UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

ASTERISK E ALTA DISPONIBILIDADE

WINFRED KWABLA GBEDEMAH

JUIZ DE FORA
DEZEMBRO, 2009

ASTERISK E ALTA DISPONIBILIDADE

WINFRED KWABLA GBEDEMAH

Monografia de Conclusão do Curso de Ciência da Computação apresentada à Universidade Federal de Juiz de Fora como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Orientador: Eduardo Pagani Julio

JUIZ DE FORA

DEZEMBRO, 2009

ASTERISK E ALTA DISPONIBILIDADE

Winfred Kwabla Gbedemah

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada em ___ de _____ de 2009.

BANCA EXAMINADORA

_____

Eduardo Pagani Julio

Mestre em Computação/UFF

_____

Marcelo Lobosco

Dr. em Engenharia de Sistemas e Computadores/UFRJ

_____

Eduardo Barrere

Dr. Em Engenharia de Sistemas e Computadores/UFRJ

AGRADECIMENTOS

Primeiramente, agradeço a Deus, a minha família, pela paciência e compressão nos momentos mais difíceis; e a memória da minha mãe. Meus agradecimentos também ao meu orientador, Eduardo Pagani Julio, pelo apoio e incentivo durante a realização deste trabalho.

SUMÁRIO

CAPÍTULO 1 – INTRODUÇÃO 13
CAPÍTULO 2 - VOZ SOBRE IP 15
2.1 FUNCIONAMENTO 15
2.1.1 *Codecs* (Codificadores/Decodificadores) 16
2.2 ARQUITETURA VOIP 18
2.2.2 Arquitetura com *gateway* 20
2.2.3 Arquitetura híbrida 21
2.3 CONCLUSÃO 21
CAPÍTULO 3 - QUALIDADE DA VOZ NA TECNOLOGIA VoIP 23
3.1 FATORES QUE AFETAM A QOS DO VOIP 24
3.1.1 Latência 24
3.1.2 *Jitter* 24
3.1.3 Perda de pacotes 25
3.1.4 Largura da banda 25
3.1.5 Equipamento 25
3.2 CONCLUSÃO 27
CAPÍTULO 4 – PROTOCOLOS 28
4.1 PROTOCOLOS H.323 E SIP 28
4.2 COMPLEXIDADE 28
4.4 EXTENSIBILIDADE 29
4.4 ESCALABILIDADE 31
4.5 SERVIÇOS 33
4.6 CONCLUSÃO 34
CAPÍTULO 5 - ASTERISK 35
5.1 HISTÓRIA 36
5.2 TIPOS DE LICENCIAMENTO 37
5.3 MOTIVOS DE USO DO ASTERISK 38
5.3.1 Redução de Custos 38
5.3.3 Ter Controle de Sistema Telefonia 39
5.3.4 Limitações da Arquitetura 39
5.4 ARQUITETURA 39
5.5 CONCLUSÃO 40
CAPÍTULO 6 - ALTA DISPONIBILIDADE 42
6.1 CONCEITOS 42
6.2 CONFIABILIDADE E DISPONIBILIDADE 43
6.3 TIPOS DE DISPONIBILIDADE 44
6.4 MÉTODOS E TÉCNICAS DE IMPLEMENTAR A ALTA DISPONIBILIDADE 45
6.4.1 *Realtime e Heartbeat* 46
6.4.2 *Realtime, Heatbeat* e *Gateway* SIP –E1 46
6.4.3 *Realtime, Heartbeat, Gateway* SIP-E1 e *DNS Round Robin* 47
6.5 CONCLUSÃO 48
CAPÍTULO 7 - ESTUDO DE CASO 49
7.1 INTRODUÇÃO 49
7.2 ESTUDO DE CASO 49

7.2.1 Estrutura 49
7.2.2 VMware 50
7.2.3 MeucciBE 51
7.2.4 X-Lite 51
7.2.5 Heartbeat 52
7.3 DESCRIÇÃO DA INSTALAÇÃO 53
7.4 DESCRIÇÃO DO FUNCIONAMENTO 55
7.5 TESTE NO SISTEMA 57
7.6 CONCLUSÃO 62
CAPÍTULO 8 - CONSIDERAÇÕES FINAIS 63
REFERÊNCIAS BIBLIOGRÁFICAS 64
APÊNDICE A 66
ANEXO A 71
ANEXO B 74

LISTA DE FIGURAS

Figura 1: Codificação e Decodificação. 16
Figura 2: Arquitetura PC-a-PC 20
Figura 3: Arquitetura com gateway 20
Figura 4: Arquitetura Hibrida 21
Figura 5: D-Link DVG-1402SL Express EtherNetwork Broadband Lingo Phone Service VoIP Router 26
Figura 6: Linksys WRT54G Wireless Router 26
Figura 7: Arquitetura do Asterisk 40
Figura 8: Realtime e Heartbeat 46
Figura 9: Realtime, Heartbeat e Gateway SIP-E1 47
Figura 10: Realtime, Heartbeat, Gateway SIP-E1 e DNS Round Robin 47
Figura 11: Mapeamento geral da rede de alta disponibilidade 50
Figura 12: Apresentação inicial do MeucciBE 53
Figura 13: Licenciamento do MeucciBE A 53
Figura 14: Licenciamento do MeucciBE B 54
Figura 15: Licenciamento do MeucciBE C 54
Figura 16: Servidor primário inativo 56
Figura 17: Servidor secundário ativo 56
Figura 18: Tela inicial do X-Lite 57
Figura 19: Configuração de conta no X-Lite 58
Figura 20: Configuração das propriedades da conta no X-Lite 58
Figura 21: Configuração final da conta no X-Lite 59
Figura 22: Informações do sistema MeucciBE 60
Figura 23: Criação de ramais no MeucciBE 60
Figura 24: Lentidão do servidor Asterisk e os recursos 61

LISTA DE TABELAS

Tabela 1: Comparação entre os codecs..... 18
Tabela 2: Comparação de controle de chamada entre H.323 e SIP..... 33
Tabela 3: Classificação da disponibilidade (Anual)..... 43
Tabela 4: Configuração dos servidores 50

LISTA DE SIGLAS

APA	*Analog Telefone Adapter*
ARA	*Asterisk Realtime*
ASN.1	*Abstract Syntax Notation One*
BGP	*Border Gateway Protocol*
CPU	*Central Processing Unit*
CS-ACELP	*Conjugate Structure Algebraic Codebook Excited Linear Prediction*
DNS	*Domain Name System*
DSP	*Digital Signal Processing*
IETF	*Internet Engineering Task Force*
GPL	*General Public License*
HA	*High Availability*
HTTP	*Hypertext Transfer Protocol*
IP	*Internet Protocol*
ISDN	*Integrated Services Digital Network*
ITU	*International Telecommunication Union*
LAN	*Local Area Network*
MGCP	*Media Gateway Control Protocol*
MMUSIC	*Multiparty Multimedia Session Control*
MTBF	*Mean Time Between Fail*
MTTR	*Mean Time To Repair*
OEM	*Original Equipment Manufacturer*
PABX	*Private Automatic Branch Exchange*
PCM	*Pulse Code Modulation*
PEP	*Protocol Extensions Protocol*
PER	*Packet Encoding Rules*
QoS	*Quality of Service*
RTP	*Real Time Transport Protocol*
RTSP	*Real Time Streaming Protocol*
SB-ADCPM	*Sub Band Adaptative Differencial Pulse Code Modulation*
SDP	*Session Description Protocol*

SIP	*Session Initiation Protocol*
SMTP	*Simple Mail Transfer Protocol*
TCP	*Transmission Control Protocol*
TDM	*Time Division Multiplexing*
UDP	*User Datagram Protocol*
VoIP	*Voice over IP*

RESUMO

Esta monografia analisa a telefonia voz sobre IP, uma tecnologia que permite fazer chamadas interurbanas e internacionais gratuitas com grande facilidade através da Internet em tempo real. Com o trabalho de Mark Spencer e Jim Dixon surge um *software*, o *Asterisk*, que roda em sistema operacional Linux oferecendo mais benefícios, com licença GPL, cujo custo tecnológico ficou bastante acessível a pequenas e médias empresas e ao usuário residencial. O estudo apresenta soluções e técnicas de Alta Disponibilidade utilizadas com o *Asterisk*, e um estudo de caso demonstrando como o *software* funciona, define e detalha sua história, tipos de licenciamento, seus benefícios, e sua arquitetura. O estudo de caso usa diversas ferramentas gratuitas para garantir a Alta Disponibilidade com o *Asterisk*, incorporado no MeucciBE, e o *heartbeat* que garante a disponibilidade dos servidores no caso de falha de algum deles. Apresenta-se o funcionamento do VoIP, diversos *codecs* de áudio e vídeo utilizados nesta tecnologia de telefonia, e também apresenta as suas arquiteturas que são utilizadas normalmente. O estudo explica a qualidade da voz na tecnologia VoIP, apresenta os principais meios de transmissão dos pacotes envolvidos na qualidade da voz, e os fatores que afetam sua qualidade, comparando os dois principais protocolos do VoIP, o SIP e o H.323, e detalhando os termos de complexidade, extensibilidade, escalabilidade e serviços. O estudo desenvolve o conceito de Alta Disponibilidade, cita os seus conceitos, tipos de disponibilidade e apresenta os métodos e técnicas a serem usados.

Palavras-chaves: *Asterisk*. Voz sobre IP. Alta Disponibilidade. *Heartbeat*. *Codec*.

ABSTRACT

This Thesis analyzes the Voice over IP telephony, a technology that allows long distance and international calls to be made free of charge with great facility through the Internet in a real time. With the help of Mark Spencer and Jim Dixon, arises software called Asterisk which runs on Linux operating system by offering more benefits, with the General Public License that generates the cost of technology accessible to small businesses and resident users. The study presents solutions and techniques of High Availability – HA used with Asterisk, and a case study demonstrating how the software works, details about the history, the definition, types of licensing, benefits and architecture. The case study uses several free software to ensure the HA with Asterisk designed in MeucciBE, and the heartbeat that ensures the availability of the servers in the event of a failure with any member. Also, presents the functionality of VoIP, audio and video codecs and the architectures that are commonly used within this technology. The study further explains the quality of voice, mentioned the main means of transmission of the packets involves, and the factors that affects VoIP, comparing the two main protocols, SIP and H.323 of VoIP, and explaining them in terms of complexity, extensibility, scalability and services. The study went along to develop the concept of HA, list its concepts, types of availability and presents the methods and techniques to be used.

Keywords: Asterisk. Voice over IP. High Availability. Heartbeat. Codec

CAPÍTULO 1 – INTRODUÇÃO

A telefonia voz sobre IP está se tornando cada vez mais popular, pois as vantagens de sua utilização se destacam sobre a telefonia tradicional. Com esta tecnologia é possível fazer telefonemas interurbanos e internacionais gratuita com uma facilidade enorme. Além das facilidades de uso da telefonia voz sobre IP existe também a facilidade de adoção desse sistema, pois necessita apenas de uma conexão de Internet banda larga – já adotada por mais de um bilhão de pessoas no mundo –, uma operadora de telefonia voz sobre IP de confiança e um meio físico para fazer a ligação, que pode ser um computador através de *softwares* gratuitos na Internet, telefone USB ligado ao computador, telefone comum ligado a um Adaptador de Telefone Analógico (ATA).

A migração do sistema convencional para o sistema voz sobre IP despendia um investimento inicial elevado, onde somente grandes empresas eram capazes de alcançar. Esse custo elevado era ainda alimentado pelas soluções encontradas pelos proprietários que tomavam conta do mercado. Com o trabalho de Mark Spencer e Jim Dixon surge um *software*, o *Asterisk*, que roda em sistema operacional Linux oferecendo mais benefícios, tem licença *General Public License* (GPL) que é capaz de realizar funções de um PBX IP, superando o sistema analógico, e possuindo outras funções. Com o *Asterisk*, junto com os *hardwares* criados por Jim Dixon, o custo da tecnologia voz sobre IP ficou bastante acessível a pequenas e médias empresas e ao usuário residencial.

O objetivo principal desta monografia é apresentar soluções e técnicas de Alta Disponibilidade utilizadas com o *Asterisk*, e um estudo de caso demonstrando como o *software* funciona. Tem sido difícil e até quase impossível, especialmente do ponto de vista da rede de telefonia tradicional, obter um sistema confiável, tolerante a falhas e de Alta Disponibilidade com o *Asterisk*. Ao longo dos anos, foi criado um conjunto de ferramentas oriundas do mundo do *Internet Protocol* (IP) para permitir aos integradores a capacidade de fornecer um mecanismo de tolerância à falha no lado da rede de voz sobre IP e de suas implementações. Até o momento, contudo, o fornecimento de *failover* rápido, confiável e robusto nas redes tradicionais de telefonia tem sido quase inexistente. O projeto de *High Availability* (HA) *clusters* está preenchendo esse vazio, pois permite aos integradores de sistemas implementarem o *Asterisk* em ambientes de demandas importantes, onde o tempo

de *downtime* (interrupção do sistema devido a falha ou para manutenção) é um luxo que uma empresa simplesmente não pode se permitir.

Desse modo, esta monografia foi estruturada em oito partes, e para atingir o objetivo, no capítulo 2 será apresentado o funcionamento do VoIP, diversos *codecs* de áudio e vídeo utilizados nesta tecnologia de telefonia, e também apresentará as suas arquiteturas que são utilizadas normalmente.

O capítulo 3 explica a qualidade da voz na tecnologia VoIP, apresenta os principais meios de transmissão dos pacotes envolvidos na qualidade da voz, e os fatores que afetam sua qualidade.

Dá-se enfoque, no capítulo 4, à comparação dos dois principais protocolos do VoIP, o SIP e o H.323, que detalhará os termos de complexidade, extensibilidade, escalabilidade e serviços.

O capítulo 5 é dedicado ao *Asterisk*, e procurará definir e detalhar sua história, tipos de licenciamento, seus benefícios, e finalmente a sua arquitetura.

O capítulo 6 introduz a Alta Disponibilidade, cita os principais conceitos da área, os tipos de disponibilidade e apresentará os métodos e técnicas que podem ser usados para atingir a Alta Disponibilidade.

A partir dessa pesquisa, no capítulo 7 será realizado um estudo de caso usando diversas ferramentas gratuitas para garantir a Alta Disponibilidade com o *Asterisk*, incorporado no MeucciBE, e o *heartbeat* que garante a disponibilidade dos servidores no caso de falha de algum deles.

Finalmente, no capítulo 8 será feita uma conclusão final da monografia e citados alguns pontos possíveis de melhora. Ao final do texto encontram-se Apêndices e Anexos que levaram à conclusão deste trabalho.

CAPÍTULO 2 - VOZ SOBRE IP

VoIP (Voz sobre IP), em inglês *Voice over IP*, é a transmissão de tráfego de voz sobre o protocolo IP[1] baseados em redes de dados. Essa forma de comunicação permite ao usuário realizar chamadas telefônicas utilizando uma conexão de banda larga da Internet em substituição às linhas analógicas de telefonia convencional. Segundo Hersent (*apud* SILVEIRA, 2005) pode-se definir a tecnologia VoIP como a digitalização, codificação da voz, criação de pacotes de dados IP e sua transmissão em uma rede.

A tecnologia VoIP pode ser entendida como o conceito de integração de serviços, foi explorada mais afundo e acabou ganhando força através de recomendações do *International Telecommunication Union* (ITU-T) na série de redes públicas digitais de serviços integrados (ISDN). Apesar de todo o esforço de padronização envolvendo o ISDN, essas redes não se tornaram de fato o padrão para integração de serviços, pois é necessário o uso do IP (COLCHER, 2005, *apud* GONZALEZ). Com o desenvolvimento de algumas técnicas avançadas, como digitalização de voz, protocolos de transmissão em tempo real, priorização do tráfego, mecanismos de controle e o estudo de novos padrões que contribuem na qualidade de serviço, criam-se perfeitas condições para a transmissão de voz sobre IP.

2.1 FUNCIONAMENTO

A plataforma VoIP usa um mecanismo de transformação entre sinais de voz analógicos e digitais no seu funcionamento, utilizando esta conversão para fazer a transmissão pela Internet. A digitalização de um sinal de voz permite que seu armazenamento e transmissão sejam feitos de forma mais eficiente. Um exemplo desta transformação entre os sinais seria quando um usuário-remetente realiza uma chamada telefônica para outro usuário-destinatário. A voz é digitalizada em pacotes de voz, que são comprimidos de modo a exigir menos espaço na rede, e transmitida através da Internet, ocorrendo uma reconversão da voz para que a mensagem seja recebida de forma analógica.

[1] Seqüência numérica que é atribuída aos dispositivos participante de uma rede de computadores que utilizam domínios recursivos para comunicação entre seus nós.

O VoIP faz com que as redes de telefonia se misturem às redes de dados. Dessa forma, é possível que, usando um microfone, caixas de som ou fones de ouvido, e um *software* apropriado, uma pessoa possa realizar uma ligação para telefones convencionais por meio de seu computador. A tecnologia VoIP é usada conjuntamente com os sistemas de *Private Automatic Branch Exchange* (PABX), contudo, muitas empresas estão deixando de ter gastos com centrais telefônicas, substituindo-as por sistemas VoIP (INFOWESTER, 2009).

A voz é introduzida na rede pelo microfone, gerando um sinal analógico que é convertido pelo computador em um sinal digital e codificado para transmissão via rede IP; na outra ponta o sinal passa por uma pilha IP e pelo *buffer de Jitter*[2], passando, então, pelo processo de decodificação e reconversão em sinal analógico para que se torne audível. Existem vários algoritmos usados na compressão da voz, chamado *codecs* (Codificador/Decodificador). A Figura 1 mostra o comportamento do sinal da voz durante a transmissão.

