

DOULOX LINUX LIVE: UMA DISTRIBUIÇÃO LINUX PARA AMBIENTES VOIP SIP E H.323

Trabalho de Conclusão de Curso

Engenharia da Computação

Rodrigo Fagner Brayner de Brito

Orientador: Prof. Doutor Adriano Lorena Inácio de Oliveira

Co-Orientadora: Prof. Mestra Zuleika Tenório Cavalcanti do Nascimento

Recife, 18 de maio de 2005

DOULOX LINUX LIVE: UMA DISTRIBUIÇÃO LINUX PARA AMBIENTES VOIP SIP E H.323

Trabalho de Conclusão de Curso

Engenharia da Computação

Este Projeto é apresentado como requisito parcial para obtenção do diploma de Bacharel em Engenharia da Computação pela Escola Politécnica de Pernambuco – Universidade de Pernambuco.

Rodrigo Fagner Brayner de Brito

Orientador: Prof. Dr. Adriano Lorena Inácio de Oliveira

Co-Orientadora: Prof. Mestra Zuleika Tenório Cavalcanti do Nascimento

Recife, 18 de maio de 2005

Rodrigo Fagner Brayner de Brito

**DOULOX LINUX LIVE: UMA
DISTRIBUIÇÃO LINUX PARA
AMBIENTES VOIP SIP E H.323**

Resumo

A tecnologia de telefonia *IP*, ou *VoIP* (voz sobre *IP*), vem aos poucos sendo difundida em redes acadêmicas e em empresas pelo baixo custo de ligações, sendo assim um paradigma alternativo à redes telefônicas convencionais. Este trabalho apresenta uma introdução geral a esta tecnologia, bem como aos dois protocolos mais utilizados, sendo estes o *H.323* e o *SIP*. Além disto, propomos o desenvolvimento de uma distribuição *Linux*, o *Doulox Linux Live*, contendo os programas necessários para se ter um ambiente *VoIP* operacional utilizando tais protocolos. Propomos também a união destes protocolos através de um *gateway* e central telefônica em *software*, o *Asterisk*, podendo também haver a união entre os ambientes de telefonia *IP* e convencional. Este trabalho apresenta também um sistema de gerenciamento para a simplificação da configuração de todos os serviços necessários para se ter os serviços *VoIP* em funcionamento.

Abstract

The IP telephony technology, or VoIP (voice over IP), is being spread out in academic networks and companies because its low calling costs, being an alternative paradigm to conventional telephone networks. This work presents a general introduction to this technology, the two most used protocols, the H.323 and SIP, and proposes a Linux distribution, the Doulox Linux Live, which contains the necessary softwares to have an operational VoIP environment using these protocols and the union of both through a gateway and a telephone exchange made in software, the Asterisk. This enables the union between the IP telephony and conventional environments. This work also presents a management system for the simplification of all necessary services to have VoIP running.

Sumário

Índice de Figuras	4
Índice de Quadros	5
Índice de Quadros	5
1 Introdução	8
1.1 Motivações	8
1.2 Objetivo	9
1.3 A solução proposta	9
2 Voz sobre IP	10
2.1 H.323	13
2.2 SIP	15
3 Criação do <i>Doulox</i> e implantações dos serviços <i>VoIP</i>	18
3.1 Tentativas de criação da distribuição a partir do <i>Slax</i>	23
3.2 Experiências com o <i>Slackware</i> e <i>Kernel 2.4</i>	24
3.3 Experiências com o <i>Slackware</i> e <i>Kernel 2.6</i>	25
3.4 Criando definitivamente a distribuição	26
3.5 Modificações e testes utilizando o <i>Doulox</i>	27
3.6 Configuração do <i>SER</i>	35
3.7 Configuração do <i>GnuGK</i>	42
3.8 Configuração do <i>Asterisk</i>	44
3.8.1 Canal <i>SIP</i>	44
3.8.2 Canal <i>H.323</i>	46
3.9 Instalação do <i>Doulox</i>	49
4 Sistema de gerenciamento do <i>Doulox</i>	50
4.1 Tecnologia	50
4.2 Autenticação	50
4.3 Rede, <i>DNS</i> e prefixos locais	52
4.4 <i>GnuGK</i>	55
4.5 Opções <i>SER</i> , <i>Asterisk</i> e recarregar	59
5 Conclusões e trabalhos futuros	65
5.1 Conclusões	65
5.2 Discussões	65
5.3 Trabalhos futuros	66

Índice de Figuras

Figura 1.	Sistema de telefonia tradicional.	10
Figura 2.	Sistema de telefonia <i>IP</i> .	11
Figura 3.	Exemplo de funcionalidade de <i>VoIP</i> , onde um telefone (telefonia <i>IP</i>) de Salvador liga para um telefone (telefonia convencional) do Rio de Janeiro.	12
Figura 4.	Cenários em <i>VoIP</i> [12].	12
Figura 5.	Escopo e componentes definidos em <i>H.323</i> [15].	13
Figura 6.	Comunicação entre terminais <i>H.323</i> [15].	15
Figura 7.	Exemplo do funcionamento do protocolo <i>SIP</i> [15].	16
Figura 8.	Registro em um servidor <i>SIP</i> [15].	17
Figura 9.	<i>X-Lite</i> , telefone virtual para protocolo <i>SIP</i> .	19
Figura 10.	Cenário utilizado para o desenvolvimento e testes.	19
Figura 11.	Utilização do <i>VMWare</i> para criação de máquinas virtuais com sistemas operacionais Linux.	20
Figura 12.	Configuração do <i>X-Lite</i>	29
Figura 13.	Erro ao ser iniciado o <i>Asterisk</i>	30
Figura 14.	<i>Asterisk</i> iniciado corretamente	30
Figura 15.	Marcação da opção <i>Disable Fast-Start</i> no <i>OpenPhone</i>	33
Figura 16.	Tela de autenticação do usuário.	51
Figura 17.	Tela de boas vindas e menu.	51
Figura 18.	Diagrama de casos de uso das opções Rede, DNS e Prefixos.	52
Figura 19.	Tela de configuração da rede.	53
Figura 20.	<i>Submenus</i> da opção Rede.	53
Figura 21.	<i>Submenus</i> da opção “DNS”.	54
Figura 22.	Configuração do servidor <i>DNS</i> .	54
Figura 23.	<i>Submenus</i> da opção “Prefixos”.	55
Figura 24.	Configuração dos prefixos do servidor.	55
Figura 25.	Diagrama de casos de uso do <i>GnuGK</i> .	56
Figura 26.	<i>Submenus</i> da opção “ <i>GnuGK</i> => <i>GKs</i> ”.	56
Figura 27.	<i>Submenus</i> da opção “ <i>GnuGK</i> =>Prefixos”.	57
Figura 28.	Adicionando um novo <i>GK</i> .	57
Figura 29.	Associando um prefixo a um <i>GK</i> .	57
Figura 30.	Associando um prefixo a um <i>GK Asterisk</i> em <i>Voip1</i> .	58
Figura 31.	Associando um prefixo a um <i>GK Asterisk</i> em <i>VoIP2</i> .	58
Figura 32.	<i>Submenus</i> da opção “ <i>GnuGK</i> =>Usuários”.	59
Figura 33.	Adição de novo usuário/telefone ao servidor <i>GK</i> em <i>Voip1</i> .	59
Figura 34.	Diagrama de casos de uso do <i>SER</i> , <i>Asterisk</i> e recarregar.	60
Figura 35.	<i>Submenus</i> da opção “ <i>SER</i> =>Redirecionamentos”.	60
Figura 36.	Configuração do redirecionamento do <i>SER</i> de <i>VoIP1</i> para o <i>SER</i> do <i>VoIP2</i> .	61
Figura 37.	Configuração do redirecionamento do <i>SER</i> de <i>VoIP1</i> para o <i>SER</i> do <i>VoIP2</i> .	62
Figura 38.	<i>Submenus</i> da opção “ <i>SER</i> =>Telefones”.	62
Figura 39.	Adição de um usuário ao <i>SER</i> em <i>VoIP1</i> .	63
Figura 40.	<i>Submenus</i> da opção “ <i>Asterisk</i> =>Redirecionamentos”.	63
Figura 41.	Redirecionamento de ligações de <i>Voip1</i> , destinadas ao grupo <i>Voip2</i> , ao servidor <i>SER</i> local através do <i>Asterisk</i> .	64

Índice de Quadros

Quadro 1. Cabeçalho de uma requisição <i>INVITE</i> do protocolo <i>SIP</i> .	16
Quadro 2. Erro ao ser executado o <i>GnuGK</i> sem as bibliotecas do <i>OpenH323</i> .	21
Quadro 3. Arquivo <i>Makefile</i> do diretório fonte do <i>OpenH323</i> .	21
Quadro 4. Execução dos comandos presentes no arquivo <i>Makefile</i> do diretório fonte do <i>OpenH323</i> .	22
Quadro 5. Erro ao ser executado o <i>GnuGK</i> sem o <i>PWLib</i> .	22
Quadro 6. Iniciando o <i>GnuGK</i> através de linha de comando.	23
Quadro 7. Arquivo de configuração inicial (básico) do <i>GnuGK</i> .	23
Quadro 8. Erro obtido ao executar o <i>GnuGK</i> como imagem no <i>Slax</i> .	24
Quadro 9. Pacotes necessários para a instalação do <i>kernel 2.6.9</i> .	25
Quadro 10. Erro <i>Kernel Panic</i> .	25
Quadro 11. Erros obtidos ao executar a primeira versão do <i>Doulox</i> .	26
Quadro 12. Erro ao iniciar o primeiro teste com o <i>SER</i> utilizando o <i>X-Lite</i> .	28
Quadro 13. Linha de configuração do <i>SER</i> para identificação de <i>URI</i> através da identificação do próprio domínio.	28
Quadro 14. Linha de configuração do <i>SER</i> para identificação de <i>URI</i> por extenso.	28
Quadro 15. Erro ao criar o pacote do <i>PWLib</i> utilizando o <i>checkinstall</i> .	31
Quadro 16. Pacotes criados com o utilitário <i>checkinstall</i> através dos códigos fonte.	32
Quadro 17. Alteração do arquivo <i>inetd.conf</i> .	32
Quadro 18. Conteúdo do arquivo <i>/etc/rc.d/rc.server</i> .	32
Quadro 19. Erro ao iniciar o <i>MySQL</i> .	33
Quadro 20. <i>Log</i> do erro relativo ao início do <i>MySQL</i> .	34
Quadro 21. Linha descomentada no arquivo <i>/etc/apache/httpd.conf</i> .	34
Quadro 22. Linhas descomentadas no arquivo <i>/etc/apache/php.ini</i> .	34
Quadro 23. Linhas adicionadas ao final do arquivo <i>/etc/profile</i> .	35
Quadro 24. Senhas padrão do banco de dados <i>SER</i> .	35
Quadro 25. Erro ao iniciar o <i>SER</i> com suporte ao <i>MySQL</i> .	35
Quadro 26. Criação de um link simbólico para a biblioteca <i>libmysqlclient.so</i> .	36
Quadro 27. Erros relacionados à inicialização do <i>SER</i> com suporte ao <i>MySQL</i> , utilizando o <i>ser-mysql</i> .	36
Quadro 28. Adição de um <i>alias</i> no arquivo <i>ser.cfg</i> .	36
Quadro 29. Remoção de comentários no arquivo <i>ser.cfg</i> relacionados aos <i>loadmodules</i> .	37
Quadro 30. Alterações ao <i>ser.cfg</i> quanto ao parâmetro <i>usrloc</i> .	37
Quadro 31. Remoção de comentários dos parâmetros <i>auth_db</i> no arquivo <i>ser.cfg</i> .	37
Quadro 32. Remoção de comentários dos parâmetros <i>auth_db</i> no arquivo <i>ser.cfg</i> .	37
Quadro 33. Bloco de redirecionamento para o grupo <i>Voip2</i> .	38
Quadro 34. Bloco de redirecionamento para o canal <i>SIP</i> do <i>Asterisk</i> .	38
Quadro 35. Bloco de redirecionamento para clientes <i>PBX</i> .	39
Quadro 36. Formato de um registro <i>SRV</i> .	39
Quadro 37. Registro <i>SRV</i> no servidor <i>DNS</i> de <i>Voip1</i> .	39
Quadro 38. Arquivo de zona do servidor <i>DNS</i> de <i>Voip1</i> .	40
Quadro 39. Bloco adicionado ao arquivo <i>/etc/named.conf</i> .	40
Quadro 40. Utilizando o <i>dig</i> para verificar o registro <i>SRV</i> adicionado.	40
Quadro 41. Resposta ao comando <i>dig</i> no registro <i>SRV</i> do servidor.	41
Quadro 42. Sintaxe do comando <i>serctl</i> para adição de novos usuários.	41
Quadro 43. Sintaxe do comando <i>serctl</i> para adição de <i>aliases</i> .	41
Quadro 44. Exemplo do uso do <i>serctl</i> .	41
Quadro 45. Adição de um usuário com valores numéricos ao <i>SER</i> .	42
Quadro 46. Adição do usuário administrador ao <i>SER</i> .	42
Quadro 47. Erro ao compilar o utilitário <i>addpasswd</i> .	42
Quadro 48. Erro ao compilar o utilitário <i>addpasswd</i> .	43
Quadro 49. Configuração do <i>gnugk.ini</i> com relação à autenticação.	43
Quadro 50. Configuração do <i>gnugk.ini</i> com relação ao nome do <i>GK</i> .	43
Quadro 51. Configuração do <i>gnugk.ini</i> com relação ao vizinhos.	44
Quadro 52. Bloqueio de telefones celulares.	44

Quadro 53. Seção <i>[general]</i> do arquivo <i>sip.conf</i> .	45
Quadro 54. Seção de autenticação do canal <i>SIP</i> em <i>sip.conf</i> .	45
Quadro 55. Seção de identificação do servidor <i>SER</i> local.	45
Quadro 56. Definição do grupo e das regras de redirecionamento em <i>extensions.conf</i> .	46
Quadro 57. Aplicação de um <i>patch</i> ao fonte do <i>OpenH323</i> .	47
Quadro 58. Erros obtidos na compilação do <i>Asterisk-oh323</i> .	47
Quadro 59. Programas compilados para o funcionamento do <i>Asterisk-oh323</i> .	47
Quadro 60. Configurações no arquivo <i>oh323.conf</i> .	48
Quadro 61. Configurações de redirecionamento no arquivo <i>extensions.conf</i> quanto ao canal <i>H.323</i> .	48

Agradecimentos

Primeiramente agradeço ao meu Deus, por me amar tanto e ter me capacitado para fazer tal trabalho, sem ele eu não estaria capacitado a escrever qualquer linha deste projeto. Por ter derramado tantas bênçãos em minha vida e por estar me ensinando a ser um verdadeiro servo, sendo o nome da distribuição deste projeto em homenagem a Ele, onde Doulos significa servo em grego. Todo este projeto foi feito em especial para Ele.

Agradeço também à minha família, em especial minha mãe (Márcia Gomes Brayner), pai (José Ricardo Figueiredo de Brito), irmão (Erasmus Eustáquio Brayner Neto) e avós (Erasmus Eustáquio Brayner e Shirley Gomes Brayner), que deu bastante força, me educando para a vida, sempre me colocando nos melhores colégios, preocupados com minha educação e me proporcionando amor, o mais importante de todos.

À minha namorada e futura esposa, Fernanda Lins da Fonseca, que sempre acreditou em mim, me deu forças, carinho e amor quando precisei em momentos difíceis do desenvolvimento deste projeto, acalmando meu coração e retirando minhas inseguranças, me entendendo quando precisei ficar dias ou semanas sem vê-la e por ser a pessoa mais preciosa e especial em minha vida aqui na terra.

Ao meu líder espiritual, Marinho, a quem amo muito e sempre acreditou em mim, e que me reforçou, através do seu exemplo, de que tudo que fizermos que façamos para a glória e honra do Senhor, portanto, como disse acima, este trabalho vai em homenagem ao meu Deus e à Marinho, por ser exemplo de um cristão e líder também do ministério Doulos, ministério esse que forma jovens servos cristãos para o mundo.

Agradeço aos meus amigos da universidade, que sempre foram meus defensores em todos os problemas que ocorreram comigo na POLI, passando até a ser o mascote temporário deles 😊. Sempre me ajudavam também quando precisei, e foram realmente amigos nas horas difíceis. Em especial agradeço às pessoas na POLI que passei mais tempo, como Fernando Antônio, Carolina Cavalcanti, Allan Bruno, Bruna Bunzen, Adélia Barros, Rodrigo Cursino, Rodrigo Gomes, Túlio Campos, Cláudio Cavalcanti, Zé Guiga, Livia Brito, Juliana Lima, Edivadno Vasconcelos, Moka, entre outros, a lista é grande. Sou grato por ter os professores que tive, tal como Ricardo Massa e Carlos Alexandre, por serem meus primeiros professores e grandes mestres, dando a mim exemplos do que é ser um verdadeiro educador, mesmo tendo alguns desentendimentos durante o curso 😊. Agradeço ao meu orientador, por ter me guiado durante o desenvolvimento deste projeto e ter acreditado no meu potencial, aceitando o papel de ser meu orientador. Agradeço ao pessoal do GT-VoIP da UFRJ, em especial Peixoto, por ter me iniciado na tecnologia VoIP e respondido aos meus e-mails com dúvidas, aos usuários do fórum do Slax pelo suporte em suas páginas e à minha co-orientadora e chefe do PoP-PE Zuleika, que me ajudou bastante quando precisei faltar e quando me apoiou no projeto, acreditando em mim, me dando forças e me incentivando a fazer o mestrado nesta área. Por fim, agradeço à RNP por ter me disponibilizado recursos para a finalização deste projeto.

Capítulo 1

Introdução

Este capítulo contém uma introdução ao projeto proposto, destacando as motivações para o desenvolvimento deste, o objetivo principal a ser alcançado e as soluções propostas para atingir o objetivo.

1.1 Motivações

Atualmente a tecnologia de voz sobre *IP*, ou *VoIP* [1], vem se destacando em meio às outras tecnologias de informação, mais especificamente nas áreas de telefonia, onde a busca crescente por menores custos nesta área vem sendo uma das maiores motivações no estudo e crescimento desta.

VoIP, ou telefonia *IP*, é um sistema de telefonia alternativo, onde a Internet vem a ser o meio de comunicação utilizado, diminuindo bastante o custo da comunicação, uma vez que a Internet é um meio bastante barato e difundido em todo o mundo. Portanto a idéia de *VoIP* é utilizar o transporte de pacotes, realizado pelos protocolos *TCP/IP* [2], para transportar voz.

Dois protocolos abertos (gratuitos) assumem a hegemonia na área de *VoIP*, sendo estes o *H.323* [3] e o *SIP* [4]. O primeiro vem a ser um protocolo mais antigo, mais maduro e bastante utilizado, sendo o maior motivo disto o tempo de sua existência. O *SIP* é um protocolo mais atual, sendo seu projeto mais voltado à Internet, uma vez que o *H.323* a partir da telefonia convencional, porém com o foco em telefonia *IP* e a comunicação com redes de telefonia convencionais. Através de um programa chamado *Asterisk*, uma central telefônica e *gateway* desenvolvido em *software*, é possível se ter os dois protocolos funcionando em uma mesma rede, fazendo a ligação entre tais protocolos e os protocolos de telefonia convencional, sendo esta uma grande motivação para utilização desta tecnologia, pois ligações poderiam ser realizadas sem praticamente nenhum custo ou custo de ligações locais. Este trabalho poderá ser um ponto inicial para abertura de empresas em *VoIP* no Brasil, uma vez que existem poucas empresas na área atualmente.

1.2 Objetivo

O objetivo deste trabalho é desenvolver uma distribuição *Linux* que conterà todos os programas necessários para a utilização dos serviços *VoIP* com protocolos *SIP*, *H.323* e de telefonia convencional. Um sistema também será desenvolvido para dar suporte à configuração de ambos os programas (servidores), uma vez que tal configuração não é trivial, passando a ser uma configuração demorada e trabalhosa. Com tal sistema, a configuração de todos os servidores será simplificada, sendo possível assim a implantação de uma rede de voz sobre *IP* em tempo e complexidade reduzidas.

Um grupo de pesquisa da RNP / UFRJ, chamado GT-VoIP, desenvolveu uma distribuição *Linux*, o *Voipix*, capaz de gerenciar uma rede de telefonia *IP* utilizando o protocolo *H.323*. O objetivo proposto neste trabalho vem abordar tanto o protocolo *H.323* quanto o *SIP* e a comunicação entre os dois.

A distribuição, que será gratuita, constará dos servidores *SER*, *GnuGK* e *Asterisk*, sendo estes necessários para prover a comunicação entre usuários *SIP*, *H.323* e telefonia convencional. Embora neste projeto não fossem realizados testes com telefonia convencional, toda a sua estrutura foi pensada para incluir tal funcionalidade em trabalhos futuros. O sistema de gerenciamento constará de várias funcionalidades incluindo o gerenciamento do programa *SER*, *GnuGK*, *Asterisk*. Será possível, ainda, configurar os servidores *DNS* e fazer configurações de rede.

No Capítulo 2 são apresentados uma introdução aos conceitos em *VoIP*, aos protocolos *SIP* e *H.323* e algumas informações quanto ao projeto a ser realizado.

No Capítulo 3 são mostrados todos os processos para o desenvolvimento da distribuição, apresentando os problemas encontrados e soluções para tais problemas, bem como a configuração e testes com os servidores sem o sistema de gerenciamento. Isto para que pudesse haver uma idéia dos pontos mais complexos da configuração e necessidades em todos os processos e assim desenvolver funcionalidades para a automatização destes.

No Capítulo 4 são abordadas as funcionalidades do sistema de gerenciamento, explicando cada uma para um maior entendimento e para tornar possível a implantação dos serviços *VoIP* sem complicações.

No Capítulo 5, são discutidas as conclusões do projeto e apresentados alguns trabalhos futuros a serem realizados tendo como base este projeto.

1.3 A solução proposta

A solução proposta para os objetivos apresentados na seção anterior serem alcançados será desenvolver o *Doulox Linux Live*, uma distribuição *Linux* que pode ser executada diretamente do CD, sem haver a necessidade de instalação, podendo assim prover uma simples demonstração em apresentações, e sendo capaz de ser instalada também no computador para se obter um servidor real. Esta distribuição conterà todos os servidores necessários, como explicado anteriormente. Além disso, ela será capaz de prover um serviço *VoIP* entre várias instituições, desde que a estrutura de rede esteja implantada e a distribuição esteja disponível.

Para a configuração de todos os servidores, um sistema de gerenciamento será desenvolvido para facilitar a implantação com êxito da comunicação entre as várias instituições interligadas via rede, alcançando assim o objetivo principal do projeto, prover um ambiente *VoIP* com facilidade e incluindo os dois protocolos mais utilizados atualmente em voz sobre *IP*.

Capítulo 2

Voz sobre IP

Voz tem sido transmitida através das tecnologias de telefonia convencional *PSTN* (*Public Switched Telephone Network*) [5] desde 1878. O desejo das empresas e consumidores de redução dos custos com telefonia convencional, junto com o investimento em tecnologias baseadas em *TCP/IP*, tem produzido um interesse na transmissão de voz utilizando o sistema de voz sobre IP (*VoIP* ou *Voice over IP*).

VoIP envolve a transmissão da voz em forma de pacotes de dados utilizando o protocolo *IP*, onde a voz do usuário é convertida em um sinal digital, comprimida, e quebrada em uma série de pacotes. Os pacotes são então transmitidos através de redes públicas ou privadas, e são agrupados e decodificados no lado do receptor. Desde a década de 70, pesquisadores vêm realizando testes em redes de pacotes. Esta tecnologia vem passar a ter um campo comercial a partir da metade da década de 90.

Portanto, a tecnologia *VoIP* surge como um novo paradigma em relação a sistemas de telefonia convencionais como ilustrado na Figura 1. Neste exemplo temos dois assinantes de operadoras não necessariamente iguais, conversando através de uma rede de telefonia tradicional.

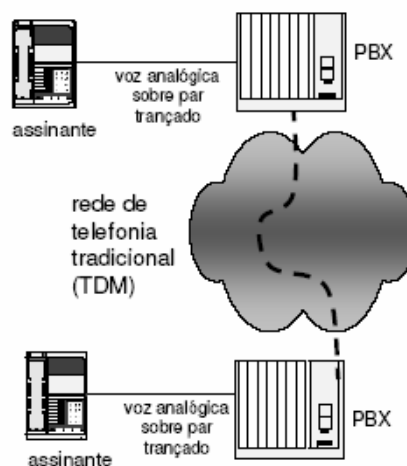


Figura 1. Sistema de telefonia tradicional.

(figura reproduzida do material do curso de *VoIP* realizado pela RNP no 10º seminário RNP de capacitação realizada no PoP-PE)

O novo paradigma introduzido está exemplificado na Figura 2. Neste exemplo, a mesma rede de telefonia ainda é utilizada por questões de flexibilidade, porém novos conceitos são introduzidos em *VoIP*, como por exemplo o uso de redes locais como meio de comunicação de voz. Neste sistema é possível haver a comunicação entre ambas as tecnologias, *VoIP* e os sistemas de telefonia tradicionais.

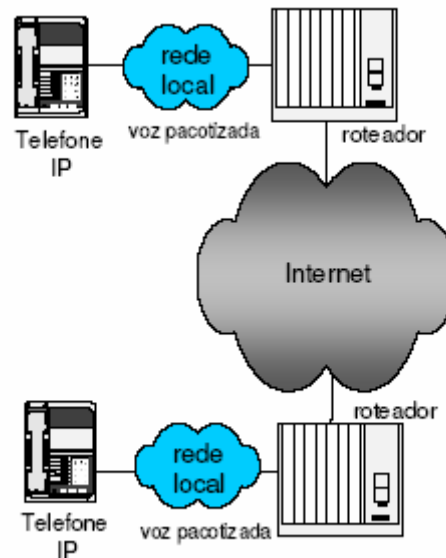


Figura 2. Sistema de telefonia *IP*.

