 5 Muda sua senha
 - * Retorna ao menu principal
- 1 Ler as mensagens de voicemail
 - 3 Opções avançadas

- 1 Reply
- 2 Call Back
- 3 Envelope
- 4 Chamada de saída
- 4 Toca a mensagem anterior
- 5 Repete a mensagem atual
- 6 Toca a próxima mensagem
- 7 Apaga a mensagem atual
- 8 Re-encaminha a mensagem para outra caixa
- 9 Salva a mensagem para um folder
- * Ajuda
- * Sair
- 2 Mudança de pastas
- Ajuda
- # Saída

Após gravar uma mensagem (mensagem de entrada, ocupado, não disponível ou nome)

1 – Aceita

2 – Revisão

3 – Regravar

0 - Vai para telefonista

11.3.4 Sintaxe do Voicemail():

```
VoiceMail([flags]boxnumber[@context][&boxnumber2[@context]][&boxnumber3])
```

Grava o canal, salvando um arquivo de áudio em um dado número de caixa postal, que deve estar configurado no arquivo voicemail.conf. O número de caixa postal deve estar precedido por uma ou mais opções.

s: A letra s, se presente, faz com que as instruções ("Please leave your message after the tone. When done, hang up, or press the pound key.") não seja executada. (Em Português "Por favor, deixe sua mensagem após o bip, quando terminar, desligue ou pressione a tecla sustenido")

u: A letra u, se presente, faz com que a mensagem não disponível seja tocada. Por default, a mensagem diz, "A pessoa na extensão 1234.... não está

disponível” mas o proprietário da caixa postal pode gravar sua própria mensagem de não disponível com o comando **Voicemailmain**.

b: A letra **b**, se presente, faz com que a mensagem de ocupado seja tocada. Por default a mensagem diz: “A pessoa na extensão 1234 está ocupada”

Você não pode especificar ambos o **u** e o **b** juntos. Você pode no entanto, combiná-los com **s**, dando seis possibilidades.

- **s**: Não toca nada
- (Sem opções): Toca instruções
- **su**: Toca mensagem não disponível
- **u**: Toca mensagem não disponível, então instruções
- **sb**: Toca a mensagem de ocupado
- **b**: Toca a mensagem de ocupado, então instruções.

Em todos os casos, o arquivo `beep.gsm` será tocado, antes de começar a gravar.

As mensagens de correio de voz serão gravadas no diretório `inbox` para aquele número de caixa postal de correio de voz.

```
/var/spool/asterisk/voicemail/context/boxnumber/INBOX/
```

Se o originador pressionar **0** (zero) durante o anúncio, ele será movido para a extensão `'o'` (Out) no contexto atual do voicemail. Isto pode ser usado como saída para a recepcionista. Se você não quer isto, apenas configure a extensão para voltar ao voicemail, senão a chamada será desligada.

Também durante o áudio, se o originador pressionar:

`'*'`—A chamada pula para a extensão `'a'` no contexto atual do correio de voz.

`'#'`—A mensagem de boas vindas e/ou instruções são paradas e a gravação começa imediatamente.

Quando usando o zero '0' e estrela '*' é importante notar que o contexto que você colocou a aplicação voicemail é irrelevante, o contexto onde foi colocada a caixa postal do correio de voz é que será usado para pular para as extensões 'a' ou 'o'.

Se a caixa postal não existir, e existir a prioridade n+101, então esta prioridade será a próxima.

Se durante a gravação o originador pressionar '#' ou o limite definido de silêncio estourar, a gravação é parada e a chamada continua na prioridade+1.

Certifique-se de fazer algo com o originador depois que ele voltar do voicemail(), pelo menos:

```
exten=>somewhere,5,Playback(Goodbye)
exten=>somewhere,6,Hangup
```

11.3.5 Códigos de retorno

Retorna -1 em erros ou caixa postal não encontrada, se o usuário desligar. Senão retorna 0.

11.4 Arquivo de configuração do VoiceMail.

11.4.1 Configurações da seção [general].

As configurações da seção geral se aplicam para todos os usuários. As seguintes configurações estão disponíveis.

attach

Faz com que o asterisk copie uma mensagem de voicemail pra um arquivo de áudio e envie-a ao usuário como um anexo em um e-mail. O padrão é não (no).

delete

Se configurado para 'yes' a mensagem será apagada da caixa de voicemail (após ter sido enviada por e-mail).