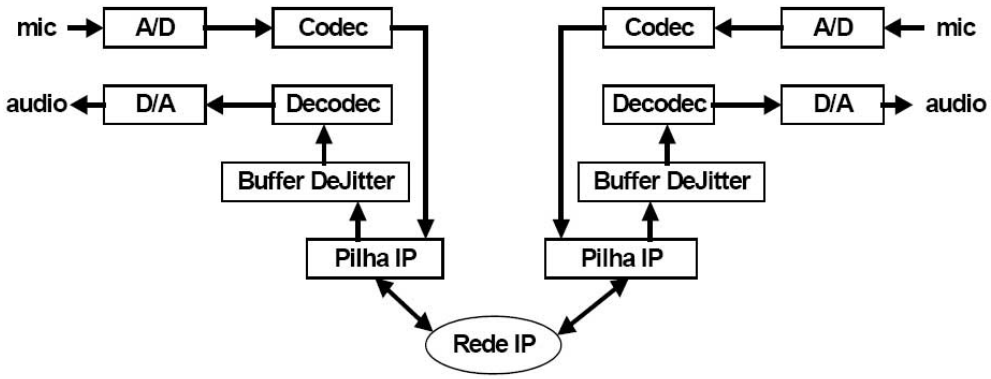

Figura 1: Codificação e Decodificação.
Fonte: PEIXOTO *apud* ALVES, 2004.

2.1.1 *Codecs* (Codificadores/Decodificadores)

A função básica dos *codecs* é manter a qualidade do sinal original o máximo possível, e reduzir a taxa de transmissão dos *bits*. Segundo Gonçalves (*apud* REICHERT, 2004), os *codecs* são responsáveis pela transformação de um sinal analógico em dados para serem transmitidos por uma rede digital, armazenados em algum meio digital ou usados em

[2] Área de dados compartilhados em que pacotes de voz podem ser recolhidos, armazenados e enviados para o processador de voz.

sistema de processamento digital. O codificador converte um sinal analógico de áudio em sinal digital e o decodificador o reconverte em sinal de áudio similar ao sinal original. A qualidade do sinal de áudio reconstituído depende da complexidade do sinal e da eficiência do *codec* utilizado. Segundo Kurose e Ross (2003), em geral, o sinal analógico reconstruído é diferente do sinal original.

Apesar de existirem diversos *codecs* de áudio e vídeo, este trabalho considerará apenas alguns especificamente usados na transmissão de pacotes de voz através da rede de dados. Os principais *codecs* de voz padronizados para a arquitetura voz sobre IP são (SILVEIRA, 2005):

- **G.711** – Utiliza a técnica PCM (*Pulse Code Modulation*) para digitalização do sinal de voz. A taxa de transmissão é de 64kbps. O G.711 é um padrão reconhecido internacionalmente, largamente utilizado na conversão de sinais de voz analógicos para transmissão em redes digitais. A qualidade resultante é adequada para sinais de voz, mas não é considerada boa para sinais de áudio;
- **G.722** – Utiliza uma variante da técnica ADPCM, denominada SB-ADCPM (*Sub Band Adaptative Differencial Pulse Code Modulation*). É utilizado nos canais B (64kbps) da RDSI (Rede Digital de Serviços Integrados) para transmissão de sinais de áudio de média qualidade. O atraso gerado na codificação é pequeno, cerca de 5ms;
- **G.723.1** - O padrão ITU-T G.723.1 (uma combinação de G.721+G.723) produz níveis de compressão digital de voz de 10:1 e 12:1, operando respectivamente a 6.3 kbps e 5.3 kbps, com maior qualidade para a taxa mais alta. A característica de largura de banda reduzida é ideal para telefonia pela Internet em tempo real e para aplicações sobre linhas telefônicas convencionais. O G.723.1 se tornou um padrão emergente para a interoperabilidade da transmissão de voz em plataformas distintas. Testes demonstraram uma qualidade equivalente à qualidade comercial dos serviços de telefonia convencional com apenas 1/10 de largura de faixa utilizada pelos sistemas PCM atuais;
- **G.729** – Utiliza a técnica de codificação denominada CS-ACELP (*Conjugate Structure Algebraic Codebook Excited Linear Prediction*) para codificar um sinal analógico de voz em um sinal digital de 8 Kbps. Uma versão mais enxuta do padrão G.729 pode ser encontrada no padrão G.729a. Este é compatível com o G.729 em termos de taxa de bits e de atraso. Por esse bom desempenho com pouca exigência

de capacidade de processamento, a técnica G.729a tem sido muito utilizada nos sistemas comerciais de VoIP.

A Tabela 1 mostra uma comparação resumida entre os *codecs* mais utilizados na tecnologia VoIP.

Tabela 1: Comparação entre os codecs.

Recomendação ITU	**Método de Compressão**	**Saída do *Codec* (kbps)**	**Atraso de Compressão (ms)**
G.711	PCM	64	0.75
G.728	LD – CELP	16	3 a 5
G.729	CS – CELP	10	10
G.729[a]	CS – CELP	10	10
G.723.1	MP – MLQ	6.3	30
G.723.1	ACELP	5.3	30
G.726	ADPCM	32	1

Depois de realiza a digitalização e a codificação, a voz estará pronta para ser encapsulada e enviada na rede através das arquiteturas definidas para isso.

2.2 ARQUITETURA VOIP

O crescimento vertiginoso da Internet nos últimos anos colocou o protocolo IP em uma posição de destaque no contexto das redes de telecomunicações. Não há dúvida que a tecnologia voz sobre IP possui um grande potencial para fornecer um excelente serviço aos usuários de telefonia e Internet, além de proporcionar uma diminuição dos custos com serviços de telefonia, particularmente de longa distância. Existem três arquiteturas de aplicação da transmissão de VoIP, tais como PC-a-PC, arquitetura com *gateway* e arquitetura híbrida.

Embora as arquiteturas ilustrem o transporte de voz sobre a Internet, existe um consenso que ao menos no curto/médio prazo a aplicação de voz sobre IP para serviços de telefonia, denominado Telefonia IP, se dará apenas em redes privadas ou Intranets, ou ainda na rede de acesso do assinante à central de comutação local. A dificuldade de se imaginar, no momento, serviços de telefonia na Internet reside no fato desta rede ser hoje

do tipo melhor esforço, impedindo que se possa oferecer *Quality of Service* (QoS) adequada ao tráfego de telefonia.

As arquiteturas podem ser incorporadas com a telefonia convencional e a rede local existente. Desta forma, as empresas podem usar esta solução de voz sobre IP para conectar as redes de telefonia da matriz com as filias gerando bastantes benefícios. A seguir são apresentados alguns benefícios que se destacam na utilização do VoIP:

- diminuição de custos com telefonia;
- tarifação reduzida com as ligações internacionais se comparada com telefonia convencional;
- facilidade de integrar a infraestrutura existente;
- não há dependência das operadoras de longa distância;
- utiliza a conexão banda larga à Internet;
- chamada sem custo de VoIP para VoIP.

A disponibilidade e benefícios destas arquiteturas para os usuários, tanto residenciais e corporativos, evidencia o uso crescente da tecnologia VoIP com qualidade de voz satisfatória. A seguir, descrevem-se as três arquiteturas estudadas.

2.2.1 Arquitetura PC-a-PC

Exigem-se, nesta arquitetura, computadores com recursos multimídia (placa de som e microfone) conectados a uma rede de computadores (tipicamente no ambiente corporativo) como, por exemplo, uma rede local (LAN) ou Rede Pública de Telefonia, utilizando um provedor de serviços Internet (tipicamente um ambiente residencial), e que se comunicam para a troca de sinais de voz. Todo o tratamento do sinal de voz (amostragem, compressão e empacotamento) é feito nos computadores, sendo a chamada de voz estabelecida com base no endereço IP do receptor. A Figura 2 apresenta esta arquitetura.

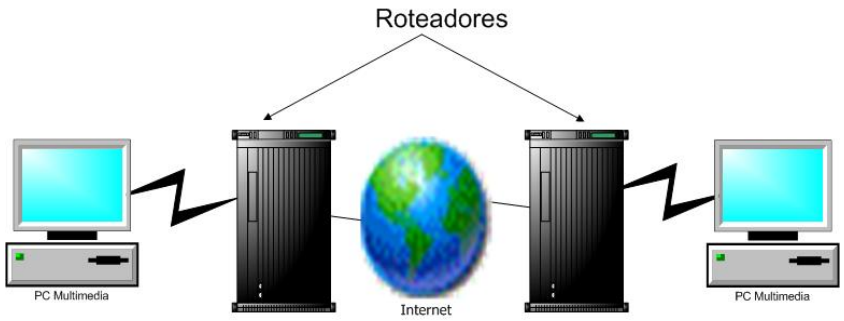

Figura 2: Arquitetura PC-a-PC

2.2.2 Arquitetura com *gateway*

Nesta arquitetura, um telefone analógico é utilizado para gerar e receber a chamada telefônica sobre a Internet. O usuário que está chamando disca para o *gateway* de telefonia IP mais próximo de sua central telefônica local; este *gateway* reconhece e valida o número telefônico do usuário chamador (para fins de autenticação e bilhetagem) e solicita a este que forneça o número do usuário de destino.

O *gateway* de entrada identifica o *gateway* de saída mais próximo do usuário do destino e inicia com este uma sessão para transmissão de pacotes de voz. O *gateway* de saída aciona o telefone receptor e, após a chamada ser atendida, a comunicação fim-a-fim tem inicio, com o sinal de voz sendo enviado através de datagrama IP entre *gateways*. A codificação e empacotamento do sinal de voz são realizados no *gateway* de origem, enquanto a decodificação e desempacotamento são realizados no *gateway* de destino. A digitalização do sinal de voz pode ser realizada na central, no *gateway*, ou mesmo no telefone, caso do RDSI, por exemplo, (BRITO, 1996). A Figura 3 descreve o cenário.

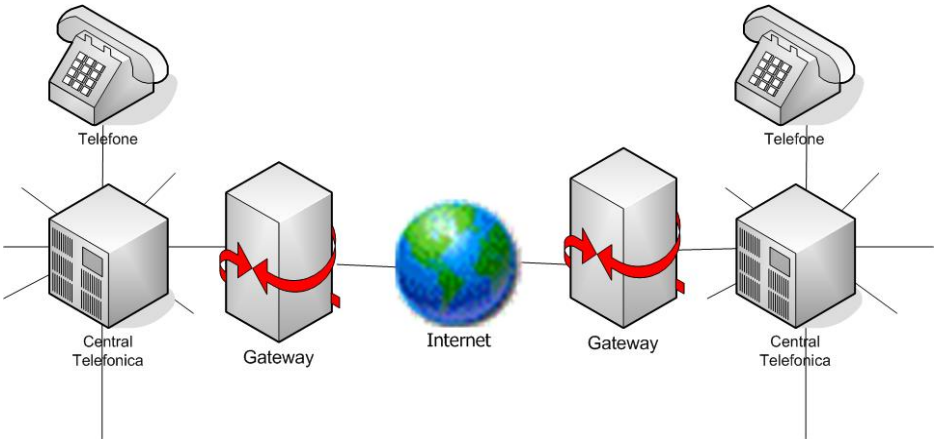

Figura 3: Arquitetura com gateway

2.2.3 Arquitetura híbrida

A arquitetura híbrida é a possibilidade de se fundir as duas arquiteturas descritas anteriormente. Nesta arquitetura, que é apresentada na Figura 4, o usuário de um telefone padrão (analógico convencional) origina (ou recebe) uma chamada para um usuário de PC (ou telefone IP). Em tais situações, obrigatoriamente, ou deve haver um serviço de mapeamento ou a translação de endereços IP em números telefônicos. Existem quatro caminhos unidirecionais neste caso: PC-a-PC, *gateway*-a-*gateway*, PC-a-*gateway*, *gateway*-a-PC. Em todas estas arquiteturas os pontos terminais (PC ou *gateways*) devem empregar o mesmo esquema de codificação de voz (CUNHA, 2009 *apud* SCHIOCHET, 2001).

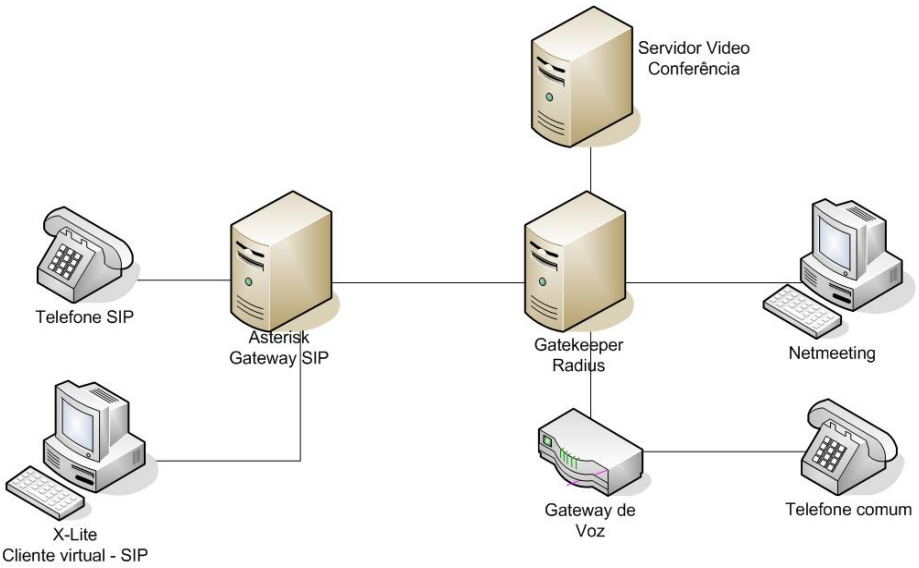

Figura 4: Arquitetura Hibrida

2.3 CONCLUSÃO

VoIP é um típico exemplo de como a Internet está mudando as comunicações, reduzindo o seu custo e simplificando a infraestrutura das empresas. Com a possibilidade de usar uma arquitetura diferentes e previsão de aumento no uso de VoIP, é provável que os invasores busquem cada vez mais formas de explorar essa tecnologia. Que por sua vez já está sujeita

a maioria das ameaças que colocam em risco as redes de dados e a QoS. Se uma empresa optar por adotar VoIP, deverá estar preparada para lidar com a falta de recursos de segurança e QoS, que estão integrados nos sistemas VoIP atuais. A empresa pode tirar proveito da redução de custos que a VoIP oferece com consciência e compromisso com a QoS.

CAPÍTULO 3 - QUALIDADE DA VOZ NA TECNOLOGIA VoIP

A exigência da qualidade da voz na tecnologia VoIP pode ser considerada uma demanda básica devido a alguns fatores, tais como, latência, *jitter*, perda de pacotes, largura da banda e equipamentos; da mesma forma, esta tecnologia corresponde a um desafio técnico enfrentado até o momento. Como diversos tipos de tráfego (dados e voz) são transmitidos no mesmo circuito, onde os de voz são mais sensíveis a rede do que os de dados, a distinção entre seus pacotes gera uma diferença de qualidade, cujo tráfego de voz é inferior com relação ao de dados.

Segundo Gomes (2005), a expressão QoS pode oferecer margem a diferentes tipos de interpretações e definições, havendo, no entanto, um certo consenso entre elas, cuja característica é a diferenciação entre tráfego e tipo de serviços, para que o usuário possa tratar uma ou mais classes de tráfego diferente das demais. Um determinado serviço pode ser caracterizado pelo seu grau de qualidade quando atender às exigências de um determinado patamar, especificado segundo um conjunto de parâmetros mensuráveis.

As novas aplicações (VoIP, multimídia, etc.) necessitam de qualidade de serviço nas redes IP, que é o aspecto fundamental nas suas operações. Assim, é importante entender os seus princípios, parâmetros, mecanismos, algoritmos e protocolos desenvolvidos e utilizados para a obtenção de uma QoS. Inicialmente, é necessário considerar que não são todas as aplicações que legitimamente necessitam de garantias fortes e rígidas de QoS para que seu desempenho seja suficiente.

No caso de VoIP, os pacotes utiliza em *Real Time Transport Protocol* (RTP) que estão dentro do pacote *User Datagram Protocol* (UDP). A tecnologia VoIP não usa o *Transmission Control Protocol* (TCP), pois ele recupera os dados perdidos por retransmissão. Assim, o fornecimento dos dados, deve esperar por todas as retransmissões, gerando grandes atrasos para aplicações em tempo real. O UDP não tem esse problema, pois fornece um serviço orientado a datagrama. O protocolo UDP não tem um mecanismo confiável de entrega dos pacotes em ordem seqüencial, nem garantem a qualidade do serviço ou o tempo de recebimento do destinatário. Ambos os protocolos são muito importantes para a qualidade da voz global (entender da mensagem) e a qualidade da conversa (facilidade do entendimento). O RTP resolve este problema permitindo que o receptor reenvie os pacotes na ordem correta e não espere tempo demasiado pelos pacotes

extraviados ou lentos, pois não precisa de todos os pacotes de voz, mas precisa de um fluxo contínuo e ordenado (ARCOMANO, 2002).

Ao longo dos últimos anos, os dois pontos que mais degradaram a reputação do VoIP são a qualidade e a confiabilidade. A seguir são apresentados os principais problemas que afetam a qualidade da voz em VoIP e o que pode ser feito para maximizar a qualidade.