(figura retirada do material do curso de *VoIP* realizada pela RNP no 10º seminário RNP de capacitação realizada no PoP-PE)

No exemplo acima, temos telefones *IP* (sendo estes telefones específicos para tecnologias *VoIP*) ligando para um outro telefone *IP* utilizando a Internet como meio de transporte de voz.

Dois protocolos de comunicação (sinalização) assumem a hegemonia nesta nova tecnologia, o *H.323* proposto pela *ITU-T* (*International Telecommunication Union Telecommunications Standardization Sector*) [6] e o mais recente protocolo, o *SIP* (*Session Initiation Protocol*), proposto pela *IETF* (*The Internet Engineering Task Force*) [7]. Com ambos os protocolos, seria possível criar um sistema de telefonia como exemplificado na Figura 3, onde se teria um telefone convencional no Rio de Janeiro ligando para um telefone analógico ligado a uma rede *IP* em Salvador com custos reduzidos a pulsos locais. Isto poderia também ser realizado a nível mundial. Neste exemplo, pode-se observar a conexão entre as duas redes. Os *PBXs* (*Private Branch Exchange*) [8] são as centrais telefônicas das operadoras, o *gateway* de voz é responsável por fazer a ligação entre as redes *PSTN* e *IP*, os roteadores encaminham os pacotes *IP* contendo tráfego de voz e o adaptador seria um equipamento adaptador (*ATA*) para fazer ligações de telefones analógicos à rede *IP*. É possível também realizar chamadas entre ambos os protocolos, *SIP* e *H.323*, sendo este o objetivo principal ao longo deste trabalho.

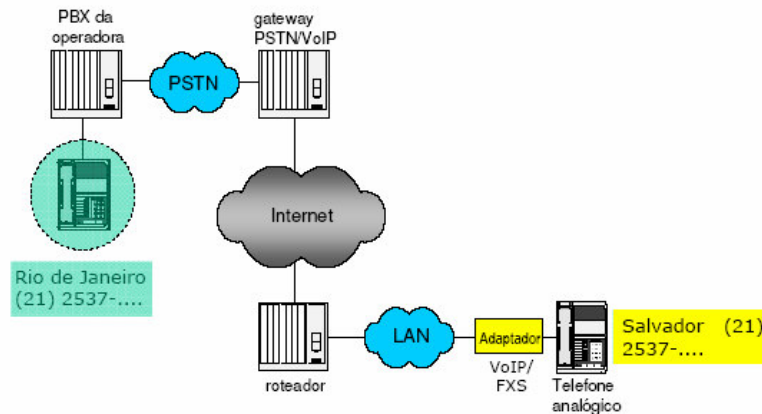


Figura 3. Exemplo de funcionalidade de *VoIP*, onde um telefone (telefonia *IP*) de Salvador liga para um telefone (telefonia convencional) do Rio de Janeiro.
(figura retirada do material do curso de *VoIP* realizado pela RNP no 10º seminário RNP de capacitação realizada no PoP-PE)

Algumas ferramentas são necessárias para a implantação de tal ambiente em ambos os protocolos de sinalização. Para ambientes *H.323*, é necessário o uso de um servidor gerenciador de chamadas. Este servidor pode ser chamado de *gatekeeper* ou simplesmente *GK*. Utilizamos uma solução gratuita, o programa *GnuGK* [9]. Para ambientes *SIP*, é necessário um *proxy*, ou redirecionador de chamadas, utilizamos o programa *SER* (*SIP Express Router*) [10] para tal funcionalidade.

O *Asterisk* [11] é utilizado como um *gateway* de voz e *PBX* desenvolvido em *software*, minimizando o custo de \$11 mil dólares (equipamentos *gateway* em *hardware*) para apenas \$350 dólares, custo de uma placa com suporte a duas interfaces *FXS* (*Foreign Exchange Subscriber*) e duas *FXO* (*Foreign Exchange Office*), podendo ser utilizado com o *Asterisk* para emular um *gateway* em *software*. Uma placa com suporte ao tronco digital *E1* custa em torno de \$ 2.000 dólares. Portas *FXS* são ligadas a telefones analógicos enquanto que as *FXO* e *E1* são ligadas aos *PBXs* com suporte a troncos digitais (*E1*)(30 ligações simultâneas) ou troncos analógicos (*FXO*). O *Asterisk* funcionará também como *gateway* entre clientes *H.323* e *SIP*.

Resumindo, podemos obter um cenário como exemplificado na Figura 4. Neste exemplo, pode-se observar que as ligações podem ser realizadas entre telefones virtuais (programas telefones ou *softphones*), utilizando a Internet entre telefones virtuais e telefones convencionais e entre telefones convencionais utilizando a Internet ao invés do sistema de telefonia convencional.

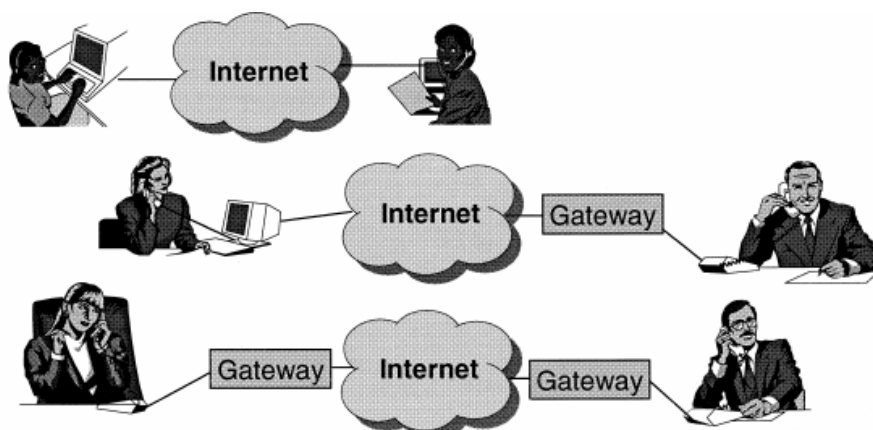


Figura 4. Cenários em *VoIP*[12].

A diferença fundamental entre os protocolos está nas suas origens. O *H.323* vem de um mundo *PBX*, sendo este orientado à tecnologia *ISDN* (*Integrated Services Digital Network*) [13], enquanto que o *SIP* surgiu de um mundo orientado a pacotes, o mundo *IP*. Chamadas em *SIP* podem ser realizadas através de *URLs* (*Uniform Resource Locator*) ou *URIs* (*Uniform Resource Identifier*), onde se pode realizar tais chamadas através do número ou e-mail de uma pessoa. A maioria das aplicações na área de *VoIP* atualmente estão focadas em *H.323* por sua maior maturidade, uma vez que este possui mais tempo no mercado do que o *SIP*, sendo isto uma das vantagens do *H.323*. Porém, muitas aplicações começam a surgir no mundo *SIP* e este passa a ser aos poucos um dos padrões mais utilizados em concorrência com o *H.323*.

2.1 H.323

A série *H.323* envolve várias recomendações propostas pela *ITU-T* em seus trabalhos com vídeo telefonia e conferência multimídia. Na série *H.320*, a *ITU-T* focou em vídeo conferência em *ISDN* acima de 2 Mbit/s enquanto que em *ATM* (*Asynchronous Transfer Mode*) [14] surgem as séries *H.310* e *H.321*. Para a tecnologia *PSTN* também surge uma nova série de recomendações, o *H.324*. Porém a recomendação mais adotada foi o *H.323* em 1995 para redes locais, focando assim em telefonia *IP*. O principal motivo foi a capacidade de interligação entre a rede de telefonia convencional e redes de computadores.

Um escopo geral e alguns componentes vistos em *H.323* estão especificados na Figura 5. Como mencionado acima, esta recomendação possibilita interligar os ambientes de telefonia convencional (*ISDN*, *PSTN*, *GSM*) e redes baseadas em *IP*.

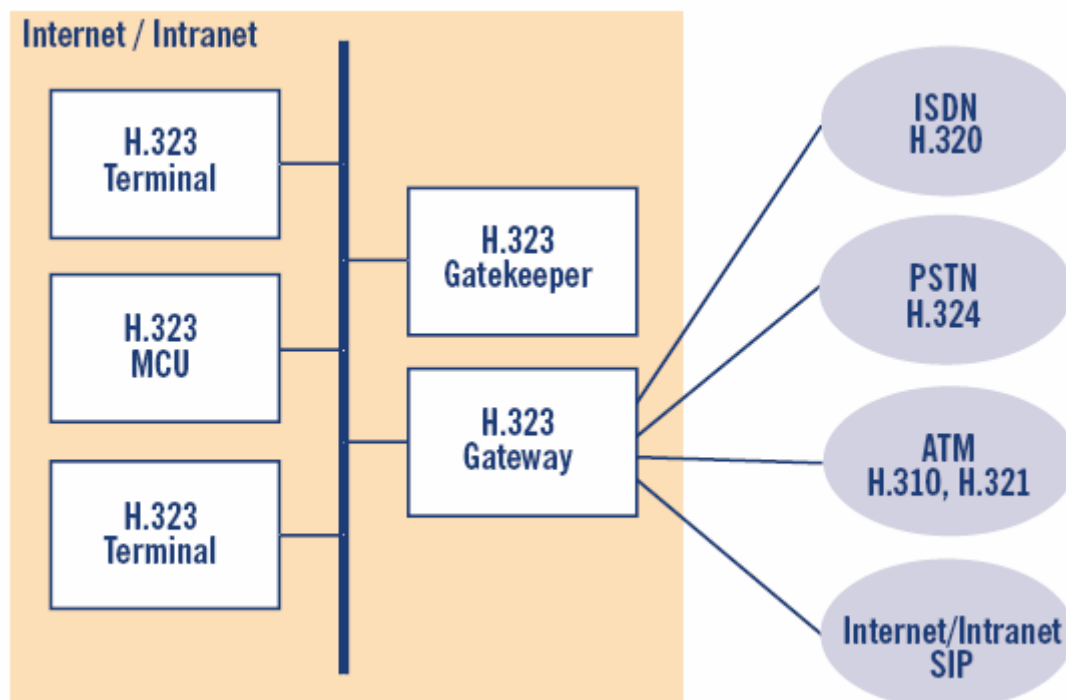


Figura 5. Escopo e componentes definidos em *H.323*[15].

Abaixo exemplificaremos alguns desses componentes:

- **Terminal:** São telefones virtuais utilizando computadores (*PCs*), mais especificamente aplicativos como, por exemplo, o *OpenPhone* [16] ou telefones *IP*, que são aparelhos específicos para *VoIP*. Pode-se também utilizar telefones analógicos utilizando um equipamento adaptador (ou *ATA*). A estes são designados um nome de usuário e números de telefone, a partir da montagem de um plano de discagem.
- **Gateway:** Os *gateways* interconectam os componentes da rede interna ao mundo exterior que utilizam outras tecnologias, tais como *GSM*, *ISDN*, *PSTN*, *ATM* e *SIP*. Utilizaremos o *Asterisk* para tal propósito.
- **Gatekeeper:** É a entidade núcleo em um ambiente *H.323*. É o responsável pelo controle de acesso, resolução de endereços, gerenciamento da rede *H.323* e policiamento de acesso.
- **MultiPoint Control Unit (MCU):** Faz possível a comunicação multiponto entre terminais *H.323*.

O *H.323* utiliza basicamente serviços como *Multicast* [17] e protocolos básicos de redes, tais como *IP*, *TCP* e *UDP (User Datagram Protocol)* [18], e faz uso de alguns outros protocolos diferenciados para sinalização de chamadas e controle. Tais protocolos são:

- **H.225.0 Registro, Admissão e Status (RAS):** O canal *RAS* é utilizado em comunicações entre *gatekeepers* ou entre terminais e *gatekeepers*. Os terminais utilizam o *RAS* para fazer autenticação nos *gatekeepers*, obter o número de um outro terminal etc., enquanto que os *gatekeepers* se comunicam entre si para manter *status* durante a comunicação entre os terminais e coletar informações como, por exemplo, a utilização do recurso após a finalização da ligação. Portanto, o *RAS* provê serviços para autenticação e autorização de chamadas.
- **H.225.0 Sinalização de chamada:** É utilizado para sinalizações entre componentes, podendo ser enviado sinais de *SETUP* que serão explicados mais adiante, sinais de sucesso, falhas, e etc. Pode também carregar operações adicionais, tais como a capacidade de realizar múltiplas chamadas e mantê-las.
- **H.245 Controle de conferência:** Pode ser utilizado, por exemplo, para o estabelecimento e controle das chamadas entre dois terminais. Como funcionalidade, ele pode determinar os modos para troca de mídia (codificação de mídia) e configurar *streams* de mídia.

Para que possamos entender melhor o funcionamento do protocolo *H.323*, na Figura 6 apresentaremos um exemplo simples de comunicação entre clientes *H.323*. Neste exemplo, podemos ver a interoperabilidade que o *H.323* suporta, envolvendo telefones analógicos ligados à rede de telefonia convencional e clientes *H.323*, possibilitando assim a heterogeneidade. As linhas em verde tratam-se dos canais *RAS*, as vermelhas do canal de controle de mensagens e o lilás do canal de sinalização de chamadas, onde todos foram descritos acima.

Na ilustração, temos que o telefone da esquerda (A) deseja fazer uma ligação para o telefone da direita (B). A primeira mensagem é a de *ARQ (Admission Request)* que contém a estimativa da banda a ser utilizada, o modelo desejado de chamada (se direto ou roteado pelo *GK*), informação do destino, identificador da ligação, dentre outras informações. A resposta que o *gatekeeper* retorna para A é um *ACF (Admission Confirm)* que informa a banda a ser utilizada, o endereço de transporte e porta para a sinalização de chamadas (*H.225*). Após a autenticação do telefone A no *gatekeeper*, este envia um sinal *SETUP* contendo informações para a conexão ao

telefone B. Este faz requisições ao *gatekeeper* com a finalidade de autenticação e envia um sinal de *CONNECT* de volta para o telefone A, onde faz terminar o canal *H.225* para sinalização de chamadas e inicia o protocolo *H.245* para controle de conferência que é ilustrado como a mensagem 7 na figura. Antes de enviar este sinal de *CONNECT*, este envia duas mensagens: o *ALERTING* e o *CALL PROCEEDING*. O primeiro indica ao telefone A que o telefone B está chamando (*ringing*) enquanto que o segundo avisa que o estabelecimento da chamada foi iniciado e que nenhum outro estabelecimento de chamada será permitido.

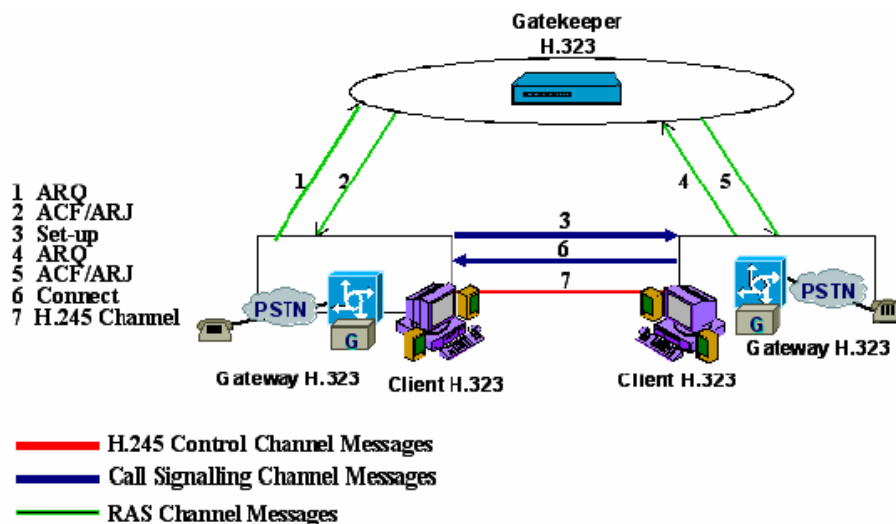


Figura 6. Comunicação entre terminais *H.323*[15].

2.2 SIP

O protocolo *SIP* (*Session Initiation Protocol*) [4] surge como um protocolo para comunicação que possibilita aos usuários se comunicarem através de vários tipos de mídia, fazendo automaticamente o tratamento de sessões adequado para o tipo de mídia a ser utilizado e possibilitando a localização dos usuários de um modo simples, utilizando, por exemplo, servidores *DNS* (*Domain Name Server*) [19] para tal aplicação. O protocolo *SIP* é bastante ágil, uma ferramenta para muitas aplicações, modificando e terminando sessões que trabalham independentemente do tipo de sessão que está sendo utilizada.

Na Figura 7 está ilustrado um exemplo do funcionamento do *SIP*. Pode-se observar a tentativa de localização de um telefone, sinalizando um pedido para se comunicar, negociando os parâmetros de sessões para que seja possível o estabelecimento da mesma e a finalização da sessão quando um dos telefones é desligado. Os servidores de *proxy* têm como funcionalidade a facilitação do estabelecimento da sessão entre ambos os usuários Alice e Bob como visualizado na figura. Neste exemplo Alice quer fazer uma ligação para Bob, que está conectado à Internet. Por sua vez, Alice está utilizando um telefone virtual (um aplicativo *SIP* no *PC* como, por exemplo, o *X-Lite* [20]) enquanto Bob utiliza um telefone *IP* com suporte a *SIP* (normalmente com uma atualização do *firmware* do telefone, pois tais telefones vêm com suporte aos protocolos do fabricante). Alice utiliza um *SIP URI* (*Uniform Resource Identifier*) para localizar Bob. Neste caso, uma *URI* para Bob seria *SIP:bob@biloxi.com*, onde *biloxi.com* é o domínio do servidor *proxy* de Bob. Alice também tem um *SIP URI* *SIP:alice@atlanta.com*.

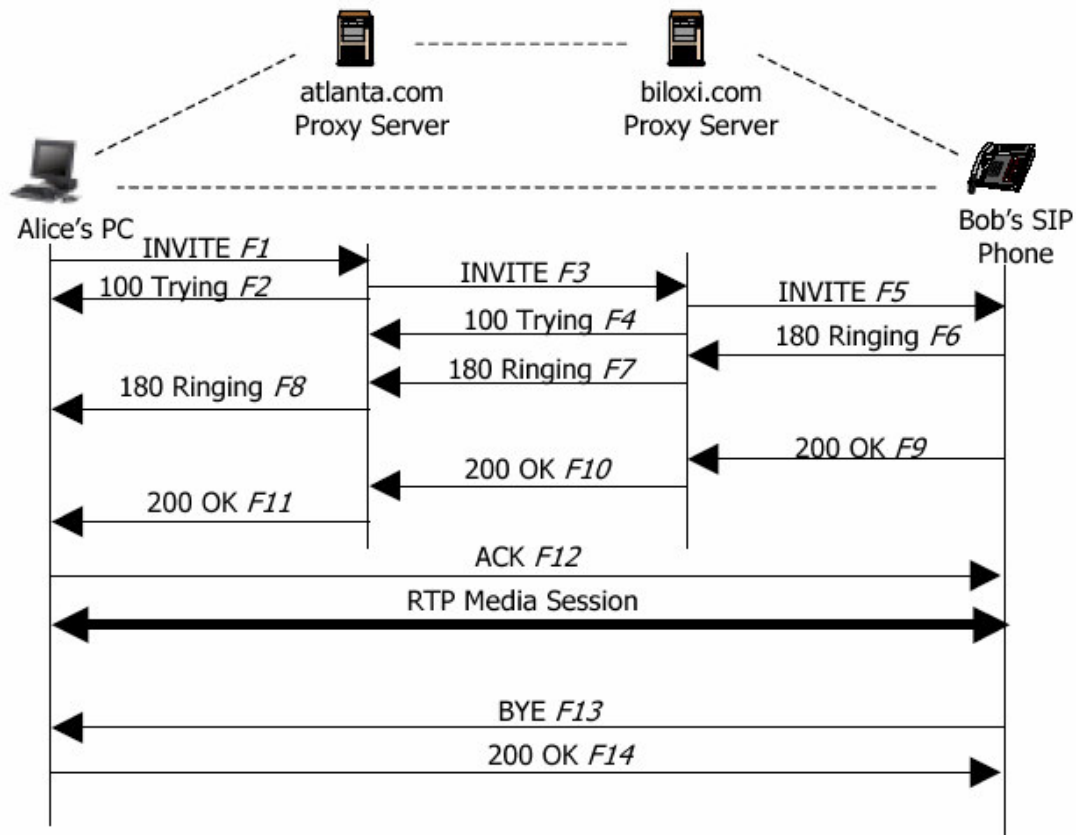


Figura 7. Exemplo do funcionamento do protocolo SIP[15].

Como primeiro sinal, Alice envia um *INVITE* para Bob através do seu servidor *proxy* que se encarrega de procurar pela *URI SIP:bob@biloxi.com*. Esta sinalização é passada adiante até que o ponto final, ou seja, Bob, seja atingido. Como exemplo de uma requisição SIP, pode-se observar no Quadro 1 um cabeçalho de uma requisição *INVITE*.

```

INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP 10.1.3.3:5060
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@10.1.3.3
CSeq: 314159 INVITE
Contact: <sip:alice@10.1.3.3>
Content-Type: application/sdp
Contact-Length: 142
  
```

Quadro 1. Cabeçalho de uma requisição *INVITE* do protocolo SIP.

Neste cabeçalho, temos o campo *INVITE*, que indica o destino a ser chamado, o campo *Via* indica o tipo de transporte a ser utilizado, o *UDP*, o endereço *IP* e a porta 5060 de Alice. O campo *To* indica a *URI* de Bob enquanto que o campo *From* especifica a *URI* de Alice. Pode-se

notar que ambos os campos *To* e *From* possuem um nome de visualização, sendo estes Bob e Alice. O Campo *Call-ID* identifica um identificador global da ligação, muitas vezes chamado de diálogo. *CSeq* especifica um número de seqüência que será incrementado a cada sinalização e *Contact* indica uma *URI* que contém um roteamento direto para identificações futuras. *Content-Type* contém uma especificação do corpo da mensagem e *Contact-Lenght* o tamanho do corpo da mensagem em *bytes*.

Na Figura 7, ao receber um *INVITE*, o servidor *proxy atlanta* informa a Alice que irá tentar identificar Bob. Atlanta envia um *INVITE* para o servidor *biloxi* que também envia um sinal de *Trying*. Em seguida *biloxi* envia um *INVITE* para Bob que responde como *Ringin*, ou seja, está chamando. Este sinal é passado até Alice. Enquanto isso, um sinal de *OK* é enviado para Alice que atende ao telefone e envia um sinal de *ACK* para Bob indicando que atendeu. Em seguida uma comunicação é realizada utilizando o protocolo *RTP (Real-Time Transport Protocol)* [21]. Em seguida Bob desliga o telefone enviando um *BYE* para Alice que retorna um *OK* finalizando assim a conversa. Um procedimento registro precisa ser executado para que estes procedimentos acima possam ocorrer com sucesso. Este procedimento encontra-se na Figura 8. O usuário *jan*, através do *IP* 1.2.3.4 e porta 5060 deseja se registrar no servidor *SIP* ou *proxy*, onde utilizou-se o programa *SER* para tal. O servidor registra sua localização no banco de dados [22] local, contendo o *IP* atual e porta do cliente associado ao usuário. Todas as vezes em que o cliente é registrado no servidor, os dados são atualizados. Em seguida o servidor retorna uma mensagem de *OK* para o cliente.

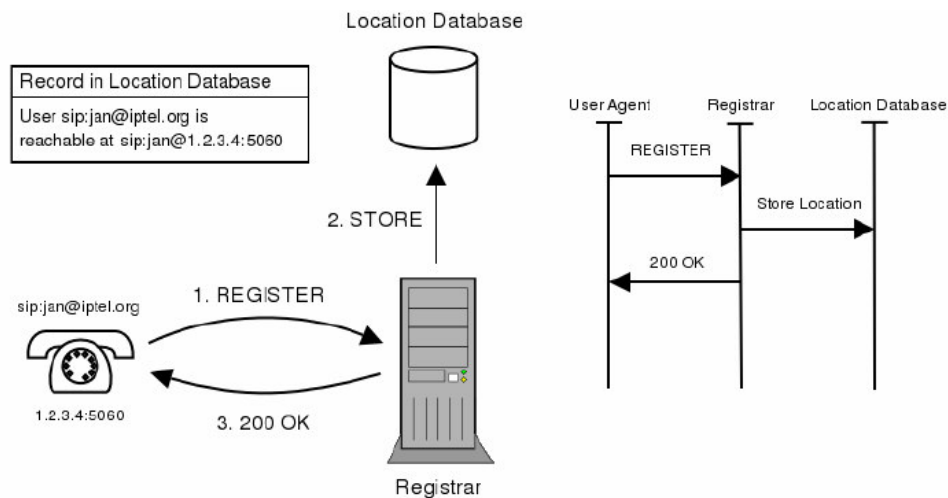


Figura 8. Registro em um servidor *SIP* [15].

Capítulo 3

Criação do *Doulox* e implantações dos serviços *VoIP*

Este projeto tem como finalidade prover a comunicação entre os clientes do protocolo *SIP* e *H.323*, utilizando algumas soluções disponíveis atualmente com programas de código fonte livre. A princípio, foi desenvolvida uma distribuição *Linux* [23], o *Doulox Linux Live*, contendo tais *softwares* pré-instalados e pré-configurados para tornar possível a fácil implantação dos serviços *VoIP* em empresas e redes acadêmicas, sendo estes o principal público alvo desta distribuição. Neste capítulo abordaremos como foi desenvolvida a distribuição *Doulox*, a configuração dos servidores necessários para tornar possível a comunicação entre os protocolos *H.323* e *SIP* e, por fim, a instalação do *Doulox*.

Os servidores a seguir foram compilados e adicionados à distribuição *Doulox*: *SER*, *GnuGK* e *Asterisk*. O *SER* funciona como um servidor *proxy* para clientes *SIP* e o *GnuGK* como *gatekeeper* para clientes *H.323*. O *Asterisk* faz a união de ambos os mundos, servindo então como *gateway*, não só para tais protocolos, mas também para a telefonia convencional, funcionando assim também como uma central telefônica desenvolvida em *software*. Em seguida foi desenvolvido um sistema utilizando as tecnologias *PHP* [24], *Shell Script* [25] e *MySQL* [26], entre outras, mais bem especificadas no Capítulo 4, para o gerenciamento destes servidores.