Nota: Esta configuração nem sempre funciona como uma configuração global. É recomendado que você coloque isto em cada caixa postal que a mensagem deve ser apagada após enviada por e-mail.

mailcmd

Mailcmd permite ao administrador, alterar o comando do e-mail por um comando definido pelo usuário. Mailcmd recebe como comando uma string com a linha de comando desejada. O padrão é `‘/usr/sbin/sendmail -t’`

maxsilence

Maxsilence define quanto tempo o Asterisk irá esperar por um período contínuo de silêncio antes de terminar uma chamada para o voicemail. O valor padrão é 0, o que significa que o detector de silêncio fica desabilitado e o tempo de espera é infinito. Maxsilence recebe um número inteiro que é o número de segundos que ele irá esperar antes de desconectar.

envelope

Envelope controla se o Asterisk irá ou não tocar uma mensagem de envelope (data/hora) antes de tocar a mensagem de voicemail. (yes/no).

externnotify

Quer rodar um programa externo sempre que um chamador deixar uma mensagem em correio de voz para o usuário. Existe o comando `externnotify`. Recebe o valor de uma string com o comando a ser executado.

silencethreshold

As vezes quando usamos o **maxsilence**, é necessário ajustar a detecção de silêncio para eliminar um disparo falso causado por ruído de fundo. Números maiores permitem mais ruído de fundo. É necessário experimentar até chegar nos valores apropriados.

serveremail

Esta configuração pode ser usada para identificar a fonte de uma notificação de mensagem de voz. O valor é uma string que pode ser codificada de uma de duas formas.. Se a string está na forma *alguem@host.com*, então a string será usada como endereço fonte nas notificações por e-mail. Se a string estiver na forma de *alguém*, então o nome do host da máquina rodando o asterisk será adicionado ao final da string após a inserção do @.

maxmessage

Define o tempo máximo em segundos de uma mensagem de entrada. Use quando o espaço em disco for limitado. O padrão é 0 que significa sem limite.

minmessage

Esta configuração pode ser usada para eliminar mensagens que são mais curtas que um determinado número de segundos. O padrão é 0 que significa sem tempo mínimo definido.

format

Permite a seleção de formato de áudio usado para armazenar as mensagens do correio de voz. O formato é uma string e pode ser wav49|gsm|wav.

- **wav49:** Neste formato o tamanho do arquivo é pequeno e a qualidade boa. É uma boa escolha para mandar mensagens de notificação por e-mail, pois tem mais suporte nas estações com Windows.
- **gsm:** Basicamente o mesmo tamanho e qualidade de voz, mas menos suportado pelo cliente.
- **wav:** Formato não compactado, grande no tamanho e não é recomendado.

maxgreet

Permite ao administrador limitar o comprimento de uma saudação do usuário gravada pelo usuário. O valor é um inteiro definindo o tempo máximo em segundos. O padrão é 0 que significa sem limite.

skipms

Define o intervalo em milisegundos para usar quando passando a frente ou para trás enquanto uma mensagem está sendo tocada. O valor entrado deve ser positivo e inteiro. O padrão é 3000 (3 segundos)

maxlogins

Define o número de tentativas de login antes que o Asterisk desconecte o usuário. O valor deve ser inteiro e positivo. O padrão é 3.

cidinternalcontexts

Define os contextos internos usados para determinar o tipo de anúncio de voz a ser tocado quando lendo a identificação da chamada ou quando o recurso avançado saycid estiver habilitado. Isto altera mensagem de “Chamada de ‘12345678’ para “Chamada da extensão”.

review

Às vezes é interessante deixar o chamado rever a sua mensagem antes de passar a mensagem à sua caixa postal. Os valores podem ser ‘yes/no’.

operator

Permite ao usuário ir para a telefonista enquanto a mensagem está sendo gravada ou uma vez que a mensagem tenha sido enviada, se a opção *review* tiver sido configurada. A telefonista deve ser configurada na extensão ‘o’ no contexto do voicemail no arquivo `extensions.conf`.

saycid

Lê de volta o número do chamador antes de tocar a mensagem de entrada, após ser tocada a mensagem de data e hora.

dialout

Especifica o contexto a ser usado para que uma chamada de saída possa ser usada. Recebe uma string com o contexto.

callback

Especifica o contexto à ser usado no recurso “retornar a chamada”. Recebe uma string especificando o contexto de saída a ser usado.

dbuser

Especifica o nome do usuário na base de dados MySQL a ser usado na aplicação de correio de voz. O valor é uma string, o padrão é ‘test’. Note que o Asterisk deve ser compilado para suporte ao MySQL.

dbpass

Especifica a senha na base dados MySQL a ser usada para a aplicação de correio de voz

dbhost

Especifica o servidor de base de dados MySQL. O valor é uma string.

dbname

Especifica o nome da base de dados MySQL. O valor é uma string.

dboption

Especifica a opção de base de dados Postgres. O valor é uma string. Note que o Asterisk tem de ser compilado para o Postgres para que esta opção funcione.

