ESCOLA POLITÉCNICA
DE PERNAMBUCO

Departamento de
Sistemas
Computacionais

ESTUDO SOBRE SEGURANÇA EM VOIP EM UM PC-PBX CONFIGURADO PARA UM AMBIENTE CORPORATIVO

Trabalho de Conclusão de Curso

Engenharia da Computação

Laura Macedo Arahata Moraes
Orientador: Prof. Dr. Ricardo Massa Ferreira Lima

Recife, 22 de maio de 2006

UNIVERSIDADE
DE PERNAMBUCO

ESCOLA POLITÉCNICA
DE PERNAMBUCO

Departamento de
Sistemas
Computacionais

ESTUDO SOBRE SEGURANÇA EM VOIP EM UM PC-PBX CONFIGURADO PARA UM AMBIENTE CORPORATIVO

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.

Laura Macedo Arahata Moraes
Orientador: Prof. Dr. Ricardo Massa Ferreira Lima

Recife, 22 de maio de 2006

UNIVERSIDADE
DE PERNAMBUCO

Laura Macedo Arahata Moraes

ESTUDO SOBRE SEGURANÇA EM VOIP EM UM PC-PBX CONFIGURADO PARA UM AMBIENTE CORPORATIVO

Resumo

Atualmente, as diversas atividades executadas em uma empresa dependem crescentemente da utilização de *softwares* e da comunicação digital, que traz consigo importantes benefícios, mas também alguns riscos. Conseqüentemente, a preocupação com ameaças à segurança da informação tem se tornado constante e necessária. VoIP, que é o tráfego de voz em uma rede de dados, é uma das fortes tendências no mercado de telecomunicações. Segundo Winn Schwartau [Schwartau05], a comunicação do mundo está convergindo para VoIP, porém ainda não se tem o conhecimento de segurança que será necessário para enfrentar os problemas do mundo real. Para que a segurança da informação seja garantida, três fatores precisam ser considerados: confidencialidade, integridade e disponibilidade. As ameaças a esses fatores vêm crescendo a cada dia devido ao aumento do número de acessos à Internet e à facilidade na obtenção de ferramentas e informações que possibilitam explorar falhas de segurança. Neste trabalho foi realizado um estudo sobre segurança em VoIP em um ambiente preparado conforme implementações habituais utilizando o Asterisk, que é uma solução em software livre de um PC-PBX. Inicialmente foi realizada uma revisão bibliográfica de alguns protocolos VoIP, de conceitos e de tecnologias necessárias para a realização do estudo. Após a preparação do ambiente de estudo, foram demonstradas as fragilidades do mesmo e, posteriormente, implementadas algumas técnicas de segurança em rede para avaliar as implicações dessas implementações no ambiente. Diferentemente do que foi apresentado por Thomas Walsh e Richard Kuhm [Walsh05], não foram encontradas grandes dificuldades na implementação de segurança em VoIP nesse ambiente. Ao final, foram apresentadas algumas recomendações, devidamente explicadas, que poderão ser seguidas, em cenários semelhantes ao estudado neste trabalho, para que as implementações de VoIP possam ter sucesso em relação a alguns requisitos de segurança.

Palavras-chave: Segurança, Asterisk, VoIP, H.323, SIP, IAX.

Abstract

Nowadays, many activities executed in a company environment depend increasingly on the use of software and network communication, which brings benefits but also introduces risks. Consequently the concerns with information security threats have become constant and necessary. VoIP, which is voice transmitted over data network, is a strong telecommunication trend. According to Winn Schwartau [Schwartau05], the communications world is moving toward VoIP but does not have the security expertise it needs to meet the real-world stress it will encounter. Three factors must be considered to assure the information security: confidentiality, integrity and availability. The threats to these factors is growing each day due to the increasing number of Internet accesses and the facility to obtain tools and information that can be used to exploit security flaws. In this work, a study about VoIP security in an environment prepared according usual implementations through Asterisk, a PC-PBX free software implementation, was performed. Initially, a bibliographical review of some VoIP protocols, concepts and technologies needed for the study are presented. After preparing the environment, its fragilities were exposed and, then, some network security techniques were applied to evaluate its implications. In contrast with the arguments presented by Thomas Walsh and Richard Kuhm [Walsh05], the implementation of security in our VoIP environment was relatively simple. Eventually, the work suggests a number of recommendations to implement VoIP environment that fulfill some security requirements.

Keywords: Security, Asterisk, VoIP, H.323, SIP, IAX.

Sumário

Índice de Figuras

Índice de Quadros

Índice de Tabelas

Tabela de Símbolos e Siglas

(Dispostos por ordem de aparição no texto)

FTC – Federal Trade Commission
VoIP – Voice over IP (voz sobre IP)
IP – Internet Protocol (protocolo de Internet)
IDG – International Data Group
SS7 – Signaling System 7
SIP – Session Initiation Protocol
TCP – Transfer Control Protocol
PBX – Private Branch eXchange
RTP – Real-time Transfer Protocol
UDP – User Datagram Protocol
IAX – Inter Asterisk eXchange
PC-PBX – Personal Computer-Private Branch eXchange
QoS – Quality of Service
VPN – Virtual Private Network
ITSP – Internet Telephony Service Provider
DDD – Discagem Direta a Distância
DDI – Discagem Direta Internacional
MSN – Microsoft Network
RTCP – RTP Control Protocol
MGCP – Media Gateway Control Protocol
RTPC – Rede Telefônica Pública Comutada
SIPS – SIP Secure
SRTP – Secure RTP
ITU-T – International Telecommunications Union-Telecom Standardization Sector
MCU – Multipoint Control Unit
MC – Multipoint Controller
MP – Multipoint Processors
RAS – Registration, Admission, Status
IETF – Internet Engineering Task Force
RTSP – Real-time Streaming Protocol
SDP – Session Description Protocol
UA – User Agent
UAC – User Agent Client
UAS – User Agent Server
ATA – Analog Telephone Adaptor
UTF – Universal Transformation Format
HTTP – Hyper Text Transfer Protocol
SMTP – Simple Mail Transfer Protocol
MIME – Multipurpose Internet Mail Extensions
CODEC – Coder/Decoder

RFC – Request For Comment
URI – Universal Resource Identifier
NAT – Network Address Translation
ACK – Acknowledge
GSM – Global System for Mobile Telecommunication
ILBC – Internet Low Bitrate Codec
LPC – Linear Predictive Coding
PABX – Private Automatic Branch eXchange
CDR – Call Detail Record
URA – Unidade de Resposta Audível
GNU – GNU's not UNIX
API – Application Programming Interface
IO – Input/Output
IVR – Interactive Voice Response
SCCP – Signaling Connection Control Part
ADPCM – Adaptive Differential Pulse Code Modulation
OSI – Open Systems Interconnection
HTTPS – HTTP Secure
DES – Data Encryption Standard
DEA – Data Encryption Algorithm
3DES – Triple DES
AES – Advanced Encryption Standard
IDEA – International Data Encryption Algorithm
RC5 – Rivest Cipher 5
DSA – Digital Signature Algorithm
PGP – Pretty Good Privacy
SVPN – Secure VPN
TVPN – Trusted VPN
IPSec – IP Security
SSL – Secure Socket Layer
TLS – Transport Layer Security
PPTP – Point-to-point tunneling protocol
DNS – Domain Name System
FTP – File Transfer Protocol
DHCP – Dynamic Host Control Protocol
ADSL – Asymmetric Digital Subscriber Line
FXS – Foreign eXchange Subscriber
DTMF – Dual Tone Multi Frequential
FXO – Foreign eXchange Office
PSTN – Public Switched Telephone Network
ICMP – Internet Control Message Protocol
DoS – Denial of Service
ms – milisegundo
FFT – Fast Fourier Transform
Mbps – megabit por segundo
kbps – kilobit por segundo
MAC – Media Access Control
ARP – Address Resolution Protocol
RPM – Red Hat Package Manager
YUM – Yellowdog Update Modified

VLAN – Virtual Local Area Network
BIND – Berkeley Internet Name Domain
IDS – Intrusion Detection Systems
CA – Certification Authority
GUI – Graphical User Interface
IMQ – Intermediate Queuing Device
EUA – Estados Unidos da América
SSH – Secure Shell

Agradecimentos

Agradeço a Deus, por ter me dado vida, por me dar saúde e guiar meus passos todos os dias.

Aos meus pais, Ciro Arahata (*in memoriam*) e Iracema Macedo Arahata, por terem me dado todo amor e apoio necessários.

Ao meu marido Joel Barbosa Moraes, pelo amor, paciência e companheirismo em todas as horas.

À minha família, em especial, a minha avó Maria Angelina, minha tia Ivete Macedo, minhas irmãs Carla e Érika Arahata, que se mantêm presente nos momentos de alegria e nos momentos de dificuldades.

Ao meu orientador, Ricardo Massa Ferreira Lima, pela amizade e disponibilidade em atender às minhas dúvidas e dificuldades.

Aos meus amigos da Escola Politécnica de Pernambuco, em especial, a César Oliveira, Cleyton Rodrigues, Marcelo Nunes, Milena Rodrigues, Nívia Quental e Petrônio Braga, pelos momentos de descontração e pelas demonstrações de amizade ao longo do curso de Engenharia da Computação.

Capítulo 1

Introdução

Atualmente, as diversas atividades executadas em um ambiente corporativo dependem crescentemente da utilização de *softwares*, tais como: *software* para controle financeiro, administrativo e operacional. Conseqüentemente a preocupação com as ameaças à segurança da informação tem se tornado constante e necessária [Marcink03].