Utilizamos vários equipamentos, incluindo um *switch* [27] (recomendável utilizar *switches* para uma melhor qualidade de voz), telefones *IPs*, telefones virtuais (*softphones*) e computadores, sendo tais computadores com sistemas operacionais instalados em máquinas virtuais ou reais. Com o *switch*, foi possível agrupar todos os outros componentes em uma rede. Utilizamos a rede do *backbone* PoP-PE (Ponto de Presença de Pernambuco) da RNP (Rede Nacional de Ensino e Pesquisa). Dois telefones Cisco 7905G foram utilizados com atualizações de *firmware* para o suporte a *SIP* ou *H.323*. Os telefones virtuais são programas que instalamos no sistema operacional a fim de fazer ligações a partir deste. Na Figura 9 podemos observar o *X-Lite*, um telefone virtual para protocolo *SIP*. Os computadores foram necessários para instalar estes telefones virtuais (telefones em programa instalado no computador) e também para servirem de servidor dos serviços *VoIP*. Utilizou-se também computadores virtuais (máquinas virtuais), utilizando o *VMWare* [28] como programa suporte.



Figura 9. X-Lite, telefone virtual para protocolo SIP.

O cenário utilizado para o desenvolvimento e testes encontra-se na Figura 10. Um *switch* foi utilizado para fazer a comunicação entre todos os componentes apresentados na figura. Dois grupos distintos foram definidos, um chamado *Voip1* e o outro *Voip2*. No grupo *Voip1*, tendo este o domínio DNS *voip1.pop-pe.rnp.br*, tem-se um telefone virtual SIP, um telefone IP H.323 e o servidor principal destes telefones, o servidor *Voip1*, onde constam os servidores *SER*, *GnuGK* e *Asterisk*, *Apache*, *BIND*, *MySQL* e *TFTP*, necessários para o funcionamento do sistema, utilizando a distribuição *Linux* desenvolvida, o *Doulox*. O outro grupo segue o mesmo raciocínio, a diferença é que este grupo apresenta um telefone virtual H.323, um telefone IP SIP e um servidor com o domínio *voip2.pop-pe.rnp.br*. Para tais domínios funcionarem, um servidor DNS, o servidor com nome “rosa” e domínio *pop-pe.rnp.br*, foi necessário, servindo como servidor DNS com um nível de hierarquia maior do que os servidores *Voip1* e *Voip2*. Com todos estes componentes, foi possível desenvolver a distribuição, o sistema de gerenciamento do *Doulox* (Capítulo 4), e colocar em prática os serviços, realizando testes entre os telefones.

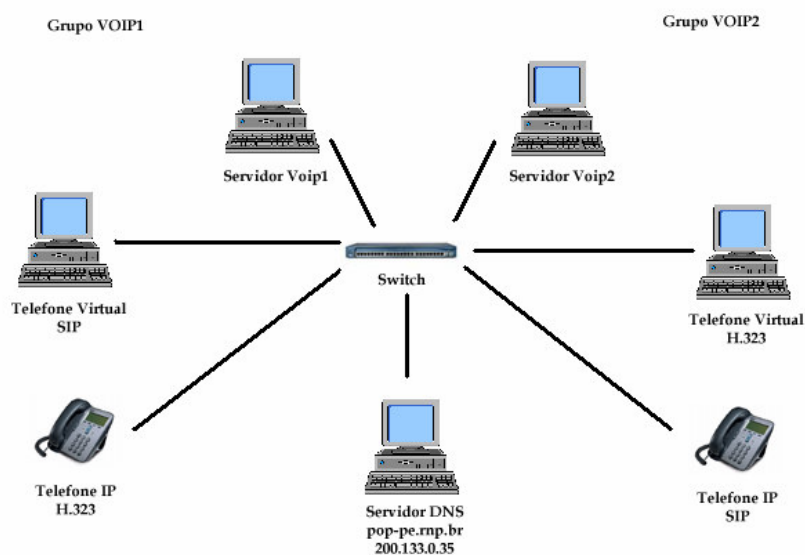


Figura 10. Cenário utilizado para o desenvolvimento e testes.

Neste capítulo serão abordados os processos realizados para o desenvolvimento da distribuição *Doulox Linux Live*, sistema operacional base para que fosse possível o funcionamento dos servidores *SER* (*ser-0.8.14-i386-1*), *GnuGK* (*gnugk-2.2.1-i386-1*) e *Asterisk* (*Asterisk-1.0.5-i386-1*). Nesta distribuição foi implantado o programa de gerenciamento dos serviços *VoIP* desenvolvido para esta distribuição que será abordada no Capítulo 4.

Para que o desenvolvimento deste projeto fosse possível, utilizaram-se máquinas virtuais, ou seja, computadores virtuais instalados em computadores reais. O programa *VMWare* foi utilizado como ilustrado na Figura 11.

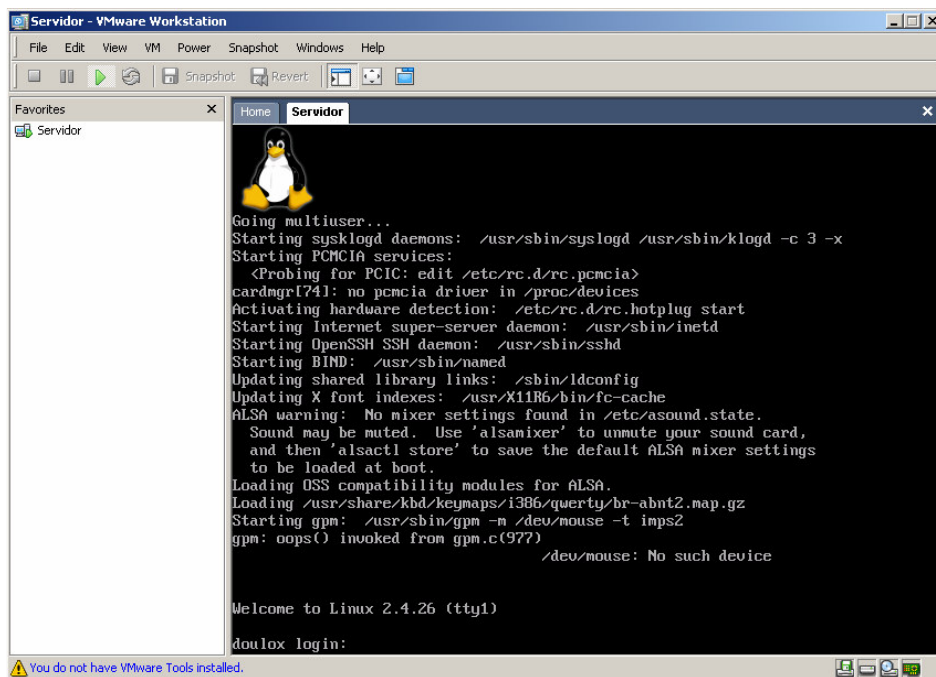


Figura 11. Utilização do *VMWare* para criação de máquinas virtuais com sistemas operacionais Linux.

Com o *VMWare*, foi possível trabalhar com vários computadores interligados via rede, possibilitando assim possuir vários servidores para testes. A Figura 11 ilustra um dos servidores iniciado. Poderíamos ter vários outros abertos em abas, levando em conta a capacidade de memória do computador. O uso deste programa foi de extrema necessidade pelo fato de não haver quantidade de computadores suficientes para o desenvolvimento do projeto. Outra utilidade foi a capacidade de trabalhar com computadores no nível de diretórios. Por exemplo, seria possível ter uma máquina virtual contendo uma distribuição *Linux* completa (com todos os pacotes instalados) enquanto que se poderia ter uma distribuição *Linux* reduzida, estando estes em um mesmo diretório do *Windows XP* [29] (sistema operacional base para a instalação do *VMWare*) e podendo ser feito *backups* destes diretórios (cada *Linux* corresponde a um diretório), prevenindo-se caso fosse obtido algum erro durante a utilização destes.

Uma outra facilidade seria a capacidade de ser instalado um sistema operacional em um terminal (máquina virtual) ou compilado algo em uma outra máquina virtual e, acrescentando, a capacidade de serem feitos testes em diferentes plataformas, como, por exemplo, fazer ligações entre clientes *Windows* e *Linux* em uma única máquina real. Praticamente todos os sistemas operacionais *Linux* mencionados neste capítulo serão máquinas virtuais.

Primeiramente foram instalados dois sistemas operacionais *Slackware Linux 10.0* [30], sistema base para o desenvolvimento do *Doulox*. A idéia inicial seria instalar o *GnuGK* em um

destes sistemas operacionais, empacotar os arquivos compilados, como, por exemplo, binários e tentar descompactar no outro sistema operacional, para que fosse possível ter um pacote padrão compilado do *GnuGK* especialmente para o *Slackware Linux* 10.0.

Alguns pacotes foram necessários para a compilação do *GnuGK*. São estes os programas: **PWLib** (*Portable Windows Library*) versão 1.5.2 e o **OpenH323** versão 1.12.2. Estes foram também compilados no *Slackware Linux* 10.0, antes de ser compilado o *GnuGK*, e foram empacotados usando os utilitários *tar* e o *gz* do *Linux*. Após serem obtidos os três programas compilados, os arquivos foram transferidos, já compilados e empacotados em um arquivo *tgz*, para o outro computador (máquina virtual) com uma versão *Linux* limpa, ou seja, uma distribuição recentemente instalada. Primeiramente apenas o pacote do *GnuGK* foi desempacotado, para a análise da sua execução sem as bibliotecas dos outros programas, a fim de tornar o *Doulox* uma distribuição com o menor tamanho possível. Mas não foi obtido êxito. Obeve-se o erro apresentado no Quadro 2.

```
./gnugk: error while loading shared libraries: libh323_linux_x86_r.so.1.12.2: cannot  
open shared object file: No such file or directory
```

Quadro 2. Erro ao ser executado o *GnuGK* sem as bibliotecas do *OpenH323*.

O *OpenH323* foi desempacotado e foi executado o binário do *GnuGK*, tendo este o nome *gnugk*. Obeve-se o mesmo erro. Primeiramente foi necessário identificar o erro analisando o arquivo *Makefile* do *OpenH323*, uma vez que a mensagem forneceu a breve pista de que esta biblioteca provavelmente poderia pertencer a este programa. Este arquivo, o *Makefile*, contém todas as linhas de comandos que seriam executadas após ser executado o comando *make* do *Linux* no diretório que o possui. O Comando *make* é utilizado para compilar arquivos fontes, que foi o caso do *GnuGK*, *OpenH323* e *PWLib* adquirido. Como não foi possível carregar tal arquivo, o *libh323_Linux_x86_r.so.1.12.2* de acordo com a mensagem de erro no Quadro 2 citado, então foi observado o *Makefile* para que tornasse possível identificar onde provavelmente esse arquivo poderia ser encontrado. Abrindo tal arquivo, o conteúdo descrito no Quadro 3 foi observado.

```
PREFIX=/usr/local  
exec_prefix = ${PREFIX}  
LIBDIR=${exec_prefix}/lib  
...  
install:  
    mkdir -p $(PREFIX)/include  
    mkdir -p $(LIBDIR)  
    chmod a+r lib/*so*  
    cp -df lib/*so* $(LIBDIR)  
    mkdir -p $(PREFIX)/include/openh323/  
    cp -rf include/* $(PREFIX)/include/openh323/  
    mkdir -p $(PREFIX)/share/openh323/  
    cp openh323u.mak $(PREFIX)/share/openh323  
    ln -s $(LIBDIR)/$(OH323_FILE) $(LIBDIR)/libopenh323.so  
    chmod -R a+r $(PREFIX)/include/openh323 $(PREFIX)/share/openh323
```

Quadro 3. Arquivo *Makefile* do diretório fonte do *OpenH323*.

Quando é executado o comando *make install* no diretório onde contém o arquivo *Makefile*, automaticamente os arquivos compilados são colocados em seus respectivos diretórios no sistema operacional para que os arquivos binários possam executar normalmente. Como a idéia de realizar a compilação, empacotamento e transferência de tais pacotes para uma outra distribuição recém instalada não funcionou como deveria, foi necessário utilizar o comando *make install* para que fosse observado onde os arquivos seriam copiados. A saída deste comando gerou as linhas mostradas no Quadro 4 de acordo com o *Makefile* do Quadro 3.

```
mkdir -p /usr/local/include
mkdir -p /usr/local/lib
chmod a+r lib/*so*
cp -df lib/*so* /usr/local/lib
mkdir -p /usr/local/include/openh323/
cp -rf include/* /usr/local/include/openh323/
mkdir -p /usr/local/share/openh323/
cp openh323u.mak /usr/local/share/openh323
ln -s /usr/local/lib/libh323_linux_x86_r.so /usr/local/lib/libopenh323.so
chmod -R a+r /usr/local/include/openh323 /usr/local/share/openh323
```

Quadro 4. Execução dos comandos presentes no arquivo *Makefile* do diretório fonte do *OpenH323*.

Os diretórios criados em */usr/local* foram observados e conseqüentemente foi encontrado o arquivo *libh323_Linux_x86_r.so.1.12.2*, mas o mesmo erro permanecia. Uma análise também foi realizada no arquivo */etc/ld.so.conf*, arquivo que contém as bibliotecas carregadas durante a inicialização (*boot*) do *Linux*. O diretório */usr/local/lib* estava constando neste arquivo, sendo este o diretório que continha o arquivo que estava faltando. Após a descompactação do *OpenH323*, apenas o comando *ldconfig* foi executado para recarregar todas as bibliotecas, incluindo as do *OpenH323*, para que o binário *gnugk* pudesse ser executado sem erros. É importante ressaltar que também foi obtido erro quando o *PWLib* não foi instalado na versão limpa do *Linux* (Quadro 5), sendo este erro bastante semelhante ao anterior.

```
./gnugk: error while loading shared libraries: libpt_linux_x86_r.so.1.5.2: cannot open
shared object file: No such file or directory
```

Quadro 5. Erro ao ser executado o *GnuGK* sem o *PWLib*.

A biblioteca *libpt_Linux_x86_r.so.1.5.2* foi necessária para o funcionamento do *GnuGK*, portanto foi realizado o mesmo procedimento feito com o *OpenH323*, a diferença é que neste caso utilizamos o *PWLib*. Após todos estes procedimentos terem sido efetuados, o *GnuGK* funcionou normalmente. Realizamos testes utilizando o *OpenPhone* (*softphone*) como cliente *H.323* para se registrar no servidor *GnuGK*. O telefone virtual **OhPhone** foi também compilado e criado o pacote, sendo este um cliente *H.323* em modo texto para sistemas operacionais *Linux*. Portanto, realizou-se com êxito um teste entre o *OpenPhone* instalado no *Windows* e o *OhPhone* no *Linux*, fazendo ligações utilizando tal servidor instalado e configurado com opções bastante básicas. Iniciou-se o *GnuGK* utilizando o comando do Quadro 6.

```
/usr/local/bin/gnugk -c /etc/gnugk.ini -ttttt -o /var/log/gnugk.log &
```

Quadro 6. Iniciando o *GnuGK* através de linha de comando.

Este comando utiliza como arquivo de configuração o arquivo *gnugk.ini*. Qualquer *log* gerado é escrito no arquivo *gnugk.log* e o *GnuGK* será executado em segundo plano (*background*) para que o terminal possa ser liberado para a execução de outros programas. A princípio, as configurações básicas citadas acima são encontradas no arquivo *gnugk.ini*, podendo ser visualizadas no Quadro 7.

```
[Gatekeeper::Main]
Fourtytwo=42
[GkStatus::Auth]
rule=allow
```

Quadro 7. Arquivo de configuração inicial (básico) do *GnuGK*.

Na seção *[Gatekeeper::Main]*, apenas a linha *Fourtytwo=42* foi adicionada, necessária para testar se o arquivo de configuração existe e mostrar uma mensagem de advertência na tela caso não exista. Na seção *[GkStatus::Auth]*, a linha *rule=allow* define que qualquer conexão pode ser realizada na porta de status, porta de controle do *GnuGK* via *Telnet* [31], um programa de emulação do terminal para redes *TCP/IP*, tais como a Internet. Mais detalhes de configuração do *GnuGK* serão abordados em seções futuras.

3.1 Tentativas de criação da distribuição a partir do *Slax*

Todos estes procedimentos acima mencionados foram realizados para um propósito. Existe uma distribuição *Linux* que pode ser executada direta do CD (por isso o nome *Live*), chamada *Slax*, mini-distribuição gerada tendo como base o *Slackware Linux 10*, sendo removidos alguns pacotes desnecessários e acrescentando outros, a fim de tornar o sistema operacional pequeno, porém bastante funcional.

Utilizou-se a versão 4 (quatro) do *Slax* para testes. Novos módulos, ou pacotes, podem ser criados para essa distribuição e adicionados a ela, diretamente ao arquivo imagem (ISO) do CD do *Slax*, colocando-as em um diretório chamado *modules*. Estes módulos são chamados de imagens, e estas são geradas a partir de programas compilados ou pacotes binários pré-compilados. Criou-se tal imagem utilizando o programa *Linux Live 4.2.4* com a ferramenta *dir2im*. Todos os arquivos gerados pós-instalação de qualquer programa no *Linux* foram copiados para um diretório temporário com os caminhos completos.

A partir dessa ferramenta são geradas as imagens a adicionar à imagem do CD do *Slax* a serem carregadas durante o boot do sistema. Observe que existem dois tipos de imagens aqui tratadas, a imagem que seria um pacote com extensão *img* criado a partir da compilação do programa fonte através da ferramenta *dir2img* e a imagem do CD que servirá para a gravação *Doulox Linux Live* no CD. A princípio, antes de editar a imagem do CD a fim de colocar a imagem do *GnuGK*, executou-se o sistema operacional *Slax* sem nenhuma imagem adicionada, e

o comando *moduse modulo.img* foi executado, onde *modulo.img* refere-se a qualquer imagem criada. Carregou-se o *gnugk.img* utilizando esta ferramenta, porém outro erro foi obtido semelhante a alguns vistos anteriormente, podendo ser visto no Quadro 8.

```
./gnugk: error while loading shared libraries: libmysqlclient.so.12: cannot open shared  
object file: No such file or directory
```

Quadro 8. Erro obtido ao executar o *GnuGK* como imagem no *Slax*.

O *MySQL* foi necessário para a compilação e execução do *GnuGK*, embora não foi necessário utilizar o *MySQL*, pois os usuários não seriam autenticados via banco de dados e sim via texto com senhas criptografadas, embora não tenha sido utilizada este tipo de autenticação, como será explicado mais adiante. Este erro não foi apresentado nas etapas iniciais (com o *OpenH323* e o *PWLib*) porque estes foram compilados em uma versão completa do *Slackware*, tendo este todos os pacotes disponíveis nos seus dois CDs de instalação, incluindo o *MySQL*. Como o *Slax* precisa ser uma distribuição bastante reduzida, o *MySQL* não seria tão importante em estar em uma distribuição normalmente voltada para usuários domésticos. O procedimento tomado para corrigir esse erro foi obter a imagem do *MySQL*, o *mysql.img*, do próprio site do *Slax* (criado pela comunidade). Utilizou-se o comando *moduse* no *MySQL* e *GnuGK* e este funcionou corretamente. Realizaram-se testes também adicionando tais imagens no diretório *modules* do *Slax* diretamente na imagem de CD, iniciando em seguida a distribuição através do CD. Os resultados foram positivos. A princípio bastaria, portanto, fazer o mesmo procedimento para os outros servidores, o *SER* e o *Asterisk*.

Depois de feitos todos os procedimentos para todos os servidores, compilando e gerando as imagens, uma série de testes foi realizada nos servidores e alguns erros foram encontrados. Um dos erros foi o problema do comando *find*, comando utilizado para procurar arquivos em um determinado diretório especificado. Ao utilizá-lo, a maioria dos arquivos, seja qual fosse o diretório, desaparecia. Os diretórios mais importantes não listavam mais, nem o */etc*, nem o */var* etc. Isto também acontecia caso fosse executado o *Asterisk* em modo console (mais detalhes na seção 3.5), onde saindo de tal modo, nenhum arquivo era listado e nenhum arquivo binário poderia ser executado. Portanto novas soluções precisavam ser tomadas.

3.2 Experiências com o *Slackware* e *Kernel 2.4*

Procurou-se gerar *Doulox Linux Live CD* a partir do *Slackware* puro, assim como o desenvolvedor do *Slax* o fez, a partir de uma versão do *Slackware 10.0* e *kernel 2.4* [32]. Precisou-se compilar e instalar o *ovlfs* (*Overlay Filesystem*) para ser possível a criação da imagem do CD. Durante a compilação deste, alguns erros foram obtidos, mas o suporte a esses erros não foi encontrado. O próprio desenvolvedor do *Slax* obteve os mesmos erros e os ignorou, não comprometendo o sistema. Obtivemos do site do *Slax* um arquivo de atualização (*patch*), tendo este o nome *slax-patch-4.2.0.tar.gz*.

Este arquivo foi descompactado e logo em seguida executado o comando *patch.sh* para que fosse possível a remoção de arquivos desnecessários, incluindo arquivos de bibliotecas, binários etc. Estes testes foram realizados em uma versão completa do *Slackware*, para que fosse possível a observação e análise dos resultados, a fim de obter a quantidade de redução de tamanho. Não obtivemos um bom resultado, visto que a imagem se mantinha grande. Obteve-se esta imagem executando o comando *runme.sh* do *patch*. Se o resultado fosse positivo, seriam

removidos alguns pacotes desnecessários e manter-se-ia tal idéia, mas, como citado anteriormente, realmente não foi obtido êxito, portanto uma outra solução deveria ser pesquisada.

3.3 Experiências com o *Slackware* e *Kernel 2.6*

Realizaram-se testes com o *kernel 2.6* do *Linux* a fim de obter êxito, em oposição aos problemas encontrados nas seções anteriores. O *ovlfs* foi ignorado por não ser necessário para tal *kernel*, e editamos, com uma ferramenta de edição de imagens de CDs (*UltraISO*), os CDs (ISOs) originais do *Slackware* a fim de apagar pacotes desnecessários e deixar a distribuição o menor possível para a instalação. Pesquisas foram realizadas a fim de analisar os pacotes importantes para o sistema. Algumas distribuições afirmavam não conter os pacotes *kernel-ide* e *kernel-modules*, mas sem estes o *Linux* não foi instalado corretamente, como consequência, a placa de rede do computador não foi reconhecida. Portanto, estes pacotes foram essenciais para o funcionamento do sistema por completo.

O *Slackware 10.0* não possui, durante a instalação, a opção de instalar com o *kernel 2.6*, portanto precisou-se instalar a versão 2.4 e atualizá-la para 2.6.9. Alguns arquivos foram necessários para a instalação e configuração deste de acordo com o Quadro 9.

```
alsa-driver-1.0.6a_2.6.9-i486-1.tgz
kernel-generic-2.6.9-i486-1.tgz
kernel-headers-2.6.9-i386-1.tgz
kernel-modules-2.6.9-i486-1.tgz
kernel-source-2.6.9-noarch-1.tgz
```

Quadro 9. Pacotes necessários para a instalação do *kernel 2.6.9*.

Procurou-se instalar tal versão do *kernel* sem a necessidade de compilação deste, mas após tais pacotes serem instalados com o comando *installpkg*, ao reiniciar o *Linux*, uma mensagem de *Kernel Panic* foi apresentada a tela, ou seja, um erro grave, travando assim todo o sistema. Tentou-se também compilar o *kernel*, mas a distribuição do *Slackware* instalada estava bastante reduzida, portanto muitos arquivos de definições de funções essenciais para a compilação estavam faltando, incluindo o *stdio.h*, que é bastante utilizado em programas escritos na linguagem de programação *C* [33]. Assim, foi necessário instalar a versão completa do *Slackware* e compilar o *kernel* versão 2.6.9.

Transferiram-se os arquivos do Quadro 9 para o *Slackware*, com todos os pacotes, e os programas *Linux Live* versão 5.0.7, *SquashFS* e *UnionFS*, necessários para a criação da distribuição. Nos arquivos de compilação do programa *Linux Live*, havia alguns arquivos com a extensão *ko* do *SquashFS* e *UnionFS*. Provavelmente, não seria necessário instalar ambos a partir do código fonte, bastaria apenas incluir esses módulos (*squashfs.ko* e *unionfs.ko*) no *Kernel* e recompilá-lo. Mas, após pesquisas feitas, não foi possível identificar uma maneira de tornar isto possível. Portanto, instalou-se tudo a partir dos arquivos fontes. Após a compilação do *kernel* e a instalação dos programas mencionados acima, o mesmo erro de *Kernel Panic* foi apresentado à tela, podendo ser visualizado no Quadro 10.

```
kernel panic - not syncing: VFS: Unable to mount root fs on unknown-block(3,2)
```

Quadro 10. Erro *Kernel Panic*.

Observando a mensagem, pode-se perceber que o erro refere-se ao sistema de arquivos (*FileSystem* ou *FS*). O *Slackware* foi reinstalado, todos os procedimentos citados acima foram refeitos e o suporte ao sistema de arquivos ReiserFS foi adicionado, antes da compilação do *kernel*, que correspondia ao sistema de arquivos que estava sendo utilizado no momento, sendo este o mais atual em sistemas operacionais *Linux*. Mas ainda foram obtidos erros: a placa de rede que não funcionava. Marcamos todas as opções de *drivers* de rede (no menu *Network Devices*) antes de recompilar novamente. Com isto tudo funcionou normalmente e foi possível rodar o *Slackware* com o *kernel* na versão 2.6.9. Instalamos o *UnionFS* apenas utilizando os comandos *make* e *make install* e o *SquashFS* foi instalado antes da compilação do *kernel* de acordo com o manual do *SquashFS*, o *SquashFS-HOWTO*.