pbxskip

Esta configuração muda a linha do assunto *Subject:* em uma mensagem de notificação. Esta configuração recebe 'yes/no'. O padrão 'no'. Quando em 'yes' a linha do assunto *Subject:* terá "Subject: New message M em Mailbox B". Quando setado para 'no' a linha do assunto terá "Subject: [PBX]: New message M in mailbox B".

fromstring

Esta configuração permite ao administrador substituir a porção da linha "From:" na mensagem de notificação. Recebe uma string.

emailsubject

Esta configuração substitui completamente a linha assunto *Subject:* . O valor passado é uma string.

emailbody

Substitui a mensagem normal de texto vista no corpo da mensagem de notificação do voicemail. Também suporta a substituição de variáveis. para torná-la mais legível. As variáveis são abordadas mais a frente.

exitcontext

Contexto de saída a deixar o usuário após ter pressionado * ou 0 para sair do voicemail.

nextaftercmd

Se configurado para "yes" após apagar a mensagem de voicemail, o sistema automaticamente irá tocar a próxima mensagem.

11.4.2 Variáveis para *emailsubject* e *emailbody*.

- VM_NAME
- VM_DUR
- VM_MSGNUM
- VM_MAILBOX
- VM_CIDNUM
- VM_CIDNAME
- VM_CALLERID
- VM_DATE

11.4.3. Configurações para as seções [CONTEXT]

A parte final do voicemail.conf contém um ou mais contextos.

```
[context_section]
extension_number =>
voicemail_password,user_name,user_email_address,user_pager_email_address,user_option(s
)
```

Após a definição da seção contexto uma entrada para cada caixa de correio deve existir neste contexto. Os parâmetros estão descritos abaixo:

- **extension_number:** é o número da extensão que será designada para esta caixa de correio de voz. Existem cinco parâmetros que definem a configuração do correio de voz para esta entrada:
- **voicemail_password:** Contém a senha numérica para esta caixa de correio de voz.
- **user_name:** Campo alfanumérico com o primeiro e último nome do usuário.
- **user_email_address:** Endereço de e-mail para notificação.

- **pager_email_address:** pode ser configurado para o endereço de e-mail de um pager de forma a quando receber um correio de voz ser notificado em um celular.
- **user_option(s):** Campo a ser usado para substituir valores definidos na seção geral. Existem 9 pares do tipo “*configuração=valor*” que podem ser especificados. As opções devem ser separados pela barra vertical (|). As nove configurações são: **attach, serveremail,tz,saycid, review, operator, callback, dialout, e exitcontext.**

11.5 Interface Web para o Correio de voz.

Existe um script em Perl na distribuição fonte chamado `vmail.cgi`, localizado em `/usr/src/asterisk/vmail.cgi`.

Note que o `make install` não instala esta interface à menos que você rode **make webvmail** separadamente. Este script requer que o Perl e Perl-suidperl estejam instalados além do Web Server.

Este script pode precisar alguma edição antes de funcionar na sua instalação. Copie `/usr/asterisk/images/*.gif` para `/asterisk` na árvore html. O script precisa passar a executável:

```
chmod +x vmail.cgi
```

11.6 Sumário

Neste capítulo você aprendeu os passos necessários para a configuração do correio de voz. Conheceu também os arquivos de configuração e os principais arquivos correlatos.

11.6 Questionário

1. Os arquivos envolvidos para a configuração do Correio de voz são:

- sip.conf
- iax.conf
- asterisk.conf
- voicemail.conf
- vmail.conf
- extensions.conf

2. Na aplicação VoiceMail(), os parâmetros “u” e “b” são _____ e _____ respectivamente. Eles servem para determinar que tipo de mensagem o correio de voz irá tocar para o usuário.

- Ocupado, Livre
- Ocupado, Não atende
- Não atende, Ocupado
- Livre, Preso

3. A aplicação VoiceMailMain() é usada para que o usuário que chamou deixe sua mensagem. A afirmativa está:

- Correta
- Incorreta

4. Para sair do VoiceMailMain você deve digitar:

- 0
- *
- #
- 9999

5. Escreva abaixo a sintaxe do comando VoiceMail()

6. Na seção [general] do arquivo voicemail.conf, o parâmetro attach=yes vai fazer com que uma notificação por e-mail seja enviada ao usuário com o arquivo de áudio anexado. A afirmação está:

- Correta

Incorreta

7 A opção “delete” faz com que todas as mensagens sejam apagadas da caixa postal antes de terem sido enviadas para a caixa de correio do usuário.