A crescente utilização da comunicação digital traz riscos. Sistemas algumas vezes são comprometidos e podem, até mesmo, ser utilizados para operações ilegais. Seguem alguns exemplos que atestam esta afirmação:

- Seisint, uma unidade da LexisNexis, empresa provedora de informações para os mercados jurídico, corporativo, governamental e acadêmico, teve as informações de 310 mil pessoas comprometidas. Dentre essas informações estavam o número da seguridade social e carteira de motorista [Lexisnex05];
- Um servidor da Novell, empresa provedora de soluções em tecnologia da informação, que os funcionários aparentemente utilizavam para jogos, foi invadido e utilizado para escutar portas vulneráveis em milhões de computadores do mundo todo [Vijayan05];
- Em 25 de maio de 2004, o *site* da Microsoft na Inglaterra foi *defaced* (alteração nas páginas *web*) por um grupo de *crackers* chamado de OutLaw Group [Leyden04];
- A ChoicePoint, que reconheceu que os dados financeiros de mais de 163 mil clientes foram comprometidos, foi multada em 15 milhões de dólares pela FTC (*Federal Trade Comission*) por ter sido negligente com a segurança e gerenciamento dos dados dos clientes [Federal06].

Esses são apenas alguns exemplos, dentre muitos, que demonstram a importância da segurança da informação. Para que ela seja garantida, três fatores precisam ser considerados:

- confidencialidade – garantir que a informação apenas esteja disponível para pessoas devidamente autorizadas;
- integridade – garantir que o conteúdo da informação esteja completo, ou seja, não tenha sido alterado nem destruído;

- disponibilidade – assegurar que a informação e o meio de acesso a esta informação estejam disponíveis sempre que solicitados.

As ameaças a esses fatores vêm crescendo a cada dia. Contribuem para isso o aumento do número de acessos à Internet e a facilidade na obtenção de ferramentas e informações para explorar falhas de segurança [Cunvin05].

Voz sobre IP, ou simplesmente VoIP, é a tecnologia que permite o tráfego de voz em uma rede de dados [Davidson00] [Fong02]. Essa é uma das fortes tendências no mercado de telecomunicações [Walsh05].

O mercado de VoIP apresenta altas taxas de crescimento a cada ano. Em 2004, a IDC (subsidiária da IDG – *International Data Group*) estimou até 2008 um crescimento médio anual de 282%. Entretanto, tendo atingido cerca de três milhões de assinantes no mercado residencial americano no final de 2005, a consultoria reviu a previsão para um crescimento médio de 900% até 2009 [Bantel05].

As redes de telefonia convencional foram desenvolvidas inicialmente para trafegar exclusivamente voz, enquanto as redes de dados são mais versáteis, suportando o transporte de outros tipos de informação. Protocolos como o SS7 [Davidson00], utilizado para a sinalização das chamadas na rede de telefonia pública, estão sendo substituídos por protocolos como o SIGTRAN [IEC05] (SS7 sobre IP), H.323 [Davidson00] e SIP (*Session Initiation Protocol*) [Davidson00], que funcionam em redes TCP/IP (*Transfer Control Protocol/Internet Protocol*) [Davidson00]. Além disso, sistemas proprietários que rodam as aplicações PBX (*Private Branch eXchange*) estão sendo substituídos por aplicações que rodam em sistemas operacionais conhecidos como o Windows e o Linux. A migração das tecnologias, anteriormente utilizadas, para áreas do conhecimento bem exploradas pelos *crackers* ocasiona um aumento da preocupação dos administradores de segurança em redes.

A rede de telefonia pública convencional é normalmente considerada segura já que é necessário o acesso físico às linhas telefônicas para a realização de grampos e, conseqüentemente, o comprometimento da informação.

Eventualmente, administradores de segurança em rede de computadores podem deduzir que, uma vez que a voz trafega em pacotes de dados, se utilizarem os equipamentos de VoIP em uma rede já segura, obter-se-á uma rede VoIP segura e estável. Entretanto, Thomas Walsh e Richard Kuhm afirmam que as ferramentas que são utilizadas para fazer a segurança em redes, normalmente não se aplicam a esse novo ambiente [Walsh05].

Winn Schwartau [Schwartau05] concluiu que a comunicação de voz do mundo está convergindo para VoIP, porém ainda não se tem o conhecimento da segurança que será necessária para enfrentar os problemas do mundo real. Para a implementação de uma rede VoIP segura e estável é preciso considerar os seguintes problemas [Walsh05]:

- *o delay*, que corresponde ao atraso da chegada do pacote ao seu destino. Aplicações VoIP são bastante sensíveis ao atraso. Quando se visita um *site*, por exemplo, um pouco de atraso no *download* de uma figura não causa problemas, porém um atraso de 500 ms em uma conversa pode torná-la inviável;
- *o jitter*, que corresponde à variação no atraso dos pacotes. Um sinal de voz é dividido em pequenos pacotes que são enviados pela rede de dados. O atraso não uniforme desses pacotes pode fazer com que eles cheguem fora da seqüência correta, fazendo com que eles precisem ser reordenados pela aplicação, já que não existe esse controle na camada de transporte, que é feito utilizando o RTP (*Real-Time Transport Protocol*) [Davidson00], um protocolo baseado em UDP (*User*

Datagram Protocol) [Davidson00]. Problemas com o *jitter* também podem tornar a comunicação impraticável;

- os protocolos utilizados em VoIP, como o H.323, o SIP e o IAX (*Inter Asterisk eXchange*) [Spencer05] normalmente não fazem a cifragem do seu conteúdo, ou seja, eles podem ter o conteúdo interceptado e/ou alterado antes de chegar ao destino. Essas interceptações podem ser realizadas através de ferramentas disponíveis na Internet que possibilitam capturar os pacotes VoIP e, por exemplo, gerar um arquivo de áudio com a conversa.

A utilização de VoIP, juntamente com um PC-PBX (*Personal Computer-Private Branch eXchange*), surgiu para colaborar na redução dos custos operacionais [Goncalv05] [Walsh05] de telefonia de uma empresa e para agregar facilidades como, por exemplo, a existência de ramais virtuais para *road warriors*, que são profissionais que trabalham a maior parte do tempo viajando. Através desses ramais virtuais os profissionais têm à disposição um ramal como se estivesse localizado na sua sala (*roaming users*).

Algumas empresas têm a telefonia como um fator fundamental para o pleno funcionamento do seu negócio. Devido à inexistência de soluções simples e disponíveis para a segurança da informação em VoIP [Walsh05], algumas delas se sentem temerosas em adotar essa tecnologia, mesmo com os benefícios que essa adoção possa trazer. Essas empresas estão aguardando soluções em VoIP seguras para que possam se beneficiar dessa tecnologia.

Este trabalho tem como objetivo realizar um estudo sobre a segurança da informação em VoIP em um PC-PBX configurado para um ambiente corporativo, abordando os três aspectos da segurança da informação: confiabilidade; integridade; e disponibilidade. O estudo visa oferecer informações importantes para:

- administradores de segurança em redes que pretendem utilizar soluções VoIP mais seguras e mais estáveis nas empresas;
- pesquisadores e desenvolvedores que queiram implementar técnicas específicas para a segurança em VoIP.

Inicialmente serão demonstradas algumas fragilidades de uma implementação VoIP comum utilizando o Asterisk, que é uma solução em software livre de um PC-PBX (central telefônica implementada em *software*), e, depois, serão analisadas as implicações da implementação de algumas técnicas convencionais básicas de segurança em rede, como a utilização de *Firewalls* [Cobb04] e VPN (*Virtual Private Network*) [Davidson00], além da implementação de técnicas para tentar garantir a qualidade de serviço (QoS – *Quality of Service*) [Colcher05].

A estrutura deste trabalho foi dividida em 5 capítulos no total. No Capítulo 1 foi feita uma introdução ao tema proposto neste trabalho. Para que seja realizado o estudo é necessário inicialmente ter o conhecimento básico dos protocolos, de algumas tecnologias e de alguns conceitos utilizados em VoIP. Essa apresentação é realizada no Capítulo 2. O Capítulo 3 traz informações, conceitos e tecnologias utilizadas na segurança da informação em redes de dados, tais como, *firewall* e criptografia. No Capítulo 4 é encontrado o estudo de caso propriamente dito. Esse traz a preparação dos ambientes, respectivos estudos realizados e conclusões. Um resumo das conclusões e trabalhos futuros é descrito no Capítulo 5.

Capítulo 2

Conceitos Básicos

Para o início do estudo proposto neste projeto, é necessário o conhecimento sobre diversos conceitos, tecnologias e padrões que são objeto de estudo desta seção.

2.1 VoIP

VoIP (*Voice over Internet Protocol*), ou Voz sobre IP é uma tecnologia que permite a transmissão de voz em tempo real sobre uma rede de dados que utiliza o protocolo IP. Esta é uma das fortes tendências do mercado de telecomunicações.

O conceito de VoIP tem sua origem em meados da década de 90, quando se iniciou o reconhecimento do potencial de enviar voz através da Internet, ao invés de utilizar os meios tradicionais. Em 1995, surgiu o primeiro *softphone*, que é uma aplicação multimídia que utiliza VoIP para realizar chamadas a partir de um computador. Inicialmente, era necessário que ambos os computadores participantes da chamada possuíssem a mesma aplicação, além de uma placa de som com um microfone e caixas de som ligadas a ela. Esses *softphones* tinham péssima qualidade de som e conectividade, mas já indicavam o potencial da tecnologia.

Devido ao crescente aumento na largura de banda disponibilizada para a transmissão de dados, conjuntamente com a comercialização de produtos específicos para VoIP, ainda no final da década de 90, a qualidade da comunicação VoIP teve um grande avanço.

VoIP, diferentemente da tecnologia utilizada nas redes telefônicas convencionais, não utiliza uma rede específica para a transmissão da voz, pois, na rede IP, diversos tipos de mídia podem ser transmitidos, tais como, textos, imagens estáticas, áudio e vídeo.

Em uma rede IP, todas as informações são enviadas em pacotes, que são pedaços de mensagens encapsuladas por cabeçalhos de protocolos [Colcher05].