Os pacotes desnecessários foram removidos e o comando *runme.sh* do *Linux Live* foi executado para criar a imagem de CD do *Doulox*, mas alguns erros foram obtidos ao iniciar o CD. Portanto não foi obtido êxito utilizando o *kernel* 2.6. Visitando a página do *Slax*, obtivemos a informação de que esta distribuição desenvolveu com êxito sua versão com o *kernel* 2.6, sendo este o *Slax* versão 5, porém, não existe ainda documentação ou um bom suporte para criação de distribuições *Live* utilizando *kernels* com versões 2.6.x.

3.4 Criando definitivamente a distribuição

Como o suporte e documentação à criação de distribuições *Live* utilizando o *Linux Live* como programa para *kernel* 2.4 foi menos difícil de encontrar, procurou-se seguir este caminho. Na página do *Slax* havia uma breve documentação de como o desenvolvedor criou tal distribuição. Seguimos os mesmos procedimentos: um pacote de um *kernel* com suporte a *ovlfs* e *alsa* (drivers para dispositivos de som) foi adquirido, chamado de *kernel-2.4.26-ovlfs-devfs-alsa-1.0.4-i486-1* e executado o *patch slax-patch-4.2.0.tar.gz*, também presente na página do *Slax*, que seria útil para remover arquivos desnecessários do *Slackware*, incluindo os do *kernel*. Após criarmos e executarmos o que poderia ser a primeira versão do *Doulox*, alguns erros foram obtidos de acordo com o Quadro 11.

INIT:	Id	"1"	respawning	too	fast:	disabled	for	5	minutes
INIT:	Id	"2"	respawning	too	fast:	disabled	for	5	minutes
INIT:	Id	"3"	respawning	too	fast:	disabled	for	5	minutes
INIT:	Id	"6"	respawning	too	fast:	disabled	for	5	minutes
INIT:	Id	"5"	respawning	too	fast:	disabled	for	5	minutes
INIT: Id "4" respawning too fast: disabled for 5 minutes									

Quadro 11. Erros obtidos ao executar a primeira versão do *Doulox*.

Pesquisando em fóruns da página do *Slax*, identificamos a solução para tais erros, e a imagem após a criação desta distribuição funcionou normalmente, resultando em nova versão primitiva e básica do *Doulox*. O erro foi eliminado removendo o pacote *udev-026-i486-1* e instalando o pacote *devfsd-1.3.25-i486-4* no lugar deste. Abaixo segue um resumo dos 5 (cinco) passos para que a distribuição base fosse criada:

1. O *Slackware* 10.0 foi instalado com os CDs modificados, retirando vários pacotes desnecessários ao propósito do *Doulox*, como, por exemplo, servidores diversos, algumas

- interfaces gráficas, editores de textos, aplicativos. Foi acrescentado a um destes CDs o pacote *devfsd-1.3.25-i486-4* e removido o pacote *udev-026-i486-1*.
2. Instalou-se um *kernel* contendo módulos compilados para *ovlfs* e *alsa* disponível no site do *Slax*. O nome do pacote era *kernel-2.4.26-ovlfs-devfs-alsa-1.0.4-i486-1*. Bastou executar o comando `installpkg kernel-2.4.26-ovlfs-devfs-alsa-1.0.4-i486-1`, sem necessidade de recompilação do *kernel*.
 3. O pacote *slax-patch-4.2.0.tar.gz* foi obtido e descompactado, disponível também no sítio do *Slax*. Após a descompactação, no diretório `./rootpatch` executou-se o comando `patch.sh` para a remoção de arquivos desnecessários. Em seguida copiaram-se os arquivos deste diretório (*rootpatch*) para o diretório raiz do *Linux (I)* e copiou-se todo o conteúdo de `./Slax-cd` para o diretório `./bootfiles` do pacote *linux-Live-4.2.4* já descompactado.
 4. Entrando no diretório do *linux-Live-4.2.4*, executou-se o comando `runme.sh`. A imagem do CD foi criada em `/tmp/Livecd.iso` por padrão.
 5. Transferiu-se o arquivo *Livecd.iso* para o *Windows*, montou-se tal arquivo com o programa **Damon Tools** e o sistema operacional foi executado normalmente. O *GnuGK* funcionou no *Doulox* a partir da imagem (módulo) criada anteriormente. O único problema foi a necessidade de utilizar o comando `ldconfig` após o início do *Doulox* para o funcionamento do *GnuGK*.

3.5 Modificações e testes utilizando o *Doulox*

Todos os momentos em que se criava a distribuição, ao reinicializar o sistema base onde era gerada a distribuição, ocorria um erro durante o *boot*, mais especificamente durante a seleção do sistema operacional após o *boot*, ou seja, no LILO [34]. O sistema apenas travava e exibia uma mensagem de erro. Realizaram-se então testes utilizando o sistema de arquivos EXT3 ao invés do *ReiserFS* e não mais foram obtidos tais erros.

Após a criação da imagem do *GnuGK* e suas dependências, precisamos criar a imagem do programa *SER*, sendo este o servidor para protocolos *SIP*. A imagem (módulo) foi criada e testada no *Doulox* e durante a sua compilação, o diretório base para armazenamento dos arquivos de configuração e binários ficou em `/usr/local`, tendo como prefixo de instalação este mesmo diretório. Primeiramente o *BIND* foi instalado, programa que servirá como servidor *DNS*, sendo este servidor utilizado pelo *SIP*. O domínio atribuído aos clientes do grupo *Voip1* estão localizados no servidor *Voip1*, tendo este o domínio `voip1.pop-pe.rnp.br`. Realizaram-se testes locais para a fácil demonstração em seminários ou empresas. Apenas para teste básico, para se ter a certeza de que o *SER* foi compilado corretamente, executou-se o *SER* sem a edição do arquivo de configuração, o `ser.cfg`, localizado em `/usr/local/etc/ser`, mantendo o arquivo padrão de instalação. Dois clientes *SIP* foram configurados, utilizando o programa *X-Lite*, fazendo-os se registrar neste servidor. Porém, obtivemos um erro. O *X-Lite* gera um arquivo de *logs*, onde é possível identificar erros. Obtendo as linhas finais desse arquivo de *log*, foi possível a identificação do erro descrito no Quadro 12.

...

```
RECEIVE TIME: 18309250
RECEIVE << 200.133.0.212:5060
SIP/2.0 483 Too Many Hops
Via: SIP/2.0/UDP
200.133.0.209:5060;rport=5060;branch=z9hG4bK0B0681B079DE43F3B071389928F174DE
From: rodrigo <sip:8132218785@voip1.pop-pe.rnp.br>;tag=2728648980
To: rodrigo <sip:8132218785@voip1.pop-pe.rnp.br>;tag=b27e1a1d33761e85846fc98f5f3a7e58.92bc
Call-ID: 28D6B5FB88EB4AA5A3CA9B4EFE367FD9@voip1.pop-pe.rnp.br
CSeq: 25337 REGISTER
Server: Sip EXpress router (0.8.14 (i386/linux))
Content-Length: 0
Warning: 392 200.133.0.212:5060 "Noisy feedback tells: pid=8020 req_src_ip=200.133.0.212
req_src_port=5060 in_uri=sip:voip1.pop-pe.rnp.br out_uri=sip:voip1.pop-pe.rnp.br via_cnt==71"
```

Quadro 12. Erro ao iniciar o primeiro teste com o *SER* utilizando o *X-Lite*.

Não entraremos em detalhes dos *IPs* utilizados, nome de usuários ou plano de numeração ainda, portanto pode-se ignorar tais dados no *log* acima. O dado que foi procurado está indicado na linha onde está contida a mensagem *Too Many Hops*, sendo este o problema a ser resolvido.

Procurou-se na Internet, em sites de busca, pelo erro acima citado, e encontrou-se que o problema estaria no arquivo de configuração, o *ser.cfg*. Editando tal arquivo, a linha descrita no Quadro 13 foi encontrada.

```
if (uri==myself)
```

Quadro 13. Linha de configuração do *SER* para identificação de *URI* através da identificação do próprio domínio.

A linha do Quadro acima foi substituída pela linha apresentada no Quadro 14.

```
if (uri=~"[@:\.]voip1\pop-pe.rnp.br\([:;].*)*")
```

Quadro 14. Linha de configuração do *SER* para identificação de *URI* por extenso.

Ou seja, o *SER* não conseguia identificar o seu próprio domínio. Isso acontece porque a variável de sistema do *Linux*, o *SIP_DOMAIN*, não estava configurada. Esta fase de configuração será verificada quando falarmos da configuração do *SER* utilizando autenticação de usuários. Após tais modificações, dois clientes foram configurados. Um deles possuía as configurações ilustradas na Figura 12.

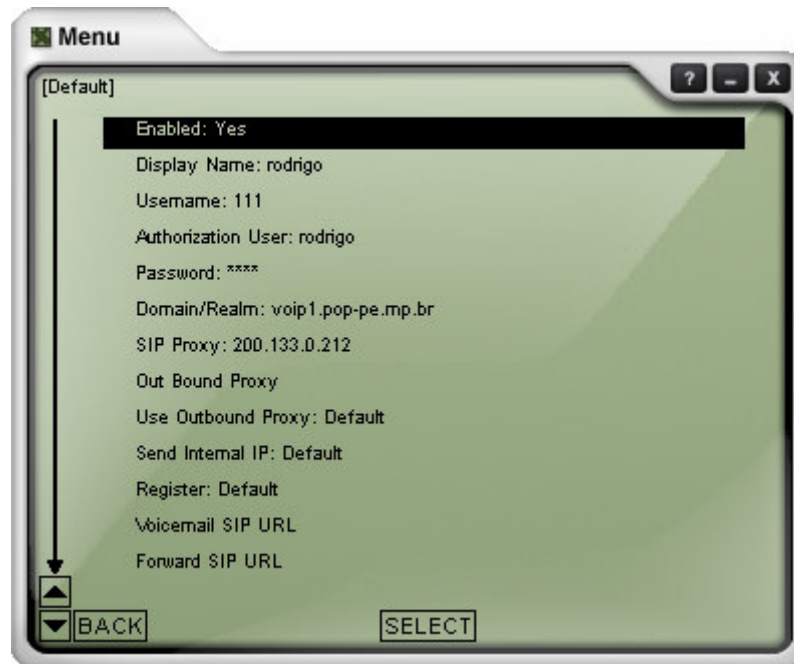


Figura 12. Configuração do X-Lite

Como nome em tela (*Display Name*), um nome qualquer foi adicionado, neste caso *rodrigo*, um usuário, o *111*, que será o número desse cliente, um usuário de autenticação, que será autenticado pelo servidor, uma senha para este usuário, o domínio correspondente e o endereço *IP* do servidor *SER*. Configurações semelhantes foram feitas em outro cliente, com número *222* e chamadas foram realizadas entre tais clientes normalmente, sem nenhum erro.

Em seguida a imagem para o Asterisk foi criada. Primeiramente compilamos este programa junto com suas dependências, os programas *Zaptel* (*zaptel-1.0.4-i386-1*) e o *Libpri* (*libpri-1.0.4-i386-1*). Inicialmente o programa *Zaptel* foi compilado e instalado com o prefixo (diretório de instalação) */* e o *Libpri* como prefixo */*. Em seguida compilamos o *Asterisk*, tendo o prefixo como */usr/local/Asterisk*, mas uma atualização do pacote *mpg123* foi requerida durante a instalação. Realizamos tal atualização, obtendo um novo pacote do *mpg123* e este foi compilado apenas com os comandos *make linux* e não o *make generic*, pois erros estavam sendo obtidos com este último. É importante lembrar que para a criação da imagem, é necessário copiar todos os arquivos e seus caminhos completos do programa, portanto, por exemplo, para instalação do *Libpri*, para o diretório *root* raiz (*/*), foi preciso identificar todos os arquivos copiados do *Libpri* a partir do diretório raiz, observando o *Makefile* deste.

Iniciamos o *Asterisk*, digitando o comando *Asterisk -vvvdddc*, onde o “-c” indica ao *Asterisk* para iniciar em modo *console*, o “-v” em modo *verbose* (visualização de *logs*) e o “-d” indica o nível de *debug*, ou seja, o nível em que serão apresentados os *logs*. Obtivemos alguns erros e advertências ao executar tal comando, de acordo com o que mostra a Figura 13, através de um cliente *SSH* (*Secure Shell*) [35] no servidor virtual. O *SSH* é uma ferramenta bastante importante em sistemas operacionais baseados em *Unix* [36] para administração remota do sistema.


```

200.133.0.211 - PuTTY
== Loaded firmware 'iaxy.bin'
  -- Loaded provisioning template 'default'
[chan_local.so] => (Local Proxy Channel)
[chan_mgcp.so] => (Media Gateway Control Protocol (MGCP))
== MGCP Listening on 0.0.0.0:2727
== Using TOS bits 0
[chan_modem_i4l.so] => (ISDN4Linux Emulated Modem Driver)
[chan_oss.so] => (OSS Console Channel Driver)
== Console is full duplex
[chan_phone.so] => (Linux Telephony API Support)
[chan_sip.so] => (Session Initiation Protocol (SIP))
== SIP Listening on 0.0.0.0:5060
== Using TOS bits 0
== Registered application 'SIPDtmfMode'
[chan_skinny.so] => (Skinny Client Control Protocol (Skinny))
Feb 21 16:58:57 WARNING[1523]: chan_skinny.c:2584 reload_config: Unable to get our IP address, Skinny disabled
Feb 21 16:58:57 WARNING[1536]: chan_oss.c:239 sound_thread: Read error on sound device: Resource temporarily unavailable
[chan_zap.so]Feb 21 16:58:57 WARNING[1523]: loader.c:258 ast_load_resource: libtonezone.so.1: cannot open shared object file: No such file or directory
Feb 21 16:58:57 WARNING[1523]: loader.c:440 load_modules: Loading module chan_zap.so failed!
root@voip:/usr/local/asterisk/usr/sbin#
  
```

Figura 13. Erro ao ser iniciado o Asterisk

Pode-se notar que uma das advertências refere-se ao arquivo *libtonezone.so.1* que não foi encontrado. Realizamos uma busca procurando pelos arquivos *libtonezone.so.** e alguns arquivos semelhantes foram encontrados. Apenas foi necessário criar um link de *libtonezone.so.1* para o arquivo *libtonezone.so* original. Feito isto, o Asterisk funcionou corretamente, mostrando o modo console, de acordo com a Figura 14.

```

Other Linux 2.4.x kernel 3 - VMware Workstation
File Edit View VM Power Snapshot Windows Help
Snapshot Revert
Favorites: Other Linux 2.4.x kernel, Other Linux 2.4.x kernel 2, Other Linux 2.4.x kernel 3
Home Other Linux 2.4.x kernel Other Linux 2.4.x kernel 2 Other Linux 2.4.x kernel 3
[Suffix]
== Registered application 'Suffix'
[Wait]
== Registered application 'Wait'
[WaitExten]
== Registered application 'WaitExten'
Asterisk Dynamic Loader Starting:
[chan_modem.so] => (Generic Voice Modem Driver)
== Loading modem driver chan_modem_aopen.so => (A/Open (Rockwell Chipset) ITU-2 VoiceModem Driver)
[res_musiconhold.so] => (Music On Hold Resource)
Feb 21 17:11:37 WARNING[1289]: res_musiconhold.c:565 moh_register: Unable to open pseudo channel for timing... Sound may be choppy.
Feb 21 17:11:37 WARNING[1294]: res_musiconhold.c:124 spawn_mp3: /var/lib/asterisk/mohmp3 is not a valid directory
Feb 21 17:11:37 WARNING[1294]: res_musiconhold.c:278 monmp3thread: unable to spawn mp3player
== Registered application 'MusicOnHold'
== Registered application 'WaitMusicOnHold'
== Registered application 'SetMusicOnHold'
Parsing /usr/local/asterisk/etc/asterisk/logger.conf
Asterisk Event Logger restarted
== RTP Allocating from port range 10000 -> 20000
Asterisk Ready.
*CLI> _
  
```

Figura 14. Asterisk iniciado corretamente

Algumas advertências foram identificadas, mas ignoradas pelo fato do modo console ter iniciado corretamente. Colocamos todas as imagens criadas, de todos os servidores e dependência, no diretório *modules* do CD do *Doulox* e iniciamos tal distribuição. Colocamos o comando *ldconfig* para iniciar durante a inicialização do *Linux* no arquivo */etc/rc.d/rc.local* para as bibliotecas do servidores serem recarregadas ao sistema. Testes foram realizados em todos os servidores, mas o *Asterisk* retornou os mesmos problemas da Figura 13. Apenas quando o comando *find* era executado, procurando por qualquer palavra, o *Asterisk* funcionava, mas apresentava os erros citados na Seção 3.1 com relação ao problema de listagem e execução após a saída do *Asterisk*, executando o comando *stop now* no console.

Pesquisamos este tipo de erro e nada foi encontrado, tentamos adicionar todas as imagens em uma só, mas não obtivemos sucesso. Procuramos não mais utilizar o sistema de imagens, mas sim procurar criar pacotes de instalação do *Slackware*, arquivos com extensão *tgz*, e instalá-los antes da criação do *Doulox* e verificar os resultados obtidos. Para isto encontramos uma ferramenta chamada ***checkInstall***. Com esta ferramenta poder-se-ia criar pacotes para o *Slackware* (*.tgz*), *Red Hat* [37] (*.rpm*) e *Debian* [38] (*.deb*), e com isto poderia se instalar facilmente um pacote em qualquer outro servidor contendo o *Slackware* 10.0 utilizando o comando *installpkg pacote.tgz*. Procurou-se criar o pacote do *PWLib* e obteve-se o erro ilustrado no Quadro 15.

```
cp tools/asnparsers/obj*/asnparsers /usr/local/bin/  
cp: will not overwrite just-created `/usr/local/bin/asnparsers' with  
`tools/asnparsers/obj_Linux_x86_r/asnparsers'  
make: *** [install] Error 1  
  
**** Installation failed. Aborting package creation.  
  
Restoring overwritten files from backup...OK  
Cleaning up...OK  
Bye.
```

Quadro 15. Erro ao criar o pacote do *PWLib* utilizando o *checkinstall*.

Para corrigir tal problema editou-se o arquivo *Makefile* do *PWLib* e a linha *cp tools/asnparsers/obj*/asnparsers /usr/local/bin/* foi modificada para *cp tools/asnparsers/obj_Linux_x86_r/asnparsers /usr/local/bin/*, especificando assim o diretório correto onde os arquivos a serem copiados estão localizados, obtendo os arquivos do diretório *obj_Linux_x86_r* (*release*) e ignorando os do diretório *obj_Linux_x86_d* (*debug*). Após ter criado a distribuição *Doulox*, os mesmos erros de desaparecimento de arquivos continuavam, mas não mais ocorria este erro ao sair do *Asterisk*, ocorrendo somente ao utilizar os utilitários *find* ou *updatedb*. Portanto foram mantidos tais procedimentos, uma vez que este obteve o melhor resultado, ou seja, menos falhas (*bugs*). Portanto os pacotes foram criados com o *checkinstall* e são estão listados no Quadro 16.

```
asterisk-1.0.5-i386-1
mysql-4.0.20-i486-1
ser-0.8.14-i386-1
ohphone-ohphone-i386-1
zaptel-1.0.4-i386-1
gnugk-2.2.1-i386-1
openh323-openh323-i386-1
libpri-1.0.4-i386-1
pwlib-pwlib-i386-1
```

Quadro 16. Pacotes criados com o utilitário *checkinstall* através dos códigos fonte.

Após iniciar o *Doulox*, verificou-se que o servidor *TFTP* (*Trivial File Transfer Protocol*) [39] simplesmente não era iniciado. Portanto modificou-se o arquivo */etc/inetd.conf* a fim de habilitar tal servidor, e apenas foi descomentada a linha visualizada no Quadro 17. Observe que foi necessário criar o diretório */usr/local/tftpboot* especificado na linha.

```
tftp dgram udp wait root /usr/sbin/in.tftpd in.tftpd -s /usr/local/tftpboot -r blksize
```

Quadro 17. Alteração do arquivo *inetd.conf*.

O servidor *TFTP* será necessário ao utilizar telefones *IP*, onde normalmente são realizadas atualizações de *firmware*, sendo tais atualizações realizadas através de tal servidor. Ainda assim não foi possível iniciar o servidor após o boot, a edição de mais um arquivo foi necessária, o */etc/rc.d/rc.server*, cujo conteúdo é apresentado no Quadro 18.

```
#!/bin/bash
# search for "server" boot option
# if not found, disable all network daemons
#
# Author: Tomas Matejcek <http://slax.linux-live.org>
#
if grep server /proc/cmdline 1> /dev/null 2> /dev/null ; then
  ATTR="a+x"
else
  ATTR="a-x"
fi

#chmod $ATTR /etc/rc.d/rc.{sshd,sendmail,httpd,ip_forward,bind} 2>/dev/null
chmod a+x /etc/rc.d/rc.{sshd,gnugk,ser,asterisk,inetd,ip_forward,httpd,bind}
2>/dev/null
```

Quadro 18. Conteúdo do arquivo */etc/rc.d/rc.server*.

A linha que continha o *chmod* foi comentada e adicionada outra linha semelhante. A diferença foi que o servidor *inetd* foi adicionado para ser iniciado, pois sem este não seria

possível obter um servidor *TFTP* funcionando. Retirou-se também o *sendmail* (servidor de e-mail) [40], pois este não é necessário ao propósito do projeto. Adicionamos o *GnuGK*, *SER* e *Asterisk*.

Com o servidor *TFTP* ativo, foi possível transferir os arquivos de atualização do telefone *Cisco IP Phone 7905G* para o diretório */usr/local/tftpboot* e alterar um arquivo de configuração que será lido pelo telefone para a configuração automática deste. Configurou-se também o *OpenPhone* e tentou-se realizar ligações entre este e o telefone *IP*. Fazendo ligações do telefone *IP* para o *OpenPhone* (telefone virtual), obteve-se êxito mas o inverso não foi possível, o telefone *IP* simplesmente não tocava. Após um bom tempo de pesquisa para tentar identificar o erro, conseguiu-se identificar que apenas habilitando a opção *Disable Fast-Start* no *OpenPhone*, mais especificamente no menu *options->general*, resolveria o problema. Tal opção pode ser conferida na Figura 15. Acessou-se a porta de controle 7000 via *Telnet* do servidor *GnuGK* para a visualização de *logs* e gerenciamento deste e funcionou normalmente.

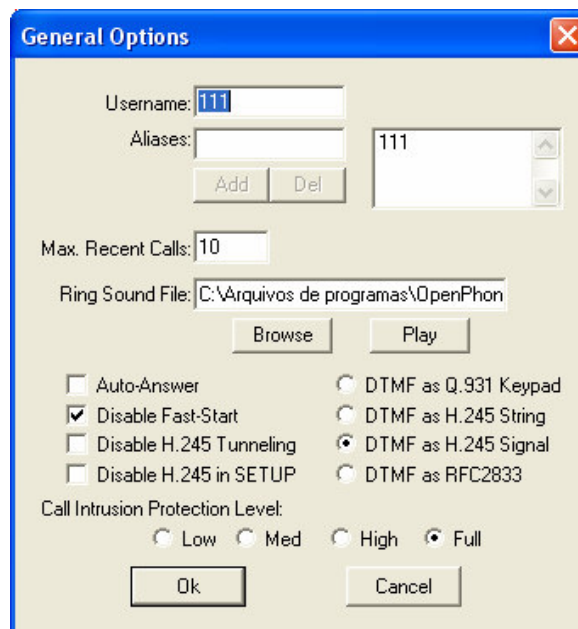


Figura 15. Marcação da opção *Disable Fast-Start* no *OpenPhone*

Tratando-se agora do servidor *SER*, para que fosse possível haver autenticação de usuários, foi necessário instalar o *MySQL*. Como já se havia instalado este, por conta do *GnuGK* (embora não fosse necessário para este servidor), necessitou-se apenas verificar se a sua instalação estava correta. Tentou-se iniciar o servidor *MySQL*, utilizando o comando */etc/rc.d/rc.mysql start*, e obteve-se o erro mostrado no Quadro 19.

```
root@servidor:/etc/rc.d# Starting mysqld daemon with databases from /var/lib/mysql
root@servidor:/etc/rc.d# 050224 16:14:11 mysqld ended
```

Quadro 19. Erro ao iniciar o *MySQL*.

Observando o arquivo */var/lib/mysql/servidor.err*, os seguintes *logs* no Quadro 20 foram obtidos.

```
050224 16:19:17 mysqld started
050224 16:19:18 InnoDB: Started
050224 16:19:18 Fatal error: Can't open privilege tables: Can't find file:
'./mysql/host.frm' (errno: 13)
050224 16:19:18 Aborting

050224 16:19:18 InnoDB: Starting shutdown...
050224 16:19:20 InnoDB: Shutdown completed
050224 16:19:20 /usr/libexec/mysqld: Shutdown Complete

050224 16:19:20 mysqld ended
```

Quadro 20. Log do erro relativo ao início do *MySQL*.

Pode-se observar que o erro acima destacado refere-se a privilégio. Executou-se o comando `chown -R mysql.users /var/lib/mysql` para que as tabelas pudessem ser editadas pelo próprio usuário *mysql*. A princípio não haverá senha para acesso no *MySQL*. Quando se criou a distribuição *Doulox*, erros ao iniciar o *MySQL* foram obtidos rodando o *Doulox* direto do CD. Obteve-se uma imagem do *MySQL* (*mysql.img*), descompactou-se em um diretório utilizando o comando *img2dir* do *Linux Live* e tais arquivos criados foram movidos para o diretório raiz. Feito isto, o *MySQL* funcionou corretamente na inicialização do *Doulox*.

O programa para gerenciamento do *Doulox* foi desenvolvido em *PHP*, portanto foi necessária a instalação deste através do pacote do *PHP* originado do próprio CD do *Slackware*. Para que fosse possível o seu funcionamento, a linha apresentada no Quadro 21 precisou ser descomentada no arquivo */etc/apache/httpd.conf*.

```
Include /etc/apache/mod_php.conf
```

Quadro 21. Linha descomentada no arquivo */etc/apache/httpd.conf*.

No arquivo */etc/apache/php.ini* foram modificadas as linhas abaixo, para que as variáveis funcionassem corretamente, deixando os parâmetros *register_globals* e *output_buffering* como no Quadro 22.

```
register_globals = On
output_buffering = On
```

Quadro 22. Linhas descomentadas no arquivo */etc/apache/php.ini*.