Correta
 Incorreta

8. O formato mais adequado para as mensagens de áudio é o “WAV”, pois tem mais suporte de estações Windows.

Correta
 Incorreta

9. É possível personalizar as mensagens de correio com emailsubject e emailbody. Que variável pode ser usada para indicar o CallerID na mensagem:

10. O nome do CGI para instalar a interface Web do voicemail é o _____.

Página deixada intencionalmente em branco

Resposta dos Exercícios

Respostas do Capítulo 1

1. Marque as opções corretas. O Asterisk tem quatro componentes básicos de arquitetura

- CANAIS
- PROTOCOLOS
- AGENTES
- TELEFONES
- CODECS
- APLICAÇÕES

2. Se for necessário criar um PABX com 4 troncos e oito telefones, você pode usar um PC com Linux e três placas TDM400P uma com quatro canais FXO e duas com quatro canais FXS cada. A afirmação acima está:

- CORRETA
- INCORRETA

3. Um canal FXS gera tom de discagem, enquanto um canal FXO recebe o tom vindo da rede pública ou de um outro PABX. A afirmação acima está:

- CORRETA
- INCORRETA

4. Marque as opções corretas, O Asterisk permite os seguintes recursos:

- Unidade de Resposta Automática
- Distribuição automática de chamadas
- Telefones IP
- Telefones Analógicos
- Telefones digitais de qualquer fabricante.

5. Para tocar música em espera o Asterisk necessita de um CD Player ligado em um ramal FXO. A afirmação está:

- CORRETA
- INCORRETA

6. É responsável pelo atendimento automático de clientes, normalmente toca um “prompt” e espera que usuário selecione uma opção. Am alguns casos pode ser usada em conjunto com um banco de dados e conversão texto para fala.

Estamos falando de uma:

- URA
- IVR
- DAC
- Unified Messaging

7 – Marque as opções corretas, Um banco de canais é conectado ao Asterisk através de uma interface:

- E1
- T1
- FXO
- FXS

8 – Marque a opção correta. Um canal E1 suporta ___ canais de telefonia enquanto um T1 suporta ___ canais.

- 12, 24
- 30, 24
- 12,12
- 1,1

9 – Nas plataformas de telefonia convencional, normalmente URA, DAC e Correio de voz estão incluídos no PABX. Esta afirmação está:

- CORRETA
- INCORRETA

10 – Marque as opções corretas, É possível interligar usando o Asterisk várias filiais através de voz sobre IP reduzindo a despesa com ligações de longa distância. Em uma filial:

- O Asterisk pode ser a central telefônica para todos os usuários.
- O Asterisk pode integrar uma central telefônica existente
- Podem ser usados apenas telefones IP ligados à um Asterisk centralizado
- Redundância e confiabilidade não são importantes quando se ligam ip fones.

Respostas do Capítulo 2

1. Qual a configuração mínima para o Asterisk.

Não existe uma configuração mínima, mas é recomendável no mínimo:

Pentium 300 Mhz
256 MB RAM
100 MB de disco

2. As placas de telefonia para o Asterisk têm um processador próprio (DSP), não precisando assim de muita CPU do servidor.

- Correto
 Incorreto

3. Para que a telefonia IP funcione com perfeição é necessário que à rede possua QoS fim-a-fim.

- Correto
 Incorreto

4. É possível obter uma boa qualidade de voz em uma rede que não esteja congestionada com switches de 100 Mbps.

- Correto
 Incorreto

5. Liste abaixo as bibliotecas necessárias para compilar o Asterisk.

gcc - GNU C Compiler and Support Files
cvs – Concurrent Versions System
ncurses – New curses libraries
ncurses-devel – Bibliotecas para desenvolvimento com ncurses
bison – The GNU parser generator
Termcap – Termcap library
openssl – Secure Sockets and TLS Layer Security
openssl-developer – Bibliotecas do openssl.

6. Se você não tem uma placa zaptel, você precisa de uma fonte de tempo. O driver ztdummy faz este papel aproveitando uma biblioteca USB. Isto é

necessário, pois algumas aplicações como o **Meetme()** e o **MusicOnHold()** precisam de uma referência de tempo.

7. O CVS é o sistema de controle de versões do Asterisk. Desta forma você só pode baixar a última versão.

- Correto
- Incorreto

8. Quando você faz uma instalação do Asterisk, o melhor é não instalar os pacotes gráficos como o KDE e GNOME, pois o Asterisk é sensível na questão de CPU e interfaces gráfica roubam muitos ciclos de CPU do servidor.