Em VoIP, o sinal de voz é transformado em sinal digital e este é dividido em pacotes para serem enviados pela rede IP. Cada pacote possui o endereço de destino e é transmitido pela rede nó a nó, ou seja, em cada nó ele é integralmente recebido e o próximo caminho da rota é determinado com base no endereço contido no mesmo. O caminho que foi determinado para o pacote pode não estar disponível para enviá-lo imediatamente. Dessa forma, o pacote precisará aguardar numa fila até chegar a sua vez de ser enviado, causando atraso no seu recebimento. Esse atraso, pode trazer prejuízos para a qualidade do serviço VoIP, causando até a impossibilidade da comunicação.

Os requisitos de qualidade de serviço (QoS - *Quality of Service*) variam de acordo com o tipo de serviço, ou seja, em geral, é uma função do tipo de aplicação e da mídia transmitida. A transmissão de um arquivo de texto, por exemplo, pode ter atraso nos seus pacotes que são transmitidos através da rede IP, porém, não admitirá perdas dos mesmos, pois tais perdas poderão causar prejuízos não-aceitáveis à informação transmitida. Em contrapartida, há outros tipos de aplicações que não são tão sensíveis à perda de pacotes. Ou seja, no caso de perda de uma pequena porcentagem, a informação transmitida não perderá o seu valor, como é o caso de algumas aplicações de tempo real. Já com relação ao atraso dos pacotes, essas aplicações são muito sensíveis, não admitindo tempos de transmissões muito altos e variações nos atrasos de cada pacote.

Uma chamada telefônica utilizando VoIP, por exemplo, pode se tornar inviável caso haja grandes atrasos na transmissão de pacotes. Além disso, uma vez que cada segundo de voz é dividido em pequenos pacotes para serem enviados pela rede de dados, os atrasos não uniformes desses pacotes (*jitter*) podem fazer com que eles cheguem fora da seqüência correta. Nesse caso, é necessário que sejam reordenados pela aplicação, já que o transporte dos dados é baseado em UDP (*User Datagram Protocol*), que não faz a ordenação na camada de transporte. Problemas com o *jitter* também podem tornar a comunicação inviável pois adiciona atraso na comunicação, devido à reordenação dos pacotes e, mesmo que os pacotes cheguem em ordem, fará com que os pacotes cheguem ao seu destino em rajadas. Alguns protocolos implementam *buffers* específicos para o tratamento do *jitter*. Esses *buffers* armazenam uma certa quantidade de pacotes para tentar solucionar o problema da variação no atraso. Caso alguns pacotes cheguem muito atrasados, eles serão descartados. Outro fator que pode inviabilizar a chamada é a perda de pacotes. Apenas 5% de perda de pacotes já pode comprometer uma chamada VoIP [Walsh05].

Em *links* dedicados, pode-se garantir a qualidade de uma chamada VoIP através de alguns recursos, como, por exemplo, a reserva de largura de banda necessária entre os nós por onde a chamada passa. Ao utilizar a Internet para realizar chamadas VoIP é possível adotar alguns recursos para tentar minimizar os problemas da chamada, porém como não se detém o controle de todos os nós do caminho da chamada, não é possível garantir a inexistência de problemas como se pode fazer em *links* dedicados.

A principal vantagem na utilização de VoIP é a redução nos custos de ligações telefônicas para empresas que possuem filiais em diferentes estados ou países e que estão interconectadas por rede IP através de um *link* dedicado ou Internet. Nesses casos, as ligações realizadas entre elas terão custo zero.

Empresas que, por alguma necessidade específica, ou por pertencerem a um determinado ramo de negócios, precisam estar constantemente efetuando ligações para outros estados ou países, podem também utilizar VoIP através de um ITSP (*Internet Telephony Service Provider*) para realizar ligações DDD/DDI com um custo menor que os normalmente cobrados pelas concessionárias tradicionais.

ITSP são provedores de telefonia pela *Internet*, que permitem que usuários utilizem sua estrutura para realizar uma chamada para a RTPC (Rede Telefônica Pública Comutada) através da rede. Por concentrar muitas ligações, eles conseguem descontos nas tarifas normais aplicadas pelas concessionárias telefônicas e repassam esses descontos para os seus usuários, fazendo com que os seus custos sejam inferiores aos das concessionárias tradicionais.

A popularização do uso residencial da tecnologia VoIP iniciou-se com a queda dos preços das conexões de banda larga e alguns *softwares* como, por exemplo, o Skype [Skype06] e o MSN Messenger [Messenger06] que permitem que duas pessoas falem entre si sem custo adicional.

As aplicações VoIP utilizam um modelo hierárquico de camadas de protocolos, ou seja, as funções desempenhadas num processo de comunicação são distribuídas em camadas. Vários protocolos são utilizados nas diversas fases da comunicação. Dessa forma, cada protocolo

adiciona um cabeçalho contendo informações de controle para a transmissão do pacote. Além dos protocolos de rede mais usuais, tais como, IP, TCP, UDP e RTP/RTCP (*RTP Control Protocol*) [Colcher05], uma chamada VoIP necessita de outros protocolos ou recomendações. Entre esses podem ser citados, por exemplo, o H.323, SIP, MGCP (*Media Gateway Control Protocol*) [Colcher05] e IAX. Na Figura 1 é apresentada uma estrutura em camadas de alguns protocolos utilizados em VoIP, separada pelas diferentes fases da comunicação: sinalização; controle de gateway e transporte de mídia. Os pacotes são encapsulados hierarquicamente de cima para baixo.

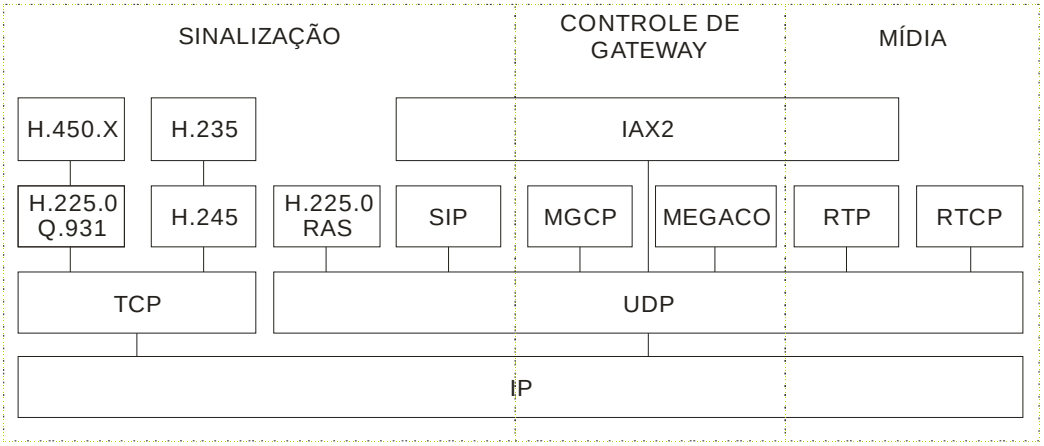

Figura 1. Estrutura em camadas de alguns protocolos utilizados em VoIP.

Diferentes graus de complexidade podem ser encontrados em implementações VoIP dependendo diretamente do cenário proposto. Segue, como exemplo, alguns cenários em ordem crescente de complexidade de implementação:

- VoIP entre terminais IP – neste cenário, terminais VoIP, como, por exemplo, *softphones*, se comunicam através de uma rede IP;
- VoIP de terminal IP para a RTPC (Rede Telefônica Pública Comutada) – neste caso, terminais IP se comunicam com um telefone comum na rede de telefonia pública. Para que isso seja possível é necessário o uso de *gateways* de voz, que serão explanados nas próximas sessões. Como exemplo, pode-se citar um usuário utilizando um *softphone* em uma estação que liga para uma linha telefônica convencional;
- VoIP da RTPC para a RTPC – neste caso, a ligação é originada a partir de um telefone da rede pública convencional com destino a um *gateway* de voz. Esse *gateway* encaminha a chamada para um outro *gateway* que, então, finaliza a chamada em uma linha telefônica convencional. Um exemplo deste cenário é a disponibilização de número telefônicos pelos ITSP para que sejam realizadas chamadas para outros destinos utilizando VoIP (menor custo), sem a necessidade da aquisição de equipamentos VoIP específicos.

Apesar de existirem protocolos que têm o objetivo de tornar a comunicação mais segura como, por exemplo, o SIPS (*SIP Secure*) e o SRTP (*Secure RTP*), os mesmos não foram objetos de estudo deste projeto devido à inexistência de implementação desses protocolos para uso no cenário proposto neste estudo.

2.1.1 H.323

A recomendação H.323 foi definida pela ITU-T (*International Telecommunications Union Telecom Standardization Sector*) e especifica sistemas de comunicação multimídia em redes comutadas por pacotes que não provêem uma qualidade de serviço (*Quality of Service* - QoS) garantida. Esta recomendação define os componentes presentes em um sistema H.323, além dos fluxos de informações previstos entre esses componentes. Os protocolos utilizados no transporte desses fluxos também são indicados pela recomendação H.323.

O escopo da recomendação H.323 é extenso e flexível, dando suporte a uma série de situações. Alguns de seus benefícios são citados a seguir:

- suporte a conferências ponto-a-ponto e multiponto: uma chamada H.323 multiponto pode ser realizada sem a utilização de um *software* ou equipamento multiponto específico, ou seja, a utilização de uma unidade de controle multiponto (*Multipoint Control Unit* – MCU) para oferecer um serviço de controle centralizado para estabelecimento da chamada não é obrigatória;
- heterogeneidade: uma chamada H.323 pode envolver equipamentos com configurações multimídia diferentes. Equipamentos em um sistema H.323 somente têm a obrigação de suporte à comunicação de áudio, não sendo necessário o suporte a vídeo, texto e imagem estática. No caso de uma conferência em que as capacidades dos equipamentos sejam heterogêneas, os equipamentos trocam informações sobre as suas capacidades e a comunicação é estabelecida com base no maior denominador comum entre as capacidades;
- suporte a contabilidade e gerência: a recomendação H.323 provê mecanismos que permitem a delimitação do número de chamadas simultâneas e da largura de banda utilizada para aplicações H.323. Além disso, possibilita a contabilização dos recursos de rede utilizados, podendo ser usada na tarifação dos serviços;
- segurança: a recomendação H.323 prevê suporte à autenticação de usuários, à confidencialidade e ao não-repúdio da informação trocada entre os participantes de uma chamada;
- serviços suplementares: a recomendação H.323 permite a elaboração e o desenvolvimento de serviços adicionais, tais como chamada em espera e identificação da chamada.