Temos, até o momento, todos os servidores instalados e basicamente configurados. Necessitar-se-ia instalar alguns pacotes extras para uma melhor configuração dos serviços *VoIP*. As seções a seguir tratarão destes aspectos e incluirão os passos para a configuração dos servidores *SER*, *Asterisk* e *GnuGK*.

3.6 Configuração do *SER*

Essa seção, como também as outras, serão aplicadas a dois servidores de teste. Para não ser tão repetitivo, apenas a configuração de um servidor será apresentada, o *voip1.pop-pe.rnp.br*.

Algumas variáveis de sistema deveriam ser iniciadas durante o *boot* do sistema. As variáveis são o *SIP_DOMAIN* e *LD_LIBRARY_PATH*. O arquivo responsável por fazer esse tipo de trabalho no *Linux* é o */etc/profile*. A variável *SIP_DOMAIN* contém o domínio da máquina em que se está trabalhando, enquanto que *LD_LIBRARY_PATH* contém o diretório onde os bancos do *MySQL* estão armazenados. Portanto, colocou-se no final do arquivo */etc/profile* o conteúdo do Quadro 23 para iniciar tais variáveis.

```
export SIP_DOMAIN=voip1.pop-pe.rnp.br
export LD_LIBRARY_PATH=/usr/lib/mysql
```

Quadro 23. Linhas adicionadas ao final do arquivo */etc/profile*.

Após esta configuração, criou-se o banco do *SER* no *MySQL* existente, utilizando o comando */usr/local/sbin/ser_mysql.sh*. O usuário e senha relacionados ao banco *SER* estão especificados no próprio arquivo *ser_mysql.sh*, como pode ser observado nos destaques no Quadro 24.

```
DBNAME=ser
DBHOST=localhost
USERNAME=ser
DEFAULT_PW=heslo
ROUSER=serro
RO_PW=47serro11
```

Quadro 24. Senhas padrão do banco de dados *SER*.

Após adicionar o banco, verificou-se se realmente este foi criado. Observe que a senha padrão não foi alterada, nem se mudou a senha do *MySQL*, que por padrão de instalação não tem senha. Isto seria um problema de segurança, mas as implementações de segurança não serão aplicadas neste projeto, deixando-as para trabalhos futuros.

Seria necessário instalar o pacote *ser-mysql*, que permitiria a conexão entre o *SER* e o *MySQL*. Porém, este pacote apenas existia para distribuições *Debian*, sendo este do tipo “.deb”. A primeira tentativa foi instalar um programa chamado *Alien*, que faz a transformação de um tipo de pacote para o outro. Este realiza transformações entre os pacotes “.deb”, “.tgz” e “.rpm”. Antes de compilar o *Alien*, instalou-se para o *Slackware* o pacote *alien-extra.tgz* e em seguida transformou-se um pacote do *Debian* para o *Slackware* do *ser-mysql*. Ao iniciar o *SER*, o erro especificado no Quadro 25 era obtido.

```
Feb 25 10:52:45 servidor ser: ERROR: load_module: could not open module
</usr/lib/ser/modules/mysql.so>: libmysqlclient.so.10: cannot open shared object
file: No such file or directory
```

Quadro 25. Erro ao iniciar o *SER* com suporte ao *MySQL*.

Para corrigir tal erro, criou-se um link simbólico em `/usr/lib/mysql` como indicado no Quadro 26 através de um comando `ls -lah`.

```
lrwxrwxrwx    1 root root    39 Feb 25 10:55 libmysqlclient.so.10 ->
/usr/lib/mysql/libmysqlclient.so.12.0.0*
```

Quadro 26. Criação de um link simbólico para a biblioteca `libmysqlclient.so`.

Após a configuração do arquivo `ser.cfg` que será vista a seguir, estávamos obtendo os erros relacionado ao `ser-mysql`, mostrados no Quadro 27.

```
Feb 25 12:25:17 servidor ser[2898]: connect_db(): No enough memory
Feb 25 12:25:17 servidor ser[2898]: db_init(): Error while trying to connect database
Feb 25 12:25:17 servidor ser[2898]: auth_db:init_child(): Unable to connect database
Feb 25 12:25:17 servidor ser[2898]: init_mod_child(): Error while initializing module
auth_db
Feb 25 12:25:17 servidor ser[2898]: init_child failed
```

Quadro 27. Erros relacionados à inicialização do `SER` com suporte ao `MySQL`, utilizando o `ser-mysql`.

Após bastante pesquisa, conseguiu-se corrigir o erro. Este pacote `ser-mysql` não precisaria ser instalado dessa forma. A dificuldade foi que na documentação do `SER` não constava o procedimento certo para instalar tal suporte. Ao compilar o `SER`, entrou-se no diretório `./modules/mysql` (dentro do diretório de fontes de `SER`) e executou-se o comando `gmake` dentro deste diretório. Este comando gerou um arquivo `mysql.so`, sendo este copiado para o diretório `/usr/lib/ser/modules/`. Mais tarde este arquivo foi adicionado direto no pacote do `SER`, sem haver a necessidade de copiá-lo após a instalação deste pacote.

Editou-se o arquivo `/usr/local/etc/ser/ser.cfg`, sendo este o arquivo de configuração do `SER`, e realizou-se algumas modificações. Adicionou-se ao início do arquivo a linha em destaque presente no Quadro 28.

```
#debug=3      # debug level (cmd line: -dddddddd)
#fork=yes
#log_stderr=no  # (cmd line: -E)

alias="voip1.pop-pe.rnp.br"
```

Quadro 28. Adição de um `alias` no arquivo `ser.cfg`.

As linhas do Quadro 29 foram descomentadas.

```
loadmodule "/usr/lib/ser/modules/mysql.so"  
loadmodule "/usr/local/lib/ser/modules/auth.so"  
loadmodule "/usr/local/lib/ser/modules/auth_db.so"
```

Quadro 29. Remoção de comentários no arquivo *ser.cfg* relacionados aos *loadmodules*.

A primeira linha indica que o *MySQL* será utilizado para o cadastro dos usuários. As segunda e terceira linhas dão suporte à autenticação das mensagens. As linhas apresentadas no Quadro 30 indicam que as alterações no banco de dados seriam escritas periodicamente. Portanto, comentou-se uma linha e descomentou-se a outra.

```
#modparam("usrloc", "db_mode", 0)  
# Uncomment this if you want to use SQL database  
# for persistent storage and comment the previous line  
modparam("usrloc", "db_mode", 2)
```

Quadro 30. Alterações ao *ser.cfg* quanto ao parâmetro *usrloc*.

Ao criar um usuário no banco de dados, são armazenadas no banco a senha em texto claro e *hash's MD5* (campos *password*, *hal* e *halb*, respectivamente). Durante a autenticação do usuário pode ser definido que a senha em texto claro será utilizada. Isto torna a configuração inicial e os testes mais simples. Para isto, devem ser removidos os comentários das linhas presentes no Quadro 31.

```
modparam("auth_db", "calculate_hal", yes)  
modparam("auth_db", "password_column", "password")
```

Quadro 31. Remoção de comentários dos parâmetros *auth_db* no arquivo *ser.cfg*.

Estas linhas trabalham em conjunto. A primeira define que o *SER* deve gerar um *hash* com base no nome do usuário, a senha e o domínio (*realm*). A segunda define em que coluna da tabela do banco de dados, o *SER* deve procurar pela senha no formato texto claro.

As requisições *REGISTER* devem ser autenticadas, só permitindo o registro de usuários autorizados. Para que isto ocorresse, teve-se que remover o comentário das linhas no Quadro 32 e alterar conforme a necessidade do projeto, utilizando o domínio *voip1.pop-pe.rnp.br*.

```
if (!www_authorize("voip1.pop-pe.rnp.br", "subscriber")) {  
www_challenge("voip1.pop-pe.rnp.br", "0"); break; };
```

Quadro 32. Remoção de comentários dos parâmetros *auth_db* no arquivo *ser.cfg*.

Após o bloco do *if (method="REGISTER")*, foram adicionadas algumas regras de redirecionamento de discagem. Tomamos como prefixo para o grupo *voip1.pop-pe.rnp.br* o prefixo *08132720[0-9]{3}*, onde o "3272" está relacionado com o grupo e o número "0" do final do prefixo indica que o número pertence a um cliente utilizando o protocolo *SIP*. O grupo *voip2.pop-pe.rnp.br* terá o prefixo *08132730[0-9]{3}* para protocolo *SIP*. Para os clientes *H.323*, serão utilizados os prefixos *08132721[0-9]{3}* e *08132731[0-9]{3}* onde o número "1" indica que tal número pertence a um cliente *H.323*. O "[0-9]{3}" especifica que poderão conter

quaisquer 3 números restantes no intervalo de 0 a 9. O prefixo 081 indica que o número corresponde ao estado de Pernambuco. Observe que poderíamos utilizar o prefixo **55** para indicar um número Brasileiro, mas os testes se detêm a ligações estaduais. O primeiro redirecionamento que foi adicionado pode ser visualizado no Quadro 33.

```
# Redirecionamento para o servidor SER de Voip2
if (uri=~"^SIP:0813273[0-9]{4}") {
    rewritehost("voip2.pop-pe.rnp.br");
    log("Redirecionamento para o servidor SER de
Voip2");
    forward(200.133.0.212,5060);
    break;
};
```

Quadro 33. Bloco de redirecionamento para o grupo *Voip2*.

Neste bloco, estamos redirecionando as chamadas feitas aos números 0813273[0-9]{4} para o servidor *SER* responsável por ele, ou seja, o servidor responsável por este prefixo (3273) encontra-se no *IP* 200.133.0.212 e porta 5060, que corresponde ao servidor *voip2.pop-pe.rnp.br*. Portanto a função *rewritehost* altera o *realm* (*voip1.pop-pe.rnp.br*) da *URI* para *voip2.pop-pe.rnp.br*. Uma mensagem de *log* (“Redirecionando para o servidor *SER* Voip2”) é registrada no arquivo de *log* gerado pelo *SER* e o comando *break* faz com que o programa não passe por outras regras vindas a seguir, pois apenas uma regra pode ser aplicada. Com relação à numeração 0813273[0-9]{4}, não importa se a ligação será feita para um servidor *SIP* ou *H.323*, o servidor apenas repassa para o outro servidor *SER*, e neste outro servidor serão tratadas ligações direcionadas à clientes *H.323*. As ligações realizadas de *SIP* para *H.323* são tratadas através do bloco presente no Quadro 34. Observe que todas as configurações estão sendo mostradas para o servidor *Voip1*.

```
# Redirecionamento de chamadas para clientes H.323
if (uri=~"^SIP:08132721[0-9]{3}") {
    log("Redirecionando para o canal H.323 do Asterisk");
    forward(0.0.0.0,5061);
    break;
};
```

Quadro 34. Bloco de redirecionamento para o canal *SIP* do *Asterisk*.

No Quadro 34, pode-se observar que a regra está relacionada ao grupo *Voip1*, ou seja, grupo local, pois temos o prefixo “3272”. Pode-se também identificar que as ligações são de clientes *SIP* para clientes *H.323*, pois o número contém o número “1” após o prefixo do grupo (3272). Portanto esta ligação será repassada para o *Asterisk*, que é o *gateway* que fará a intermediação entre protocolos *SIP* e *H.323*, ou seja, *SER* e *GnuGK*. Observe que a porta que o *Asterisk* está rodando para o canal *SIP* é a porta 5061. O *IP* 0.0.0.0 representa o *IP* local.

Para finalizar esta seção de regras, definiu-se mais uma regra para o redirecionamento de chamadas *SIP* para ramais de telefones convencionais. Embora esse projeto não abordasse a conexão da telefonia convencional com a telefonia *IP*, porque a placa *Digium* necessária não foi

conseguida, ainda assim colocou-se este bloco no Quadro 35 para entender o funcionamento básico.

```
# Redirecionamento de chamadas para clientes PBX
if (uri=~"^SIP:08132729[0-9]{3}") {
    rewritehost("voip1.pop-pe.rnp.br");
    log("Redirecionando para o canal PBX do Asterisk");
    forward(0.0.0.0,5061);
    break;
};
```

Quadro 35. Bloco de redirecionamento para clientes *PBX*.

Temos que todas as ligações feitas a um telefone contendo o número “9” do grupo local serão redirecionadas para o *Asterisk* local que conterà a placa e assim ligará para o ramal contido em [0-9]{3} e funcionará como *gateway* entre os dois mundos da telefonia. Um exemplo de ramal seria 9242, 9331 etc. Caso o ramal começar com 4 (quatro), então mudar-se-ia a regra para 08132724[0-9]{3}.

Terminado a configuração do *ser.cfg*, seria necessário configurar o *DNS*. As consultas *SIP* podem ser feitas utilizando o *DNS*, uma facilidade que permite o uso de servidores redundantes sem que seja necessária qualquer configuração no cliente. Isto pode ser obtido através do registro *DNS* do tipo *SRV*, disponível a partir do *BIND* 8.X. O formato do registro *SRV* está representado no Quadro 36.

_service._protocol	SRV	Priority	Weight	Port	Hostname
--------------------	-----	----------	--------	------	----------

Quadro 36. Formato de um registro *SRV*.

Por exemplo, pode ser definido o servidor *SIP* primário do domínio *voip1.pop-pe.rnp.br* através do registro descrito no Quadro 37.

_sip._udp	SRV	0	0	5060	servidor.voip1.pop-pe.rnp.br.
-----------	-----	---	---	------	-------------------------------

Quadro 37. Registro *SRV* no servidor *DNS* de *Voip1*.

Havendo mais de um servidor, o parâmetro *Priority* pode ser utilizado para selecionar os servidores mais prioritários, no caso os que tiverem o parâmetro *Priority* de menor valor. Quando os servidores possuírem a mesma prioridade, o parâmetro *Weight* define como será o balanceamento de carga entre os servidores. O arquivo de configuração de zona *voip1.pop-pe.rnp.br* (*/var/named/pri/dominio.zone*) ficou com a forma presente no Quadro 38.

```
$TTL 600
@           IN SOA  voip1.pop-pe.rnp.br. root.voip1.pop-pe.rnp.br. (
                2005041701 ; serial
                600   ; refresh
                300   ; retry
                1800  ; expire
                600   ; minimum
            )

            NS     voip1.pop-pe.rnp.br.
            TXT    'DOULOX'

localhost   A      127.0.0.1
voip1.pop-pe.rnp.br.  A      200.133.0.211
servidor    A      200.133.0.211
_sip._udp   SRV    0 0 5060 servidor.voip1.pop-pe.rnp.br.
www         CNAME  voip1.pop-pe.rnp.br.
```

Quadro 38. Arquivo de zona do servidor *DNS* de *Voip1*.

Observe que os *TTLs* estão com valores baixos, *TTLs* como *refresh*, *retry*, *expire*, *minimum* e *TTL*. Mais detalhes destes parâmetros serão apresentados no Capítulo 4. Estes valores mantiveram valores baixos apenas para testes, pois se poderia mudar de domínio a qualquer momento e os servidores *DNS* vizinhos estariam em pouco tempo atualizados desta mudança. No arquivo */etc/named.conf* foi inserido o bloco do presente no Quadro 39.

```
zone "voip1.pop-pe.rnp.br" {
    type master;
    file "pri/dominio.zone";
};
```

Quadro 39. Bloco adicionado ao arquivo */etc/named.conf*.

Para verificar se o *DNS* foi configurado corretamente com o registro *SRV*, o utilitário *dig* foi executado com o comando indicado no Quadro 40.

```
dig -t SRV _sip._udp.voip1.pop-pe.rnp.br. @127.0.0.1
```

Quadro 40. Utilizando o *dig* para verificar o registro *SRV* adicionado.

Observamos os resultados e eles estavam corretos, identificando assim o *IP* com autoridade sobre este domínio de acordo com os resultados obtidos. Tais resultados encontram-se no Quadro 41 e as linhas destacadas indicam o sucesso do resultado do comando *dig*.

```
;; ANSWER SECTION:
_SIP._udp.voip1.pop-pe.rnp.br. 86400 IN SRV 0 0 5060
servidor.voip1.pop-pe.rnp.br.

;; AUTHORITY SECTION:
voip1.pop-pe.rnp.br. 86400 IN NS voip.pop-pe.rnp.br.
voip1.pop-pe.rnp.br. 86400 IN NS rosa.pop-pe.rnp.br.

;; ADDITIONAL SECTION:
servidor.voip1.pop-pe.rnp.br. 86400 IN A 200.133.0.211
rosa.pop-pe.rnp.br. 86397 IN A 200.133.0.35
voip1.pop-pe.rnp.br. 86400 IN A 200.133.0.211
```

Quadro 41. Resposta ao comando *dig* no registro *SRV* do servidor.

Após todas as configurações acima apresentadas, seria necessário adicionar os usuários para testes serem realizados. O comando para adicionar um usuário pode ser visualizado no Quadro 42.

```
serctl add <usuário> <senha> <email>
```

Quadro 42. Sintaxe do comando *serctl* para adição de novos usuários.

Os usuários poderiam ter como nome de usuário um valor alfanumérico. Mas alguns problemas foram obtidos utilizando este modo. Não havendo sucesso na identificação da causa deste problema, todos os usuários passaram a ter apenas valores numéricos. Caso conseguíssemos utilizar valores alfanuméricos, seria necessário utilizar o comando para adicionar *aliases*, ou seja, todo usuário teria associado a ele um “alias”, que no caso seria um número de telefone. A sintaxe deste comando encontra-se no Quadro 43.

```
serctl alias add <alias> <uri>
```

Quadro 43. Sintaxe do comando *serctl* para adição de *aliases*.

Um exemplo dos procedimentos citados acima pode ser encontrado no Quadro 44. Neste, associamos um “alias” 08132720001 ao usuário virtual1.

```
root@servidor:~# serctl add virtual1 virtual1p virtual1@voip1.pop-pe.rnp.br
MySQL password:
new user added
new user into uri table added
root@servidor:~# serctl alias add 08132720001 sip:virtual1@voip1.pop-pe.rnp.br
```

Quadro 44. Exemplo do uso do *serctl*.

Mas, como não foi possível utilizar a questão do *alias*, por estar obtendo erros ao verificar o *alias* no banco de dados, os usuários foram adicionados apenas como apresentado no Quadro 45, utilizando a senha *heslo* do usuário *ser*. Neste caso o nome de usuário seria 08132720001 ao invés de apenas virtual1 como apresentado no Quadro 44.

```
root@servidor:~# serctl add 08132720001 virtual1p virtual1@voip1.pop-pe.rnp.br
MySQL password:
new user added
new user into uri table added
```

Quadro 45. Adição de um usuário com valores numéricos ao SER.

Uma conta para o administrador foi adicionada para que o programa **SERWEB** pudesse funcionar corretamente. O **SERWEB**, programa para acesso de clientes, a fim de obter informações de ligações e duração, não foi instalado, mas ainda assim foi adicionada tal conta para trabalhos futuros. O administrador foi criado com o comando presente no Quadro 46.

```
root@servidor:~# serctl add administrador voipp admin@voip1.pop-pe.rnp.br
MySQL password:
new user added
new user into uri table added
```

Quadro 46. Adição do usuário administrador ao SER.

Com isto finalizamos a configuração do servidor **SER**. Um *script* (*/etc/rc.d/rc.ser*) desenvolvido para gerenciamento do **SER**, pode ser utilizado para iniciá-lo com o comando */etc/rc.d/rc.ser start*, pará-lo (*stop*) ou reinicia-lo (*restart*).

3.7 Configuração do *GnuGK*

O *GnuGK* funciona normalmente com uma configuração bastante simples. Para adicionar autenticação por usuário ou *IP*, alguns procedimentos são necessários. Para autenticação por usuário, foi necessário compilar o utilitário *addpasswd*. Antes de compilar o *GnuGK*, entrou-se no diretório fonte do *GnuGK* e foi executado o comando *make addpasswd*. Este comando gerou um erro de compilação, podendo ser visualizado no Quadro 47.

```
/usr/lib/libz.so: undefined reference to `errno'
```

Quadro 47. Erro ao compilar o utilitário *addpasswd*.

Após pesquisas realizadas, o pacote *zlib1g* do *Debian* foi obtido e transformado em um pacote *Slackware* (*tgz*) utilizando o *Alien*. Tal pacote, o *zlib_1.2.2.orig.tar.gz*, foi compilado e instalado. Alguns *links* simbólicos foram criados com relação ao módulo *libz.so* em */usr/lib* e

conseguiu-se compilar o *addpasswd* utilizando novamente o comando *make addpasswd*. Inseriu-se tal utilitário no pacote (*tgz*) do *GnuGK*. O comando apresentado no Quadro 48 pode ser utilizado para criar novos usuários ao *GnuGK*.

```
addpasswd /etc/gnugk.ini SimplePasswordAuth virtual3 virtual3password
```

Quadro 48. Erro ao compilar o utilitário *addpasswd*.

O usuário *virtual3* possuirá a senha *virtual3password* e tais informações serão armazenadas na seção *[SimplePasswordAuth]* no arquivo de configuração do *GnuGK*, o */etc/gnugk.ini*. Como não foi possível fazer o telefone *IP Cisco* funcionar com este tipo de autenticação (conseguiu-se apenas com telefones virtuais), configurou-se autenticação por *IP* a princípio apenas (*AliasAuth*), comentando a linha do *SimplePasswordAuth*, ficando esta parte da configuração como visualizado no Quadro 49.

```
[Gatekeeper::Auth]
#SimplePasswordAuth=required;RRQ,ARQ
AliasAuth=required;RRQ
default=allow

[RasSrv::RRQAuth]
asterisk=allow
virtual3=sigIP:200.133.0.209
phone1=sigIP:200.133.0.210
```

Quadro 49. Configuração do *gnugk.ini* com relação à autenticação.

Neste exemplo, apenas os telefones *phone1* e *virtual3* poderão se registrar no *gatekeeper* (*GnuGK*). Cada servidor *gatekeeper* possui um nome, portanto colocou-se o nome *VOIP1GK* no *gnugk.ini* na seção *[Gatekeeper::Main]* como indicado no Quadro 50.

```
[Gatekeeper::Main]
Fourtytwo=42
Name=VOIP1GK
UseBroadcastListener=0
TimeToLive=300
StatusPort=7000
```

Quadro 50. Configuração do *gnugk.ini* com relação ao nome do *GK*.

No Quadro 51, mais especificamente na seção *[RasSrv::Neighbors]*, foi colocado um *IP* do *DGK* (*Directory Gatekeeper*), *gatekeeper* que autenticará os prefixos de todos os estados e instituições de uma rede, ou seja, o *gatekeeper* de maior nível na hierarquia dos *gatekeepers*. Pelo fato da rede utilizada ser o da RNP, o *DGK* do PoP-RJ do Rio de Janeiro foi adicionado e também o endereço local do *Asterisk*, como este funcionando não como *DGK*, mas, sim, como *GK* adicional para tratamento de ligações *SIP* ou *PBX*. Desse modo, todas as ligações direcionadas ao prefixo 08132720 ou 08132729 seriam encaminhados do *gatekeeper* para o *Asterisk* que encaminharia para o servidor *proxy SER* local onde trataria se a ligação seria para

algum usuário registrado no próprio servidor *SER* ou para telefones ramais. Observe que ligações para o grupo Voip2 (0813273) também são redirecionados da mesma forma. Tal configuração encontra-se no Quadro 51.

```
[RasSrv::Neighbors]
UFRJGK=146.164.247.202;*
Asterisk=200.133.0.211:1620;08132720,08132729,0813273
```

Quadro 51. Configuração do *gnugk.ini* com relação ao vizinhos.

Todos os prefixos, portanto, que não sejam locais e nem direcionados ao *Asterisk* serão redirecionados ao DGK (*UFRJGK*). Não foi configurado o serviço *VoIP* com telefones convencionais, como mencionado anteriormente, mas a seção presente no Quadro 52 bloquearia as chamadas a telefones celulares.

```
[PrefixAuth]
55819=deny ipv4:ALL
819=deny ipv4:ALL
818=deny ipv4:ALL
55818=deny ipv4:ALL
ALL=allow ipv4:ALL
```

Quadro 52. Bloqueio de telefones celulares.

Com isto está finalizada a configuração do *GnuGK*. Um arquivo *Shell Script* (*/etc/rc.d/rc.gnugk*) foi desenvolvido para iniciar o *GnuGK* durante o *boot* do sistema. Com este *Script* pode-se iniciar o *GnuGK* utilizando o comando */etc/rc.d/rc.gnugk start*, parar (*stop*) e reiniciar (*restart*).

3.8 Configuração do *Asterisk*

Nesta seção abordaremos a configuração do servidor *Asterisk*, servidor que será responsável pela função, em software, de um *gateway* e uma central telefônica, sendo este último não abordado por completo no escopo deste capítulo.

3.8.1 Canal *SIP*

Para a configuração do canal *SIP* foi necessário a edição de dois arquivos, o */etc/asterisk/sip.conf* e o */etc/asterisk/extensions.conf*. O canal *SIP* será responsável por redirecionar as chamadas originadas de clientes *SIP* para outros canais, tais como *H.323* ou *PBX*. No primeiro arquivo é definido como o servidor *SIP* irá operar, e no segundo, o plano de discagem que irá determinar como as chamadas *SIP* serão encaminhadas.

Na seção *[general]* do *sip.conf* é definido como será o comportamento do servidor *SIP*. Nesse arquivo foi criado um contexto onde serão tratadas as chamadas recebidas. A configuração desta seção pode ser visualizada no Quadro 53.

```
[general]
context=chamadas-a-encaminhar

port=5061
bindaddr=0.0.0.0
srvlookup=yes

disallow=all
allow=ulaw
allow=alaw
```

Quadro 53. Seção *[general]* do arquivo *sip.conf*.