- Correto
- Incorreto

9. Os arquivos de configuração do Asterisk ficam em **/etc/asterisk**.

10. Para instalar os arquivos de configuração de exemplo você precisa executar o seguinte comando.

#make samples

Respostas do Capítulo 3

1. São exemplos de arquivos de configuração de canais Asterisk.

- zaptel.conf
- zapata.conf
- sip.conf
- iax.conf

2. É importante definir o contexto no arquivo de canais, pois quando uma ligação deste canal (sip, iax, zap) chegar ao Asterisk ele será tratado no arquivo extensions.conf neste contexto.

- Correto
- Incorreto

3. O parâmetro switchtype no arquivo zapata.conf define o tipo de PABX ao qual o Asterisk está ligado. Isto é válido para conexões no padrão E1 com sinalização ISDN PRI. Normalmente no Brasil e na Europa este padrão deve ser definido como **National**.

- Correto
- Incorreto

4. Apesar de ser considerado uma linha digital, o E1 pode ser configurado com sinalização associada ao canal (CAS) neste caso cada timeslot pode se comportar como um canal analógico FXS ou FXO, por exemplo. Isto é útil para a conexão à channel-banks (bancos de canal).

- Correto
- Incorreto

5. SIP Session Initiated Protocol é o protocolo da ITU usado para conexões de voz sobre IP. Ele é bastante antigo e vem sendo substituído recentemente pelo H.323.

- Correto
- Incorreto

6. Dado a configuração abaixo do arquivo sip.conf, na seção [general] está definido o endereço IP 10.1.30.45, onde o SIP estará esperando por conexões.

Se fosse necessário que todas as placas de rede da máquina esperassem por uma conexão SIP, `bindaddr` deveria estar configurado para: **0.0.0.0**

```
[general]
port = 5060
bindaddr = 10.1.30.45
context = default
disallow = speex
disallow = ilbc
allow = ulaw
maxexpirey = 120
defaultexpirey = 80
```

7. No arquivo abaixo, os telefones 8000 e 8001 foram definidos com a opção `canreinvite=no`. Com isto, quando uma ligação é feita de um telefone para o outro, o áudio vai diretamente de um telefone para outro sem passar pelo Asterisk.

```
[8000]
type=friend
secret=8000
host=dynamic
canreinvite=no
```

```
[8001]
type=friend
secret=8000
host=dynamic
canreinvite=no
```

- Correto
 Incorreto

8. A principal diferença entre o comando `Playback()` e o comando `Background()` é que o `Playback()` simplesmente toca uma mensagem e passa ao comando seguinte, enquanto o `Background` aguarda que você digite algo e desvia para algum lugar no plano de discagem baseado nos dígitos discados.

- Correto
 Incorreto

9. Quando uma ligação entra no Asterisk por uma interface de telefonia (FXO) sem identificação de chamada, esta ligação é desviada para a extensão especial:

- '0'
- '9'
- 's'
- 'i'

10. Os formatos válidos para o comando Goto() são:

- Goto(contexto, extensão, prioridade)
- Goto(prioridade, contexto, extensão)
- Goto(extensão, prioridade)
- Goto(prioridade)

Respostas do Capítulo 4

1. Cite pelo menos quatro benefícios do uso de Voz sobre IP
Redução das tarifas, Mobilidade, URA Integrada em IP, Agentes Remotos.

2. Convergência é a unificação das redes de voz, vídeo e dados em uma única rede e seu principal benefício é a redução com os custos de manutenção de redes separadas.

- Correto
- Incorreto

3. O Asterisk não pode usar simultaneamente recursos de PSTN (Rede pública de telefonia e de voz sobre IP, pois os codecs não são compatíveis).

- Correto
- Incorreto

4. A Arquitetura do Asterisk é de um SIP proxy com possibilidade outros protocolos.

- Correto
- Incorreto

5. Dentro do modelo OSI, os protocolos SIP, H.323 e IAX2 estão na camada de:

- Apresentação
- Aplicação
- Física
- Sessão
- Enlace

6. SIP é hoje o protocolo mais aberto (IETF) sendo implementado pela maioria dos fabricantes.

- Correto
- Incorreto

7. O H.323 é um protocolo sem expressão, pouco usado foi abandonado pelo mercado em favor do SIP.

- Correto
- Incorreto

8. O IAX2 é um protocolo proprietário da Digium, apesar da pouca adoção por fabricantes de telefone o IAX é excelente nas questões de:

- Uso de banda
- Uso de vídeo
- Passagem por redes que possuem NAT
- Padronizado por órgãos como a IETF e ITU

9. “Users” podem receber chamadas

- Correto
- Incorreto

10. Sobre codecs assinale o que é verdadeiro

- O G711 é o equivalente ao PCM (Pulse Code Modulation) e usa 64 Kbps de banda.
- O G.729 é gratuito por isto é o mais utilizado, usa apenas 8 Kbps de banda.
- GSM vem crescendo pois ocupa 12 Kbps de banda e não precisa de licença.
- G711 ulaw é comum nos EUA enquanto a-law é comum na Europa e no Brasil.
- G.729 é leve e ocupa pouca CPU na sua codificação.