Uma comunicação H.323 é formada por quatro componentes principais. A Figura 2 mostra a organização típica de um sistema de comunicação H.323.

A seguir são relacionados os quatro componentes de um sistema H.323:

- terminais: são fontes ou destinos dos fluxos de informação de um sistema H.323. Exemplos de terminais: telefone IP, *softphone* (aplicação que executa em um computador pessoal com recursos multimídia), etc.;
- *gateways*: são necessários quando se deseja estabelecer uma comunicação entre terminais de padrões diferentes, como, por exemplo, numa comunicação entre um terminal H.323 e terminais H.3xx, ou numa comunicação entre terminais H.323 e aparelhos em uma rede telefônica comutada por circuitos. Para possibilitar essa interoperabilidade o *gateway* faz a conversão do formato de codificação de mídias e a tradução dos procedimentos de estabelecimento e encerramento de chamadas;

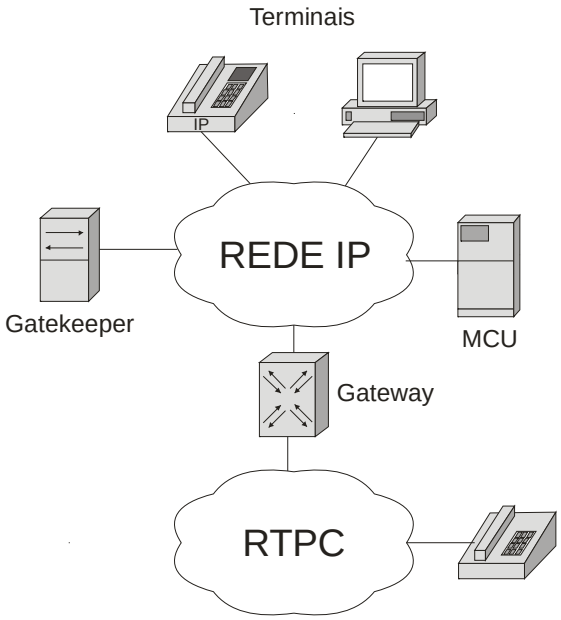

Figura 2. Organização típica de um sistema H.323.

- *gatekeepers*: são componentes opcionais. Apesar disso, sua utilização permite um controle centralizado do sistema. Ou seja, ele atua como centralizador para todas as chamadas dentro de sua zona, que é o conjunto de todos os terminais, *gateways* e MCUs gerenciados por um único *gatekeeper*. Mensagens de sinalização e controle em uma zona H.323 podem ser roteadas pelo seu *gatekeeper*, possibilitando melhor suporte a funções de contabilidade, gerência, segurança, re-roteamento de chamadas, etc.;
- Unidades de Controle Multiponto (MCUs): são componentes opcionais. Uma MCU suporta conferências entre três ou mais participantes e consiste de um controlador multiponto (*Multipoint Controller* – MC) e zero ou mais processadores multiponto (*Multipoint Processors* – MPs). O MC controla o processo de negociação de parâmetros em uma chamada multiponto. Os MPs são responsáveis por encaminhar os fluxos de informações (áudio, vídeo, texto e imagem estática) entre os participantes de uma conferência. Além disso, possibilitam o processamento desses fluxos, podendo, por exemplo, mesclar dois fluxos de áudio gerados por participantes distintos em um único fluxo para que o mesmo possa ser distribuído para os demais participantes de uma conferência.

A recomendação H.323 depende de vários outros padrões. Abaixo estão listados alguns protocolos e recomendações adotadas em sistemas H.323:

- T.120
  A recomendação T.120 [Colcher05] é referenciada pela recomendação H.323 como padrão para operar o transporte de dados (texto e imagens estáticas) em conferências multimídia. A T.120 dá suporte a uma série de atividades, tais como transferência de arquivos, envio de mensagens instantâneas, entre outras.
- RTP, RTCP (IETF)

O protocolo RTP em conjunto com o RTCP é referenciado pela recomendação H.323 como padrão para o transporte fim a fim em tempo real de pacotes de áudio e vídeo. O RTCP monitora a entrega dos dados, provendo funções mínimas de controle, uma vez que o RTP não garante qualidade de serviço em tempo real.

- H.245, H.225.0
  A recomendação H.245 [Colcher05] é referenciada pela H.323 para o procedimento de controle de mídias. Esse controle exige algumas funcionalidades principais, tais como:
  - troca de informação sobre as capacidades dos terminais com o intuito de estabelecer algumas características para transmissão e recepção de uma chamada, tais como, tipos de mídia aceitos e *codecs* [Colcher05] utilizados;
  - estabelecimento e encerramento de canais lógicos;
  - determinação dos terminais mestre e escravo, para evitar conflitos entre os mesmos;
  - notificações entre terminais sobre condições especiais nos canais lógicos estabelecidos entre eles.

  A recomendação H.225.0 [Colcher05] é referenciada pela H.323 para a sinalização de chamadas. Esta recomendação define um mecanismo específico para troca de informações de registro, admissão e estado entre terminais e *gatekeepers*. Essa troca de informações é realizada através de mensagens RAS (*Registration, Admission, Status*) [Colcher05].
- H.235
  Sistemas H.323 que dão suporte à Recomendação H.235 [Colcher05], permitem as seguintes funcionalidades:
  - mensagens de sinalização H.225.0 podem ser enviadas por um canal seguro;
  - usuários podem ser autenticados durante o estabelecimento da chamada, durante o processo de estabelecimento de privacidade no canal de controle H.245, ou pela troca de certificados nesse canal;
  - possibilidade de troca de capacidades criptográficas através da adição de campos nas mensagens de troca de capacidades definidas na recomendação H.245;
  - uso de chaves criptográficas nos canais de mídia.
- H.450.x
  A série de recomendações H.450 [Colcher05] define um modelo geral para o desenvolvimento de serviços suplementares, tais como transferência e redirecionamento de chamadas, chamada em espera, identificação da chamada, entre outros.

Os padrões referenciados na recomendação H.323 formam uma pilha organizada conforme Figura 3.

Uma chamada H.323 pode ser dividida em cinco fases:

- estabelecimento da chamada – As chamadas são estabelecidas através da troca de mensagens, tais como: *Setup* (solicitação de estabelecimento de chamada), *CallProceeding* (informação sobre o início do processo de chamada) e *Connect* (indicação para o requisitante que a chamada foi estabelecida);

<table>
<tr><td colspan="9">Aplicações Multimídia</td></tr>
<tr><td colspan="3">Transporte de dados</td><td colspan="3">Transporte de áudio e vídeo</td><td colspan="3">Controle e sinalização</td></tr>
<tr><td rowspan="2">V.150.1</td><td rowspan="2">T.120</td><td rowspan="2">T.38</td><td>Codecs Áudio</td><td>Codecs Vídeo</td><td rowspan="2">RTCP</td><td rowspan="2">H.225.0 (sinalização)</td><td rowspan="2">H.245</td><td rowspan="2">H.225.0 mensagens RAS</td></tr>
<tr><td colspan="2">RTP</td></tr>
<tr><td>UDP</td><td>TCP</td><td>TCP/UDP</td><td colspan="3">UDP</td><td>TCP/UDP</td><td>TCP</td><td>UDP</td></tr>
<tr><td colspan="9">IP</td></tr>
</table>

Figura 3. Pilha organizada dos padrões referenciados na recomendação H.323.

- estabelecimento do canal de controle H.245 – Após a primeira fase, pode ser estabelecida uma camada de controle H.245 sobre a chamada;
- estabelecimento de canais lógicos – Após a troca das capacidades, os canais lógicos são estabelecidos;
- uso dos serviços da chamada – Diversos serviços podem ser executados durante o curso da chamada. Dentre eles, alguns são obrigatórios, tais como: alteração de banda passante e a verificação do estado dos terminais. Serviços suplementares são referenciados na recomendação H.323 para indicar serviços que podem ser providos opcionalmente;
- encerramento da chamada – Mensagens são enviadas para indicar o encerramento dos canais de mídia e o de controle.

2.1.2 SIP

O protocolo SIP, definido pelo IETF (*Internet Engineering Task Force*), é um protocolo do nível de aplicação que permite o estabelecimento, gerenciamento e encerramento de sessões multimídia, como, por exemplo, chamadas telefônicas através da rede IP. O SIP negocia as características de uma sessão, definindo, por exemplo, os tipos de mídia utilizados na sessão. Além disso, ele pode auxiliar na localização dos participantes de uma chamada.

Ao contrário do H.323, o SIP não é um sistema verticalmente integrado. O SIP é um elemento que pode ser usado em conjunto com outros protocolos e componentes na construção de uma arquitetura multimídia completa.

Normalmente, o protocolo SIP trabalha conjuntamente com outros protocolos, tais como:

- RTP e RTCP, que são usados para transportar dados em tempo real e monitorar esse transporte;
- RTSP (*Real-time Streaming Protocol*) [Colcher05] que é utilizado para a distribuição de conteúdo multimídia em tempo real;
- MGCP (*Media Gateway and Control Protocol*) [Colcher05], que é utilizado no controle de *gateways* de mídia sobre redes IP;
- Megaco [Colcher05], que é um protocolo de sinalização utilizado entre *Media Gateways* e *Media Gateways Controllers.* O Megaco é uma versão mais atualizada e com mais recursos da MGCP;
- SDP (*Session Description Protocol*) [Colcher05], que especifica o formato para a descrição das informações em sessões multimídia.