Analisando tal seção, o contexto “*chamadas-a-encaminhar*”, será encontrado no arquivo *extensions.conf*, onde serão tratadas as regras de redirecionamento de chamadas *SIP* e *H.323*. O servidor *SIP* estará funcionando na porta 5061, pois o *SER*, que também será o servidor *proxy SIP*, já utiliza a porta 5060. Estão habilitadas as opções *srvlookup* (*DNS*) e os *codecs ulaw* (*G711U*) e *alaw* (*G711A*). O canal *SIP* do *Asterisk* utilizará um usuário para registrar-se no servidor *SIP* do *SER*, para que seja estabelecido o canal. Portanto um usuário chamado *Asterisk* (08132720999) com senha *voip* foi criado no *SER* e foi adicionado à seção *[general]*, ainda em *sip.conf*, as linhas de acordo com o Quadro 54, para que o *Asterisk* possa efetuar o registro com sucesso no servidor *SER*.

```
[08132720999]
type=friend
username=08132720999
secret=voip
host=dynamic
mailbox=08132720999
```

Quadro 54. Seção de autenticação do canal *SIP* em *sip.conf*.

Definiu-se também, no arquivo *sip.conf*, ainda na seção *[general]*, como serão encaminhadas as chamadas destinadas aos clientes *SIP*, especificando o endereço do servidor *SER* de acordo com o Quadro 55.

```
[voip1.pop-pe.rnp.br]
type=friend
host=200.133.0.211
username=SER
```

Quadro 55. Seção de identificação do servidor *SER* local.

O arquivo *extensions.conf* foi editado e inserido algumas linhas nas seções *[globals]* e *[default]* de acordo com o Quadro 56.

```
[globals]
GRUPO => voip1.pop-pe.rnp.br

[default]
exten => _,1,Congestion

[chamadas-a-encaminhar]
include => to-sip

[to-sip]
exten => _0813272[02]xxx,1,Dial(SIP/${EXTEN}@${GRUPO},20,tT)
exten => _0813272[02]xxx,2,Hangup
exten => _0813273xxxx,1,Dial(SIP/${EXTEN}@${GRUPO},20,tT)
exten => _0813273xxxx,2,Hangup
```

Quadro 56. Definição do grupo e das regras de redirecionamento em *extensions.conf*.

Estas linhas especificam o plano de discagem. Caso nenhuma regra seja aplicada a uma chamada, o resultado da chamada será um sinal de congestionamento (*exten => _,1,Congestion*). Uma seção *[chamadas-a-encaminhar]* foi adicionada onde colocou-se as regras de discagem para clientes *SIP*. A seção *[to-sip]* define as regras de redirecionamento para o canal *SIP*. As duas primeiras linhas (*exten*) definem que se o número da chamada de destino contiver os dígitos 0813272[02] (0 – *SIP*, 2 - *PBX*), ou seja, a ligação está sendo direcionada a algum cliente *SIP* local, tal chamada é redirecionada para o servidor *SER* local. Caso o *Asterisk* não consiga encaminhar tal ligação, ou o cliente *SIP* esteja ocupado, este enviará ao cliente que está fazendo a ligação um sinal de *Hangup*, ou seja, indisponível. As outras duas linhas definem que se a ligação for direcionada a outro grupo, o *Voip2*, contendo os dígitos 0813273, tal ligação será encaminhada para o servidor *SER* local, que conterà outro redirecionamento para o servidor *SER* responsável por este prefixo, ou seja, o servidor *SER* do servidor *Voip2*. Com estas configurações aplicadas, foi iniciado o *SER* (*/etc/rc.d/rc.ser restart*) e ligações foram realizadas, apenas para teste, entre o usuário *asterisk* registrando-se no *Asterisk* e o usuário criado *virtual1* (08132720001). Os testes foram bem sucedidos, e foi finalizada assim a configuração deste canal.

3.8.2 Canal H.323

Primeiramente foi necessário compilar e instalar o pacote *Asterisk-oh323*. A recompilação do programa *OpenH323* foi necessária, mas desta vez com uma atualização através de um patch vindo no pacote *Asterisk-oh323*. As versões mais recentes do *OpenH323* e *PWLib* foram baixadas como indicado no manual. As versões foram o *openh323-v1_13_5-src-tar.gz* e *pwlib-v1_6_6-src.tar*.

O patch foi aplicado utilizando o comando indicado no Quadro 57, dentro do diretório fonte do *OpenH323*.

```
$OPENH323DIR # patch -p1 < /root/Asterisk-oh323-0.7.1/OpenH323_1.13.5-make.patch
```

Quadro 57. Aplicação de um *patch* ao fonte do *OpenH323*.

Muitos erros de compilação foram relatados quando o *Asterisk-oh323* foi compilado, após a compilação do *OpenH323* já com o *patch* aplicado. Erros como os apresentados no Quadro 58.

```
gcc -Wall -pipe -Wall -Wstrict-prototypes -Wmissing-prototypes -Wmissing-declarations -D_REENTRANT -D_GNU_SOURCE -I/root/asterisk-1.0.5/include/asterisk -I../wrapper -g -c -o chan_oh323.o chan_oh323.c
In file included from /usr/include/string.h:33,
      from chan_oh323.c:34:
/usr/lib/gcc-lib/i486-slackware-linux/3.3.5/include/stddef.h:213: error: syntax error before "typedef"
In file included from chan_oh323.c:34:
/usr/include/string.h:38: error: syntax error before "extern"
/usr/include/string.h:39: error: parse error before "__THROW"
/usr/include/string.h:43: error: parse error before "__THROW"
/usr/include/string.h:56: error: parse error before "__BEGIN_NAMESPACE_STD"
/usr/include/string.h:58: error: syntax error before "extern"
/usr/include/string.h:58: error: parse error before "__THROW"
/usr/include/string.h:62: error: parse error before "__THROW"
```

Quadro 58. Erros obtidos na compilação do *Asterisk-oh323*.

Bastantes versões do *Asterisk-oh323* foram compiladas e instaladas, mas simplesmente não se tinha sucesso. Outras versões do GCC e versões do *OpenH323* e *PWLib* foram compiladas, mas ainda assim erros semelhantes ao acima apareciam. Pesquisas foram realizadas e não se conseguiu identificar o problema principal. Desconfiava-se de que a versão 10 do *Slackware* era o problema, portanto a versão 9.1 do *Slackware* foi instalada. Versões mais recentes do do *PWLib* e *OpenH323* foram baixadas e foi possível a recompilação, com sucesso, do pacote *Asterisk-oh323* versão 6.5, pois com a versão 7.1 erros eram obtidos. Os pacotes utilizados foram os citados no Quadro 59.

```
Pwlib 1.6.6
Openh323 1.13.5
Asterisk-oh323-6.5
```

Quadro 59. Programas compilados para o funcionamento do *Asterisk-oh323*.

O *GnuGK* no *Slackware* 10 foi novamente compilado e instalado, utilizando o *PWLib* 1.5.2 e *OpenH323* 1.12.2, pois com as versões do Quadro 59, erros de *segmentation fault* (erros de memória, provavelmente) eram encontrados ao ser iniciado o *GnuGK*. Portanto o *Doulox* possui as duas versões instaladas do *PWLib* e *OpenH323*, a primeira sendo utilizadas pelo *Asterisk-oh323* e a última sendo utilizado pelo *GnuGK*.

Para a configuração deste canal, precisaram-se editar dois arquivos. O */etc/asterisk/oh323.conf* e o */etc/asterisk/extensions.conf*. Adicionou-se ao primeiro o conteúdo do Quadro 60.

```
[general]
listenAddress=0.0.0.0
listenPort=1620
h245Tunnelling=yes
gatekeeper=200.133.0.211
context=chamadas-a-encaminhar

[register]
context=chamadas-a-encaminhar
alias=asterisk
gwprefix=55
gwprefix=08132720
gwprefix=08132729
gwprefix=0813273
codec=G711A
frames=20
codec=G711U
frames=20
```

Quadro 60. Configurações no arquivo *oh323.conf*.

O servidor *Asterisk* iniciará este servidor na porta 1620 no *IP* local (*0.0.0.0*), onde está especificado no arquivo *gnugk.ini* tal porta. Adicionou-se o *IP* do *gatekeeper*, habilitou-se a opção *h245Tunneling* para que chamadas no modo roteadas sejam aceitas, e definiu-se o contexto onde serão tratadas as regras de discagem no arquivo *extensions.conf*. Na seção *[register]* descreveu-se novamente o contexto, definiu-se um “alias” que será registrado no *GK*, onde este já foi autorizado na configuração do *gatekeeper*, definiu-se que todos os prefixos iniciados com 55, 08132720, 08132729 e 0813273 (*Voip2*) sejam redirecionados para o *Asterisk* e que os *codecs* de áudio aceitos para a conversação são o *G711A* e *G711U*. Em seguida, configurou-se o arquivo */etc/asterisk/extensions.conf* e adicionou-se as linhas presentes no Quadro 61.

```
[chamadas-a-encaminhar]
include => to-sip
include => to-h323

[to-h323]
exten => _08132721xxx,1,Dial(OH323/${EXTEN},20)
exten => _08132721xxx,2,Hangup
```

Quadro 61. Configurações de redirecionamento no arquivo *extensions.conf* quanto ao canal *H.323*.

A linha (*include => to-sip*) foi adicionada durante a configuração do canal *SIP*. Na seção *[to-h323]*, as regras indicam que qualquer ligação realizada para o prefixo 08132721xxx, ou seja, para clientes *H.323* locais, serão encaminhadas para o *gatekeeper*. Caso a regra falhe, a ligação será interrompida (*Hangup*). Por fim o *Asterisk* foi reiniciado (*/etc/rc.d/rc.asterisk restart*) e

ligações foram realizadas e testadas entre clientes *SIP* e *H.323*, obtendo sucesso nas ligações entre ambos os protocolos.

Os arquivos e diretórios necessários para a instalação e configuração de todo o *Doulox*, incluindo o sistema de gerenciamento, encontra-se disponível no *CD-ROM*. A partir destes arquivos foi possível executar o *install.sh*. Um arquivo chamado *doulox.tar.gz* pode também ser encontrado no *CD-ROM*, que contém todos estes arquivos compactados, incluindo o *install.sh*.

3.9 Instalação do *Doulox*

Como mencionado anteriormente, o *Doulox* é uma distribuição *Linux Live*, ou seja, pode ser iniciada direto do CD. Porém, pode-se instalar tal distribuição no *HD* utilizando o utilitário *slax-install* ao ser iniciada a distribuição.

Para utilizar o *slax-install*, é necessário que seja iniciada a interface gráfica do *Doulox*, o *KDE* [41]. Inicialmente é apresentada uma tela de *login* onde serão fornecidos usuário e senha. Utilizamos um usuário e senha padrão, sendo estes *root* e *doulos* respectivamente, como apresentado em uma tela de boas vindas ao iniciar o CD. Em seguida deve-se executar o comando *startx* para iniciar o *KDE*. Ao entrar no sistema, deve-se iniciar um terminal e utilizar o utilitário *cfdisk* (*curses based disk partition table manipulator for Linux*) [42] para particionar o HD presente no computador.

Após adicionar uma partição raiz (*/*) com partição do tipo *EXT3* e uma *swap*, ambos utilizando o *cfdisk* como utilitário, deve-se formatar tais partições utilizando o utilitário *mkfs.ext3* para formatar a partição raiz. Se o HD estiver na primeira IDE e for primário (não escravo), deve-se utilizar o comando *mkfs.ext3 /dev/hda1* para formatar a partição raiz. Feito isto, é necessário montar tal partição, criando primeiramente um diretório *hda1* em */mnt* com o comando *mkdir /mnt/hda1*. Em seguida, deve-se montar a partição formatada em tal diretório, com o comando *mount /dev/hda1 /mnt/hda1*. Para finalizar, utiliza-se o utilitário *slax-install* para instalar todo o sistema. Este pedirá apenas a partição a instalar o sistema, e no caso do exemplo citado nesta seção, deve-se escolher a partição */dev/hda1* que provavelmente será a única. Após todo o sistema ser instalado, o computador deverá ser reiniciado e o CD removido da bandeja para que o sistema inicie sem a necessidade deste, finalizando assim a instalação do *Doulox*.

Capítulo 4

Sistema de gerenciamento do *Doulox*

Para a configuração de todos os servidores necessários para a utilização dos serviços de voz sobre *IP*, foi necessário editar vários arquivos de configuração. Um programa foi desenvolvido para um melhor gerenciamento destes arquivos de configuração através de uma interface gráfica, que facilitou bastante o gerenciamento dos servidores, permitindo criar um ambiente *VoIP* entre duas ou mais redes em poucos minutos.

4.1 Tecnologia

Para o desenvolvimento de tal programa, foram necessárias as tecnologias *PHP*, linguagem de programação para *Web*, *HTML* (utilizando *tableless*) [43], *CSS* [44], *JavaScript* [45], *Shell Script* (linguagem de *scripts* em ambientes *Linux*), *Apache* [46] como servidor *HTTP*, suporte ao *PHP* e o banco de dados *MySQL*. Utilizou-se tal banco pelo fato do *SER* já utilizá-lo, portanto procurou-se manter homogeneidade. Alguns casos de uso foram criados baseados em *UML* [47], para explicar o funcionamento do sistema, sendo estes apresentados nas próximas seções.

4.2 Autenticação

Para iniciar o sistema, é necessário iniciar a interface gráfica do *Linux* (*KDE*), e utilizar o navegador padrão deste, o *konqueror*. Para abrir o sistema, deve-se digitar o endereço *http://localhost/* e a tela de autenticação será exibida. O sistema está controlado por autenticação de usuário. A princípio, apenas um usuário foi criado para o sistema, o usuário *admin*. Outros usuários não foram adicionados, pois inicialmente, somente o usuário administrador do sistema terá acesso a este sistema. Portanto, para o usuário *admin*, uma senha padrão foi definida, a senha "*Doulox*". É importante lembrar de que detalhes de segurança avançada não foram desenvolvidas neste projeto, deixando esta preocupação para projetos futuros. Portanto, um usuário apenas e uma senha padrão foram utilizados para testes do sistema. A tela de autenticação pode ser observada na Figura 16.

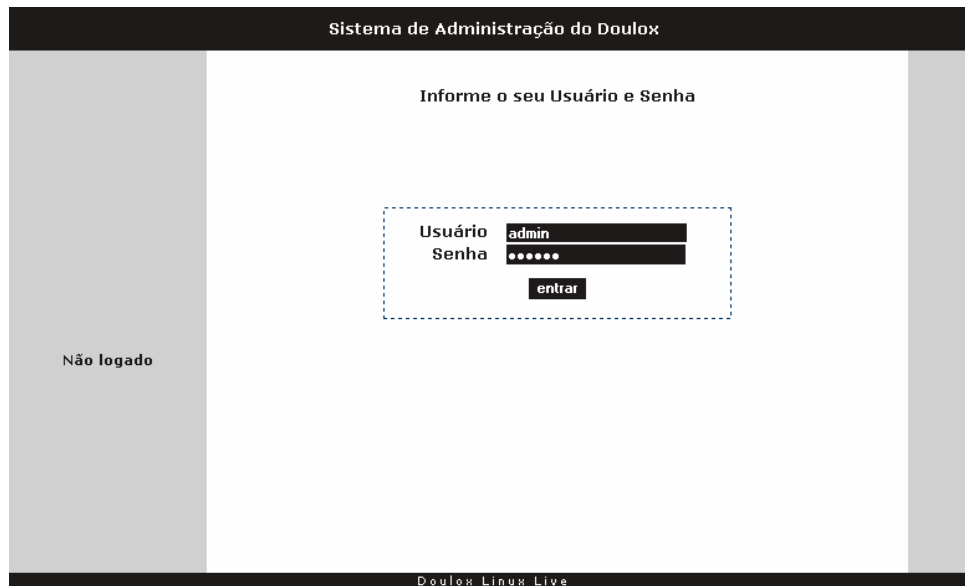


Figura 16. Tela de autenticação do usuário.

Verificando a figura, observamos que existe uma mensagem de “não logado” localizado à esquerda da tela. Esta mensagem indica que o usuário não está autenticado, pois foram utilizadas as técnicas de sessões em *PHP* para prover certa segurança com relação à autenticação. Todas as páginas do programa serão autenticadas utilizando o nome de usuário e senha passados no início. Para alterar tal senha, deve-se alterar direto utilizando o terminal do banco de dados *MySQL*, uma vez que o sistema não contém a funcionalidade de gerenciamento de usuários. Após a autenticação, uma página de boas vindas é exibida e um menu é apresentado, contendo todas as funcionalidades do sistema como exibidos na Figura 17.

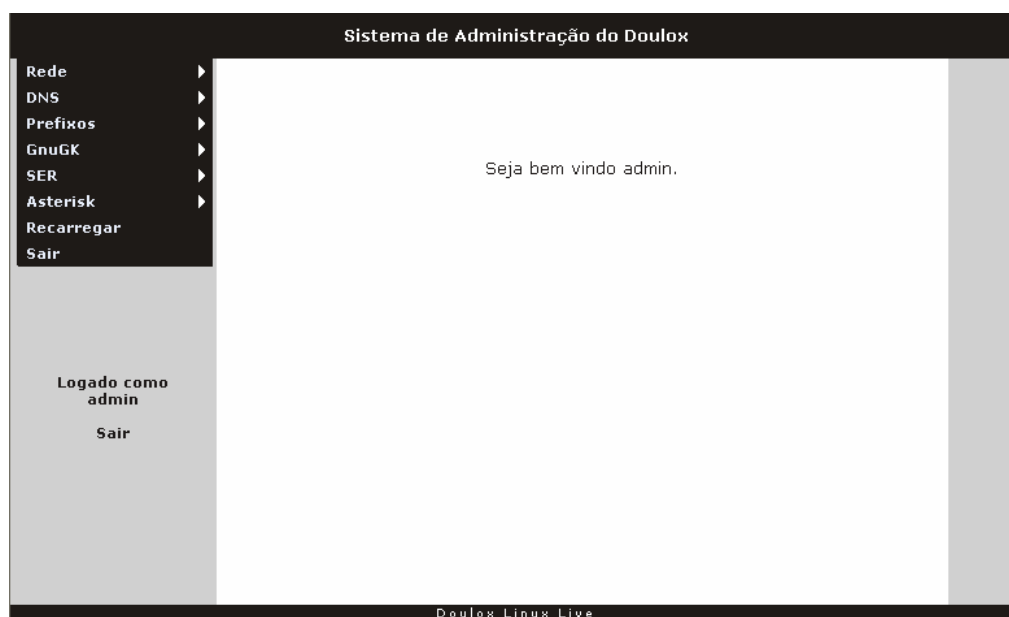


Figura 17. Tela de boas vindas e menu.

Pode-se então observar que uma nova mensagem foi mostrada à tela, a mensagem “Logado como admin” e “Sair”. Esta indica o usuário atualmente autenticado e um link “Sair” caso o usuário não utilize mais o sistema. O mesmo link pode ser encontrado no menu principal. A seguir veremos em detalhes o funcionamento das outras seções do menu.

4.3 Rede, DNS e prefixos locais

As três primeiras opções do menu são “Rede”, “DNS” e “Prefixos”. As suas funcionalidades podem ser representadas em um diagrama de caso de uso definida na Figura 18.

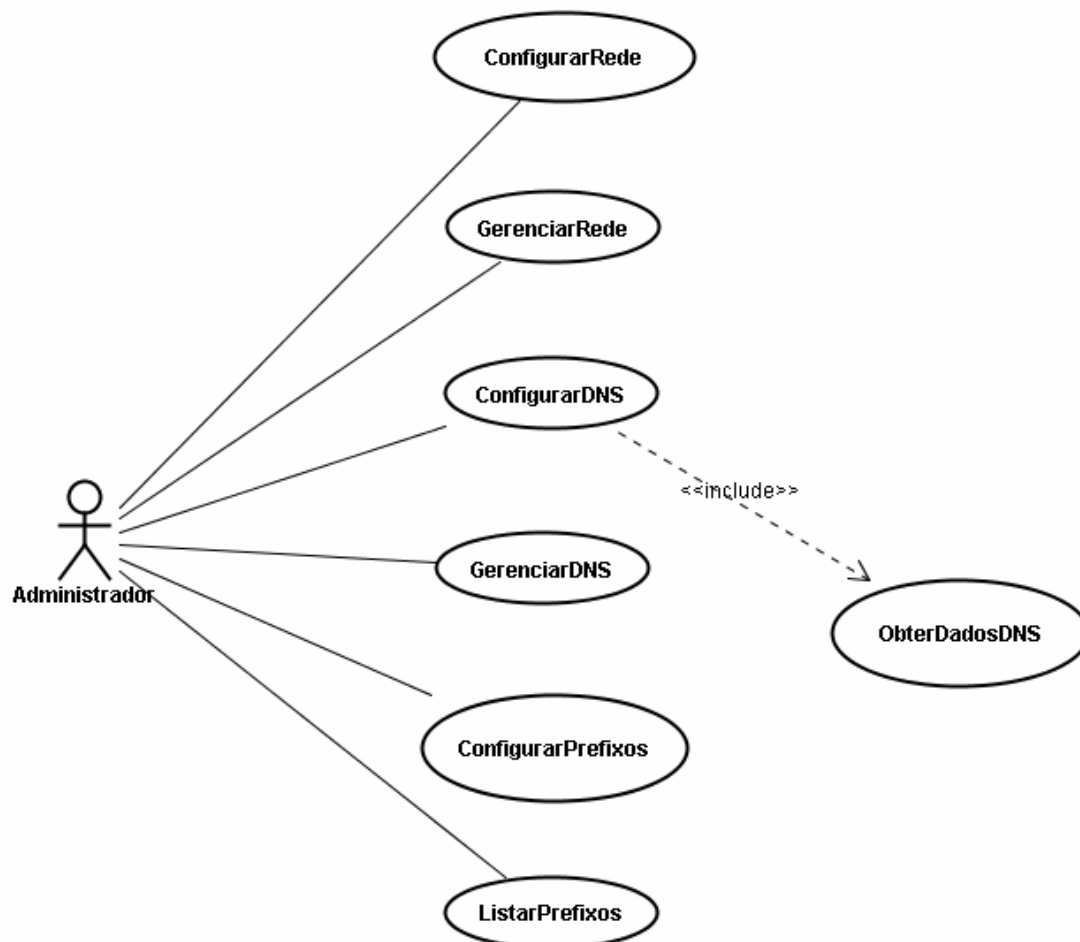


Figura 18. Diagrama de casos de uso das opções Rede, DNS e Prefixos.

O usuário único e principal, o administrador (*admin*), poderá configurar a rede do *Linux* (*ConfigurarRede*), inserindo dados como *IP*, máscara e *gateway*, como mostra o exemplo da Figura 19. Esta funcionalidade pode ser acessada através do *submenu* Rede=>Configurar como ilustrada na Figura 20. O administrador poderá também reiniciar, parar, listar ou iniciar o serviço de rede, através do *submenu* Rede=>Gerenciar (*ConfigurarRede*). Todas as funcionalidades acima ilustradas são autenticadas através da inclusão do caso de uso *ValidarUsuário*. Este é uma generalização do caso de uso *ChecarSenha*.

Configurar Rede

IP: 200.133.0.211

Máscara: 255.255.255.0

Gateway: 200.133.0.45

Configurar

Figura 19. Tela de configuração da rede.

Rede	Configurar
DNS	Gerenciar
Prefixos	
GnuGK	
SER	
Asterisk	
Recarregar	
Sair	

Figura 20. Submenus da opção Rede.

Na opção DNS, o administrador poderá configurar o servidor *DNS (BIND)*, através desta interface. Esta opção pode ser encontrada no *submenu DNS=>Configurar (ConfigurarDNS)*(Figura 21). Pode-se também reiniciar, parar, listar e iniciar o servidor *DNS* no *submenu DNS=>Gerenciar (GerenciarDNS)*. Esta funcionalidade requer uma obtenção dos dados do *DNS* pré-configurados, portanto há uma inclusão do caso de uso *ObterDadosDNS*.

Como ilustrado na Figura 22, pode-se identificar vários campos a serem preenchidos. No campo *Hostname* deve-se colocar o nome do servidor, no campo *Domínio* o domínio registrado correspondente do servidor e no *IP* o mesmo valor inserido no campo *IP* durante a configuração da rede, ou seja, deve-se inserir o valor *IP* do servidor. Os campos a seguir são detalhes de configuração de servidores DNS, portanto não nos aprofundaremos em tais campos. O *Serial* conterá um valor numérico qualquer, sendo este responsável pela atualização dos servidores *DNS* escravos ou secundários, que não serão abordados no escopo deste trabalho. Uma vez que este é alterado para um valor maior do que o corrente, o servidor *DNS* escravo será informado que houve alterações nos arquivos de configuração do *DNS* primário e atualizará o seu cachê, ou seja, o seu banco contendo informações de servidores *DNS*. Os administradores utilizam o seguinte padrão para esta variável: ano+mês+dia+contador. Seguindo o exemplo, a configuração foi executada no ano 2005, mês 04, dia 17 e o contador é 01, pois corresponde à primeira atualização. Caso seja alterado qualquer valor nesta seção, o valor do serial terá que ser alterado para 2005041702 e assim por diante, como um contador.

Os campos de TTL são os *Refresh*, *Retry*, *Expire*, *Minimum* e *TTL*. Estes possuem valores em segundos. O *Refresh* indica aos servidores secundários o intervalo de tempo que estes levarão para atualizar a sua cachê. Caso os servidores secundários não consigam atualizar seus dados com o servidor primário que estamos configurando, estes tentam novamente atualizar a cachê com o valor em segundos do campo *Retry*. Estes continuam tentando até completar o número de

segundos identificados no campo *Expire*. O campo *Minimum* especifica aos servidores *DNS* que contêm na sua cachê os dados do servidor *DNS* que estamos configurando, que tais dados serão válidos pela quantidade de segundos identificada neste campo. Caso este tempo seja atingido, os servidores buscarão novamente os dados no servidor. O valor do campo *TTL* servirá para os registros de subdomínios caso não contenham o *TTL* especificado. Este identifica que certos subdomínios serão válidos por *TTL* segundos. Colocou-se como padrão valores muito pequenos dos *TTLs*, uma vez que esta distribuição servirá para testes. Caso se deseje instalar tal distribuição e utilizá-la, devem-se aumentar tais valores. Neste caso, poderíamos colocar o valor 21600 (6 horas) para o *Refresh*, 3600 (1 hora) para o *Retry*, 1728000 (20 dias) para para o *Expire*, 21600 (6 horas) para o *Minimum* e 21600 (6 horas) para o *TTL*.