3.1 FATORES QUE AFETAM A QOS DO VOIP

A QoS é a qualidade de uma chamada através de uma rede, e também se refere à capacidade de priorizar certos tipos de tráfego sobre uma rede IP. No caso do VoIP, isso geralmente significa priorizar o tráfego de voz a um nível maior do que outras formas de tráfego, como o de dados, de modo que o tráfego de voz não seja atrasado ou descartado. A maioria das soluções de QoS focam tanto na reserva, quanto na priorização de recursos. A seguir são apresentados os principais fatores.

3.1.1 Latência

A latência de voz sobre IP provoca atrasos na entrega dos pacotes. A distância física, o número de nós do roteador, a criptografia, e a conversão de voz e dados impactam na latência. Os usuários percebem a latência como uma queda de nível de serviço quando a latência no circuito é maior que 250ms. A ITU recomenda que a latência nunca exceda 150 ms no intervalo de uma comunicação (SCHIOCHET, 2005).

3.1.2 *Jitter*

O *jitter* é a variação no intervalo entre a chegada de pacotes introduzidos pelo comportamento aleatório de atraso na rede (SCHIOCHET, 2005). Dentro do VoIP, o *jitter* ocorre quando os pacotes são enviados e recebidos com variações de tempo, e é efetivamente uma variação de atraso de pacotes, que realmente impacta na qualidade da conversa. Como resultado, muitos prestadores de serviços já oferecem um nível maior de controle de *jitter*.

3.1.3 Perda de pacotes

Diversos fatores acarretam a perda de um pacote IP, tornando-se um fator crítico que implica diretamente na qualidade da voz. Os motivos para a perda de pacotes são vários, dentre eles, o estouro de *buffer* em roteador e *switch*, onde os pacotes são descartados na medida em que encontram o *buffer* cheio; e a grande quantidade de tráfego de acessos à rede, provocando a queda, a lentidão e ruído na conversa. A perda de pacotes não deve exceder 1%, e a maioria dos provedores prestam o serviço com garantia no nível de 0,5% ou menos com relação à perda de pacotes (ALVES, 2004).

3.1.4 Largura da banda

O fator principal que afeta a qualidade de voz em VoIP é a sua conexão à Internet. A banda disponível para VoIP é a chave para a qualidade de voz. Por exemplo, a conexão *dial-up* (discada), não oferece grande qualidade. Uma conexão de banda larga funcionará perfeitamente se não houver irregularidade na transmissão e nem compartilhamento com muitas aplicações.

3.1.5 Equipamento

Os equipamentos de VoIP utilizados podem gerar um grande impacto na qualidade, pois, equipamentos de qualidade inferior são normalmente o mais baratos. Dessa forma, é sempre importante ter informação sobre o roteador, ATA ou Telefonia IP antes do investimento. Às vezes, pode acontecer do *hardware* escolhido ser de excelente qualidade, porém não se encaixa nas necessidades do usuário. Para escolher ATA/Roteador deve-se levar em conta a existência da compressão tecnológica *codecs* e o mecanismo de cancelamento e diminuição de ECO. Existe diferença entre um ATA e um roteador em suas funções e capacidades, pois, um ATA não fornece acesso à Internet, é apenas um dispositivo que prepara a voz para ser transmitida convertendo os sinais analógicos em sinais digitais de dados e, posteriormente, fazer a fragmentação em pacotes que contém informações importantes sobre o seu destino, junto com os dados de voz.

Quando um ATA recebe pacotes, os remonta e os converte de volta aos sinais analógicos. No caso de um roteador, a conexão é feita primeiramente com a Internet. Esse

dispositivo faz a fragmentação, a remontagem e o roteamento dos pacotes aos seus destinos. Ao contrario do ATA, um roteador se comunica com outros roteadores na Internet, pois a voz enviada passa por muitos roteadores antes de chegar ao seu destino.

Portanto, o ATA servir para VoIP caso ele não será usado para fornecer acesso a Internet e também não seja necessário adquirir telefones IPs. Pois bem, ele atua como uma interface entre o telefone e a conexão a Internet via roteador; caso contrário, deve-se usar um roteador. A seguir pode-se ver um roteador na Figura 5.

Figura 5: D-Link DVG-1402SL Express EtherNetwork Broadband Lingo Phone Service VoIP Router

Este modelo atua como um roteador e permite ligar até quatro computadores, e possui um software *Firewall*. Existem vários modelos de roteadores para utilização no serviço de VoIP. A Figura 6 mostra outro modelo de roteador.

Figura 6: Linksys WRT54G Wireless Router

Este roteador é baseado no protocolo Wi-Fi 802.11g, e é um dispositivo três-em-um, cuja plataforma pode conectar 4 computadores diretamente, ou outros roteadores em cascata, como *hubs* ou *switches* para grande escalabilidade.

3.2 CONCLUSÃO

Não é sempre boa a qualidade de serviço nesta tecnologia como esperado e os usuários devem ter a noção que o preço certamente é reduzido, porem nem sempre a qualidade da chamada é alcançada como na telefonia convencional em virtude dos fatores citados anteriormente. Um problema do VoIP é certamente a qualidade da voz nas transmissões, mas este problema está sendo suprimido pela evolução dos *codecs* de voz e também pelo o custo acessível da banda larga. Apesar do avanço significativo da Internet, ainda há muitas falhas no protocolo que estabelece a comunicação, talvez com grande evolução nos principais protocolos (SIP e H.323) a qualidade pode se tornar melhor.

CAPÍTULO 4 – PROTOCOLOS

A fim de prestar serviços de qualidade, o VoIP requer um conjunto de protocolos de controle para o estabelecimento da conexão, capacidades de troca de informação, e controle de conferência. Atualmente, são usados dois protocolos que atendem essa necessidade: o H.323 e o *Session Initiation Protocol* (SIP). Este capítulo comparará os dois protocolos a partir da complexidade, extensibilidade, escalabilidade e serviços.

4.1 PROTOCOLOS H.323 E SIP

A série de recomendações da ITU H.323 define os protocolos e os procedimentos de comunicações multimídia sobre a Internet, incluindo os protocolos H.245 (controle), o H.225 (estabelecimento de conexão), o H.332 (grandes conferências), o H. 235 (segurança), o H. 246 (interoperabilidade com serviços de comutação de circuitos), e os H.450.1, H.450.2, H. 450.3 (serviços suplementares) (SCHULZRINNE; ROSENBERG, 1998).

Segundo Schulzrinne e Rosenberg (1998), o H.323 começou como um protocolo de comunicação multimídia em um segmento de LAN sem QoS, mas tem evoluído para tentar atender às necessidades mais complexas de VoIP. Desenvolvido no *Multiparty Multimedia Session Control* (MMUSIC), pelo grupo de trabalho do *Internet Engineering Task Force* (IETF), o SIP tem uma abordagem diferente de sinalização do VoIP, reutilizando muitos do campo de cabeçalho, regras de codificação, códigos de erro, e mecanismos de autenticação de *Hypertext Transfer Protocol* (HTTP).

Em ambos os casos, os dados de multimídia provavelmente serão trocados através de RTP, de modo que a escolha do conjunto de protocolos não influencia a QoS do VoIP.

4.2 COMPLEXIDADE

O H.323 é um protocolo muito complexo, pois apenas a soma total de sua base de especificações totaliza 736 páginas. O SIP, por outro lado, juntamente com suas extensões de controle de chamada e os protocolos de descrição de sessão, totaliza 128 páginas. O

H.323 define centenas de elementos, enquanto o SIP tem apenas 37 campos e cabeçalhos (32 na especificação base e 5 nas extensões de controle de chamada), cada um com um pequeno número de valores e parâmetros, mas que contém mais informações (SCHULZRINNE; ROSENBERG, 1998).

Conforme Schulzrinne e Rosenberg (1998), o H.323 usa uma representação binária para suas mensagens, baseando-se em *Abstract Syntax Notation One* (ASN.1), que geralmente requer um gerador de código especial para efetuar as análises, e o *Packet Encoding Rules* (PER). O SIP, por outro lado, codifica as mensagens em texto, similar ao HTTP e o *Real Time Streaming Protocol* (RTSP), cujas análises e geração são simples, especialmente quando realizadas com a poderosa linguagem de processamento de texto Perl. A codificação textual também simplifica a depuração, permitindo a entrada manual e analisando as mensagens. A sua semelhança com o HTTP também permite a reutilização de código, e os analisadores existentes HTTP podem ser rapidamente modificados para o uso do SIP.

A complexidade do H.323 decorre, também, pela sua utilização de vários componentes de protocolos, não existindo uma separação clara entre os componentes, contudo, muitos serviços exigem uma interação entre vários deles. O *Call Forward*, por exemplo, exige os componentes de H.450, H.225.0, e H.245. A utilização de vários protocolos diferentes também complica a passagem do *firewall*. O SIP, por outro lado, usa um único pedido que contenha todas as informações necessárias.

O H.323 também prevê um arranjo de opções e métodos de realizar uma única tarefa, pois existem três maneiras distintas pelas quais os H.245 e H.225.0 podem ser utilizados em conjunto. Um aspecto adicional da complexidade do H.323 é a sua duplicação de algumas das funcionalidades presentes em outras partes do protocolo, em particular, fazendo uso do RTP e do RTCP.

4.4 EXTENSIBILIDADE

A extensibilidade é uma forma de medir o protocolo de sinalização do VoIP. Atualmente, a telefonia é tremendamente popular, cujo serviço é critico. O VoIP está pronto para substituir os existentes circuitos comutados de infraestrutura. Como qualquer serviço muito utilizado, os recursos fornecidos evoluem ao longo do tempo e novas aplicações são desenvolvidas. Isso faz com que a compatibilidade entre versões seja uma questão

complexa. Como a Internet é aberta pode-se esperar que as extensões aos protocolos de VoIP sejam difundidas e disseminadas. Isso torna essencial a construção de poderosos mecanismos de extensão desde o início.

O SIP incorporou as lições do HTTP e do *Simple Mail Transfer Protocol* (SMTP), foi construído num conjunto rico de funções de extensibilidade e compatibilidade. Para melhorar ainda mais a extensibilidade os códigos de erros estão hierarquicamente organizados como no HTTP. Existem seis classes básicas que são identificadas por centenas de dígitos no código da resposta, e o protocolo básico de operação é ditado exclusivamente pela classe; os terminais só precisam compreender a classe da resposta; e os outros dígitos fornecem informações adicionais, geralmente úteis, mas não criticas. Como o SIP é similar ao HTTP, os mecanismos a serem desenvolvidos, para a extensibilidade do HTTP, também podem ser usados no SIP. Entre estes, está o *Protocol Extensions Protocol* (PEP), que contém ponteiros para a documentação de vários recursos dentro das próprias mensagens do HTTP (SCHULZRINNE; ROSENBERG, 1998).

O H.323 também fornece mecanismos de extensibilidade, que geralmente são campos de parâmetros não padronizados colocados em vários locais no ASN.1. Estes parâmetros contem um código do fornecedor, seguidos por um valor que só tem significado para o vendedor. Isso permite que cada fornecedor desenvolva suas próprias extensões. No entanto, estas extensões têm algumas limitações, quais sejam: as extensões são limitadas àqueles lugares onde um parâmetro não padronizado foi acrescentado; o H.323 não tem mecanismo para permitir que os terminais troquem informações sobre quais extensões cada um suporta; além disso, o H.323 exige a compatibilidade entre as versões atualizadas. Como algumas características são substituíveis e variáveis ao longo do tempo, o tamanho das codificações aumenta; entretanto, o SIP permite que os cabeçalhos mais antigos e outros recursos desapareçam gradualmente, conforme se tornem desnecessários, mantendo o protocolo e a sua codificação limpa e concisa.

Uma questão crítica para a extensibilidade são os *codecs* de áudio e vídeo. Existem centenas de *codecs* desenvolvidos, muitos dos quais são proprietários. O SIP usa a *Session Description Protocol* (SDP) para transmitir os *codecs* suportados por um ponto final em uma sessão. No H.323 todos os *codecs* devem ser centralmente registrados e padronizados.

Outro aspecto da extensibilidade é a modularidade. O VoIP requer um grande número de funções diferentes, que incluem sinalização básica, controle de conferência,

QoS, acesso de diretório, descoberta de serviços, etc. É certo que os mecanismos para realizar estas funções irão evoluir ao longo do tempo (especialmente com relação ao QoS), pois é complexo repartir as funções de separar e modular os componentes ortogonais, que podem ser trocados ao longo do tempo. O recurso de um protocolo geral para várias funções permite o seu uso em outras aplicações com facilidade. Por exemplo, é mais eficiente ter um único mecanismo de QoS, que é um aplicativo independente, do que inventar um novo protocolo ou mecanismo de QoS para cada aplicação (SCHULZRINNE; ROSENBERG, 1998).

Deve-se considerar que o SIP é razoavelmente modular, pois engloba chamada básica de sinalização, localização de usuário e registro. A sinalização avançada é parte do SIP, mas dentro de uma única extensão. QoS, acessos de diretório, descoberta de serviços, descrição do conteúdo da sessão e controle de conferência, são todos ortogonais, e residem em protocolos separados. Por exemplo, é possível utilizar a capacidade do elemento de descrição do H.245, no SIP, sem nenhuma alteração no mesmo (SILVA, 2009).

O H.323 é menos modular, e define um protocolo de integração vertical para uma única aplicação. A mistura de serviços prestados pelos componentes do H.323 abrange a capacidade de permuta, controle de conferência, operações de manutenção, sinalização básica, qualidade de serviços, registros e descoberta de serviços. No entanto, estes serviços são interligados dentro dos vários sub-protocolos no âmbito do H.323.

A modularidade do SIP permite que seja usado em conjunto com o H.323. Um usuário pode usar o SIP para localizar outro usuário, aproveitando das suas múltiplas possibilidades de pesquisa. Quando o usuário está finalmente localizado, pode usar uma resposta de redirecionamento para uma URL do H.323, indicando que a comunicação real deve ocorrer com o H.323.

4.4 ESCALABILIDADE

Os protocolos H.323 e SIP também diferem em termos de escalabilidade, que pode ser observada em níveis diferentes (SCHULZRINNE; ROSENBERG, 1998):

- **grandes números de domínios:** o H.323 foi originalmente concebido para uso em uma única rede local, e questões como ampla área de endereçamento e localização do usuário não eram uma preocupação. A versão mais recente define o conceito de uma zona, e define os procedimentos para localização do usuário em torno de uma

zona de nomes de email. No entanto, para um grande número de domínios as operações de localização complexa têm problema de escalabilidade. O H.323 não fornece uma maneira fácil de executar a detecção de *loop* numa busca múltipla de domínio, que pode ser feito com monitoração de estado por armazenamento de mensagens, o que não é escalável. O SIP, no entanto, utiliza um algoritmo de detecção de *loop* semelhante ao utilizado em *Border Gateway Protocol* (BGP), que pode ser realizado *in a stateless manner*;

- **servidor de processamento:** em um sistema H.323, os *gateways* de telefonia e *gatekeepers* são obrigados a lidar com as chamadas a partir de uma infinidade de usuários. Da mesma forma, os servidores SIP e *gateways* têm necessidade de lidar com muitas chamadas. Para grandes prestadores de serviços de telefonia IP, o número de chamadas a serem tratadas por um grande servidor pode ser significativo. No SIP uma transação, através de vários servidores e *gateway*, pode ser com estado ou sem estado. No modelo sem estado, um servidor recebe um pedido de chamada, executa algumas operações, encaminha o pedido, e o esquece completamente. Mensagens do SIP contêm estado suficiente para permitir a resposta a ser transmitida corretamente. Além disso, o SIP pode ser realizado em TCP ou UDP. No caso do UDP, não é necessária uma conexão. Isso significa que os grandes servidores de *backbone* podem ser baseadas em UDP e operar sem estado, reduzindo significativamente o uso de memória e melhorando a escalabilidade. O H.323, por outro lado, exige *gatekeepers*, quando estão no *loop* de chamada, para manter o estado da chamada durante toda a ocorrência. Além disso, as conexões são baseadas em TCP, o que significa que um *gatekeeper* deve manter suas conexões TCP durante a duração total da chamada. Isso pode representar sérios problemas de escalabilidade para os *gatekeepers* grandes. Um *gateway*, ou *gatekeeper*, precisará processar a sinalização de mensagens para cada chamada. Quanto mais simples a sinalização, mais rápido ela pode ser processada, e mais chamadas um *gateway* ou *gatekeeper* pode suportar. Como o SIP é mais simples para processar do que o H.323, ele deve permitir mais chamadas por segundo;
- **tamanho de conferência:** o H.323 suporta conferência de múltiplas partes como distribuição de dados *multicast*. No entanto, o SIP possui escala para diferentes tamanhos de conferência, cuja coordenação é totalmente distribuída. Isso melhora a

complexidade da escalabilidade. Além disso, como pode usar UDP ou TCP, o SIP suporta a sinalização de *multicast*, permitindo que um único protocolo possa estabelecer uma sessão cuja escala varia de dois, a milhões de membros;

- ***feedback*:** o H.323 define os procedimentos que permitem aos receptores controlarem codificações de mídia, taxas de transmissão e recuperação de erros. Este tipo de *feedback* faz sentido no ponto-a-ponto de cenários, mas deixa de ser funcional em conferência de multiponto. O SIP depende de RTCP para fornecer *feedback* sobre a qualidade de recepção e obter listas de membros do grupo. O RTCP, como SIP, funciona de uma maneira totalmente distribuída.

4.5 SERVIÇOS

O H.323 e o SIP oferecem serviços equivalentes. Alguns dos serviços de controle de chamadas são listados na Tabela 2. Como pode ser visto, o SIP e o H.323 apóiam serviços semelhantes.

Uma comparação nessas dimensões é um pouco difícil, pois os novos serviços estão sempre sendo adicionados ao SIP e ao H.323. Além dos serviços de chamada de controle, tanto o SIP (quando usado com SDP) quanto o H.323, prestam capacidade de troca de serviços.