Os componentes da arquitetura de sinalização do SIP são definidos como clientes e servidores:

- Agente Usuário (UA – *User Agent*): é formado por uma parte cliente (*UAC – User Agent Client*), que efetua requisições SIP, e uma parte servidor (*UAS – User Agent Server*), que recebe e responde as requisições;
- Servidor Proxy (*proxy server*): atua como um servidor e como um cliente, ou seja, ele intermedia requisições de outros clientes que não podem fazer as requisições diretamente;
- Servidor de redirecionamento (*redirect server*): permite o mapeamento de um endereço em zero ou mais novos endereços associados a um cliente. Ou seja, fornece informações ao Agente Usuário sobre o endereço do servidor para que ele possa conectar diretamente;
- Servidor de registro (*registrar server*): trabalha em conjunto com o servidor de redirecionamento e o servidor *proxy* para armazenar informações sobre a localização de um terminal.

Essa existência de cliente e servidor no mesmo agente usuário SIP permite a comunicação *peer-to-peer* (P2P) com outros agentes sem a necessidade de utilizar servidores. O agente SIP é normalmente implementado em telefones IP, *softphones* ou adaptadores de telefones analógicos (ATA – *Analog Telephone Adaptor*).

Quando o escopo aumenta para sistemas de telefonia baseados em SIP, os outros componentes assumem papéis fundamentais. O servidor *proxy* pode, por exemplo, reescrever o cabeçalho das mensagens antes de enviar para o destinatário, que pode ser um outro servidor *proxy* ou um agente. O servidor de redirecionamento exibe o próximo nó para onde o a mensagem SIP deve ser encaminhada.

Existem dois tipos de mensagens SIP: as mensagens de requisição e as mensagens de respostas.

Como o protocolo é baseado em texto, as suas mensagens utilizam o conjunto de caracteres UTF-8 (*Universal Transformation Format*), similar aos protocolos HTTP (*Hyper Text Transfer Protocol*) [Kurose03] e SMTP (*Simple Mail Transfer Protocol*) [Kurose03], e devem obedecer a regra de formação do Quadro 1.

linha-de-requisição ou linha-de-status cabeçalho <linha em branco> corpo-da-mensagem

Quadro 1. Regra de formação de mensagem SIP.

A linha de requisição é formada por um método, um endereço e a identificação da versão SIP utilizada. Os métodos de requisição e respectivas funcionalidades no SIP/2.0 estão especificados na Tabela 1.

Tabela 1. Métodos de requisição e funcionalidades do SIP/2.0.

Método	Funcionalidade
ACK	Confirma o recebimento da mensagem final "200 (OK)" do INVITE.
BYE	Requisição de término da sessão.
CANCEL	Solicita cancelamento de um INVITE.
INVITE	Requisição para estabelecimento de uma sessão.
OPTIONS	Consulta sobre as capacidades de um servidor ou cliente SIP.
REGISTER	Associa uma SIP URI com um determinado agente SIP.

A mensagem de resposta SIP é caracterizada pelo uso de uma linha de *status* como uma linha de início. A linha de *status* possui a versão SIP utilizada, um código de *status* numérico e seu significado correspondente, que não é um campo obrigatório.

A Tabela 2 exibe as seis classes de resposta representadas pelo primeiro número do código de *status* especificados no SIP/2.0. Os outros dígitos não representam nenhuma categoria.

Tabela 2. Classes de respostas, funcionalidade e exemplos no SIP/2.0.

Classe	Função das Respostas	Exemplos
1xx	Informativas	100 Trying 180 Ringing
2xx	Sucesso	200 OK 202 Accepted
3xx	Redirecionamento	300 Multiple choices 302 Moved temporarily
4xx	Falha na requisição do cliente	403 Forbidden 404 Not found
5xx	Falha no servidor	500 Server internal error 503 Service unavailable
6xx	Falha Global	600 Busy everywhere 606 Not acceptable

Os cabeçalhos das mensagens SIP são constituídos de uma seqüência estruturada de campos. Esses campos se assemelham aos campos do cabeçalho da mensagem HTTP. Eles podem ser incluídos nos dois tipos de mensagens: de requisição e de resposta. Dependendo da mensagem, alguns campos podem ser obrigatórios, outros opcionais e alguns não aplicáveis. No Quadro 2 pode-se visualizar exemplos de campos de cabeçalho em uma mensagem SIP de requisição de chamada (*INVITE*) originada a partir do ramal 11 (campo *From*) do agente "Sipura/SPA2002-3.1.2(a)" (campo *User-Agent*) para o ramal 10 no servidor 10.0.0.254 (campo *To*).

```
Via: SIP/2.0/UDP 10.0.0.200:5060;branch=z9hG4bK-bf5dce6
From: Ramal 11 <sip:ramal11@10.0.0.254>;tag=2286ea32734e1306o0
To: <sip:10@10.0.0.254>
Call-ID: f35aaa2-b80cd56@10.0.0.200
CSeq: 101 INVITE
Max-Forwards: 70
Contact: Ramal 11 <sip:ramal11@10.0.0.200:5060>
Expires: 240
User-Agent: Sipura/SPA2002-3.1.2(a)
Content-Length: 420
Allow: ACK, BYE, CANCEL, INFO, INVITE, NOTIFY, OPTIONS, REFER
Supported: x-sipura
Content-Type: application/sdp
```

Quadro 2. Exemplos de campos de cabeçalho.

As informações importantes para as operações são transmitidas no corpo da mensagem SIP. O tipo da informação transportada, caso esteja presente, é indicada no campo *Content-Type*, em conformidade com o padrão MIME (*Multipurpose Internet Mail Extensions*). A informação referente ao SDP (*Session Description Protocol*), indicada pelo valor *application/sdp*, no campo *Content-Type*, é utilizada pelos participantes para negociar os parâmetros necessários para se estabelecer a sessão.

Para estabelecer uma sessão multimídia é necessário que haja uma negociação sobre qual será a mídia utilizada e sobre as informações necessárias para a transmissão da mídia escolhida, tais como, o protocolo para transmissão e o *codec* (*Coder/Decoder*) escolhido.

Enquanto o SIP especifica o processo pelo qual se anuncia a descrição dos parâmetros de inicialização de uma sessão, o SDP especifica o formato para descrição dessas informações, representada de forma textual e, ao contrário das representações binárias, não economizam largura de banda. A forma textual é adotada para facilitar a portabilidade, permitir várias formas de transporte e possibilitar que ferramentas baseadas em texto possam gerar e processar as descrições das sessões. Para diminuir a ineficiência no consumo de banda, a descrição é especificada de uma forma compacta, substituindo os nomes dos campos por caracteres únicos.

O endereçamento SIP segue as recomendações de formações de URI (*Universal Resource Identifier*) definidas pela RFC (*Request For Comment*) 3986. A sua forma geral de apresentação e exemplos da uma URI SIP é apresentada no Quadro 3.

```
sip:usuário:senha@host:porta;parâmetros-uri?cabeçalhos

sip:ramal11@10.0.0.254
sip:+55-81-3333-3333:1234@exemplo.com;user=phone
sip:host.com;method=REGISTER?to=maria%40host.com
```

Quadro 3. Forma geral e exemplos de URI SIP.

Uma comunicação baseada em SIP pode ser de dois modos:

- *peer-to-peer* – quando dois agentes SIP se comunicam diretamente um com o outro, devido à existência de uma parte cliente e uma parte servidor no mesmo agente, conforme a especificação. A Figura 4 apresenta um exemplo de fluxo de

sinalização da comunicação *peer-to-peer* de um agente A com um agente B. Inicialmente o agente A requisita uma chamada (*INVITE*) para o agente B, que responde informando que está chamando (*180 Ringing*). Quando a chamada é atendida o agente B informa para o agente A (*200 OK*), que responde informando que recebeu a mensagem (ACK). A chamada então é estabelecida até que um dos agentes desligue. Neste caso, o agente A envia uma mensagem de desconexão (BYE) e o agente B informa que a recebeu (200 OK), encerrando a chamada;

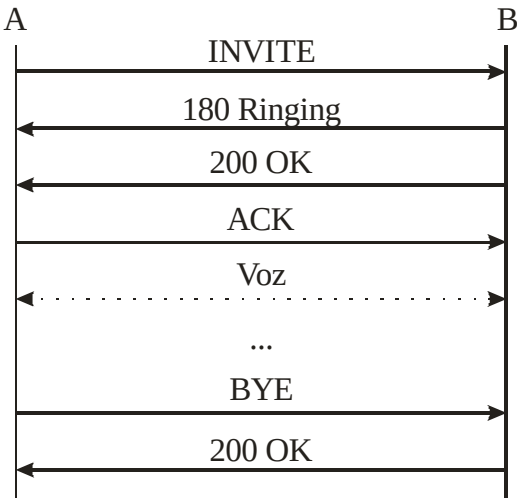

Figura 4. Exemplo de fluxo de sinalização *peer-to-peer*.

- via *proxy* – é um modo indireto de comunicação que passa por um servidor *proxy* e/ou outros servidores de apoio. A Figura 5 apresenta um exemplo de fluxo de sinalização da comunicação de um agente A com um agente B através de um *proxy*. A diferença do que foi apresentado na Figura 4, é que, neste caso, o servidor *proxy* fica responsável por encaminhar as mensagens entre os agentes e quando recebe a requisição de chamada do agente A (*INVITE*) informa-o de que irá requisitar a chamada para o agente B (*100 Trying*).