Figura 21. Submenus da opção “DNS”.

The image shows a form titled 'Configurar DNS' with the following fields and values:

- Hostname: servidor1
- Domínio: voip1.pop-pe.rnp.br
- IP: 200.133.0.211
- Serial: 2005041701
- Refresh (segundos): 600
- Retry (segundos): 300
- Expire (segundos): 1800
- Minimum (segundos): 600
- TTL (segundos): 600

At the bottom of the form is a 'Configurar' button.

Figura 22. Configuração do servidor *DNS*.

Após tais configurações, o administrador especificará os prefixos relacionados ao servidor através do *submenu* Prefixos=>Configurar (*ConfigurarPrefixos*) ilustrado na Figura 23. Um exemplo de configuração desta opção pode ser visualizado na Figura 24.



Figura 23. Submenus da opção “Prefixos”.

Neste exemplo, especificamos que o servidor será identificado como possuindo telefones com prefixo 0813272. Portanto todos os telefones com números iniciados por este prefixo serão atribuídos ao servidor que está sendo configurado. Utilizou-se um código de região, seguido pelo código do *PBX* da empresa ou departamento. Todos os telefones com prefixo 08132720 serão atribuídos aos telefones de protocolo *SIP*. O prefixo 08132721 será atribuído aos telefones utilizando o protocolo *H.323* e o 08132729 aos ramais da instituição. No primeiro campo pede-se para inserir o valor 0 (zero) no início. Isto porque ligações sem o 0 seriam encaminhadas para telefones convencionais. O tratamento dessas ligações não faz parte do escopo do nosso projeto. Tal funcionalidade foi deixada para trabalhos futuros.

Configurar Prefixos Locais

Código da região de sua cidade (para ligações internas à rede IP. Ex: 081. Importante utilizar o 0 no início como no exemplo.):

Código da central telefônica (Ex: primeiros quatro números do telefone):

Código identificador para ligações SIP (Ex: 0):

Código identificador para ligações H.323 (Ex: 1):

Código identificador para ligações PBX (Ex: 9):

Figura 24. Configuração dos prefixos do servidor.

O administrador poderá também listar os prefixos atuais através do *submenu* Prefixos=>Listar (*ListarPrefixos*).

4.4 GnuGK

O *GnuGK* possui a maior parte da configuração. Pode-se identificar as suas funcionalidades através do diagrama de casos de uso na Figura 25. A opção *ConfigurarNomeGnuGK* pode ser encontrada acessando o *submenu* GnuGK=>Configurar=>Nome enquanto que a opção *ConfigurarDGKGnuGK* no *submenu* GnuGK=>Configurar=>DGK. Um nome precisa ser

atribuído ao *gatekeeper* (GK) local. O DGK contém o nome e o IP do *directory gatekeeper*, servidor *gatekeeper* raiz de uma organização. Estes dados não são obrigatórios para ligações entre poucas empresas através do *Doulox*, apenas deve-se especificar o nome. Tal opção é mais indicada se a instituição já possui uma rede *H.323* em funcionamento, possuindo assim um DGK.

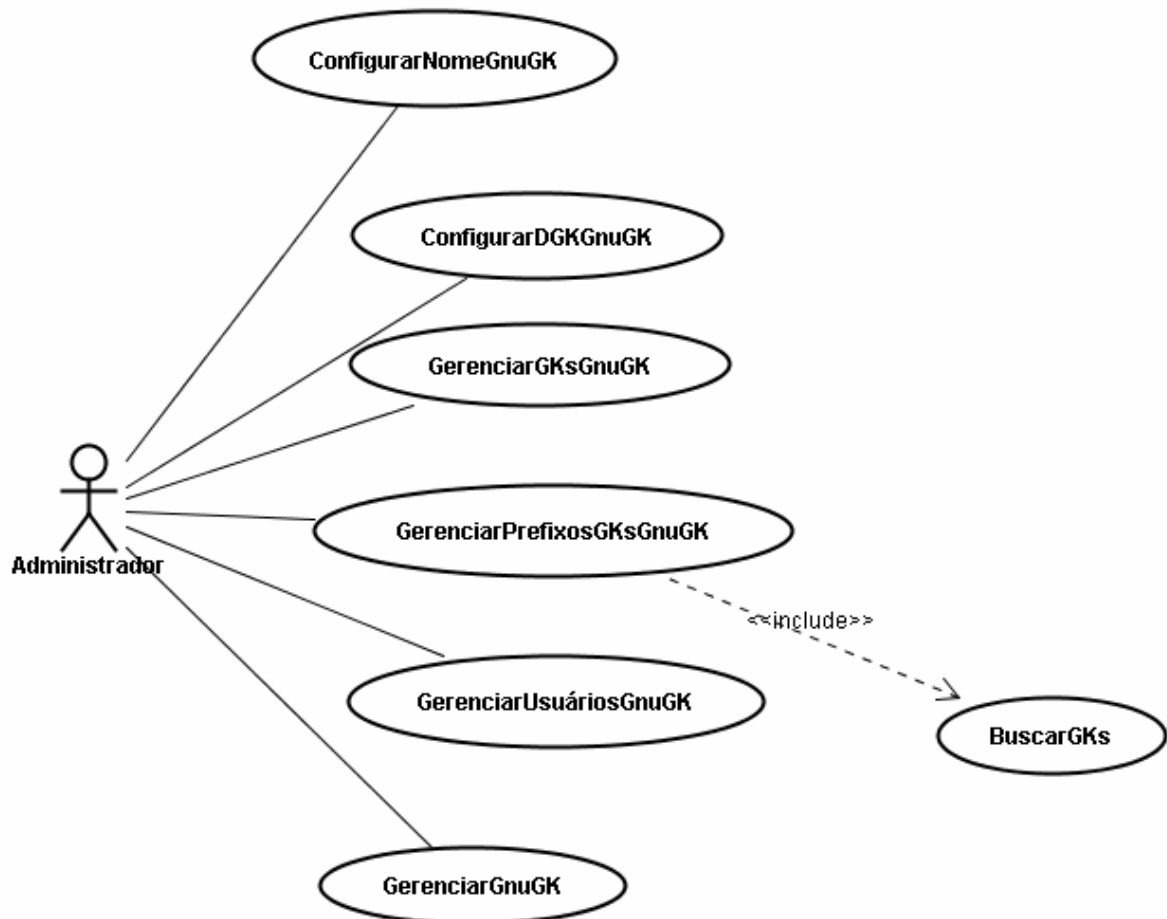


Figura 25. Diagrama de casos de uso do *GnuGK*.

O administrador poderá adicionar outros *gatekeepers* ao arquivo de configuração, identificando seus respectivos prefixos. Para adicionarmos os *gatekeepers*, pode-se utilizar o *submenu* GnuGK=>GKs=>Adicionar (*GerenciarGKsGnuGK*)(Figura 26).

GnuGK	Configurar	▶
SER	▶ GKs	Adicionar
Asterisk	▶ Prefixos	▶ Remover
Recarregar	Usuários	▶ Modificar
Sair	Gerenciar	Listar

Figura 26. Submenus da opção “GnuGK=>GKs”.

Pode-se também remover, modificar e listar tais GKs. Após a adição de um GK, pode-se associar prefixos a este, utilizando o *submenu* GnuGK=>Prefixos=>Adicionar (*GerenciarGKsPrefixosGnuGK*)(Figura 27).

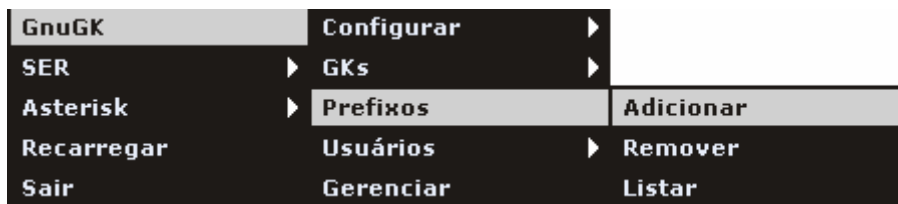


Figura 27. Submenus da opção “GnuGK=>Prefixos”.

Utilizamos as Figuras 28 e 29 como exemplo. Na Figura 28, estamos adicionando um *GK* que seria uma empresa qualquer. O nome (fictício) desta empresa foi adicionado como *EMPRESAXGK* e *IP* (fictício) *200.17.134.40*.

Adicionar novo GK para redirecionamento de chamadas

Nome do GK (não utilizar caracteres especiais ou espaços em branco):

IP do GK:

Figura 28. Adicionando um novo *GK*.

Digamos que um telefone de *Voip1*, como, por exemplo, o telefone com número 08132721001 esteja realizando uma ligação para esta empresa, começando com o prefixo 01132251, como identificado na Figura 29, esta chamada será encaminhada para a *EMPRESAXGK*. A idéia principal destas duas funcionalidades, é realizar o redirecionamento de chamadas.

Adicionar prefixos aos GKs

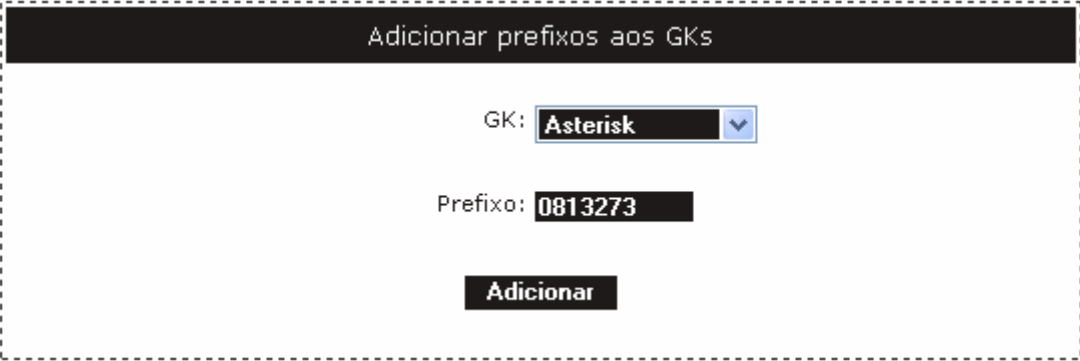
GK: ▼

Prefixo:

Figura 29. Associando um prefixo a um *GK*.

Um *GK* foi pré-adicionado ao sistema sistema de gerenciamento que não pode ser removido via próprio sistema. Este *GK* é o *Asterisk* que possui *IP* local. Nos testes, utilizamos os servidores *voip1.pop-pe.rnp.br* com prefixo 0813272 e *voip2.pop-pe.rnp.br* com prefixo 0813273. Na configuração do *Voip1*, foi adicionado ao *GK Asterisk* o prefixo 0813273 (Figura 30). Isto faz com que todas as ligações originadas de um cliente *H.323* do servidor *Voip1* para

clientes de *Voip2* sejam redirecionadas para o *Asterisk* local que se encarregará de redirecioná-las para o servidor *SER* local. Com configurações apropriadas deste servidor (*SER*) que veremos na Seção 4.5, tais chamadas são encaminhadas para o servidor *SER* da empresa *Voip2*. O servidor *SER* desta empresa verificará o prefixo novamente. Caso comece com 08132731, este redireciona para o *Asterisk* de *Voip2* que redireciona para o canal *H.323*, ou seja, para o *gatekeeper* de *Voip2* e assim é realizada a chamada. Se for para clientes *SIP*, ou seja, com prefixo 08132730, o servidor *SER* de *Voip2* envia a chamada ao cliente registrado no próprio servidor *SER* e estabelece a chamada. No servidor *Voip2* foram configurados os parâmetros semelhantes ao servidor *Voip2*, fazendo os devidos redirecionamentos. Estes redirecionamentos citados acima serão vistos mais à frente.



Adicionar prefixos aos GKs

GK: Asterisk

Prefixo: 0813273

Adicionar

Figura 30. Associando um prefixo a um *GK Asterisk* em *Voip1*.

Em *Voip2* utilizou-se a configuração apresentada na Figura 31. Todas as ligações destinadas ao prefixo 0813272 (*Voip1*) serão encaminhadas para *Voip1* através do *Asterisk*.



Adicionar prefixos aos GKs

GK: Asterisk

Prefixo: 0813272

Adicionar

Figura 31. Associando um prefixo a um *GK Asterisk* em *VoIP2*.

O gerenciamento dos *GKs* e prefixos não só envolve as opções adicionar, mas também o remover, modificar e listar. O administrador será capaz de gerenciar os usuários/telefones do ambiente (*AdicionarUsuáriosGnuGK*) através do *submenu* *GnuGK=>Usuários=>Adicionar* (Figura 32).

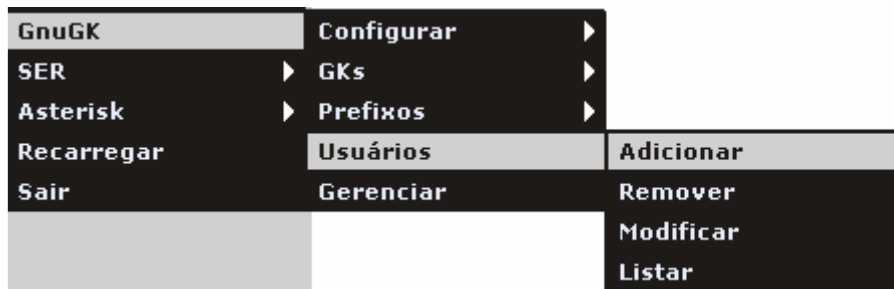


Figura 32. Submenus da opção “GnuGK=>Usuários”.

Um exemplo da adição de um usuário *H.323* em *Voip1* pode ser visto na Figura 33. O responsável por este telefone seria o Rodrigo, tendo este como nome de usuário *rodrigo1*, número [08132721001](#) e sendo o registro deste telefone restrito ao *IP 200.133.0.209*, ou seja, Rodrigo poderia apenas ligar deste *IP*. Se a opção “Liberado a qualquer *IP*” fosse marcada, Rodrigo poderia ligar de qualquer dispositivo com qualquer *IP*.

Adicionar novo usuário ao GK

Responsável:

Usuário:

Número (Ex: 001):

Permissão: Restrito ao IP:
 Liberado a qualquer IP

Figura 33. Adição de novo usuário/telefone ao servidor *GK* em *Voip1*.

Pode-se reiniciar, parar ou iniciar o *GnuGK* através do *submenu* *GnuGK=>Gerenciar*.

4.5 Opções *SER*, *Asterisk* e recarregar

O sistema possui também as opções *SER*, *Asterisk* e recarregar. No menu *SER* encontraremos opções para a configuração do *SER*, no *Asterisk* a opção de gerenciamento de redirecionamentos e a opção recarregar é utilizada para re-configurar e recarregar todos os serviços. O diagrama de casos de uso encontra-se na Figura 34.

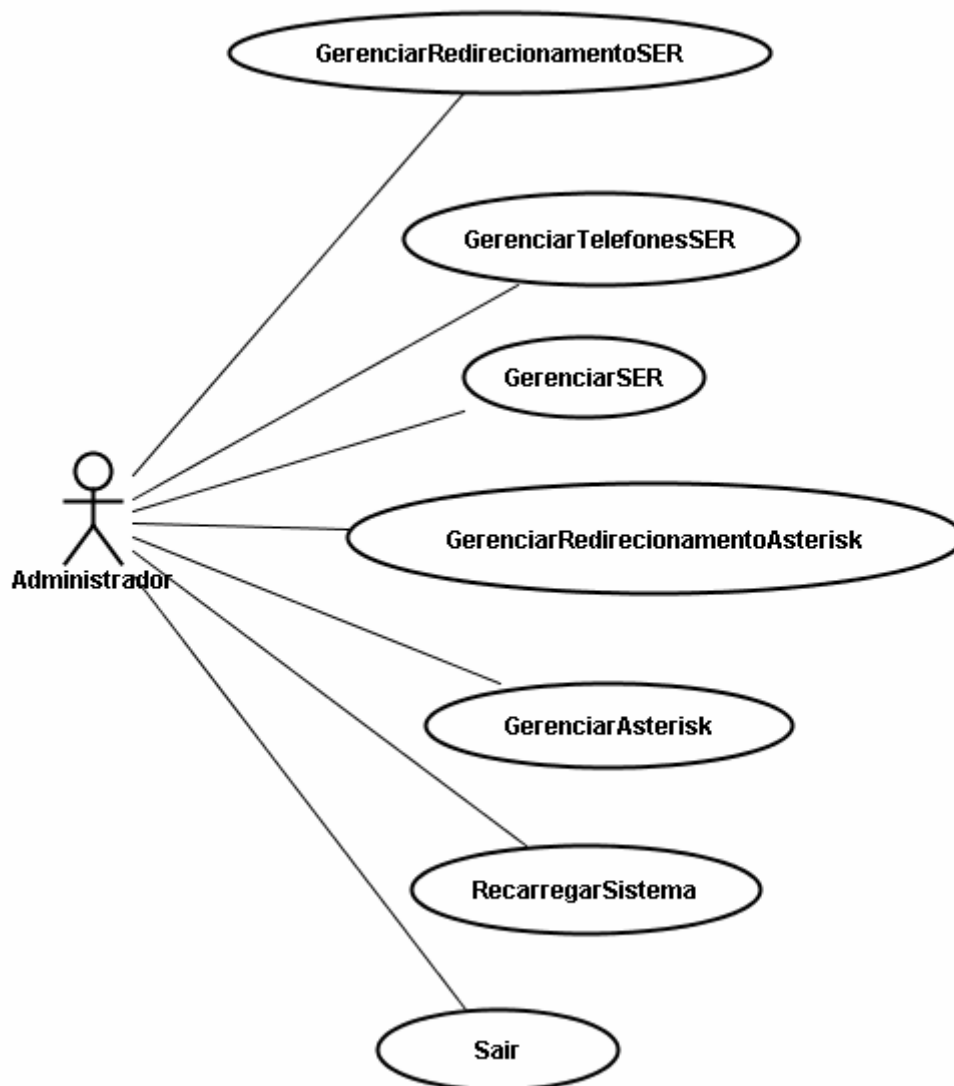


Figura 34. Diagrama de casos de uso do *SER*, *Asterisk* e recarregar.

Primeiramente, o administrador poderá gerenciar os redirecionamentos (*GerenciarRedirecionamentosSER*) através do submenu *SER=>Redirecionamentos* (Figura 35).

SER	Redirecionamentos	Adicionar
Asterisk	▶ Telefones	▶ Remover
Recarregar	Gerenciar	Modificar
Sair		Listar

Figura 35. Submenus da opção “*SER=>Redirecionamentos*”.

Voltando ao caso das ligações feitas de *Voip1* para *Voip2*, teríamos que adicionar um redirecionamento no *SER* do *Voip1* e no *SER* do *Voip2*. Um cliente *H.323* ao ligar de *Voip1* para um cliente *SIP* de *Voip2*, seria encaminhado do *GK* de *Voip1* para o *Asterisk* de *Voip1* através do prefixo adicionado na seção anterior. O *Asterisk* identificaria que a ligação não é para o servidor local e sim para outro servidor, pois o prefixo é diferente (0813273) e encaminharia para o *SER* local que encaminharia para o *SER* de *Voip2*. Tal “identificação” de chamadas do *Asterisk* será configurada durante a configuração do *Asterisk*. Este redirecionamento do *SER* local para o *SER*

de *Voip2* é realizado nesta seção. Na Figura 36, todas as ligações destinadas ao prefixo 0813273 e contendo quatro outros números restantes entre 0-9 ([0-9]{4}), serão redirecionadas para o servidor *SER* de *Voip2* que possui o IP 200.133.0.212 e domínio *voip2.pop-pe.rnp.br*. Pode-se também postar algum comentário para futuras visualizações através da opção listar.

Adicionar redirecionamento de chamadas ao SER

No campo regra de discagem, inserir uma regra que satisfaça uma chamada específica.

Caso essa regra seja satisfeita, a chamada será encaminhada para o IP e porta especificados nos campos abaixo.

Exemplo de regra de discagem: **0813221[0-9]{4}**

Neste exemplo, todas as chamadas que começarem por **0813221** e contiverem após esta cadeia quatro (**{4}**) números entre 0 e 9 (**[0-9]**), tal chamada será encaminhada para o IP e porta especificados nos campos abaixo.

Regra de discagem:

IP:

Porta :

Domínio :

Comentário:

Figura 36. Configuração do redirecionamento do *SER* de *VoIP1* para o *SER* do *VoIP2*.

Em *VoIP2*, ter-se-ia a configuração apresentada na Figura 37, onde haveria parâmetros semelhantes, porém inversos.

Adicionar redirecionamento de chamadas ao SER

No campo regra de discagem, inserir uma regra que satisfaça uma chamada específica.

Caso essa regra seja satisfeita, a chamada será encaminhada para o IP e porta especificados nos campos abaixo.

Exemplo de regra de discagem: **0813221[0-9]{4}**

Neste exemplo, todas as chamadas que começarem por **0813221** e contiverem após esta cadeia quatro (**{4}**) números entre 0 e 9 (**[0-9]**), tal chamada será encaminhada para o IP e porta especificados nos campos abaixo.

Regra de discagem:

IP:

Porta :

Domínio :

Comentário:

Figura 37. Configuração do redirecionamento do *SER* de *VoIP1* para o *SER* do *VoIP2*.

Após tal configuração, o administrador passaria a adicionar, modificar, remover ou listar os usuários/telefones (*GerenciarTelefonesSER*) através do menu *SER=>Telefones* (Figura 38).

SER	Redirecionamentos	Adicionar
Asterisk	▶ Telefones	▶ Remover
Recarregar	Gerenciar	Modificar
Sair		Listar

Figura 38. Submenus da opção “*SER=>Telefones*”.

Um exemplo de adição de um usuário ao *SER* pode ser visto na Figura 39. A pessoa responsável pelo telefone seria Fernanda Lins, tendo esta o telefone 08132720001, com uma senha específica contendo o e-mail fernanda@dominio.com.

Adicionar novo telefone ao SER

* Campos obrigatórios

Número (Ex: 001) * :

Senha * :

Redigitar Senha * :

Primeiro nome:

Segundo Nome:

Email * :

Figura 39. Adição de um usuário ao *SER* em *VoIP1*.

Após tais configurações, o usuário administrador poderá parar, iniciar ou reiniciar o servidor *SER*, através do *submenu* *SER=>Gerenciar (GerenciarSER)*. Configuramos também o *Asterisk*, sendo este necessário para haver o redirecionamento das ligações destinadas a outros grupos. Portanto, o redirecionamento realizado de *Voip1* para *Voip2* através do *Asterisk* pode ser realizado através do *submenu Asterisk=>Redirecionamento (GerenciarRedirecionamentosAsterisk)*.

SER	Redirecionamentos	Adicionar
Asterisk	▶ Telefones	▶ Remover
Recarregar	Gerenciar	Modificar
Sair		Listar

Figura 40. *Submenus* da opção “*Asterisk=>Redirecionamentos*”.

Seguindo o exemplo abordado desde então, configuramos o *Asterisk* de acordo com a Figura 41. Note que se podem adicionar quantos redirecionamentos forem necessários, tanto no *Asterisk* como no *SER* e pode-se adicionar quantos *GKs* e prefixos quisermos na seção do *GnuGK*. Isto nos permite configurar ligações entre várias empresas utilizando o *Doulox*. Neste exemplo, todas as ligações originadas de *Voip1* através de clientes *H.323* e destinadas a *Voip2* (0813273) serão destinadas, através do *Asterisk*, para o canal *SIP*, diretamente ao servidor *SER*, onde será repassado para o servidor *SER* de *Voip2* com *IP 200.133.0.212* de acordo com as configurações vistas acima. O “*xxxx*” da regra indica que o número precisa conter exatamente quatro números finais e o “*_*” que pode ser começado com qualquer prefixo inicial. É recomendado colocar o traço para que possamos utilizar prefixos internacionais, tendo o Brasil o prefixo 55. Se *Voip2* estivesse fora do país, uma ligação de *Voip2* para *Voip1* seria realizada utilizando o prefixo 550813272xxxx, passando assim por uma regra *_0813272xxxx* no *Asterisk* de *Voip2* onde qualquer prefixo inicial antecedendo o 081 pode ser aceito..

Adicionar redirecionamento de chamadas ao Asterisk

No campo regra de discagem, inserir uma regra que satisfaça uma chamada específica.

Caso essa regra seja satisfeita, a chamada será encaminhada para o servidor SIP local. Essa funcionalidade servirá apenas se houverem outros servidores espalhados pela rede, não será necessária para redirecionamentos locais, onde se tem apenas um servidor.

Exemplo 1: `_081322[02347]xxxx`

Neste exemplo, todas as chamadas que contiverem o início **081322** (o "_" indica que pode conter qualquer sequência de número antes do **081322**), contiverem após esta cadeia **UM** número no conjunto **[02347]** e possuírem quaisquer quatro números (**xxxx**), tal chamada será encaminhada para o servidor SIP local.

Exemplo 2: `0813272xxxx`

Neste caso, o número **TEM QUE** iniciar com o **0813272** e terminar com quatro números (**xxxx**) quaisquer.

Regra de discagem: `_0813273xxxx`

Adicionar

Figura 41. Redirecionamento de ligações de *Voip1*, destinadas ao grupo *Voip2*, ao servidor *SER* local através do *Asterisk*.

Em *Voip2*, a configuração seria semelhante ao da Figura 41, porém utilizar-se-ia a regra “_0813272xxxx”. Podemos também iniciar, parar e reiniciar o *Asterisk* utilizando o *submenu Asterisk=>Gerenciar (GerenciarAsterisk)*. Ao configurarmos ou alterarmos qualquer valor no sistema, precisamos utilizar a função “Recarregar” (*RecarregarSistema*). Este é necessário para *reconfigurarmos* todo o sistema e reiniciarmos todos os servidores. Este vem a ser uma funcionalidade bastante importante no sistema. Caso queiramos mudar o *DNS*, *IP* ou prefixos, após feita as modificações e executarmos esta função, todos os parâmetros de todos os arquivos de configuração dos servidores e alguns dados no banco de dados do *SER* ou do *Doulox* serão alterados. Para finalizarmos, a opção “Sair” fecha a sessão do usuário e sai do sistema, voltando à tela de autenticação. Realizados todos os procedimentos acima, usuários de *Voip1* poderiam fazer ligações para *Voip2* ou vice-versa, estando estes ligados a uma rede em pouco tempo utilizando o *Doulox* e o sistema de gerenciamento para uma rápida configuração dos serviços.