Tabela 2: Comparação de controle de chamada entre H.323 e SIP
Fonte: Schulzrinne; Rosenberg, 1998

Features	**SIP**	**H.323**
Blind Transfer	Yes	Yes
Operator Assisted Transfer	Yes	No
Hold	Yes; through SDP	Not yet
Multicast Conferences	Yes	Yes
Multi-unicast Conferences	Yes	Yes
Bridged conferences	Yes	Yes
Forward	Yes	Yes
Call Park	Yes	No
Directed Call Pickup	Yes	No

Enquanto o H.323 fornece um conjunto mais rico de funcionalidade, o SIP oferece suportes avançados para serviços pessoais de mobilidade. Quando um chamador contata

um receptor, o receptor pode redirecionar o chamador para um número de locais diferentes. Cada um destes locais pode ser um URL arbitrário, e contém informações adicionais sobre o terminal do local. Deve-se considerar que este tipo de suporte é limitado no H.323.

4.6 CONCLUSÃO

Este capítulo compara o SIP e o H.323, em termos de complexidade, extensibilidade, escalabilidade e serviços. Considera-se que o SIP oferece um conjunto de serviços semelhantes ao H.323, mas fornece uma complexidade muito inferior, rica extensibilidade, e melhor escalabilidade.

O próximo capítulo apresentar o *software Asterisk*, um PBX baseado em código livre que está revolucionando o mundo das telecomunicações, podendo substituir, inclusive, um PABX de grande porte.

CAPÍTULO 5 - ASTERISK

O *Asterisk* é um *software* PBX baseado em código livre[3] que prove todas as funcionalidades de um PABX tradicional e está revolucionando o mundo das telecomunicações. Ele pode substituir um PABX de grande porte utilizando um computador que se comunica com o mundo VoIP e com a rede pública de telefonia (GONZALEZ, 2007). Segundo Gonçalves (2007), o *Asterisk* é muito mais que um PABX padrão, pois além de ter um excepcional *upgrade* do seu PABX convencional, também adiciona novas funcionalidades a ele.

O Asterisk é o maior projeto de código livre para um PBX, é baseado num *software* que utiliza o Linux como sistema operacional. Além de muitas funcionalidades avançadas, ele custa muito mais barato e é altamente flexível, integra telefones, computadores, rede local e a Internet em uma única plataforma. Baseado em tecnologia aberta e protocolos padrões de mercado, possui as funcionalidades de:

- correio de voz;
- correio eletrônico;
- atendimento automático;
- unidade de resposta audível (URA);
- distribuição automática de chamadas (DAC);
- integração entre telefonia e computadores;
- conectividade com o PABX da empresa e como a rede pública de telefonia comutada (STFC) e celular.

Além disso, o *Asterisk* oferece muitos recursos que só eram encontrados em sistemas de mensagem unificada como:

- música em espera para clientes esperando nas filas, com suporte a *streaming* de mídia, bem como música em formato MP3;
- filas de chamada onde agentes de forma conjunta atendem as chamadas e monitoram a fila;
- integração com softwares para a sintetização da fala *"text to speech"*;

[3] Programa de computador que pode ser usado, copiado, estudado, modificado e redistribuído de acordo com a Licença Pública Geral (GPL).

- registro detalhado de chamadas para integração com sistemas de tarifação e bancos de dados SQL;
- integração com reconhecimento de voz *"automatic speech recognition"*;
- a habilidade de interfaceamento com linhas telefônicas normais.

O Asterisk suporta um volume de protocolos *Time Division Multiplexing* (TDM) para o tratamento e transmissão de voz sobre interfaces de telefonia tradicional, e protocolos de pacotes VoIP, como o SIP e o H.323.

5.1 HISTÓRIA

Como sempre, existe algum motivo que leva a invenção neste mundo. No caso do *Asterisk* não é diferente, a necessidade e o custo são os principais motivos para a sua invenção.

Em 1999, Mark Spencer, com quase quatro mil dólares de capital, abriu uma empresa de suporte técnico comercial e livre ao Linux, chamada *Linux Support Services.* Segundo Gonzalez (2007), com a grande demanda de chamados técnicos, Spencer sentiu a necessidade de um sistema telefônico que pudesse auxiliar no suporte técnico 24 horas realizando algumas tarefas, como: atender automaticamente as ligações, coletar a identificação do cliente, gravar a mensagem de suporte ou dúvida, localizar um técnico disponível e enviar a mensagem. Dessa forma, conseguiriam supostamente atender maior quantidade de chamados, de forma ágil e prática. Com o baixo capital naquela época, Spencer não tinha condições de adquirir um sistema telefônico com as funcionalidades necessárias para esta solução.

Como ele tinha uma experiência de cinco anos com o Linux, participado no desenvolvimento de diversos programas de código aberto e na completa ausência de alguma pessoa que pudesse lhe auxiliar e explicar a complexidade de tal tarefa, decidiu que iria projetar e desenvolver o seu próprio sistema telefônico utilizando equipamentos emprestados. Depois de alguns meses de desenvolvimento, Spencer possuía uma plataforma livre de telefonia que supria suas necessidades na empresa e disponibilizou na Internet com o nome de *Asterisk.* Ele escolheu este nome ao seu projeto de *Asterisk* por ser tanto uma tecla do telefone comum, como também um símbolo curinga no GNU/Linux.

Jim Dixon, do projeto *Zapata Telephony,* na mesma época, sabia que o motivo que elevava preços das placas de telefonia, eram os Processadores de Sinal Digital (DSP).

Insatisfeito, realizou algumas experiências em seu computador pessoal com a placa *Mitel3900C "ISDN Express Development Card"*, escrevendo um *driver* para o FreeBSD (GONZALEZ, 2007).

Comprovando seus estudos em uma máquina de Pentium III 600Mhz, ele concluiu que a placa utilizou pouco processamento e como isto conseguiria gerenciar de 50 a 75 canais e o que limitava era a forma ineficiente de gerenciar a Entrada/Saída (I/O). Com o sucesso deste experimento, e sabendo que o conceito era revolucionário, acabou disponibilizando as informações completas na Internet. Depois de disponibilizar os dados na Internet surgiu a necessidade de desenvolver o *driver* no módulo do *kernel* Linux. E como não tinha experiência suficiente em Linux, começou com algumas dificuldades e mesmo assim disponibilizou-o na Internet. Em algumas horas, recebeu um contato de Spencer que se ofereceu para efetivar o projeto com o uso do Asterisk.

A junção do *Asterisk*, de Spencer, com a placa e módulo do *kernel* Linux, de Dixon, desenvolvida perfeitamente, possibilitou o crescimento do projeto que se tornou um PABX real capaz de se comunicar com telefones e linhas reais (GONZALEZ, 2007 *apud* GONÇALVES, 2006). Em 2001, com o crescimento econômico, resolveram alterar o nome da empresa para Digium.

5.2 TIPOS DE LICENCIAMENTO

A Digium oferece o *Asterisk* em três formas de licenciamento:

- ***Asterisk GNU Public License* (GPL*)***: a licença GPL é a mais encontrada, e permite o uso e alteração do código. A restrição existente é que quaisquer alterações no código fonte têm de ser redistribuídas. Em outras palavras, se você altera o código fonte do Asterisk tem de fornecer as modificações;
- ***Asterisk Business Edition***: é uma licença comercial do *Asterisk*, e não possui recursos adicionais em comparação com a versão GPL, com exceção da proteção contra cópia. A grande vantagem da licença comercial é para desenvolvedores que não desejam abrir o código fonte de seus produtos e não podem ou não querem usar a versão GPL;
- ***Asterisk Original Equipment Manufacturer* (OEM)**: foi criado para fabricantes de centrais telefônicas que não desejam mostrar aos seus clientes que a central é baseada em Asterisk.

Possui uma excelente relação custo/beneficio, pois além de ser um *software* livre, o *Asterisk* pode ser instalado em diversos sistemas operacionais incluindo GNU/Linix, Mac OS, OpenBSD, FreeBSD e Sun Solaris instalados num microcomputador convencional. Esses são os principais fatores que contribuem para o Asterisk ser uma solução de PABX flexível e de baixo custo. Em casos de manutenção, encontra-se assistência técnica facilmente, bem como rápida reposição das peças.

O Asterisk é referência em soluções de telefonia IP e sua comunidade é a que possui maior contribuições para a evolução da tecnologia VoIP que está revolucionando a indústria das telecomunicações, devido à maneira como se relaciona com outras aplicações de rede (ASTERISK, 2007).

5.3 MOTIVOS DE USO DO ASTERISK

Tornou-se uma grande evolução realmente nas telecomunicações, existem diversos benefícios/custos que está levando todas grandes empresas investir nele. Ele trouxe uma mudança profunda em todo o mercado de telecomunicações e voz sobre IP.

5.3.1 Redução de Custos

Abaixo tem-se alguns itens que contribuem para o Asterisk ser uma solução de PABX de baixo custo:

- instalado em um micro-computador convencional;
- utiliza o mesmo cabeamento da rede de dados;
- não há limites para a quantidade de ramais;
- rico em aplicações e recursos avançados;
- utilização dos aparelhos telefônicos convencionais;
- custo das placas analógicas/digitais são acessíveis;
- é um software livre e sem custos;
- instalado em sistemas operacionais livres e sem custos;
- entroncamentos com operadoras VoIP;
- entroncamento entre filiais;
- flexível com rotas de menor custo;

- utilização de SoftPhone como ramal.

5.3.1 Código aberto

Uma das coisas mais fantásticas do *Asterisk* é a grande vantagem de ser um programa de código aberto, que possibilita alterações diretamente no código-fonte em situações onde a necessidade de personalização dele seja de alto nível. E também o seu aspecto importante de estar disponibilizada gratuita na Internet, possibilitando o acesso a qualquer pessoa nela conectado. O *Asterisk* é programado em C.

5.3.3 Ter Controle de Sistema Telefonia

Este é um dos benefícios mais citados, ao invés de esperar alguém configurar o seu PABX proprietário como acontece no convencional onde eles são geralmente lacrados (dificilmente é entregue ao cliente uma documentação sobre o seu funcionamento e nem mesmo uma senha de acesso) após sua ativação, no *Asterisk* existe a liberdade total de configuração e personalização.

5.3.4 Limitações da Arquitetura

O *Asterisk* usa a unidade central de processamento (CPU) do servidor para processar os canais de voz, ao invés de ter um DSP dedicado a cada canal. Enquanto isto permite que o custo fosse reduzido para as placas E1/T1, o sistema é muito dependente do desempenho da CPU (GONÇALVES, 2007).

5.4 ARQUITETURA

Um servidor *Asterisk* é um meio de comunicação entre a Internet e/ou PSTN e seus ramais, sendo responsável por controlar todo o trafego de chamadas passantes. Desta forma, todas as chamadas tendem a passar pelas camadas de sua arquitetura. Conforme se apresenta na Figura 7, a arquitetura é basicamente composta por canais, compressores protocolos e aplicações.

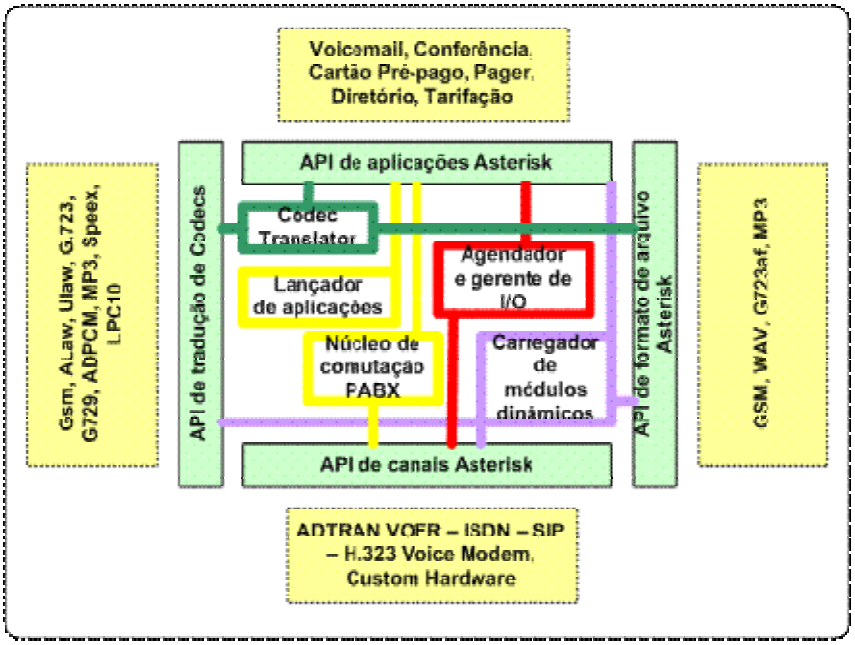

Figura 7: Arquitetura do Asterisk
Fonte: Gonçalves, 2007

Essencialmente, o *Asterisk* age como um *Middleware*[4], fazendo as ligações entre as tecnologias de telefonia e Internet no fundo e as aplicações de telefonia e Internet no topo. As tecnologias de telefonia podem incluir serviços de VoIP com SIP, H.323, IAX e *Media Gateway Control Protocol*(MGCP), usando tanto para *gateway* como para ligação. As aplicações de telefonia incluem serviços tais como, conferencia, *voicemail*, auto-atendimento, música em espera, captura de chamadas, siga-me resposta interativa de voz além de outras facilidades e utilidades para uso.

Segundo Gonçalves (2007), o ponto principal da arquitetura do *Asterisk* é que ele funciona como um *gateway* de mídia entre todos estes protocolos e não apenas com um *Proxy* de sinalização. Com isso, um canal pode estar configurado em IAX2 com CODEC GSM e se comunicar com outro com o SIP e CODEC G.711.

5.5 CONCLUSÃO

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

[4] *Software* de interface que permite interação de diferentes aplicações de *software*, geralmente sobre diferentes plataformas de hardware e infraestrutura, para troca de dados.

hardware de telefonia. Como o *Asterisk*, o *hardware* de telefonia também é aberto e foi desenvolvido por Jim Dixon no projeto *Zapata Telephony*. Com o *software Astersik*, pode implementar e ter bom aproveitamento dele com um sistema de Alta Disponibilidade. Para que se entenda a Alta Disponibilidade faz-se necessário, antes de qualquer coisa, que não é apenas um produto ou uma aplicação que se instala, e sim uma característica de um sistema computacional que são detalhados no próximo capítulo. Existem mecanismos e técnicas, blocos básicos, que podem ser utilizados para aumentar a disponibilidade de um sistema. A simples utilização destes blocos, entretanto, não garante este aumento se não for acompanhado de um completo estudo e projeto de configuração.

CAPÍTULO 6 - ALTA DISPONIBILIDADE

Um sistema de Alta Disponibilidade é aquele que visa manter a disponibilidade dos serviços prestados por um sistema computacional replicando serviço e servidores através da redundância de *hardware* e reconfiguração de *software*.

Um sistema de Alta Disponibilidade tem como principal finalidade a de estar o maior tempo possível disponível para que seus serviços, de alguma forma, não sejam interrompidos. Atualmente, as redes e hardware vêm se tornando cada vez mais rápidos e de certa forma, mais baratos. Isto muitas vezes os torna muito mais atrativos e a principal conseqüência disso é que cada dia uma empresa acaba dependendo mais de um sistema computacional para realizar as tarefas críticas, onde algum tipo de falha poderia causar danos materiais e até mesmo, em algumas ocasiões, a perda de vidas humanas (PEREIRA, 2005).

No ponto de visto computacional, todo sistema está de forma direta ou indireta, sucessível a falhas, que podem ser contornados suando algumas técnicas para garantir a continuidade desse serviço. Estas técnicas podem ser tanto no nível de *hardware* ou *software.*

Segundo Pereira (2005 *apud* Pitanga 2003), não são raras as situações em que disco, fontes de alimentação, interfaces de rede e outras partes do sistema começam a apresentar falhas entre 30.000 e 80.000 horas de uso. Quando isso acontece e seu trabalho começa a ser penalizar, isto levar na hora de pensar em uma solução que seja tolerante a falhas e que apresente uma boa relação custo/benefício. Devido ao caso de algum sistema estar suscetível a falhas, *clusters* são implementados de tal modo que quando um servidor primário falhe, outro servidor secundário tome o lugar do primário de forma transparente ao usuário.

Os conceitos fundamentais e técnicas usadas para construir um sistema de disponibilidade são redundância, manutenção e módulos de falha rápido.

6.1 CONCEITOS

É bastante necessário compreender que a redundância de recursos é um pré-requisito para atingir um alto grau de disponibilidade. Lembrando que, as falhas não são restritas aos

hardwares nem *softwares*, mas também à rede de computadores, às redes elétricas e à localização dos recursos.

Ao respeito a redes de computadores, deve ser levado em consideração os *hubs, switches,* cabos, roteadores e todos os equipamentos físico de uma rede. Já com relação as redes elétricas, deve se entender que qualquer tipo de oscilação ou indisponibilidade do serviço deve ser suprima por outra forma de geração de energia, como *no-break* e até mesmo geradores de energia. E finalmente com relação a localização física dos recursos, deve-se lembrar que um local está propício a furacões, enchentes, terremotos, roubos e até mesmo ataques terroristas, então é importante sempre ter equipamentos em locais distintos, onde na falta de um o outro supra de forma que não pare o serviço.

Contudo, quando se diz que um sistema é de Alta Disponibilidade, este sistema deve envolver não somente uma simples redundância de recursos, mas sim uma poderosa estrutura que o suporte. E pode ser visto que quanto maior o grau de disponibilidade, maior será o custo para tal.