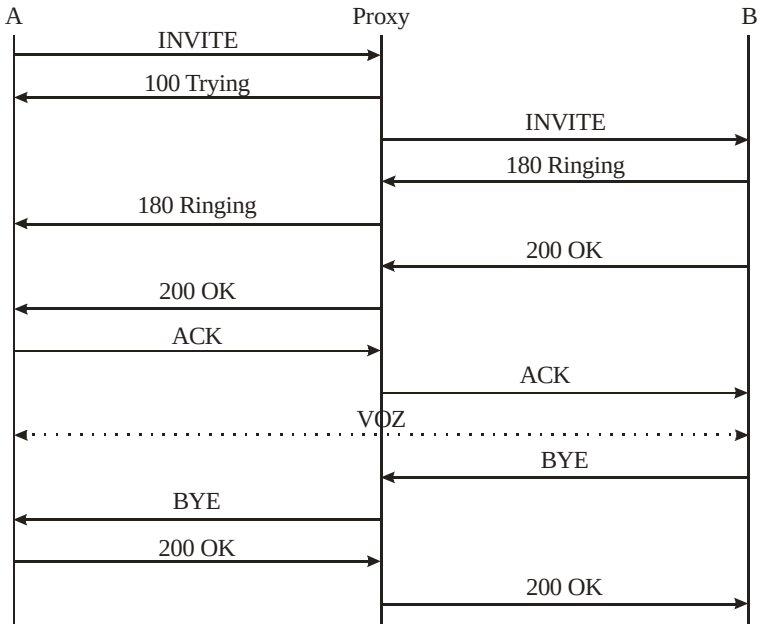

Figura 5. Exemplo de fluxo de sinalização via Proxy.

São necessários alguns cuidados na implementação de telefonia baseada em SIP, sobretudo quando estes estão por trás de um dispositivo que faça NAT (*Network Address Traslation*) [Walsh05]. NAT é uma técnica para possibilitar que dispositivos interligados em uma rede privada possam utilizar apenas um IP público válido para acessar a Internet. O dispositivo na rede local, ao tentar acessar um destino na Internet tem o seu cabeçalho modificado para o IP público válido pelo dispositivo que faz o NAT. Esse mesmo dispositivo, ao receber a resposta da requisição, substitui o cabeçalho para o endereço de rede local e devolve ao solicitante.

Como citado anteriormente, diversos parâmetros da comunicação SIP são transferidos dentro das mensagens, como, por exemplo, o IP e a porta que são utilizados para a sinalização e para a transferência dos dados. Um dispositivo SIP que utiliza NAT para acessar a Internet apenas conhece o IP e a porta da rede privada. Ao acessar a Internet o endereço do cabeçalho IP será substituído pelo IP público, enquanto o conteúdo da mensagem SIP não será substituído. Dessa, forma a mensagem SIP e o cabeçalho IP possuem IPs diferentes, fazendo com que a chamada não consiga ser estabelecida.

Um dos recursos que podem ser utilizados para solucionar esse problema é a sinalização simétrica que faz com que os dispositivos enviem e recebam as mensagens através da mesma porta.

Além da sinalização, deve-se ter a mesma preocupação com o canal de mídia. Em alguns casos de implementações incorretas de SIP, poderá ocorrer o tráfego de mídia em apenas uma direção.

2.1.3 IAX/IAX2

O protocolo IAX permite o controle e a transmissão de fluxo de dados através de redes IP. Ele permite a transmissão de qualquer tipo de fluxo de dados incluindo vídeo, porém seu objetivo principal é o controle de chamadas VoIP.

O IAX, na sua segunda versão, é utilizado pelo PC-PBX Asterisk [Goncalv05] como uma alternativa a outros protocolos VoIP, como H.323, MGCP e SIP, para se conectar a dispositivos que suportam esse protocolo, tais como: outro PC-PBX Asterisk e ATAs (*Analog Telephone Adaptor*).

O IAX ainda não é um padrão, e sim o resultado de um esforço conjunto da comunidade de *software* livre. Diferentemente dos outros protocolos, como o H.323 e o SIP, o IAX utiliza apenas uma porta UDP para o controle e a transmissão dos dados. O IAX2 utiliza a porta UDP 4569. Sua primeira versão, já obsoleta, utilizava a porta UDP 5036.

O desenvolvimento do protocolo IAX tomou como base de referência protocolos VoIP já existentes como o SIP e o MGCP para o controle das conexões e o RTP para a transmissão do fluxo de dados. Ele teve como objetivos principais: a redução do consumo de banda; suporte a NAT transparente; transmissão de informações de plano de discagem e suporte a implementação de interfones.

A sua sinalização é mais similar ao SIP do que ao MGCP, que utiliza o controle de chamadas *Master-Slave*, e o transporte dos dados não utiliza o RTP.

O IAX multiplexa a sinalização e o transporte de dados no mesmo canal UDP entre dois nós. Dessa forma ele atua como dois protocolos em um só. Essa característica o faz distinto entre os outros protocolos (H.323, SIP e MGCP), que utilizam controle e transmissão de dados separados.

Como a sinalização e a transmissão de dados utiliza a mesma porta UDP, o IAX não sofre de problemas com o NAT, como, por exemplo, o SIP. Diferentemente do SIP, caso a conexão seja estabelecida, haverá áudio em ambos os sentidos.

O IAX foi desenvolvido como um protocolo binário para ser mais eficiente em relação ao consumo de banda, além do que, ele é otimizado especificadamente para a redução do consumo de banda para chamadas de voz.

O IAX suporta autenticação através de chaves assimétricas (públicas e privadas) e *trunking*.

O *trunking* permite que múltiplos *streams* de voz compartilhem o mesmo canal, reduzindo o *overhead* causado pelos pacotes IP, ou seja, ele remove a redundância do pacote IP para cada *stream*. Isso só é possível quando as chamadas são entre os mesmos nós. Através do *trunking*, o consumo de banda não é um múltiplo simples da largura de banda utilizada por cada canal. O primeiro canal utilizará um certo valor de largura de banda, dependendo do *codec* utilizado, e, a partir do segundo canal, a largura de banda necessária é menor devido a não existência do mesmo *overhead* nos pacotes IPs, já que todo o tráfego é feito pela mesma porta UDP.

As principais diferenças entre o SIP e o IAX são:

- dispositivos com suporte a IAX estão começando a ser produzidos, enquanto a maioria dos dispositivos produzidos suportam SIP;
- o IAX tem suporte nativo a NAT, enquanto o SIP, não;
- o IAX é otimizado para a minimização do consumo de banda e suporta *trunking*, ou seja, consome menos banda do que o SIP;
- como é um protocolo binário e não texto, ele é menos suscetível a ataques de *buffer overrun*, já que não é necessária a análise léxica e sintática para interpretação das informações. Os ataques de *buffer overrun* consistem em tentar escrever dados fora do limite estabelecido de um *buffer* fazendo com que a aplicação gere resultados indesejados ou pare de funcionar [Donaldson02];
- o IAX suporta a transmissão de planos de discagem.

A Figura 6 mostra o fluxo básico para o estabelecimento de uma chamada utilizando IAX até a sua finalização. Neste caso, o agente A envia uma mensagem de requisição de chamada (NEW) para o agente B, que responde informando que aceitou (*ACCEPT*) e o agente A informa que recebeu a mensagem (*ACK*). O agente B informa que está chamando (*RINGING*) e o agente A informa novamente que recebeu a mensagem (*ACK*). Quando a chamada é atendida, o agente B informa ao agente A (*ANSWER*), que informa que recebeu a mensagem (*ACK*). A chamada, então, é estabelecida até que um dos agentes desligue. Neste caso, o agente A desliga e envia uma mensagem para o agente B (*HANGUP*), que informa que recebeu a mensagem (*ACK*) encerrando a chamada.

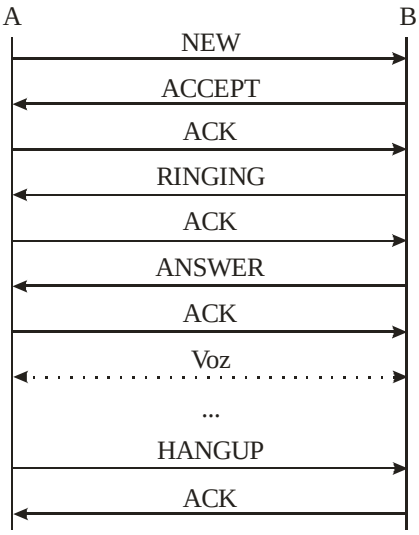

Figura 6. Fluxo básico para estabelecimento de uma chamada IAX.

O IAX utiliza um número de 15 bits para identificar o número da chamada a fim de controlar os múltiplos fluxos de mídia na mesma porta UDP entre dois nós, conforme Figura 7.

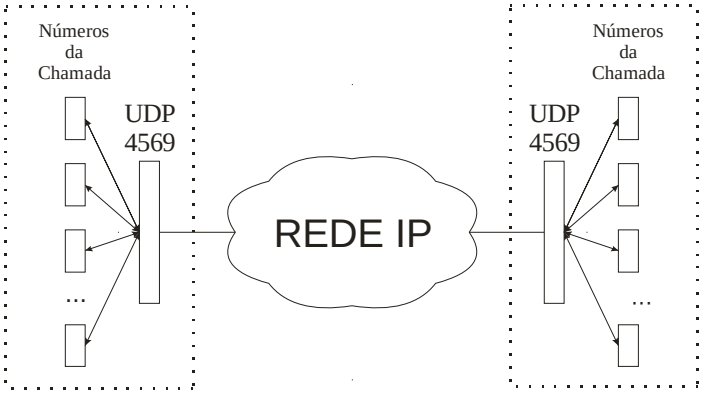

Figura 7. Várias chamadas multiplexadas por apenas uma porta UDP.