Capítulo 5

Conclusões e trabalhos futuros

5.1 Conclusões

O objetivo desse trabalho foi desenvolver uma distribuição *Linux* para ambientes *VoIP* e um programa de gerenciamento dos servidores disponíveis nesta distribuição, inicialmente dando uma noção geral na tecnologia voz sobre *IP*, mostrando uma parte do cenário tradicional, partindo para um novo paradigma, voz sendo transportada via pacotes *IP*. Os protocolos *VoIP* mais importantes, o *H.323* e *SIP*, foram basicamente apresentados, para que uma noção sobre o que estaríamos trabalhando fosse estabelecida.

Após tal introdução, vimos como foi desenvolvida a distribuição *Doulox Linux Live*, passando por todos os obstáculos enfrentados durante o desenvolvimento e solucionando-os através de pesquisas e conhecimentos prévios em *Linux*. Vários quadros contendo tais erros foram apresentados, sendo estes indispensáveis para dar um melhor suporte a este tema.

Em seguida passamos a explicar a instalação e configuração de todos os servidores (e suas dependências) responsáveis por prover um serviço *VoIP* com protocolos *SIP*, *H.323* e sua união, apresentando todos os obstáculos obtidos e soluções apresentadas. O sistema de gerenciamento de tais servidores também foi desenvolvido com êxito, e neste trabalho foram apresentadas as funcionalidades deste e os diagramas de caso de uso, para um melhor entendimento para futuras configurações.

Por fim, concluímos que o objetivo do desenvolvimento da distribuição foi concluído com sucesso, apenas com uma pequena falta na configuração com redes de telefonia convencional, por conta da falta de um equipamento, o modem da *Digium*, que não foi entregue a tempo pela RNP.

5.2 Discussões

Acreditamos que este projeto poderá servir para pessoas que queiram ter uma introdução a *VoIP*, seus protocolos e queiram realmente implantar ou testar tal tecnologia em seus locais de trabalho, podendo ser estendido a testes maiores. Isto porque a tecnologia *VoIP* ainda não é bem difundida, é bastante comentada entre as pessoas, porém, quase sempre em relação a clientes e não a

servidores, sendo o material disponível sobre servidores na Internet atualmente bastante escasso. Com isso, este trabalho poderá motivar algumas pessoas a desenvolverem a sua própria rede de telefonia *IP*.

Duas grandes empresas do estado de Pernambuco, Telemar e Corisco, mostraram um interesse inicial por este projeto, a Telemar querendo trocar arquivos de configurações e a Corisco utilizar o sistema. Portanto acredita-se que trabalhos como este poderão dar um impulso na tecnologia local, como também nacional, reduzindo bastante o custo de ligações entre instituições. Isto já vem sendo feito pelo menos entre a RNP e algumas instituições clientes.

5.3 Trabalhos futuros

Alguns problemas ainda precisam ser resolvidos. Um dos trabalhos a serem realizados futuramente refere-se resolver o problema apresentado quanto ao comando *find* utilizado durante o *Linux* rodando no CD, uma vez que não conseguimos resolver tal situação. Este problema pode ser encontrado na Seção 3.1.

A segurança do sistema não foi reforçada por completo, é necessário prover uma maior segurança quanto a regras de *firewall*, restrição do acesso ao sistema de gerenciamento e o *MySQL* apenas localmente, segurança no tráfego de voz e maior nível de autenticação dos telefones, pois, como apresentado, não foi possível autenticar os telefones *IP Cisco*, necessitando haver uma maior verificação disto. Também é necessário realizar testes entre vários *codecs* de voz diferentes e desenvolver testes de estresse em um servidor com tal distribuição, verificando a sua capacidade de processamento e qualidade, podendo isto se tornar um outro projeto. Outros trabalhos seriam procurar por *bugs* e corrigi-los, realizar testes em rede sem fio (*wireless*), adicionar um sistema de gerenciamento de chamadas e contabilidade, fazer testes de estresse no servidor em operação, testes de ocupação de banda e testes com a utilização de *QoS* (qualidade de serviço) [48].

Quanto ao sistema, é necessário desenvolver mais funcionalidades, como por exemplo, funcionalidades para tornar possível a configuração de correio de voz. Com isto, os clientes poderiam deixar uma mensagem em caixa postal de um outro telefone caso este não atendesse ao telefone. Pode-se também adicionar funcionalidades de redirecionamento tanto entre a rede de telefonia *IP* e convencional.

Isto não foi desenvolvido no sistema por questões da não possibilidade de haver testes pela falta da placa. Seria interessante desenvolver uma melhor interface (*design*) para o sistema, pois está bastante simples, porém rápida e funcional. A adição de configurações avançadas do *GnuGK*, *SER* e *Asterisk* também seria interessante, pois existem vários outros tipos de configuração, muitos parâmetros podem ser adicionados ao sistema para refinar ainda mais o comportamento dos serviços *VoIP*. Este seria o principal trabalho futuro a ser realizado, pois os arquivos de configuração de todos estes servidores possuem vários parâmetros de configuração, onde utilizamos normalmente valores padrão ou simplesmente não exploramos configurações mais avançadas ou pré-configuramos tais parâmetros, deixando inacessível aos administradores que estão utilizando o sistema de gerenciamento.

Outro trabalho a ser feito é melhorar a interface homem máquina do *Doulox*, uma vez que não se preocupou muito com este detalhe. Mudar a simples tela de *boot*, criar um logotipo melhor para o mesmo, uma tela de *login* com desenhos em ASCII e criar temas para o *KDE*, como também desenvolver papéis de parede do *Doulox*, ícones personalizados etc.

Bibliografia

- [1] VARSHNEY, UPKAR. *Voice over IP*. ACM Press, Communications of the ACM, Volume 45, Issue 1, 2002, New York, NY, USA.
- [2] COMER, DOUGLAS E. *Internetworking with TCP/IP Vol.1: Principles, Protocols, and Architecture*. Prentice Hall, 4th edition, 2000.
- [3] KUMAR, VINEET. *IP telephony with H.323*. John Wiley Professio, 1st edition, 2001.
- [4] JOHNSTON, ALAN B. *SIP: Understanding the Session Initiation Protocol*. Artech House. 2nd edition. 2004.
- [5] HARTE, LAWRENCE. *Introduction to Public Switched Telephone Networks (PSTN), Local Loop, Switching, DSL, ATM, SS7, and AIN*. ALTHOS Publishing, 2003.
- [6] ITU-T. *ITU Telecommunication Standardization Sector*. Disponível em: <<http://www.itu.int/ITU-T/>>. Acesso em: 19 abril 2005.
- [7] IETF. *The Internet Engineering Task Force*. Disponível em: <<http://www.ietf.org>>. Acesso em: 09 dezembro 2004.
- [8] HARTE, LAWRENCE. *Introduction to Private Telephone Networks: KTS, PBX, Centrex, CTI, and WPBX*. ALTHOS Publishing, 2003.
- [9] GNUGK. *The GNU Gatekeeper Manual*. Disponível em: <<http://www.gnugk.org/gnugk-manual.html>>. Acesso em: 09 dezembro 2004.
- [10] SER. *SIP Express Router Admin's Guide*. Disponível em: <<http://www.iptel.org/ser/admin.html>>. Acesso em: 19 de abril de 2005.
- [11] SCHWARZ, BRETT. *Asterisk open-source PBX system*. Specialized Systems Consultants, Inc., 2004, Seattle, WA, USA.
- [12] RINDE, JOSEPH. *Telephony in the year 2005*. Computer Networks: The International Journal of Computer and Telecommunications Networking, Volume 31, Issue 3, Elsevier North-Holland, Inc., February 1999, New York, NY, USA.
- [13] JAMES, STEVE. *Isdn Clearly Explained*. Morgan Kaufmann Pub; 2nd edition, 1997.
- [14] KYAS, OTHMAR. *ATM Networks*. Prentice Hall PTR; 1st edition, 2002.
- [15] BRANDL, MARGIT. *IP Telephony Cookbok*. TERENA (Trans-European Research and Education Networking Association) Report. Março, 2004.
- [16] OpenPhone. *Command Line H.323 Client*. Disponível em: <<http://www.openh323.org/>>. Acesso em: 11 maio 2005.
- [17] MILLER, C. KENNETH. *Multicast Networking & Applications*. Addison-Wesley Professional, 1st edition, 1998.
- [18] JAVVIN. *Network Protocols Handbook*. Javvin Technologies, 2005.
- [19] ALBITZ, PAUL. *DNS and BIND, Fourth Edition*. O'Reilly, 4th edition, 2001.
- [20] X-Lite. *A VoIP SIP Client*. Disponível em: <<http://www.xten.com/>> . Acesso em: 11 maio 2005.
- [21] PERKINS, COLIN. *RTP: Audio and Video for the Internet*. Addison-Wesley Professional; 1st edition, 2003.

- [22] OPPEL, ANDREW. *Databases Demystified*. McGraw-Hill Osborne Media; 1st edition, 2004.
- [23] WELSH, MATT. *Running Linux, Fourth Edition*. O'Reilly; 4th edition, 2002.
- [24] WELLING, LUKE. *PHP and MySQL Web Development*. Sams, 3th edition, 2004.
- [25] MEADORS, TODD. *Linux Shell Script Programming*. Course Technology, 1st edition, 2003.
- [26] DUBOIS, PAUL. *MySQL Cookbook*. O'Reilly, 1st edition, 2002.
- [27] SEIFERT, RICH. *The Switch Book: The Complete Guide to LAN Switching Technology*. Wiley, 1st edition, 2000.
- [28] WARD, BRIAN. *The Book of VMware: The Complete Guide to VMware Workstation*. No Starch Press, 1st edition, 2002.
- [29] RUSSEL, CHARLIE. *Microsoft Windows XP Professional Resource Kit*. Microsoft Press, 3rd Bk&Cdr edition, 2005.
- [30] PARKER, TIM. *Slackware Linux Unleashed*. Sams Pub, 3rd edition, 1997.
- [31] O'SULLIVAN, T. C. *RFC0139: Discussion of Telnet Protocol*. RFC Editor, United States, 1971.
- [32] BOVET, DANIEL P. *Understanding the Linux Kernel (2nd Edition)*. O'Reilly, 2nd edition, 2002.
- [33] KERNIQHAN, BRIAN W. *C Programming Language (2nd Edition)*. Prentice Hall PTR, 2nd edition, 1988.
- [34] LILO. *The Linux Boot Loader*. Disponível em: <<http://tldp.org/HOWTO/LILO.html>> . Acessado em: 11 maio 2005.
- [35] BARRET, DANIEL J. *SSH, The Secure Shell: The Definitive Guide*. O'Reilly, 1st edition, 2001.
- [36] PEEK, JERRY. *Learning UNIX Operating System, Fifth Edition*. O'Reilly, 5th edition, 2002.
- [37] NEGUS, CHRISTOPHER. *Red Hat Linux Bible: Fedora and Enterprise Edition*. Wiley, Bk&CD-Rom edition, 2003.
- [38] HARRIS, DAVID B. *Debian GNU/Linux 3.1 Bible*. Wiley, CD-Rom edition, 2005.
- [39] SOLLINS, K. *RFC1350: The TFTP Protocol (Revision 2)*. RFC Editor, United States, 1992.
- [40] HUNT, CRAIQ. *Sendmail Cookbook*. O'Reilly, 1st edition 2003.
- [41] POWELL, DENNIS E. *Practical KDE*. Que, 1st edition, 1999.
- [42] FRISCH, AELEEN. *Linux System Administration: Adding a New Disk to a Linux System*. Specialized Systems Consultants, Inc., Seattle, WA, USA, 1995.
- [43] CASTRO, ELIZABETH. *HTML for the World Wide Web with XHTML and CSS: Visual QuickStart Guide, Fifth Edition*. Peachpit Press, 5th edition, 2002.
- [44] MEYER, ERIC A. *Cascading Style Sheets: The Definitive Guide*. O'Reilly, 2nd edition, 2004.
- [45] FLANAQAN, DAVID. *JavaScript: The Definitive Guide*. O'Reilly, 4th edition, 2001.
- [46] COAR, KEN. *Apache Cookbook*. O'Reilly, 1st edition, 2003.
- [47] FOWLER, MARTIN. *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley Professional, 3rd edition, 2003.
- [48] WANG, ZHENG. *Internet QoS: Architectures and Mechanisms for Quality of Service (Morgan Kaufmann Series in Networking)*. Morgan Kaufmann, 1st edition, 2001.

Apêndice A

Pacotes pré-instalados para o desenvolvimento do *Doulox*

Apresentaremos aqui os pacotes que foram necessários para a instalação base do *Slackware*, para que fosse possível o desenvolvimento do *Doulox* a partir dele.

aaa_base-10.0.0-noarch-1
aaa_elflibs-9.2.0-i486-1
aalib-1.4rc5-i386-1
acpid-1.0.3-i486-1
alsa-driver-1.0.5a-i486-2
alsa-lib-1.0.5-i486-1
alsa-oss-1.0.5-i486-1
alsa-utils-1.0.5-i486-1
amp-0.7.6-i386-1
arts-1.2.3-i486-1
ash-0.4.0-i386-1
aspell-0.50.5-i486-1
aspell-en-0.51_0-noarch-1
atk-1.6.1-i486-1
audiofile-0.2.6-i486-1
autofs-3.1.7-i386-1
automake-1.8.5-noarch-1
bash-2.05b-i486-3
bc-1.06-i386-2
bin-9.2.0-i486-2
bind-9.2.3-i486-1
binutils-2.15.90.0.3-i486-1
bpe-2.01.00-i486-1
bzip2-1.0.2-i486-5
cdparanoia-IIIalpha9.8-i386-1
cdrdao-1.1.9-i486-1
cdrtools-2.00.3-i486-1

coreutils-5.2.1-i486-1
cpio-2.5-i386-1
cups-1.1.20-i486-1
cxxlibs-5.0.6-i486-1
dcron-2.3.3-i386-4
devfsd-1.3.25-i486-4
devs-2.3.1-noarch-21
dhcpcd-1.3.22pl4-i386-1
diffutils-2.8.1-i386-1
dvd+rw-tools-5.19_1.4.9.7-i486-1
e2fsprogs-1.35-i486-1
elvis-2.2_0-i486-2
esound-0.2.34-i486-1
etc-5.1-noarch-9
expat-1.95.7-i486-1
findutils-4.1.7-i386-1
flac-1.1.0-i386-1
floppy-5.4-i386-3
fluxbox-0.1.14-i386-1
gawk-3.1.3-i486-1
gcc-3.3.4-i486-1
gcc-g++-3.3.4-i486-1
gdk-pixbuf-0.22.0-i486-1
gettext-0.14.1-i486-1
glib-1.2.10-i386-2
glib2-2.4.2-i486-1
glibc-solibs-2.3.2-i486-6
glibc-zoneinfo-2.3.2-noarch-6
glut-3.7-i486-1
gpm-1.19.6-i486-6
grep-2.5-i386-2
groff-1.17.2-i386-3
gtk+-1.2.10-i386-3
gtk+2-2.4.3-i486-1
gzip-1.3.3-i386-2
hdparm-5.5-i486-1
hotplug-2004_01_05-noarch-3
imlib-1.9.14-i486-2
indent-2.2.9-i386-1
inetd-1.79s-i486-6
infozip-5.51-i486-1
iproute2-2.6.7_ss040608-i486-2
iptables-1.2.10-i486-1
iptraf-2.7.0-i386-1
isapnptools-1.26-i386-1
jfsutils-1.1.6-i486-1
joe-3.1-i486-1
jove-4.16.0.61-i386-1
kbd-1.12-i486-2

kdeartwork-3.2.3-i486-1
kdebase-3.2.3-i486-1
kdegames-3.2.3-i486-1
kdegraphics-3.2.3-i486-1
kdelibs-3.2.3-i486-1
kdemultimedia-3.2.3-i486-1
kdenetwork-3.2.3-i486-1
kdepim-3.2.3-i486-1
kdeutils-3.2.3-i486-1
kernel-ide-2.4.26-i486-4
kernel-modules-2.4.26-i486-3
koffice-1.3.1-i486-3
less-382-i486-1
lesstif-0.93.94-i486-1
libao-0.8.4-i486-1
libart_lgpl-2.3.16-i486-1
libexif-0.5.12-i486-1
libglade-2.4.0-i486-1
libgsf-1.9.1-i486-1
libid3tag-0.15.1b-i486-1
libieee1284-0.2.8-i486-1
libjpeg-6b-i386-4
libmad-0.15.1b-i486-1
libmng-1.0.5-i486-1
libogg-1.1-i486-1
libpng-1.2.5-i486-2
libtermcap-1.2.3-i486-6
libtiff-3.6.1-i486-2
libungif-4.1.2-i486-1
libusb-0.1.8-i486-1
libvorbis-1.0.1-i486-1
libwmf-0.2.8.2-i486-2
libxml2-2.6.9-i486-1
libxslt-1.1.6-i486-1
lilo-22.5.9-i486-2
logrotate-3.6.8-i486-1
lsof-4.71-i486-1
lynx-2.8.5rel.1-i486-2
m4-1.4.1-i486-1
make-3.80-i386-1
man-1.5m2-i486-1
man-pages-1.64-noarch-1
mc-4.6.0-i486-4
mdadm-1.6.0-i486-1
mhash-0.9.1-i486-1
module-init-tools-3.0-i486-1
most-4.9.5-i486-1
mpeg_lib-1.3.1-i386-2
mpg321-0.2.10-i486-2

mt-st-0.7-i386-1
mutt-1.4.2.1i-i486-1
nc-1.10-i386-1
ncurses-5.4-i486-2
netpipes-4.2-i386-1
netwatch-1.0a-i386-1
nmap-3.50-i486-1
normalize-0.7.6-i486-1
openssh-3.8.1p1-i486-1
openssl-0.9.7d-i486-1
openssl-solibs-0.9.7d-i486-1
pango-1.4.0-i486-1
pciutils-2.1.11-i486-5
pcmcia-cs-3.2.7-i486-3
pcre-4.5-i486-2
perl-5.8.4-i486-3
pidentd-3.0.18-i486-1
pkgtools-10.0.0-i486-1
popa3d-0.6.4-i486-1
popt-1.7-i386-1
portmap-5.0-i486-1
ppp-2.4.2-i486-2
procps-3.2.1-i486-1
qt-3.3.2-i486-2
reiserfsprogs-3.6.17-i486-1
rexima-1.4-i486-1
rp-pppoe-3.5-i386-1
rpm-4.2.1-i486-3
rzip-2.0-i486-2
screen-4.0.2-i486-1
sdl-1.2.7-i486-2
sed-4.0.9-i486-2
seejpeg-1.10-i386-1
sendmail-8.12.11-i486-2
shadow-4.0.3-i486-11
slocate-2.7-i486-3
smartmontools-5.30-i486-1
sox-12.17.4-i486-2
startup-notification-0.6-i486-1
strace-4.5.4-i486-1
sudo-1.6.6-i386-1
svgalib-1.4.3-i386-2
sysklogd-1.4.1-i486-9
sysvinit-2.84-i486-50
t1lib-1.3.1-i386-2
taglib-1.1-i486-2
tar-1.14-i486-4
tcpdump-3.8.3-i486-2
tcpip-0.17-i486-29

traceroute-1.4a12-i386-2
umsdos-progs-1.13-i386-1
usbutils-0.11-i486-2
utempter-1.1.1-i486-1
util-linux-2.12a-i486-1
vorbis-tools-1.0.1-i486-2
wget-1.9.1-i486-1
whois-4.6.16-i486-1
wireless-tools-26-i486-3
workbone-2.40-i386-3
wv2-0.2.2-i486-1
x11-6.7.0-i486-4
x11-fonts-cyrillic-6.7.0-noarch-1
x11-fonts-misc-6.7.0-noarch-1
x11-fonts-scale-6.7.0-noarch-1
xaw3d-1.5-i386-3
xfsprogs-2.6.13-i486-1
ytalk-3.1.1-i386-1
zlib-1.2.1.1-i486-1

Apêndice B

Arquivo de configuração do telefone *IP Cisco 7905G*

```
#txt - Sample Cisco IP Phone 7905G (H.323) Parameter Profile
#
UIPassword:0
upgradecode:0,0x501,0x0400,0x0100,0.0.0.0,69,0,none
dhcp:0
StaticIp:200.133.0.210
StaticRoute:200.133.0.45
StaticNetmask:255.255.255.0
Domain:phone1@voip1.pop-pe.rnp.br
GkId:VOIP1GK
Gk:200.133.0.211
AltGk:0
AltGkTimeOut:0
GkTimeToLive:300
Gateway:0
UID:08132721001
PWD:phone1p
LoginID:phone1
UseLoginID:1
RxCodec:2
TxCodec:2
AudioMode:0x00150015
NumTxFrames:2
CallFeatures:0xffffffff
PaidFeatures:0xffffffff
ConnectMode:0x00060400
Timezone:15
AutMethod:0
NTPIP:200.19.119.69
AltNTPIP:0.0.0.0
```

DNS1IP:200.133.0.35
DNS2IP:0.0.0.0
UseTftp:1
TftpURL:200.133.0.211
CfgInterval:3600
EncryptKey:0
NPrintf:200.133.0.211.9001
DialPlan:*St4-|#St4-|91111|>#t8.r9t2-|0>#t811.rat4-|^1t4>#.-
RingOnOffTime:2,4,25
DialTone:2,31538,30831,3100,3885,1,0,0,1000
BusyTone:2,30467,28959,1191,1513,0,4000,4000,0
ReorderTone:2,30467,28959,1191,1513,0,2000,2000,0
RingBackTone:2,30831,30467,1943,2111,0,16000,32000,0
CallWaitTone:1,30831,0,5493,0,0,2400,2400,4800
AlertTone:1,30467,0,5970,0,0,480,480,1920
EchoIP:200.133.0.45
MediaPort:16384
TOS:0xA0
SigTimer:0x01418564
OpFlags:0x2
VLANSetting:0x0000002b

Apêndice C

Arquivo *install.sh* criado para automatizar a criação do *Doulox* e figura dos diretórios e arquivos necessários para a sua execução

install.sh

```
#!/bin/sh

tar -xzf linux-live-4.2.4.tar.gz
tar -xzf slax-patch-4.2.0.tar.gz
cd slax-patch-4.2.0/rootpatch/
./patch.sh
cp -rf * /
cd -
cp -rf slax-patch-4.2.0/slax-cd/* linux-live-4.2.4/bootfiles/
installpkg apache-1.3.31-i486-2.tgz
installpkg asterisk-1.0.5-i386-1.tgz
installpkg asterisk-oh323-0.6.5-i386-1.tgz
installpkg gnugk-2.2.1-i386-1.tgz
installpkg kernel-2.4.26-ovlfs-devfs-alsa-1.0.4-i486-1.tgz
installpkg libpri-1.0.4-i386-1.tgz
cp -rf mysql/* /
cp dhcpd-eth0.info /etc/dhcpd/
cp -rf issue* /etc
installpkg ohphone-ohphone-i386-1.tgz
installpkg openh323-1.12.2/openh323-openh323-i386-1.tgz
installpkg openh323-1.13.5/openh323-openh323-i386-1.tgz
installpkg php-4.3.7-i486-1.tgz
installpkg proftpd-1.2.9-i486-3.tgz
```

```
installpkg pwlib-1.5.2/pwlib-pwlib-i386-1.tgz
installpkg pwlib-1.6.6/pwlib-pwlib-i386-1.tgz
installpkg ser-0.8.14-i386-1.tgz
installpkg zaptel-1.0.4-i386-1.tgz
useradd apache
mkdir /var/named/pri
mysql_install_db
chown -R mysql.users /var/lib/mysql
chown mysql.users /var/run/mysql -R
chmod +x /etc/rc.d/rc.mysql
/etc/rc.d/rc.mysql start
cp rc.* /etc/rc.d/
chmod +x /etc/rc.d/rc.asterisk /etc/rc.d/rc.gnugk /etc/rc.d/rc.ser
rm -rf /var/www/htdocs/*
cp phpMyAdmin-2.6.1-pl2.tar.gz /var/www/htdocs/
cd /var/www/htdocs/
tar -xzf phpMyAdmin-2.6.1-pl2.tar.gz
mv phpMyAdmin-2.6.1-pl2 mysql
rm -rf phpMyAdmin-2.6.1-pl2.tar.gz
cd -
mkdir /usr/local/tftpboot
cp -rf firmware.h323.configured.tar.gz /usr/local/tftpboot/
cd /usr/local/tftpboot/
tar -xzf firmware.h323.configured.tar.gz
rm -rf firmware.h323.configured.tar.gz
cd -
cd confs/
cp -rf config.inc.php /var/www/htdocs/mysql/
cp -rf extensions.conf oh323.conf sip.conf /etc/asterisk/
cp -rf gnugk.ini /etc/
cp -rf httpd.conf /etc/apache/
cp -rf php.ini /etc/apache/
cp -rf ser.cfg /usr/local/etc/ser/
cp -rf inetd.conf /etc
cp -rf profile /etc
cp -rf sudoers /etc
cp -rf proftpd.conf /etc
cp -rf hosts /etc
cp -rf HOSTNAME /etc
cd -
cp -rf slaxinstall/usr/bin/slax-install /usr/bin
cp -rf doulox-software.tar.gz /var/www
cd /var/www
tar -xzf doulox-software.tar.gz
rm -rf doulox-software.tar.gz
cd -
chmod 755 -R /var/www/htdocs
chmod 777 -R /var/www/htdocs/confs
mysqladmin create doulox
```

```
mysql doulox < /var/www/htdocs/sql/doulox.sql  
/usr/local/sbin/ser_mysql.sh create
```

Figura onde o install.sh se encontra