O fator que deve ser levado em consideração em um sistema de Alta Disponibilidade é como medir a disponibilidade dos serviços. O tempo em que os serviços ficam ativos e fora do ar. A disponibilidade de um sistema é calculada utilizando à seguinte formula:

Disponibilidade = MTBF / (MTBF + MTTR)

Onde, *Mean Time Between Fail* (MTBF) e *Mean Time To Repair* (MTTR).

A Tabela 3 apresenta uma visão de classificação de disponibilidade (PEREIRA, 2005).

Tabela 3: Classificação da disponibilidade (Anual)
Fonte: Pereira, 2005

Porcentagem	Tempo-desativado	Classificação
99.5%	3.7 dias	Convencional
99.9%	8,8 horas	Disponível
99.99%	1 minuto	Alta Disponibilidade
99.999%	6 segundos	Resistente a falhas
99.9999%	0.6 segundos	Tolerante a falhas

6.2 CONFIABILIDADE E DISPONIBILIDADE

A confiabilidade é a habilidade de se confiar em alguém ou alguma coisa, em termo computacional é a confiança no serviço e recurso computacional e a disponibilidade é a

qualidade ou estado do que é disponível. Elas são palavras ou termos muito parecidos. O objetivo básico da tolerância a falhas é aumentar a confiabilidade de certo sistema, utilizando-se tolerância a falhas, isso faz com que muitas falhas que podem ocorrer são evitadas e aumentando assim a confiabilidade nos sistemas. Então, é possível concluir que utilizando as técnicas de tolerância a falhas, aumenta-se a confiabilidade e a disponibilidade de um serviço.

6.3 TIPOS DE DISPONIBILIDADE

A Disponibilidade de um sistema computacional, indicada por A(t), é a probabilidade de que este sistema esteja funcionando e pronto para uso em um dado instante de tempo t. Esta disponibilidade pode ser enquadrada em três classes, de acordo com a faixa de valores desta probabilidade. As três classes são:

- **disponibilidade básica**: é aquela encontrada em máquinas comuns, sem nenhum mecanismo especial, em software ou hardware, que vise de alguma forma mascarar as eventuais falhas destas máquinas. Costuma-se dizer que máquinas nesta classe apresentam uma disponibilidade de 99% a 99,9%. Estes dados são empíricos e os tempos não levam em consideração a possibilidade de paradas planejadas, porém são aceitas como o senso comum na literatura da área;
- **alta disponibilidade:** adicionando-se mecanismos especializados de detecção, recuperação e mascaramento de falhas, pode-se aumentar a disponibilidade do sistema, de forma que este venha a se enquadrar na classe de Alta Disponibilidade. Nesta classe as máquinas tipicamente apresentam disponibilidade na faixa de 99,99% a 99, 999%, podendo ficar indisponíveis por um período de pouco mais de 5 minutos até uma hora em um ano de operação. Nesta classe, se encaixam grande parte das aplicações comerciais de Alta Disponibilidade, como centrais telefônicas;
- **disponibilidade contínua:** ela estende a definição de Alta Disponibilidade a um ponto que até mascara as paradas planejadas e obtém uma disponibilidade cada vez mais próxima de 100%, diminuindo o tempo de inoperância do sistema de forma que este venha a ser desprezível ou mesmo inexistente. Contudo, na Disponibilidade Contínua significa que todas as paradas planejadas e não planejadas são mascaradas, e o sistema está sempre disponível.

> Isso é possível somente na teoria, a taxa da disponibilidade seria igual a 1, ou seja, o sistema nunca pararia por nada, (PEREIRA, 2005 *apud* GARCIA, 2003).

6.4 MÉTODOS E TÉCNICAS DE IMPLEMENTAR A ALTA DISPONIBILIDADE

Sendo uma tecnologia que está substituindo o PABX usando servidores, existem técnicas e modelos de soluções em relação a sua alta disponibilidade. Alguns deles são tratados neste trabalho como, por exemplo (VOIPCENTER, 2007):

- ***Asterisk Realtime* (ARA):** armazenam as configurações do *Asterisk* em um banco de dados, principalmente os ramais e o plano de discagem;
- **DNS Round Robin:** autentica os clientes SIP (ramais) através da identificação do ambiente e não pelo endereço IP do servidor. A identificação DNS aponta para mais de um endereço IP, e o servidor entrega as requisições em formato round robin, ou seja, uma requisição para cada endereço IP;
- ***Heartbeat***: é outra técnica e modelo de solução que monitora o status de dois ou mais nodos (servidores) em um ambiente, em caso de detecção de falha, redireciona o serviço para outro servidor de forma transparente para o usuário/ aplicação;
- **CYSNC2:** utilizado para sincronizar arquivos em múltiplos servidores;
- ***Gateways* E1/T1 – SIP (Audiocodes):** elimina a necessidade de hardware adicional aos servidores, e incorpora *Codecs* G.723 e G.729;
- **OpenSER:** é um modelo que, em particular, vem ganhando mais aceitação tecnológica. Ele provê tolerância a falhas além do balanceamento de carga através de um SIP *Proxy Server*;
- **DUNDi**: é um protocolo *peer-to-peer* de pesquisa de clientes VoIP cuja pesquisa é baseada em uma tabela pré-configurada de servidores conhecidos.

A seguir serão apresentadas a combinação de algumas destas técnicas, o seu funcionamento e os recursos envolvidos.

6.4.1 *Realtime e Heartbeat*

O seu funcionamento inclui a sincronização dos bancos de dados em ambas as máquinas, e a criação de um endereço IP virtual através do *software heartbeat*, que aponta para o servidor primário, onde estão conectados os links E1 da operadora. Em caso de falha dessa máquina, automaticamente o *software heartbeat* transfere o IP virtual para a máquina secundária, que continuará disponibilizando os serviços. Neste modelo de redundância os cabos dos links E1 devem ser religados no servidor secundário manualmente. A Figura 8 apresenta um modelo de disponibilidade de servidor.

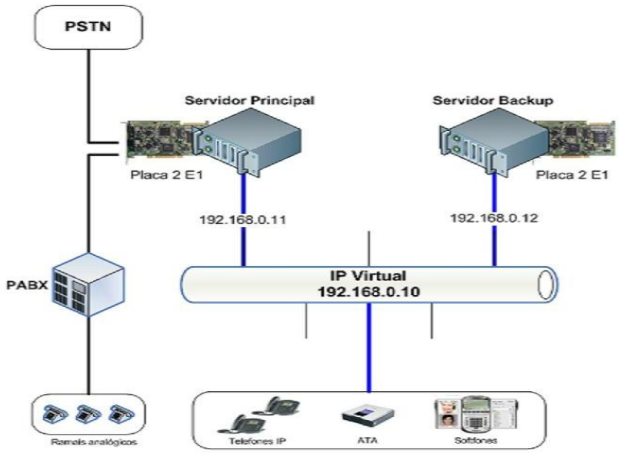

Figura 8: Realtime e Heartbeat

6.4.2 *Realtime, Heatbeat* e *Gateway* SIP –E1

A única diferença deste modelo para o modelo anterior é a remoção dos *hardwares* de comunicação dos servidores e adição de um *gateway* SIP-E1. Neste modelo, não existe nenhuma perda de comunicação com os *links* da operadora, todos os serviços continuam disponíveis, não havendo nenhuma necessidade de intervenção para reestabilização do ambiente. A Figura 9 apresenta os modelos *Realtime, Heatbeat* e *Gateway* SIP –E1.

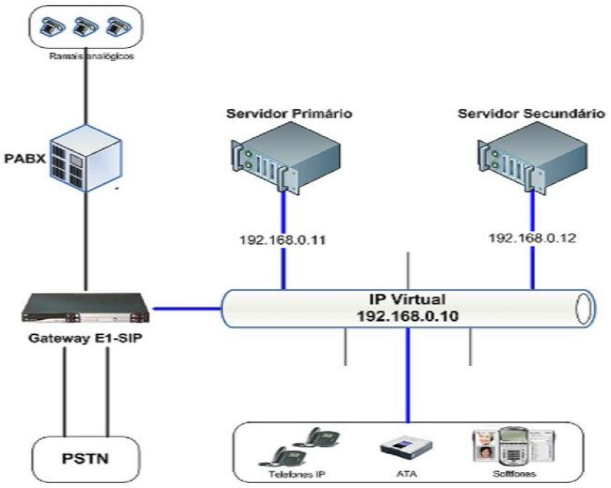

Figura 9: Realtime, Heartbeat e Gateway SIP-E1

6.4.3 *Realtime, Heartbeat, Gateway* SIP-E1 e *DNS Round Robin*

Neste modelo os servidores são configurados aos pares no sistema *heartbeat*, no caso de falha do servidor primário, o secundário continua respondendo para o IP virtual. Os clientes comunicam-se com os servidores através do nome (ipbx.empresa.com.br), cadastrado no servidor de *Domain Name Ssystem (*DNS), que deve ser configurado no formato *Round-Robin*. A Figura 10 apresenta os modelos de *Realtime, Heartbeat, Gateway* SIP-E1 e *DNS Round Robin*.

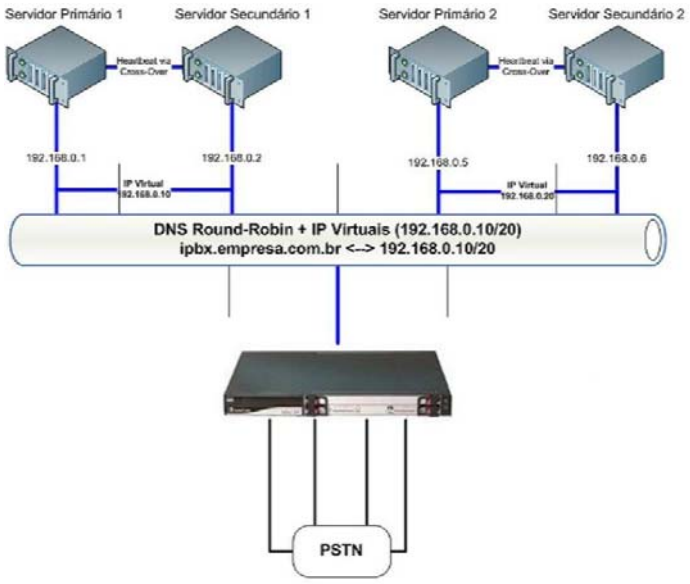

Figura 10: Realtime, Heartbeat, Gateway SIP-E1 e DNS Round Robin

6.5 CONCLUSÃO

A maioria das soluções de Alta Disponibilidade existentes nas empresas se assenta em arquiteturas proprietárias, com custos elevados de aquisição, instalação e manutenção. A Alta Disponibilidade vem se tornando um fator indispensável, pois através de seus ganhos, os sistemas se tornam mais estáveis e mais confiáveis. Sendo assim, as empresas podem estar almejando uma excelência em prestação de serviço, pois com o aumento de novas tecnologias disponiveis, se consegue uma maior satisfação perante seus clientes, fornecedores, e um bom funcionamento das áreas informatizadas da empresa. No próximo capítulo, é apresentado um estudo de caso onde a tecnologia *Asterisk* com Alta Disponibilidade foi implementada.

CAPÍTULO 7 - ESTUDO DE CASO

7.1 INTRODUÇÃO

Neste capítulo é abordada a implementação de um sistema de alta disponibilidade e é demonstrado, de forma detalhada, o funcionamento das ferramentas, das máquinas virtuais, e um IP virtual.

7.2 ESTUDO DE CASO

O estudo de caso implementa um *cluster* de alta disponibilidade para o funcionamento do *Asterisk*, utilizando duas máquinas virtuais, Srv1 (servidor primário) e Srv2 (servidor secundário), um IP virtual e algumas ferramentas. Antes de ser explicitada a configuração da estrutura das máquinas virtuais apresentar-se-á as ferramentas analisadas para que haja uma melhor compreensão de todo o cenário: *VMware Player*, *MeucciBE, X-Lite*, *Hearbeat*, que foram escolhidas devido à sua disponibilidade na Internet. É analisada mais detalhadamente a ferramenta *heartbeat*, pois a conexão entre os dois servidores usará sempre a configuração desta ferramenta para quaisquer comunicações entre si.

7.2.1 Estrutura

A estrutura montada pode ser modificada em sua forma, adaptando-se a diversas necessidades. O Srv1 tem a sua base de arquivos e dados configurados no Srv2, e caso haja qualquer problema, O Srv2 substituirá o Srv1. A configuração dos equipamentos utilizados nas maquinas virtuais, bem como suas especificações, estão apresentadas na tabela 4.

Depois definidas as especificações, foram determinadas a estrutura final e criadas as máquinas virtuais com o VMware Player; em seguida, foi feita a instalação do MeucciBE nas máquinas virtuais onde estão localizados os servidores, conforme pode-se visualizar na Figura 11.

Tabela 4: Configuração dos servidores

Dispositivos	Capacidade
Processador	Intel Celeron D CPU 3.06 GHz
Memória	512 MB DDR 667 MHz
Placa de rede	PCnet Fast 79c971
Placa de som	Creative Sound Blaster Áudio PCI64V
Disco Local	80GB

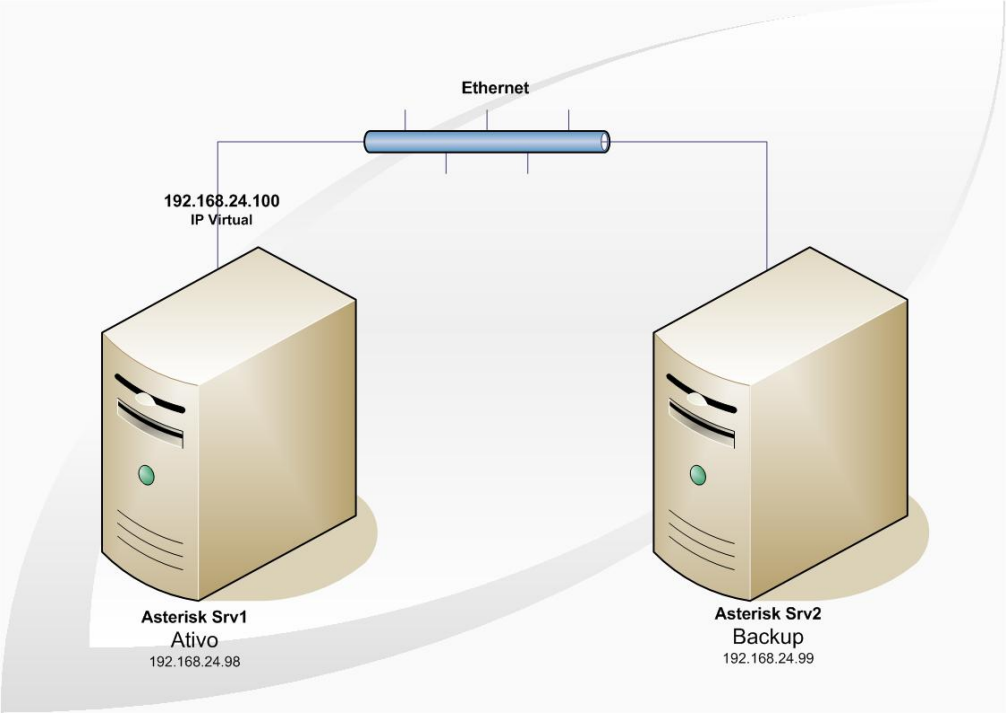

Figura 11: Mapeamento geral da rede de alta disponibilidade

7.2.2 VMware

O VMware é um *software* que permite a criação de uma máquina virtual utilizando um sistema operacional dentro de outro suporte real a *software* de outros sistemas operacionais. Os *softwares* de virtualização, como o VMware, possibilitam a utilização de um ou mais sistemas operacionais simultaneamente num ambiente isolado, criando computadores completos (virtuais) a funcionarem dentro de um computador físico, e que rodam sistemas totalmente distintos.

7.2.2.1 VMware Player

O VMware Player é uma aplicação *desktop* gratuita que permite executar uma máquina virtual em Windows e Linux. Fornece uma interface de usuário intuitiva para executar máquinas virtuais criadas com GSX Server, VMware Workstation, EasyVMX e ESX Server. Existem recursos no VMware Player que lhe permitem configurar as máquinas virtuais, otimizando o desempenho e tirando proveito dos dispositivos do PC local (VMware, 2009).

7.2.3 MeucciBE

O MeucciBE é um sistema baseado na tecnologia de placas de voz DigiVoice com sistema operacional Linux (SUSE) e *Asterisk*, totalmente adaptado às condições de telefonia nacionais e permite a inclusão constante de novas facilidades (Digivoice, 2009). Desse modo, o mundo da tecnologia da informação e telecomunicação convergem numa solução única que proporciona:

- conexões voz sobre IP;
- troncos analógicos e digitais E1;
- espera telefônica;
- atendimento automático;
- salas de conferência.

Existe uma interface gráfica do MeucciBE que é acessada através do *browser* indicando o IP do computador onde está instalado o sistema.

7.2.4 X-Lite

X-lite é um *softphone*[5] SIP desenvolvido pela CounterPath, que permite a combinação de chamadas de voz e vídeo, mensagens instantâneas e uma interface intuitiva de gerenciar. Concebido para trabalhar nos sistemas baseados em IP, o X-Lite fornece soluções VoIP

[5] Aplicativo multimídia que trabalha associado com a tecnologia VoIP dando a possibilidade de fazer chamadas diretamente do seu PC.

que utilizam servidores baseados na telefonia IP dentro de uma empresa, ou numa rede de prestador de serviços VoIP.