Respostas do Capítulo 5

1. Podemos citar como principais benefícios do IAX a economia de banda e facilidade de passar por Firewalls com NAT.

- Correto
- Incorreto

2. No protocolo IAX os canais de sinalização e mídia passam separados. Esta afirmação é:

- Correta
- Incorreta

3. O IAX emprega os seguintes tipos de frames

- Frame Completo
- Frame Incompleto
- Mini-Frame
- Trunked Frame

4. A banda passante usada pelo protocolo IAX é a soma da carga de voz (payload) mais os cabeçalhos (Marque todas as que se aplicam)

- IP
- UDP
- IAX
- RTP
- cRTP

5. Comparando o protocolo IAX e o protocolo cRTP (compressed RTP) podemos afirmar que em uma rede baseada na transmissão pela Internet como o ADSL (Marque uma opção)

O IAX2 é sempre a melhor opção.
O cRTP não pode ser implantado neste tipo de circuito.
O cRTP ocupa menos banda por isto é a melhor opção.
A partir de 16 linhas o IAX2 passa a ser a melhor opção.

6. Quando o IAX é usado no modo trunk, apenas um cabeçalho é usado para transmitir múltiplas ligações. A afirmação acima está:

- Correta
- Incorreta

7. O protocolo IAX2 é o mais comum para conectar provedores de telefonia IP, pois passa fácil pelo NAT. A afirmação acima está

- Correta
- Incorreta

8. Em um canal IAX como o abaixo, a opção <secret> pode ser tanto uma senha como uma chave digital.

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

9. O contexto é adicionado para cada cliente IAX, isto permite que diferentes clientes possuam diferentes contextos. Pode-se pensar em contexto como uma classe de ramal onde o cliente será colocado. A afirmação está

- Correta
- Incorreta

10. O comando IAX2 show registry mostra informações sobre:

- Os usuários registrados
- Os provedores ao qual o Asterisk se conectou.

Respostas do Capítulo 6

1. O SIP é um protocolo do tipo texto similar ao _____ e _____.

- IAX
- HTTP
- H323
- SMTP

2. O SIP pode ter sessões do tipo: (marque todos que se aplicam)

- Voz
- Correio Eletrônico
- Vídeo
- Chat
- Jogos

3. Podemos citar como componentes do SIP o: (marque todos que se aplicam)

- User Agent
- Media gateway
- PSTN Server
- Proxy Server
- Registrar Server

4. Antes que um telefone possa receber chamados, ele precisa se **Registrar**.

5. O SIP pode operar em modo PROXY e modo REDIRECT, a diferença entre eles é que no caso do PROXY a sinalização sempre passa pelo computador intermediário (SIP Proxy) enquanto no modo REDIRECT os clientes sinalizam diretamente.

- Correto
- Incorreto

6. No modo PROXY o fluxo de mídia e a sinalização passam pelo “SIP proxy” e não diretamente de um cliente para o outro.

- Correto
- Incorreto

7. O Asterisk atua como um SIP Proxy.

- Correto
- Incorreto

8. A opção canreinvite=yes/no é de importância fundamental pois vai definir se o fluxo de mídia vai passar pelo Asterisk ou não. A afirmação está:

- Correta
- Incorreta

9. O Asterisk suporta sem problemas supressão de silêncio em canais SIP. A afirmação está:

- Correta
- Incorreta

10. O tipo mais difícil de NAT para transpor é o:

- Full Cone
- Restricted Cone
- Port Restricted Cone
- Symmetric

Respostas do Capítulo 7

1. Na configuração da seção [general] o valor padrão da opção writeprotect=no. Sendo assim se você emitir o comando save dialplan na linha de comando CLI>. (Marque todas que se aplicam).

- O Asterisk irá sobrescrever o extensions.conf com a configuração atual.
- Todos os comentários serão perdidos
- Será feito um backup de extensions.conf em extensions.bak
- A opção static=yes deve estar configurada para pode salvar o plano de discagem

2. Normalmente as variáveis globais vão estar escritas em maiúsculas enquanto as variáveis de canal estarão começando com maiúscula e restante em minúsculas. Isto não é obrigatório, mas facilita a leitura.