A Figura 8 apresenta a máquina de estado do requisitante de uma chamada IAX e a Figura 9 apresenta a máquina de estado do requisitado. Ambos iniciam no estado *Null* até que o requisitante envie uma mensagem de requisição de chamada (*NEW*) para o requisitado. O fluxo básico para o estabelecimento de uma chamada IAX foi apresentado na Figura 6 e está destacado na Figura 8 e na Figura 9.

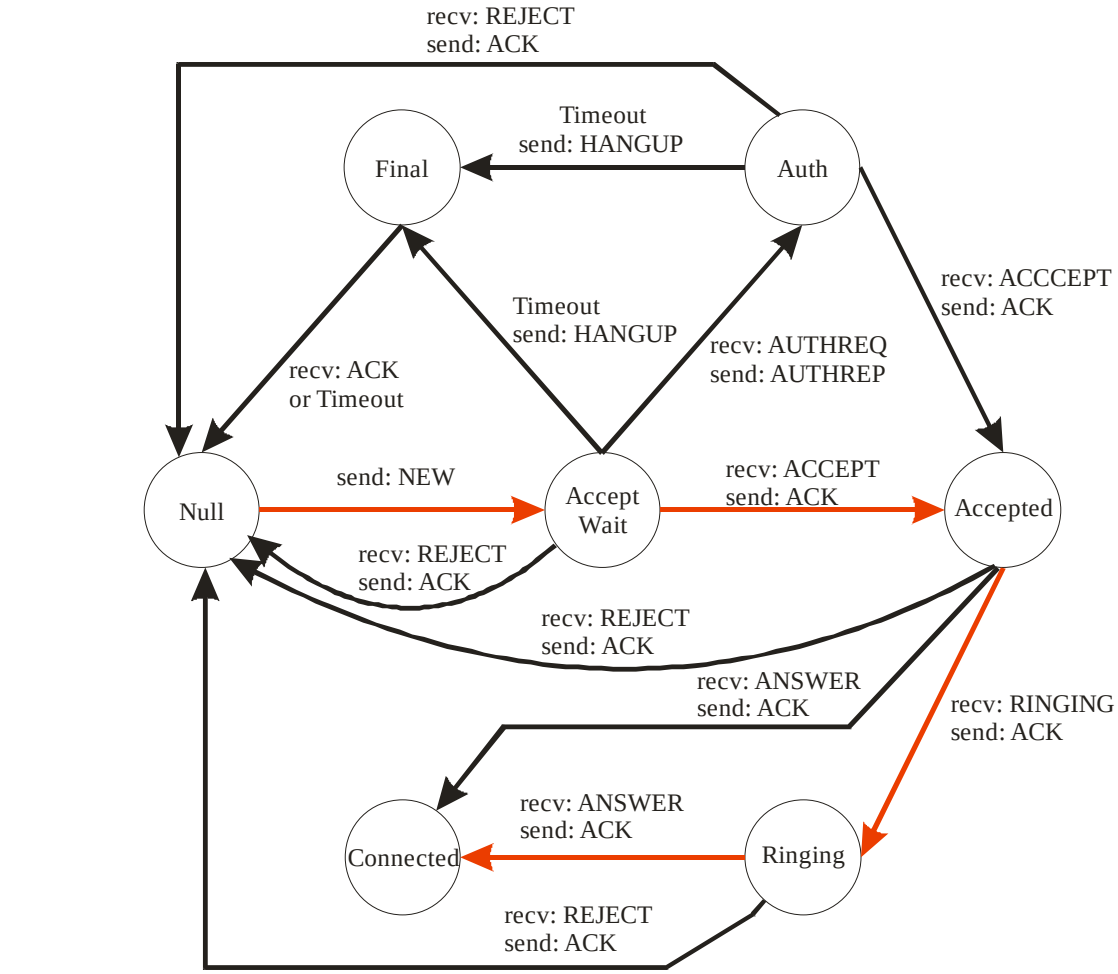

Figura 8. Máquina de estado do requisitante de uma chamada IAX.

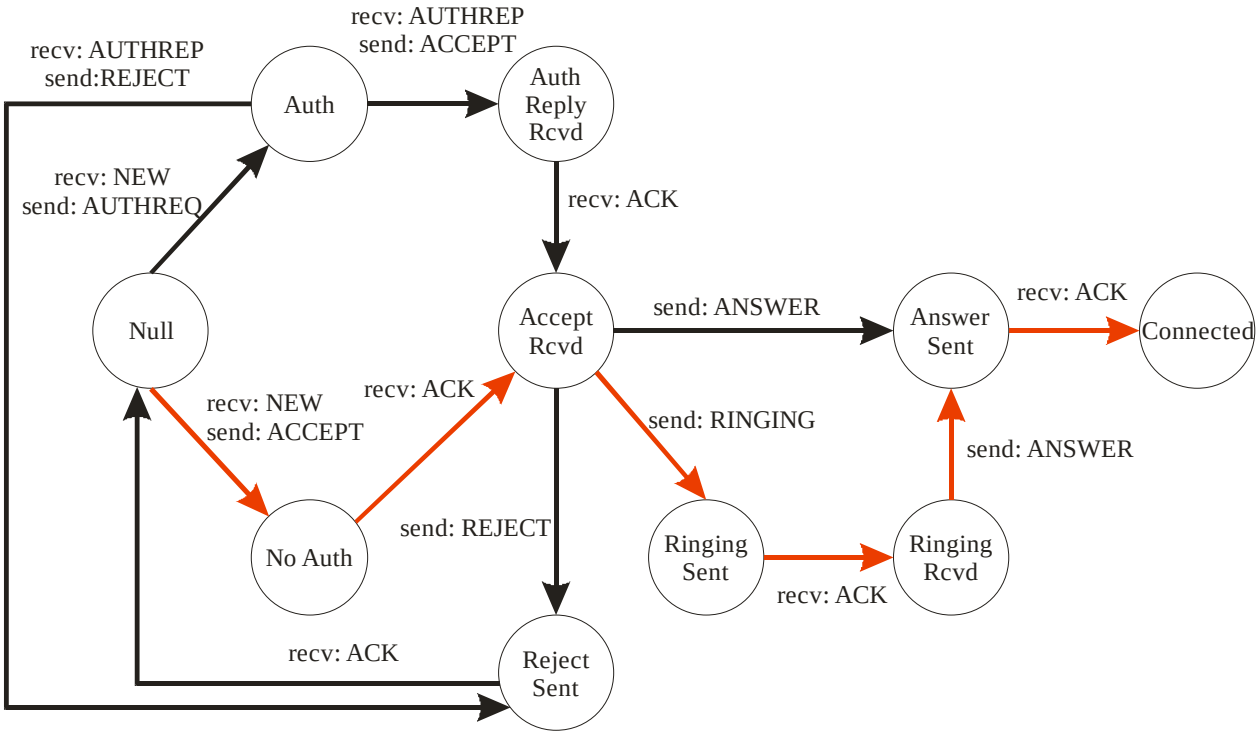

Figura 9. Máquina de estado do requisitado de uma chamada IAX.

2.1.4 CODECS

Codec (*Coder/Decoder*) originalmente referencia a função de codificação e decodificação de dados em um microprocessador. Em computação é comum sua utilização para referenciar algoritimos que fazem a compressão e descompressão (*Compression/Decompression*) de sinais digitais. Na área de telecomunicações, codec é a tecnologia que faz a conversão do sinal analógico para o sinal digital (A/D) e do sinal digital para o sinal analógico (D/A). Pode ser feita por *hardware*, por *software* ou uma combinação de ambos. [Ihets06]

Em VoIP, alguns desses codecs são utilizados para a conversão e compressão dos sinais de voz. Esses codecs diferem em relação à qualidade do áudio, largura de banda necessária para o tráfego dos pacotes convertidos e poder de processamento necessário para a conversão. Alguns codecs são proprietários, ou seja, é necessário pagar licenças para o seu uso, como, por exemplo, o G.729 [Colcher05].

Os codecs podem trabalhar com diferentes tamanhos de quadros (*frame*) e taxas. Os sinais analógicos são divididos em pequenos blocos de n-milisegundos e o tamanho de cada bloco é chamado de tamanho do quadro. A taxa (*rate*) de cada codec é definida como a quantidade de *bytes* gerados pela conversão de 1 segundo de áudio.

É comum confundir a taxa do codec com a largura de banda necessária para o tráfego de 1 segundo de áudio. A largura de banda necessária por codec é bem superior à sua taxa, já que, para o tráfego desses dados, é necessário o encapsulamento dos dados pelos protocolos de rede, principalmente os protocolos IP e UDP.

O tempo de conversão de cada segundo de áudio vai depender do algoritmo utilizado e do poder de processamento do *hardware* ou *software*.

Na Tabela 3 são apresentados alguns exemplos de codecs e suas respectivas características.

Tabela 3. Tabela com codecs e respectivas características.

Codec	Taxas/tamanho do frame
GSM	13 kbps/20 ms
iLBC	15 kbps/20 ms, 13.3 kbps/30 ms
ITU-T G.711 (alaw/ulaw)	64 kbps/10 ms
ITU-T G.723.1	5.3 kbps/30 ms, 6.3 kbps/30 ms
ITU-T G.726	16,24,32,40 kbps/5 ms
ITU-T G.729	8 kbps/10 ms
Speex	2.15 a 44.2 kbps/20 ms
LPC10	2.5 kbps/22.5 ms

2.2 PABX/PC-PBX

Normalmente, o gerenciamento da telefonia nas empresas é feito por um PABX (*Private Automatic Branch eXchange*) [Colcher05], comumente chamado de central telefônica. PABX é um equipamento utilizado para a comutação automática dos circuitos internos e externos de telefonia, ou seja, permitem a existência de uma quantidade maior de telefones internos (ramais) do que a quantidade de linhas externas existentes em uma empresa. Esse equipamento gerencia as

chamadas entre pessoas da empresa (ramal-ramal) e as chamadas externas, que utilizam as linhas externas existentes. Devido ao fim da fabricação dos PBXs, que tinham a mesma função do PABX, mas não eram automáticos, é comum a utilização do termo PBX para referenciar também os PABXs.