7.2.5 Heartbeat

Um *software* que fornece infraestrutura *cluster* de serviços aos seus clientes, permitindo-lhes que saibam sobre a presença ou o desaparecimento de grupo de processos em outras máquinas. Segundo Pereira (2005), a sua principal finalidade é monitorar o estado de um sistema e, dependendo do resultado encontrado, tomar uma decisão.

O termo *heartbeat* representa batimentos cardíacos, e foi escolhido porque tem características de envio de pacotes a outra máquina para verificar se a mesma está ainda operante. É um dos principais programas para a criação de um sistema de *cluster* de alta-disponibilidade. O seu funcionamento depende da configuração de três arquivos principais, *ha.cf, haresources* e *authkeys* cujam configurações estão em anexo.

7.2.5.1 Funcionamento do *Heartbeat*

Ao se configurar o *heartbeat*, o arquivo *ha.cf* verificará, temporariamente, as máquinas configuradas em seu arquivo de configuração, e se por algum motivo uma das máquinas não responder, o *heartbeat* poderá assumir os recursos desse computador que não está respondendo. Com apenas dois nodos IP, o *software* pode assumir os recursos e serviços de um número ilimitado de interfaces IPs. O *heartbeat* tem duas formas de verificar se um computador está ativo ou não: utilizando a própria interface de rede ou uma porta serial (PEREIRA, 2005).

Quanto à sua configuração, o *heartbeat* possui arquivos idênticos nos dois servidores, cuja exceção é o parâmetro *ucast eth1* no arquivo *ha.cf*, onde os endereços IPs são trocados entre os servidores. O arquivo *haresources* é responsável por levantar os recursos as serem monitorados. Neste trabalho constam *apache2*, *mysql*, *motdmeucci*, *dgvfifo*, *amportal* e *postfix*. O último arquivo a ser configurado é o *authkeys*, onde ficam as informações que dizem respeito à autenticação (sha1, md5 e crc).

7.3 DESCRIÇÃO DA INSTALAÇÃO

Após a efetuada a instalação completa do sistema, uma tela é apresentada indicando as informações do MeucciBE, conforme ilustrado na Figura 12.

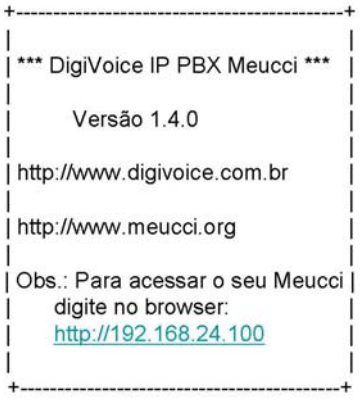

Figura 12: Apresentação inicial do MeucciBE

Conforme o IP apresentado na ilustração, http://192.168.24.100, que deve corresponder ao IP virtual, o sistema foi acessado através do navegador no endereço IP para ter acesso à interface *web* de configuração do sistema. Neste momento, conforme demonstrado na Figura 13, é necessário iniciar o licenciamento do MeucciBE.

Figura 13: Licenciamento do MeucciBE A
Fonte: DigiVoice, 2009

Durante este procedimento, deve-se preencher as informações solicitadas, tais como empresa, contato e *e-mail* válido, uma vez que a chave do licenciamento é recebida através do endereço *web*. Depois de enviadas as informações preenchidas, deve-se dar continuidade ao processo de licenciamento do MeucciBE, como demonstrado na Figura 14. Antes de configurar, a empresa DigiVoice solicita que seja feito um cadastro gratuito da distribuição para que seja fornecida a licença de uso. Ela utiliza este cadastro para ter um controle estatístico de quantos sistemas MeucciBE existem, e dessa forma é mais fácil desenvolver melhorias para o sistema e manter os usuários informados sobre as versões.

Figura 14: Licenciamento do MeucciBE B
Fonte: DigiVoice, 2009.

No momento de registro do software, é enviado junto com a chave de licenciamento (1639759cc0a3d1f8d236ac3d370de66c), um número serial (002159) (ANEXO B), para autenticar e finalizar o licenciamento do MeucciBE, conforme a Figura 15.

Figura 15: Licenciamento do MeucciBE C
Fonte: DigiVoice, 2009

Após o preenchimento do número serial e da chave, é realizado o licenciamento gratuito pela interface *web*. Como o *software* já comporta o *Asterisk*, não é necessária sua instalação, contudo, existe a necessidade de instalar o repositório em que está localizado o *heartbeat* do *OpenSuSE*. Para realizar esta tarefa é utilizada a linha de comando *zypper sa* (endereço: http://download.opensuse.org/distribuition/10.2/repo/oss/_oss), que é o gerenciador de pacotes. Depois deste procedimento, inicia-se a instalação do *heartbeat*, usando o comando de linha *zypper install heartbeat*.

7.4 DESCRIÇÃO DO FUNCIONAMENTO

A lógica de funcionamento do esquema montado, para o estudo de caso, tem os servidores, primário (endereço IP: 192.168.24.98), secundário (endereço IP: 192.168.24.99), e IP virtual (endereço IP: 192.168.24.100) que serão utilizados para acessar os serviços providos na estrutura, quais sejam: *apache2*, *mysql*, *motdmeucci*, *dgvfifo*, *amportal* e *postfix*. Quando inicializado o *heartbeat*, o servidor primário adicionará uma interface virtual com o IP virtual para ativar todos os recursos. Deve-se acrescentar que os recursos e os endereços IPs são configurados pelo *heartbeat*, e o IP virtual tornará o IP do sistema para comunicação entre os servidores. Todas as aplicações do *heartbeat*, em ambos os servidores, se comunicam entre si por meio de envio de pacotes *unicast* ou *broadcast*, através da Ethernet.

No caso da queda do servidor primário, verificado pelo IP virtual, o servidor secundário irá adicionar uma interface virtual, ficando a cargo do IP virtual a transferência dos recursos para o servidor secundário, que se tornará responsável pela resposta a esse IP, conforme visualizado pela Figura 16.

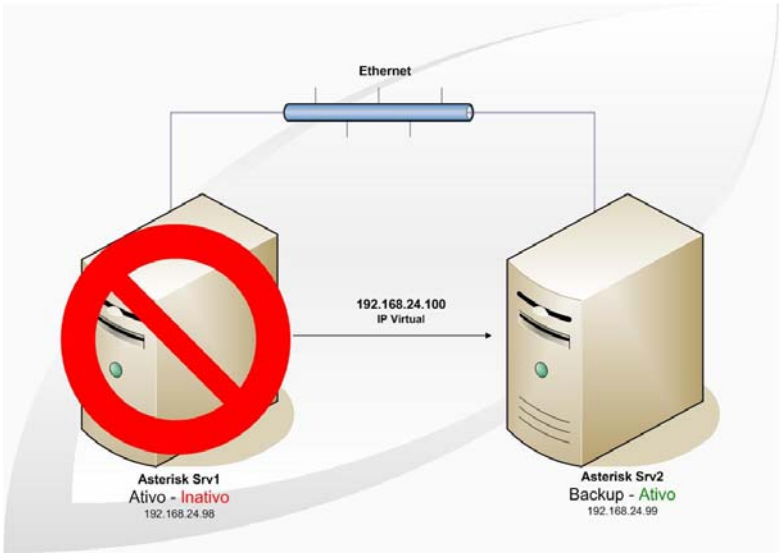

Figura 16: Servidor primário inativo

Essa queda é percebida pelo servidor secundário quando, dentro de um determinado tempo, chamado período de latência, não são recebidos pacotes *heartbeat* do servidor primário. Dessa maneira, todos os recursos, incluindo o IP virtual, se tornam ativos no servidor secundário, que os rodará normalmente sem intervenção humana, em poucos segundos, como exibido na Figura 17. Deve-se citar que os recursos são adquiridos da esquerda para direita e desativados ao contrario.

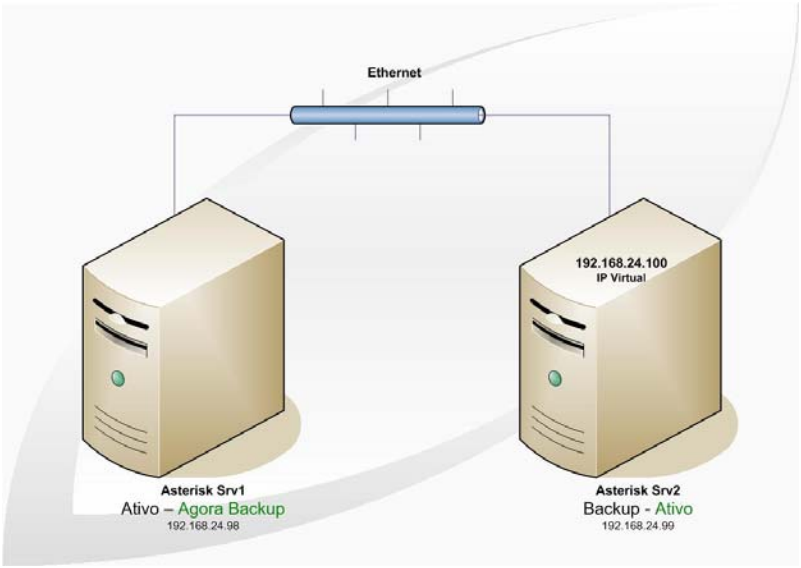

Figura 17: Servidor secundário ativo

Depois deste procedimento, o servidor primário se torna o *backup* até a sua falhaé resolvida e se tornar ativo novamente, recuperando todos os recursos do secundário. Deve-se salientar que o servidor primário só se tornará ativo novamente após a resolução do problema e, quando, na configuração do *heartbeat*, o parâmetro *auto_failback* estiver ON. Caso isso não ocorra, o primário só se tornará ativo novamente quando o secundário falhar.

7.5 TESTE NO SISTEMA

Para validar a configuração feita no estudo de caso, foram realizados testes de simulação de funcionamento, considerando as falhas ocorridas, quando o Srv2 deve tomar todas as funções exercidas pelo Srv1.

Na primeira fase do teste foi instalado e configurado o X-Lite, que realizará a chamada entre os usuários quando o sistema for inicializado. Conforme descrito na Figura 18, ao clicar no *SIP Account Settings* apresenta-se a janela de configuração de conta do X-Lite.

Figura 18: Tela inicial do X-Lite
Fonte: Programa X-Lite

Após este procedimento aparecerá a janela de configuração de conta do X-Lite, descrita na Figura 19.

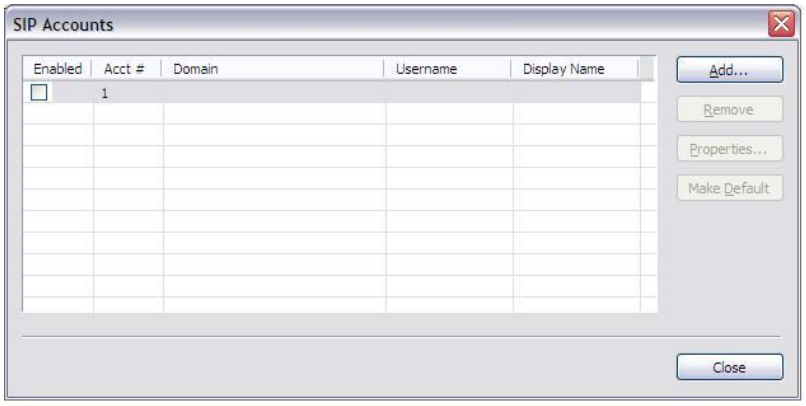

Figura 19: Configuração de conta no X-Lite
Fonte: Programa X-Lite

Com a janela de conta aberta, deve-se clicar no botão *Add* que abrirá outra janela, conforme descrito na Figura 20, em que é preciso preencher todas as informações do sistema de alta disponibilidade, descrito anteriormente no estudo de caso.

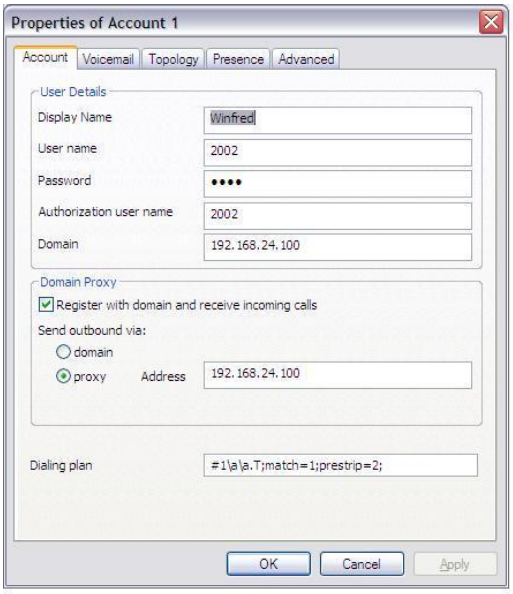

Figura 20: Configuração das propriedades da conta no X-Lite
Fonte: Programa X-Lite

A seguir são descritas as propriedades da conta de usuário:

- *Display Name* – Nome que é exibido no X-Lite;
- *User name* – Nome de usuário criado para ser utilizado na autenticação;
- *Password* – A senha para autenticar a conta no servidor;

- *Authorization user name* – O mesmo nome do usuário;
- *Domain* – IP virtual do servidor.

Depois do preenchimento das informações da conta deve-se clicar em OK, e aparecerá a configuração final, como descrita na Figura 21.

Figura 21: Configuração final da conta no X-Lite
Fonte: Programa X-Lite

Com toda a configuração do X-Lite pronta, a segunda fase será ativar o *heartbeart* no Srv1 (apêndice A.1) e no Srv2 (apêndice A.2). Feito isso, pode-se visualizar que o IP virtual está adicionado no Srv1 (*up*) e no Srv2 (*standby*).

Na próxima fase deve-se criar ramais no MeucciBE através do IP virtual, cujos recursos, *Asterisk*, *MySQL*, Servidor *Web* e Servidor SSH estão todos inicializados. Os detalhes dos recursos e a criação dos ramais podem ser visualizados nas Figuras 22. e 23, respectivamente.

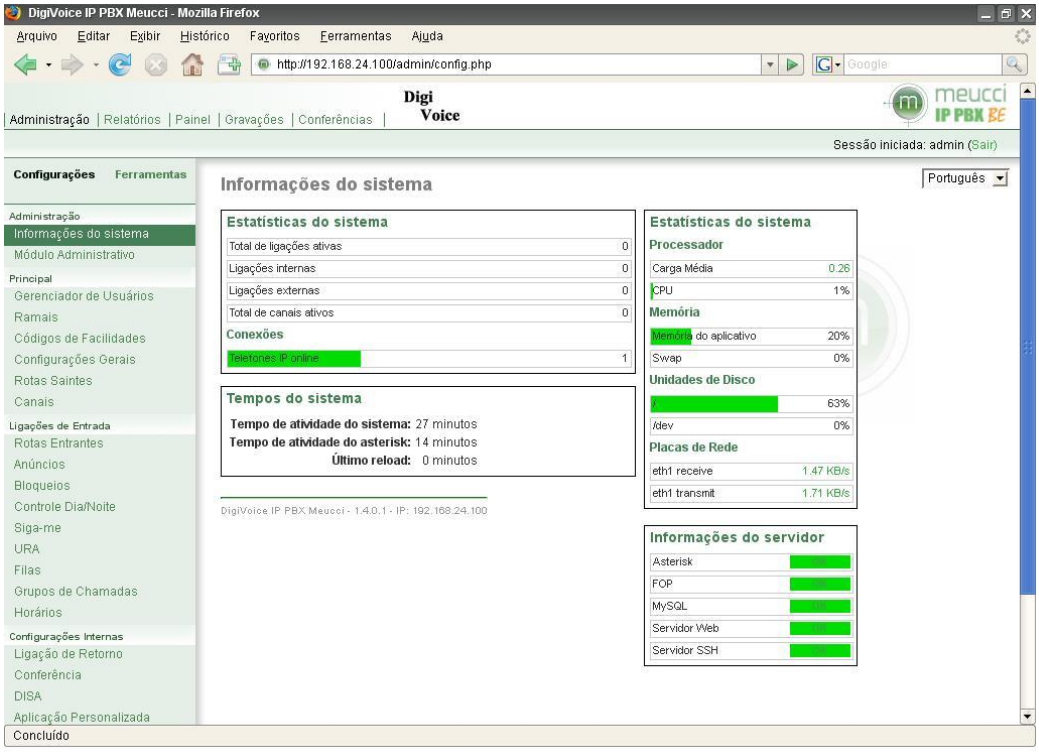

Figura 22: Informações do sistema MeucciBE

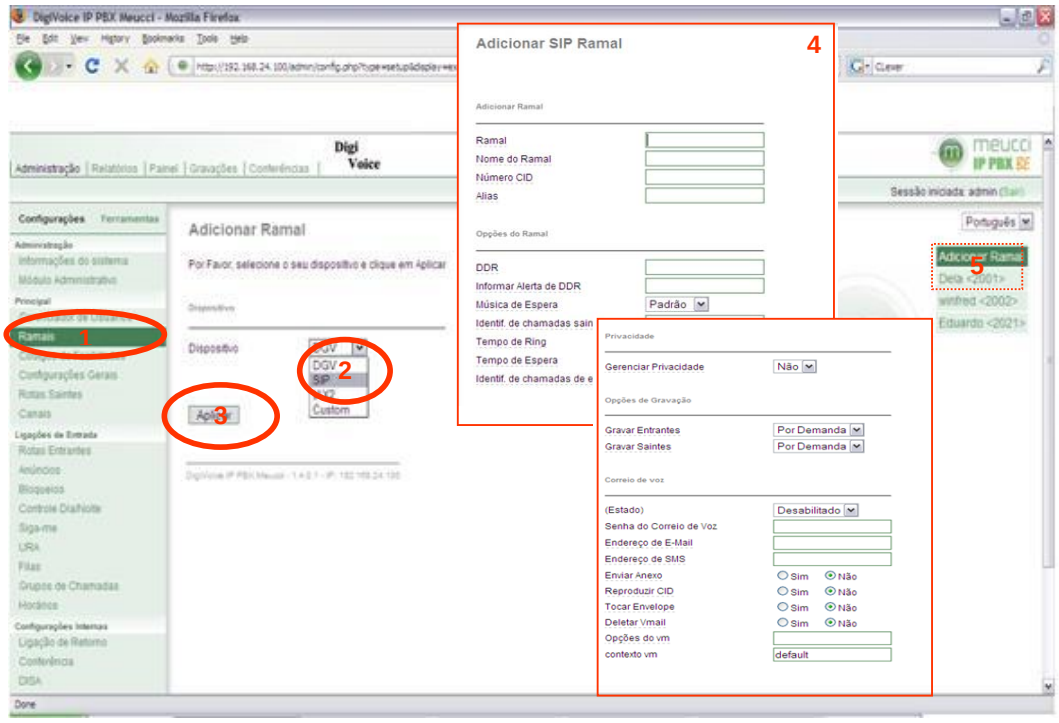

Figura 23: Criação de ramais no MeucciBE
Fonte: DigiVoice, 2009

Na quarta fase, com o sistema funcionando normalmente, foi simulada a queda do Srv1. Retirou-se o cabo de rede às 18:08:39h, conforme detalhado no apêndice A.1. Com a falha no sistema, o IP virtual avisa, às 18:08:45h, e o Srv2 que imediatamente começa a adquirir os recursos do Srv1, concluindo a tarefa às 18:09:01h, conforme detalhado no apêndice A.2. Neste intervalo, a chamada que estava em andamento chegou a cair, observando-se também, que durante a troca de recursos o servidor ficou lento, conforme apresentado na Figura 24.