- Verdadeiro
- Falso

3. E extensão ‘s’ é usada como extensão de início dentro de um contexto, ela é normalmente usada nos seguintes casos.

- Na entrada de uma chamada sem CallerID
- Como entrada de um menu chamado pelo comando Background()
- Na entrada de uma chamada com CallerID
- Como entrada de um contexto que foi direcionado pelo Goto()

4. Cite quatro situações onde contextos poderiam ser utilizados:

Implementar segurança
Funções de roteamento
Menus Multicamada
Privacidade

5. Switches são usados para direcionar para outro PABX. A afirmação acima está:

- Correta
- Incorreta

6. Para usar uma variável no plano de discagem no Asterisk você deve usar o seguinte formato:

- `[$varname]`
- `{varname}`
- `$(varname)`
- `${varname}`

7. As variáveis disponíveis no Asterisk são: (Escolha 3)

- Constantes
- Variáveis públicas
- Variáveis de ambiente
- Variáveis globais
- Variáveis privadas
- Variáveis de canal

8. Para obter o comprimento de uma string você pode usar a função `LEN(string)`.

9. Para concatenar strings basta colocar uma do lado da outra como no exemplo abaixo. A afirmação está:

```
`${foo}``${bar}`  
555`${Onumero}`  
`${PrefixoLongaDistancia}555`${ONumero}`
```

- Correta
- Incorreta

10. Uma macro pode ser usada para automatizar uma série de operações em seqüência para uma extensão específica. O primeiro argumento passado pela chamada da macro é o:

- `ARG1`
- `ENV1`
- `V1`
- `X`

Respostas do Capítulo 8

1. Para incluir um contexto que depende do horário, você pode usar:

```
include => context|<times>|<weekdays>|<mdays>|<months>
```

O comando abaixo:

```
include=>expedientel08:00-18:00|mon-fri|*|*
```

- Executa as extensões de segunda à sexta das 08:00 às 18:00
- Executam as opções todos os dias em todos os meses
- O comando é inválido

2. Quando o usuário disca “0” para pegar a linha o Asterisk automaticamente corta o áudio. Isto é ruim, pois o usuário está acostumado a discar o “0” e ouvir o tom externo de discagem. Para criar este comportamento que o usuário está acostumado, pode-se usar comando ignorepat=>.

3. Os comandos:

```
exten => 8590/482518888,1,Congestion
exten => 8590,1,Dial(Zap/1,20)
exten => 8590,2,Voicemail(u8590)
exten => 8590,102,Voicemail(b8590)
```

Faz com que um usuário que ligou para a extensão 8590:

- Receba um sinal de ocupado se o CallerID=482518888
- Receba um sinal de ocupado independente do número discado
- Vá para o canal ZAP/1 se o número não for 482518888
- Vá para o VoiceMail() se o canal ZAP/1 estiver ocupado ou não atender, exceto no caso onde o CallerID for 482518888

4. Para concatenar várias extensões basta separá-las com o sinal &.

5. Um menu de voz normalmente é criado com o comando inicial Answer() e Background().

6. Você pode incluir arquivos dentro dos seus arquivos de configuração com o comando #include.

7. O Asterisk permite que se use no plano de discagem uma base de dados baseada em:

- Oracle
- MySQL
- Berkley DB
- PostgreSQL

8. Quando você usa o comando Dial(tipo1/identificar1&tipo2/identificar2) com vários identificadores, o Asterisk disca para cada um na seqüência e espera 20 segundos ou o tempo de timeout antes de passar para o outro número. A afirmação é:

- Falsa
- Verdadeira

9. No comando Background a música de fundo tem de ser tocada inteiramente antes que o usuário possa digitar algo. A afirmação é:

- Falsa
- Verdadeira

10. Os formatos válidos par o comando Goto() são:

- Goto (context,extension)
- Goto(context,extension,priority)
- Goto(extension,priority)
- Goto(priority)

Respostas do Capítulo 9

1. Com relação à Call Parking assinale as afirmativas verdadeiras.

- Por Default a extensão 800 é usada para Call Parking
- Quando for para o outro telefone para disque 700 para recuperar a chamada
- Por Default a extensão 700 é usada para Call parking
- Digite a extensão anunciada para recuperar a chamada

2. Para que o Call Pickup funcione é preciso que as extensões estejam no mesmo GRUPO. No caso de extensões Zap isto é configurado em ZAPATA.CONF.

3. No caso de transferência de chamadas existem as transferências ÀS CEGAS, onde o ramal de destino não é consultado antes e a transferência ASSISTIDA onde é possível verificar se o usuário está na extensão.

4. Para fazer uma transferência assistida você usa o #2 enquanto para fazer uma transferência às cegas (blind) você usa #1.