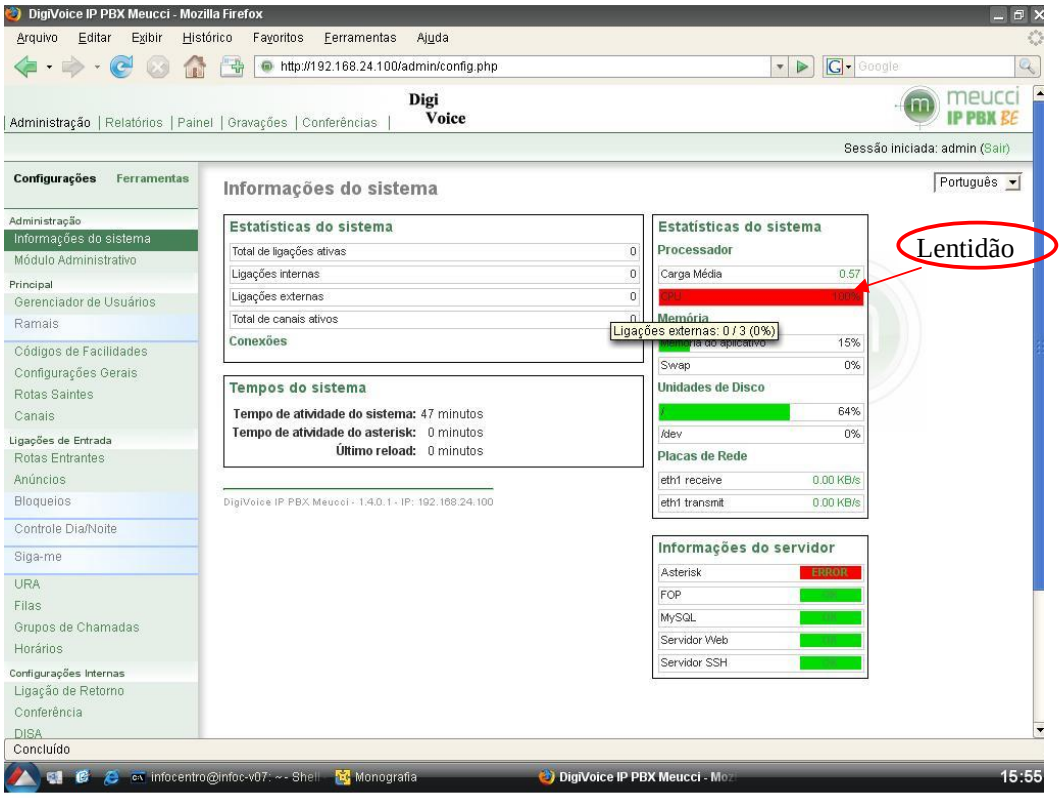

Figura 24: Lentidão do servidor Asterisk e os recursos

Após isso, foi recolocado o cabo de rede e reiniciado o Srv1, às 18:09:07h, conforme detalhado no apêndice A.1. No mesmo instante, o Srv1 aciona o próprio retorno transferindo os recursos locados no Srv2 durante o período de falha, conforme detalhado no apêndice A.2. Às 18:09:32h, o Srv2 concluiu a devolução dos recursos do Srv1 e retornou ao status de *standby*, enquanto o outro retornou ao status de *up*.

7.6 CONCLUSÃO

Como pode ser visto neste estudo de caso, é possível ter um sistema de alta disponibilidade baseada em ferramentas de *open-source,* sem gastar muito dinheiro no quesito de *software*. Foi observado que os dois servidores levam entre 25s à 30s para efetuarem a troca de recursos. A simulação feita no teste pode ser comparada a uma falha do *hardware*, queda de energia ou qualquer outro de tipo de falha que possa interromper os recursos no servidor primário onde os recursos serão transmitidos para o servidor secundário. Neste caso, haverá uma melhoria de colocar dispositivos como *no-breaks* no caso da queda de energia para o servidor não desligar.

CAPÍTULO 8 - CONSIDERAÇÕES FINAIS

Como relatado ao longo do trabalho, o grande avanço das telecomunicações, impulsionado pelo surgimento de novas tecnologias, a invenção do *Asterisk,* no mercado do VoIP, proporcionou ao homem conhecer o mundo todo, e se comunicar internacionalmente sem precisar sair de casa.

Nesta monografia apresentou-se o que existe de mais moderno na comunicação, que é a convergência de vários tipos de dados em apenas uma rede, mais especificamente da voz em uma rede IP. O custo da implementação do VoIP é elevado e acessível apenas a grandes incorporações, mas com a criação do *Asterisk,* o *software* livre tema desta monografia, esse custo inicial baixou absurdamente, tornando o VoIP acessível a todos.

Deve-se considerar que os trabalhos futuros devem ser mais completos quanto aos protocolos, e analisar as métricas de desempenho quantitativas para caracterizar essas diferenças, pois é possível se criar novas invenções de protocolos que ajudarão muito na incorporação da tecnologia VoIP nas infraestruturas existentes.

Grande parte das soluções de Alta Disponibilidade existente nas empresas se assenta em arquiteturas proprietárias, com custos elevados de aquisição, instalação e manutenção. Mas, como demonstrado pelo estudo de caso desta monografia, é possível implementar uma solução de Alta Disponibilidade de código aberto e de baixo custo. Os *clusters* implementados e testados, embora de uma forma simples, permitem ilustrar o impacto desta arquitetura em empresas que pretendam implementar uma solução de Alta Disponibilidade, que pode ser melhorada de uma forma significante para evitar a queda da chamada quando houver a falha de um dos servidores. As soluções apresentadas são de nível empresarial ou acadêmico, concluindo-se que a disponibilidade dos serviços ao utilizador é garantida de uma forma simples e eficaz.

O *heartbeat* encontra-se em atualização permanente, prevendo-se versões recentes com novas funcionalidades. É interessante integrar as funções de *backup* e balanceamento de carga utilizando o mesmo *software*. Este projeto permite uma continuidade no que diz respeito ao aumento da redundância aplicada aos cenários e a implementação de soluções para o balanceamento de carga.

REFERÊNCIAS BIBLIOGRÁFICAS

4LINUX. Disponível em. http://www.4linux.com.br/eventos/2007/asterisk-alta-disponibilidade-mantendo-seu-ip-pbx-999999.html. Acesso em : 30 de Abril, 2009.

ALVES, Carlos Alberto de Souza. VoIP – Voz Sobre IP. Uberlândia. Disponível em: http://si.uniminas.br/TFC/monografias/Monografia%20VoIP%20JULHO%202004.pdf. Acesso em: 09 de Junho, 2009.

ARCOMANO, Robert. VoIP Howto. Disponível em. http://www.bertolinux.com/voip/english/VoIP-HOWTO-4.html. Acesso em: 01 de Junho, 2009.

BRITO, José Marcos Câmara. Voz sobre IP - Projeto e análise de redes de Telecomunicações. Apostila de curso. INATEL, 1996. Disponível em. http://www.dcc.fua.br/~voip/ref_brito.html. 12 Junho, 2009.

GOMES, Anderson Ferreira. Qualidade de serviço. Disponível em http://www.ccet.unimontes.br/arquivos/monografias/66.pdf. 13 Junho, 2009.

GONÇALVES, Flavio E, Asterisk PBX, Guia de Configuração. 3ª Edição, 2007.

GONZALEZ, Felipe Nogaroto. Estudo e implementação de solução de voz sobre IP Baseadas em Softwares Livres. Joinville, 2007. 28p. Dissertação (Bacharel em Sistemas de Informações) – Instituto Superior Tupy, Sociedade Educacional de Santa Catarina.

INFOWESTER. Disponível em: http://www.infowester.com/. Acesso em: 15 de Abril, 2009.

ITU, International Telecomunication Union. 2004. About ITU - History. Disponível em. http://www.itu.int/aboutitu/overview/history.html. 19 de Junho. 2009. Acesso em: 20 de Abril, 2009.

KUROSE, James F;ROSS, Keith W. Redes de Computadores e a Internet. São Paulo: Addison Wesley, 2003. 548p.

LOPES, Ana Luiza S. V. GONÇALVES, Lucelita Ferras. PINHEIRO, Sylvio Pires. Voz Sobre IP – VOIP (TCC em Ciências da Computação) CEFET/RJ, 2003. Disponível em. http://www.cefetrio.hpg.ig.com.br/ciencia_e_educacao/8/trabalhos/rlc_1_2003/VoIP/. Acesso em: 13 de Junho, 2009.

MEUCCIBE. Disponível em: http://www.digivoice.com.br/meucciBE/. Acesso em: 14 de Abril, 2009.

PEREIRA, Roberto Benedito de Oliveira. Alta Disponibilidade em Sistemas GNU/Linux utilizando as ferramentas Drbd, Heartbeat e Mon. Lavras, 2005. 31p, Dissertação (Especialização de Administração em Redes Linux) – Pós – Graduação, Universidade Federal de Lavras.

PINHEIRO, Bruno de Oliveira. VoIP Sobre IP Utilizando Asterisk. Lavras, 2005. Disponível em: http://www.ginux.ufla.br/files/mono-BrunoPinheiro.pdf. Acesso em: 20 de Junho, 2009.

REICHERT, Romeu Hugo. Transmissão de voz sobre redes de pacotes Estudo de caso com H.323. Novo Hamburgo. 2004. Monografia. Centro Universitário Feevale.

SCHIOCHET, Karin Klayton. VOIP – Voz sobre IP e o "Asterisk" um PABX IP open source. Belém. Disponível em. http://www.cci.unama.br/margalho/portaltcc/tcc2005/PDF/011.pdf. 13 Junho, 2009.

SCHULZRINNE, Henning; ROSENBERG, Jonathan. A Comparition of SIP and H.323 for Internet Telephony. Disponível em. http://www.cs.columbia.edu/~hgs/papers/Schu9807_Comparison.pdf. 16 Junho, 2009.

SILVA, Jeremias Neves. Segurança em Protocolo SIP. Disponível em: http://www.jack.eti.br/www/arquivos/apostilas/voip/seguranca_protocolo_sip.pdf. Acesso em: 20 de agosto, 2009.

SILVEIRA, Andre Pompermayer. Analise de implementação de VoIP em um ambiente wireless. Novo Hamburgo, 2005. Trabalho de conclusão de curso. Centro Universitário Feevale.

VOIPCENTER, Asterisk e Alta Disponibilidade. Disponível em: http://albertosato.voipcenter.com.br/ wpcontent/ uploads/ 2007/11/08.11% 2004% 20Alexandre%20Keller.pdf. Acesso em: 14 de Abril, 2009

VMWARE. Disponível em: http://www.vmware.com. Acesso em 10 de Abril, 2009.

X-LITE. Disponível em: http://www.counterpath.net. Acesso em: 14 de Abril, 2009.

APÊNDICE A

A.1 - Srv1

SETTING FILE PERMISSIONS

Permissions OK

STARTING ASTERISK

Asterisk Started

1. heartbeat[10903]: 2009/09/04_18:08:10 WARN: Core dumps could be lost if multiple dumps occur
2. heartbeat[10903]: 2009/09/04_18:08:10 WARN: Consider setting /proc/sys/kernel/core_uses_pid (or equivalent) to 1 for maximum supportability
3. heartbeat[10903]: 2009/09/04_18:08:10 WARN: Logging daemon is disabled --enabling logging daemon is recommended
4. heartbeat[10903]: 2009/09/04_18:08:10 info: **********
5. **heartbeat[10903]: 2009/09/04_18:08:10 info: Configuration validated. Starting heartbeat 2.0.7**
6. **heartbeat[10904]: 2009/09/04_18:08:10 info: heartbeat: version 2.0.7**
7. heartbeat[10904]: 2009/09/04_18:08:10 info: Heartbeat generation: 74
8. heartbeat[10904]: 2009/09/04_18:08:10 info: glib: UDP Broadcast heartbeat started on port 694 (694) interface eth1
9. heartbeat[10904]: 2009/09/04_18:08:10 info: glib: UDP Broadcast heartbeat closed on port 694 interface eth1 - Status: 1
10. heartbeat[10904]: 2009/09/04_18:08:10 info: glib: ucast: write socket priority set to IPTOS_LOWDELAY on eth1
11. heartbeat[10904]: 2009/09/04_18:08:10 info: glib: ucast: bound send socket to device: eth1
12. heartbeat[10904]: 2009/09/04_18:08:10 info: glib: ucast: bound receive socket to device: eth1
13. heartbeat[10904]: 2009/09/04_18:08:10 info: glib: ucast: started on port 694 interface eth1 to 192.168.24.99
14. heartbeat[10904]: 2009/09/04_18:08:10 info: G_main_add_SignalHandler: Added signal handler for signal 17
15. heartbeat[10904]: 2009/09/04_18:08:10 info: Local status now set to: 'up'
16. heartbeat[10904]: 2009/09/04_18:08:11 info: Link srv2:eth1 up.
17. heartbeat[10904]: 2009/09/04_18:08:11 info: Status update for node srv2: status up
18. heartbeat[10904]: 2009/09/04_18:08:11 info: Link srv1:eth1 up.
19. harc[10912]: 2009/09/04_18:08:11 info: Running /etc/ha.d/rc.d/status status
20. heartbeat[10904]: 2009/09/04_18:08:12 info: Comm_now_up(): updating status to active
21. heartbeat[10904]: 2009/09/04_18:08:12 info: Local status now set to: 'active'
22. heartbeat[10904]: 2009/09/04_18:08:12 info: Status update for node srv2: status active
23. harc[10923]: 2009/09/04_18:08:12 info: Running /etc/ha.d/rc.d/status status
24. heartbeat[10904]: 2009/09/04_18:08:22 info: remote resource transition completed.
25. heartbeat[10904]: 2009/09/04_18:08:22 info: remote resource transition completed.
26. heartbeat[10904]: 2009/09/04_18:08:22 info: Initial resource acquisition complete (T_RESOURCES(us))
27. IPaddr[10963]: 2009/09/04_18:08:24 INFO: Resource is stopped
28. heartbeat[10936]: 2009/09/04_18:08:24 info: Local Resource acquisition completed.
29. harc[10989]: 2009/09/04_18:08:24 info: Running /etc/ha.d/rc.d/ip-request-resp ip-request-resp
30. **ip-request-resp[10989]: 2009/09/04_18:08:24 received ip-request-resp 192.168.24.100 OK yes**