- #1, #2
- #2, #1
- #3, #1
- #4, #2

5. Para fazer conferência no Asterisk é necessário usar o aplicativo MEETME().

6. Se for necessário administrar uma conferência, você pode usar o aplicativo _____ e tirar um usuário da sala.

- MeetMe()
- MeetMeConsole()
- MeetMeAdministrator()
- MeetmeAdmin()

7. A música em espera pode ser feita pelos seguintes aplicativos.

- mpg123
- mpg321
- mp3player

8. Cite quatro estratégias de roteamento do sistema de fila de atendimento.

ringall: Toca todos os canais disponíveis até que um atenda.
roundrobin: Distribui as chamadas pelas interfaces igualmente.
leastrecent: Distribui para a interface que menos recebeu chamadas
fewestcalls: Toca aquela com menos chamadas completadas.
random: Toca uma interface aleatória
rrmemory: Roundrobin com memória, lembra onde deixou a última chamada.

Respostas do Capítulo 10

1. Cite quatro estratégias de roteamento do sistema de fila de atendimento.

ringall: Toca todos os canais disponíveis até que um atenda.

roundrobin: Distribui as chamadas pelas interfaces igualmente.

leastrecent: Distribui para a interface que menos recebeu chamadas

fewestcalls: Toca aquela com menos chamadas completadas.

random: Toca uma interface aleatória

rrmemory: Roundrobin com memória, lembra onde deixou a última chamada.

2. É possível gravar a conversação dos agentes usando **record=yes** no arquivo agents.conf.

3. Para logar um agente usa-se o comando Agentlogin([agentnumber]). Quando o agente termina a chamada ele pode pressionar:

- * para desconectar e permanecer na fila
- Desligar o telefone e ser desconectado da fila
- pressionar #7000 e a chamada será transferida para a fila de auditoria
- Pressionar # para desligar.

4. As tarefas obrigatórias para configurar uma fila de atendimento são:

- Criar a fila
- Criar os agentes
- Configurar os agentes
- Configurar a gravação
- Colocar no plano de discagem

5. Qual a diferença entre os aplicativos AgentLogin() e AgentCallBackLogin().

O Agent Login() faz com que o usuário fique com o fone

aguardando uma ligação enquanto no AgentCallBackLogin()

o agente coloca o telefone no gancho e aguarda uma chamada.

6. Quando em uma fila de atendimento, você pode definir um determinado número de opções que o usuário pode discar. Isto é feito incluindo um _____ na fila.

- Agente
- Menu

Contexto

Aplicativo

7. As aplicações de apoio `AddQueueMember()`, `AgentLogin()` , `AgentCallBackLogin` e `RemoveQueueMember()` devem ser incluídas no:

Plano de discagem

Interface de linha de comando

Arquivo `queues.conf`

Arquivo `agents.conf`

8. É possível gravar os agentes, mas para isto é preciso de um gravador externo.

Verdadeiro

Falso

9. “Wrapuptime” é o tempo que o agente precisa após o término de uma chamada para se preparar para a próxima ou completar processos em relação a chamada atendida

Verdadeiro

Falso

10. Uma chamada pode ser priorizada dependendo do CallerID dentro de uma mesma fila. A afirmativa está:

Correta

Incorreta

Respostas do Capítulo 11

1. Os arquivos envolvidos para a configuração do Correio de voz são:

- sip.conf
- iax.conf
- asterisk.conf
- voicemail.conf
- vmail.conf
- extensions.conf

2. Na aplicação VoiceMail(), os parâmetros “u” e “b” são _____ e _____ respectivamente. Eles servem para determinar que tipo de mensagem o correio de voz irá tocar para o usuário.

- Ocupado, Livre
- Ocupado, Não atende
- Não atende, Ocupado
- Livre, Preso

3. A aplicação VoiceMailMain() é usada para que o usuário que chamou deixe sua mensagem. A afirmativa está:

- Correta
- Incorreta

4. Para sair do VoiceMailMain você deve digitar:

- 0
- *
- #
- 9999

5. Escreva abaixo a sintaxe do comando VoiceMail()

```
VoiceMail([flags]boxnumber[@context][&boxnumber2[@context]][&boxnumber3])
```

6. Na seção [general] do arquivo voicemail.conf, o parâmetro attach=yes vai fazer com que uma notificação por e-mail seja enviada ao usuário com o arquivo de áudio anexado. A afirmação está:

- Correta
- Incorreta

7 A opção “delete” faz com que todas as mensagens sejam apagadas da caixa postal antes de terem sido enviadas para a caixa de correio do usuário.

- Correta
- Incorreta

8. O formato mais adequado para as mensagens de áudio é o “WAV”, pois tem mais suporte de estações Windows.

- Correta
- Incorreta

9. É possível personalizar as mensagens de correio com emailsubject e emailbody. Que variável pode ser usada para indicar o CallerID na mensagem: VM_CallerID

10. O nome do CGI para instalar a interface Web do voicemail é o **vmail.cgi**.