31. **ResourceManager[11004]: 2009/09/04_18:08:24 info: Acquiring resource group: srv1 192.168.24.100 apache2 mysql motdmeucci dgvfifo amportal postfix**
32. **IPaddr[11028]: 2009/09/04_18:08:24 INFO: Resource is stopped**
33. **ResourceManager[11004]: 2009/09/04_18:08:24 info: Running /etc/ha.d/resource.d/IPaddr 192.168.24.100 start**
34. **IPaddr[11082]: 2009/09/04_18:08:24 INFO: Using calculated nic for 192.168.24.100: eth1**
35. **IPaddr[11082]: 2009/09/04_18:08:24 INFO: Using calculated netmask for 192.168.24.100: 255.255.248.0**
36. **IPaddr[11082]: 2009/09/04_18:08:24 INFO: Using calculated broadcast for 192.168.24.100: 192.168.31.255**
37. **IPaddr[11082]: 2009/09/04_18:08:25 DEBUG: Sending Gratuitous Arp for 192.168.24.100 on eth1:0 [eth1]**
38. **IPaddr[11073]: 2009/09/04_18:08:25 INFO: Success**
39. **ResourceManager[11004]: 2009/09/04_18:08:26 info: Running /etc/init.d/apache2 start**
40. **ResourceManager[11004]: 2009/09/04_18:08:28 info: Running /etc/init.d/mysql start**
41. **ResourceManager[11004]: 2009/09/04_18:08:30 info: Running /etc/init.d/dgvfifo start**
42. **ResourceManager[11004]: 2009/09/04_18:08:31 info: Running /etc/init.d/amportal start**
43. **ResourceManager[11004]: 2009/09/04_18:08:38 info: Running /etc/init.d/postfix start**
44. heartbeat[10904]: 2009/09/04_18:08:39 WARN: Shutdown delayed until current resource activity finishes.
45. heartbeat[10904]: 2009/09/04_18:08:39 info: Heartbeat shutdown in progress. (10904)
46. heartbeat[11751]: 2009/09/04_18:08:39 info: Giving up all HA resources.
47. **ResourceManager[11763]: 2009/09/04_18:08:40 info: Releasing resource group: srv1 192.168.24.100 apache2 mysql motdmeucci dgvfifo amportal postfix**
48. **ResourceManager[11763]: 2009/09/04_18:08:40 info: Running /etc/init.d/postfix stop**
49. **ResourceManager[11763]: 2009/09/04_18:08:40 info: Running /etc/init.d/amportal stop**
50. **ResourceManager[11763]: 2009/09/04_18:08:40 info: Running /etc/init.d/dgvfifo stop**
51. **ResourceManager[11763]: 2009/09/04_18:08:41 info: Running /etc/init.d/motdmeucci stop**
52. **ResourceManager[11763]: 2009/09/04_18:08:41 info: Running /etc/init.d/mysql stop**
53. **ResourceManager[11763]: 2009/09/04_18:08:43 info: Running /etc/init.d/apache2 stop**
54. **ResourceManager[11763]: 2009/09/04_18:08:44 info: Running /etc/ha.d/resource.d/IPaddr 192.168.24.100 stop**
55. IPaddr[12051]: 2009/09/04_18:08:44 INFO: Success
56. heartbeat[11751]: 2009/09/04_18:08:45 info: All HA resources relinquished.
57. heartbeat[10904]: 2009/09/04_18:08:45 WARN: 1 lost packet(s) for [srv2] [80:82]
58. heartbeat[10904]: 2009/09/04_18:08:45 info: No pkts missing from srv2!
59. **heartbeat[10904]: 2009/09/04_18:08:47 info: srv1 Heartbeat shutdown complete.**
60. heartbeat[12182]: 2009/09/04_18:09:07 WARN: Core dumps could be lost if multiple dumps occur
61. heartbeat[12182]: 2009/09/04_18:09:07 WARN: Consider setting /proc/sys/kernel/core_uses_pid (or equivalent) to 1 for maximum supportability
62. **heartbeat[12182]: 2009/09/04_18:09:07 WARN: Logging daemon is disabled --enabling logging daemon is recommended**
63. **heartbeat[12182]: 2009/09/04_18:09:07 info: **************************
64. **heartbeat[12182]: 2009/09/04_18:09:07 info: Configuration validated. Starting heartbeat 2.0.7**
65. **heartbeat[12183]: 2009/09/04_18:09:07 info: heartbeat: version 2.0.7**

66. heartbeat[12183]: 2009/09/04_18:09:07 info: glib: UDP Broadcast heartbeat started on port 694 (694) interface eth1
67. heartbeat[12183]: 2009/09/04_18:09:07 info: glib: UDP Broadcast heartbeat closed on port 694 interface eth1 - Status: 1
68. heartbeat[12183]: 2009/09/04_18:09:07 info: glib: ucast: write socket priority set to IPTOS_LOWDELAY on eth1
69. heartbeat[12183]: 2009/09/04_18:09:07 info: glib: ucast: bound send socket to device: eth1
70. heartbeat[12183]: 2009/09/04_18:09:07 info: glib: ucast: bound receive socket to device: eth1
71. heartbeat[12183]: 2009/09/04_18:09:07 info: glib: ucast: started on port 694 interface eth1 to 192.168.24.99
72. heartbeat[12183]: 2009/09/04_18:09:07 info: G_main_add_SignalHandler: Added signal handler for signal 17
73. heartbeat[12183]: 2009/09/04_18:09:07 info: Local status now set to: 'up'
74. heartbeat[12183]: 2009/09/04_18:09:08 info: Link srv2:eth1 up.
75. heartbeat[12183]: 2009/09/04_18:09:08 info: Status update for node srv2: status active
76. heartbeat[12183]: 2009/09/04_18:09:08 info: Link srv1:eth1 up.
77. **ResourceManager[12228]: 2009/09/04_18:09:15 info: Acquiring resource group: srv1 192.168.24.100 apache2 mysql motdmeucci dgvfifo amportal postfix**
78. IPaddr[12252]: 2009/09/04_18:09:16 INFO: Resource is stopped
79. ResourceManager[12228]: 2009/09/04_18:09:16 info: Running /etc/ha.d/resource.d/IPaddr 192.168.24.100 start
80. IPaddr[12308]: 2009/09/04_18:09:16 INFO: Using calculated nic for 192.168.24.100: eth1
81. IPaddr[12308]: 2009/09/04_18:09:16 INFO: Using calculated netmask for 192.168.24.100: 255.255.248.0
82. IPaddr[12308]: 2009/09/04_18:09:16 INFO: Using calculated broadcast for 192.168.24.100: 192.168.31.255
83. IPaddr[12308]: 2009/09/04_18:09:16 DEBUG: Sending Gratuitous Arp for 192.168.24.100 on eth1:0 [eth1]
84. IPaddr[12297]: 2009/09/04_18:09:16 INFO: Success
85. **ResourceManager[12228]: 2009/09/04_18:09:16 info: Running /etc/init.d/apache2 start**
86. **ResourceManager[12228]: 2009/09/04_18:09:19 info: Running /etc/init.d/mysql start**
87. **ResourceManager[12228]: 2009/09/04_18:09:21 info: Running /etc/init.d/dgvfifo start**
88. **ResourceManager[12228]: 2009/09/04_18:09:23 info: Running /etc/init.d/amportal start**
89. **ResourceManager[12228]: 2009/09/04_18:09:30 info: Running /etc/init.d/postfix start**
90. **heartbeat[12218]: 2009/09/04_18:09:31 info: local HA resource acquisition completed (standby).**
91. **heartbeat[12183]: 2009/09/04_18:09:31 info: Standby resource acquisition done [foreign].**
92. heartbeat[12183]: 2009/09/04_18:09:31 info: Initial resource acquisition complete (auto_failback)
93. heartbeat[12183]: 2009/09/04_18:09:32 info: remote resource transition completed.

A.2 – Srv2

1. heartbeat[10015]: 2009/09/04_18:08:10 WARN: Core dumps could be lost if multiple dumps occur
2. heartbeat[10015]: 2009/09/04_18:08:10 WARN: Consider setting /proc/sys/kernel/core_uses_pid (or equivalent) to 1 for maximum supportability
3. heartbeat[10015]: 2009/09/04_18:08:10 WARN: Logging daemon is disabled --enabling logging daemon is recommended
4. heartbeat[10015]: 2009/09/04_18:08:10 info: **********

5. **heartbeat[10015]: 2009/09/04_18:08:10 info: Configuration validated. Starting heartbeat 2.0.7**
6. heartbeat[10016]: 2009/09/04_18:08:10 info: heartbeat: version 2.0.7
7. heartbeat[10016]: 2009/09/04_18:08:11 info: Heartbeat generation: 52
8. **heartbeat[10016]: 2009/09/04_18:08:45 info: Received shutdown notice from 'srv1'**.
9. heartbeat[10016]: 2009/09/04_18:08:45 info: Resources being acquired from srv1.
10. heartbeat[10057]: 2009/09/04_18:08:45 info: acquire local HA resources (standby).
11. heartbeat[10057]: 2009/09/04_18:08:45 info: local HA resource acquisition completed (standby).
12. heartbeat[10016]: 2009/09/04_18:08:45 info: Standby resource acquisition done [all].
13. heartbeat[10058]: 2009/09/04_18:08:45 info: No local resources [/usr/lib/heartbeat/ResourceManager listkeys srv2] to acquire.
14. harc[10077]: 2009/09/04_18:08:45 info: Running /etc/ha.d/rc.d/status status
15. mach_down[10087]: 2009/09/04_18:08:45 info: Taking over resource group 192.168.24.100
16. ResourceManager[10107]: 2009/09/04_18:08:45 info: Acquiring resource group: srv1 192.168.24.100 apache2 mysql motdmeucci dgvfifo amportal postfix
17. IPaddr[10131]: 2009/09/04_18:08:46 INFO: Resource is stopped
18. ResourceManager[10107]: 2009/09/04_18:08:46 info: Running /etc/ha.d/resource.d/IPaddr 192.168.24.100 start
19. IPaddr[10185]: 2009/09/04_18:08:46 INFO: Using calculated nic for 192.168.24.100: eth1
20. IPaddr[10185]: 2009/09/04_18:08:46 INFO: Using calculated netmask for 192.168.24.100: 255.255.248.0
21. IPaddr[10185]: 2009/09/04_18:08:46 INFO: Using calculated broadcast for 192.168.24.100: 192.168.31.255
22. IPaddr[10185]: 2009/09/04_18:08:47 DEBUG: Sending Gratuitous Arp for 192.168.24.100 on eth1:0 [eth1]
23. IPaddr[10176]: 2009/09/04_18:08:47 INFO: Success
24. ResourceManager[10107]: 2009/09/04_18:08:47 info: Running /etc/init.d/apache2 start
25. heartbeat[10016]: 2009/09/04_18:08:49 WARN: node srv1: is dead
26. heartbeat[10016]: 2009/09/04_18:08:49 info: Dead node srv1 gave up resources.
27. heartbeat[10016]: 2009/09/04_18:08:49 info: Link srv1:eth1 dead.
28. ResourceManager[10107]: 2009/09/04_18:08:50 info: Running /etc/init.d/mysql start
29. ResourceManager[10107]: 2009/09/04_18:08:52 info: Running /etc/init.d/dgvfifo start
30. ResourceManager[10107]: 2009/09/04_18:08:53 info: Running /etc/init.d/amportal start
31. ResourceManager[10107]: 2009/09/04_18:09:00 info: Running /etc/init.d/postfix start
32. **mach_down[10087]: 2009/09/04_18:09:01 info: /usr/lib/heartbeat/mach_down: nice_failback: foreign resources acquired**
33. **heartbeat[10016]: 2009/09/04_18:09:01 info: mach_down takeover complete.**
34. mach_down[10087]: 2009/09/04_18:09:01 info: mach_down takeover complete for node srv1.
35. **heartbeat[10016]: 2009/09/04_18:09:07 info: Heartbeat restart on node srv1**
36. heartbeat[10016]: 2009/09/04_18:09:07 info: Link srv1:eth1 up.
37. heartbeat[10016]: 2009/09/04_18:09:07 info: Status update for node srv1: status init
38. heartbeat[10016]: 2009/09/04_18:09:07 info: Status update for node srv1: status up
39. harc[10860]: 2009/09/04_18:09:07 info: Running /etc/ha.d/rc.d/status status
40. harc[10870]: 2009/09/04_18:09:07 info: Running /etc/ha.d/rc.d/status status
41. heartbeat[10016]: 2009/09/04_18:09:09 info: Status update for node srv1: status active
42. harc[10880]: 2009/09/04_18:09:09 info: Running /etc/ha.d/rc.d/status status
43. heartbeat[10016]: 2009/09/04_18:09:09 info: remote resource transition completed.

44. heartbeat[10016]: 2009/09/04_18:09:09 info: srv2 wants to go standby [foreign]
45. heartbeat[10016]: 2009/09/04_18:09:09 info: standby: srv1 can take our foreign resources
46. heartbeat[10890]: 2009/09/04_18:09:09 info: give up foreign HA resources (standby).
47. **ResourceManager[10900]: 2009/09/04_18:09:10 info: Releasing resource group: srv1 192.168.24.100 apache2 mysql motdmeucci dgvfifo amportal postfix**
48. **ResourceManager[10900]: 2009/09/04_18:09:10 info: Running /etc/init.d/postfix stop**
49. **ResourceManager[10900]: 2009/09/04_18:09:10 info: Running /etc/init.d/amportal stop**
50. **ResourceManager[10900]: 2009/09/04_18:09:10 info: Running /etc/init.d/dgvfifo stop**
51. **ResourceManager[10900]: 2009/09/04_18:09:10 info: Running /etc/init.d/motdmeucci stop**
52. **ResourceManager[10900]: 2009/09/04_18:09:11 info: Running /etc/init.d/mysql stop**
53. **ResourceManager[10900]: 2009/09/04_18:09:13 info: Running /etc/init.d/apache2 stop**
54. **ResourceManager[10900]: 2009/09/04_18:09:14 info: Running /etc/ha.d/resource.d/IPaddr 192.168.24.100 stop**
55. IPaddr[11181]: 2009/09/04_18:09:15 INFO: Success
56. **heartbeat[10890]: 2009/09/04_18:09:15 info: foreign HA resource release completed (standby).**
57. **heartbeat[10016]: 2009/09/04_18:09:15 info: Local standby process completed [foreign].**
58. heartbeat[10016]: 2009/09/04_18:09:32 WARN: 1 lost packet(s) for [srv1] [58:60]
59. heartbeat[10016]: 2009/09/04_18:09:32 info: remote resource transition completed.
60. heartbeat[10016]: 2009/09/04_18:09:32 info: No pkts missing from srv1!
61. **heartbeat[10016]: 2009/09/04_18:09:32 info: Other node completed standby takeover of foreign resources.**

ANEXO A

ha.cf

Nele ficam as configurações gerais que dizem respeito ao *heartbeat*.

Arquivo localizado em /etc/ha.d/ha.cf

debugfile ***/var/log/ha-debug***

Esta opção indica onde será escrito as mensagens de debugação.

logfile ***/var/log/ha-log***

Esta opção especifica o local do arquivo de log.

logfacility ***local1***

Facilidade de uso para syslog/logger.

keepalive ***500ms***

Tempo de vida entre os pacotes de heartbeat.

warntime ***1000ms***

Quanto tempo após deve ser usado “O pacote heartbeat atrasado”.

deadtime ***2000ms***

Esta opção indica quanto tempo o host é declarado como morto.

initdead ***20000ms***

Verifica o tempo necessário para declarar o servidor morto quando o *heartbeat* for inicializado pela primeira vez. Deve ser pelo menos duas vezes o tempo de *deadtime*.

auto_failback ***on***

Esta opção determina se os recursos retornarão automaticamente como o nodo primário ou continuarão no nodo servindo até esse nodo falhasse.

Opções:

On: ativa

Off: desativa

*Legacy:*Ativa automaticamente o *failback* no sistema quando todos os nos não suportarem está opção.

realtime ***on***

Ativa ou desativa a execução em tempo real, o valor padrão é *on*.

udpport ***694***

Especifica a porta udp que será usada para *broadcast* e *unicast.*

ucast ***eth1*** ***192.168.24.98/99***

Configuração do *unicast* e a porta que será utilizada.

bcast ***eth1***

Indica qual interface devem ser enviados os pacotes *heartbeat*.

baud ***19200***

A velocidade da porta serial.

node ***srv1 srv2***

Esta opção especifica quais nos fazem parte do *Cluster*.

Haresources

É o arquivo responsável por levantar os recursos as serem monitorados, nesta estrutura constam apache2, mysql, motdmeucci, dgvfifo, amportal e postfix.

Arquivo localizado em /etc/ha.d/haresources

srv1 192.168.24.100 apache2 mysql motdmeucci dgvfifo amportal postfix

onde srv1 é o servidor primário seguindo pelo IP virtual e depois os recursos. Os recursos são adquiridos da esquerda para direita e desativados ao contrario.

Authkeys

O último arquivo a ser configurado é o *authkeys*, nele ficam as informações que dizem respeito à autenticação. Existem três métodos de autenticação:

sha1 – método mais seguro sem se importar com o consumo da CPU

md5 – quando a rede não é segura mas não quer economizar recursos da CPU

crc – quando o heartbeat está sendo executado em uma rede segura e este método exige menos recursos.

Foi utilizado o primeiro método nesta estrutura e após a configuração deste arquivo foi mudado a permissão dele com a seguinte linha de comando: *chmod 600 /etc/ha.d/authkeys*

Arquivo localizado em /etc/ha.d/authkeys

1 sha1 key4auth

Terminado as configurações da estrutura, foram realizados alguns testes para verificar o funcionamento do sistema de alta disponibilidade.

ANEXO B

Licenciamento de software 'MeucciBE'.
****

Data: 28. Apr 2009, 18:44
Empresa: winsystem
Contato: winfred
Email: mailto:winfred.cobby@gmail.com
Hardware: i686
Processador: i686
Ambiente de trabalho: i386
Mac Address: 00:0C:29:D1:10:82
N. Serial: 002159
Chave de licenciamento: 1639759cc0a3d1f8d236ac3d370de66c

Politicas de Suporte

De forma pratica e simples a DigiVoice disponibiliza gratuitamente os seguintes meios de comunicacao:

Forum: http://www.digivoice.com.br/forum2
FAQ: http://www.digivoice.com.br/meucciBE/faq
Manual do Usuario:
http://downloads.digivoice.com.br/pub/meucci/manual_usuario_meucci_be.pdf

A DigiVoice tambem disponibiliza ao mercado um Contrato de Suporte Tecnico para que haja um acompanhamento de todas as fases de implantacao do Meucci BE, desde a instalacao ate sua customizacao.

Para maiores informacoes entre em contato com nosso Depto. Comercial:
Telefone: (11) 2191 6363 ou e-mail: comercial@digivoice.com.br

Para usuarios da versao 1.4.0 do meucciBE que forem utilizar trunk SIP com operadoras VoIP, por favor, consulte o item 11 da FAQ em:
http://www.digivoice.com.br/meucciBE/faq