Com o desenvolvimento da tecnologia, diversos recursos foram incorporados ao PABX, como, por exemplo: autenticação, atendimento automático, atendimento remoto, caixa postal de voz, CDR (*Call Detail Record*), chamada em espera, conferência, FAX, integração com banco de dados, músicas, transferência de chamadas, redirecionamento de chamadas, URA (Unidade de Resposta Audível), etc.

Normalmente, são investidos milhares de reais na compra de PABXs para o gerenciamento da comunicação telefônica dessas empresas.

Por essa razão, uma alternativa aos PABXs são os PC-PBXs, que são computadores que desempenham a função de PABXs através da implementação de seus recursos via *software*. Utilizando um PC-PBX, as empresas que necessitam de um PABX completo, podem usufruir as mesmas funcionalidades de um PABX tradicional com um custo menor, chegando, em alguns casos, a mais de 50% de economia [Goncalv05].

2.3 ASTERISK

O Asterisk é uma solução em *software* livre de um PABX. Ele pode rodar na plataforma GNU/Linux [Siever00], implementa diversas funcionalidades de um PABX moderno e suporta VoIP com vários protocolos.

O Asterisk foi originalmente desenvolvido por Mark Spencer, da Digium Inc, para a plataforma GNU/Linux e arquitetura x86. O código recebe contribuições de desenvolvedores em *software* livre de todos os lugares do mundo.

O Asterisk foi desenvolvido com o objetivo de interoperar diversos *hardwares* e *softwares* (protocolos, codecs, etc.) diferentes. Por essa razão, ele possui um núcleo (*core*) e APIs (*Application Programming Interface*) específicas que auxiliam o núcleo a realizar a interconexão do PABX, abstraindo outros componentes, tais como: os protocolos e os codecs.

O núcleo do Asterisk é responsável internamente:

- pela comutação de chamadas;
- pela execução dos aplicativos de serviços específicos, como, por exemplo, caixa postal de voz;
- pela conversão de codecs: usa os módulos dos codecs para a conversão entre diferentes codecs;
- pelo gerenciamento de IO (*Input/Output*).

Quatro APIs são definidas para a criação de módulos. São elas:

- *Channel API* – responsável pelo tipo de conexão de recebimento;
- *Application API* – permite a chamada de módulos para a execução de diversos serviços específicos, como, por exemplo, caixa postal de voz;
- *Codec Translator API* – módulos responsáveis pelo suporte a diferentes codecs;
- *File Format API* – responsável pelo suporte a diferentes formatos de arquivos para o armazenamento de dados.

Sua arquitetura modular permite que o núcleo não se preocupe com as tecnologias utilizadas, deixando a cargo dos módulos essa função. Com isso, novas tecnologias e dispositivos podem ser facilmente integrados ao núcleo do Asterisk.

Algumas das funcionalidades do Asterisk são:

- caixa postal de voz com serviço de diretório;
- conferência;
- enfileiramento de chamadas;
- IVR (*Interactive Voice Response*) ou URA (Unidade de Resposta Audível);
- identificador de chamadas.

O Asterisk suporta os protocolos: IAX, H.323, SIP, MGCP e SCCP, e os codecs: ADPCM (*Adaptive Differencial Pulse Code Modulation*), G.711, G.723, G.726, G.729, GSM (*Global System for Mobile Telecommunication*), iLBC (*Internet Low Bitrate Codec*), Linear, LPC-10 e Speex.

Capítulo 3

Segurança da informação

Este capítulo se propõe a introduzir alguns conceitos e tecnologias que podem ser utilizados para garantir a segurança da informação em redes e que auxiliarão no estudo proposto.

3.1 *Firewall*

Firewall é o nome dado ao dispositivo de rede que tem como objetivo principal regular o acesso entre diferentes redes de dados.

Normalmente, esse dispositivo, que pode ser uma combinação de *software* e *hardware*, é utilizado para negar o acesso não autorizado entre redes distintas através da implementação de filtros nas camadas 3 e 4 do modelo OSI (*Open Systems Interconnection*) [Colcher05].

Alguns tipos de *firewall* são:

- de filtro – os pacotes, à medida que são transmitidos da camada de enlace (2ª camada do modelo OSI) para a camada de rede (3ª camada do modelo OSI), são analisados e, de acordo com as regras implementadas no *firewall*, podem, por exemplo, ser descartados, negados ou permitidos. Existem diversos tipos diferentes de regras de filtro, tais como, regras baseadas no endereço de origem, no endereço de destino, nas portas utilizadas (TCP/UDP), etc.
- de *proxy* – *software* que atua como intermediário entre um cliente e um servidor. O *proxy* fica responsável por receber requisições dos clientes, refazê-las como se ele fosse o requisitante e repassar as respostas das requisições de volta aos clientes. Um *proxy* bastante utilizado é o dos protocolos HTTP e HTTPS (*HTTP Secure*), que também pode funcionar como servidor de *cache* [Kurose03].

A Figura 10 apresenta um cenário típico onde a rede interna de uma empresa é separada da Internet pela utilização de um *firewall*.

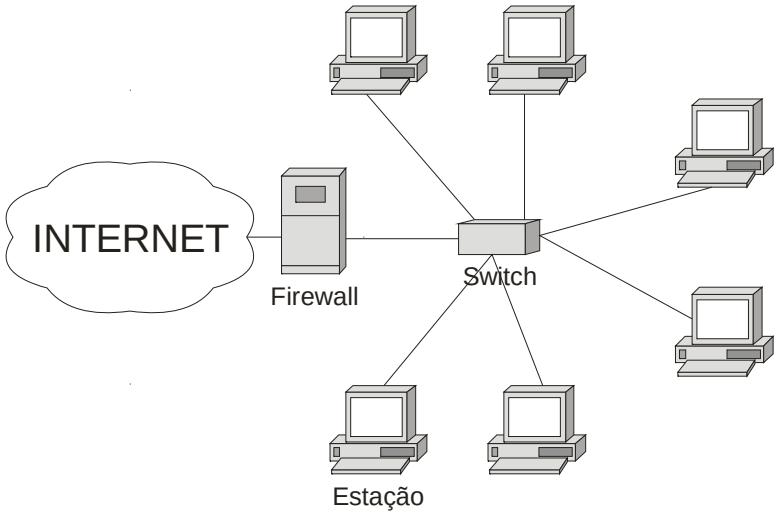

Figura 10. Cenário onde a rede interna é separada da *Internet* através de um *firewall*.

3.2 Criptografia

Criptografia ou cifragem é o nome que se dá às técnicas utilizadas para transformar dados em código secreto [Cobb04]. Essa técnica pode ser um método utilizado para garantir a confidencialidade da informação. Para se ter acesso a uma informação cifrada é necessário que se tenha uma senha ou chave secreta para permitir que a informação seja decifrada.

Os dois principais tipos de criptografia utilizam chaves simétricas e assimétricas.

Nos algoritmos que utilizam chaves simétricas, os dados são cifrados e decifrados utilizando a mesma chave. A maior vantagem da utilização de chaves simétricas é a grande velocidade de criptografia e o seu maior problema é a administração dessas chaves. As chaves precisam ser entregues de forma segura já que o acesso à chave garante o acesso à informação. Alguns algoritmos utilizados na criptografia simétrica são: DES (*Data Encryption Standard*) ou DEA (*Data Encryption Algorithm*), 3DES (*Triple DES*), AES (*Advanced Encryption Standard*), IDEA (*International Data Encryption Algorithm*), Blowfish, RC5 (*Rivest Cipher* 5) e CAST [Cobb04].

A criptografia utilizando chaves assimétricas foi inventada em 1976 por Whitfield Diffie e Martin Hellman. Por esse motivo, na literatura, esse tipo de criptografia também é referenciado como "*Diffie-Hellman encryption*" [Cobb04]. Ela é chamada de assimétrica porque utiliza duas chaves de criptografia, uma conhecida como chave pública e outra, chave privada. Uma mensagem cifrada utilizando a chave pública só poderá ser decifrada pelo dono da chave privada e, da mesma forma, uma mensagem cifrada utilizando a chave privada só poderá ser decifrada com a chave pública. A chave privada não pode ser deduzida através do conhecimento da chave pública. Uma das maiores vantagens da criptografia assimétrica é a maior facilidade na administração das chaves, se comparada ao sistema de chaves simétricas, já que, caso a chave pública seja interceptada, o conteúdo da informação cifrada por ela não poderá ser decifrado. Em contrapartida, a velocidade para a realização da criptografia é menor. O algoritmo mais popular de criptografia assimétrica foi desenvolvido por Ron Rivest, Adi Shamir e Leonard Adleman e chamado de RSA. Outros algoritmos são: DSA (*Digital Signature Algorithm*) e PGP (*Pretty Good Privacy*) [Cobb04].

3.3 VPN

VPN (*Virtual Private Network*) é uma rede privada de dados construída através da utilização de uma rede pública, como, por exemplo, a Internet. Ou seja, utiliza-se uma rede pública para conectar diferentes nós, ou redes, ao invés de *links* dedicados entre os mesmos.

A utilização da Internet para a conexão de *hosts* de redes privadas é uma boa alternativa para a redução de custos, se comparado com a utilização de links dedicados, porém falha nos três fatores da segurança da informação. Em relação à cofidencialidade, uma solução é utilizar criptografia para o tráfego de dados através da rede pública, ou seja, se estes dados forem capturados durante o trajeto pela rede pública eles não poderão ser decifrados.

As SVPNs (*Secure VPN*) funcionam como se um túnel seguro fosse criado entre os dois nós que precisam se comunicar utilizando a rede pública, ou seja, todo o tráfego é cifrado antes de ser enviado através da rede pública e, posteriormente, decifrado no outro nó. A utilização de SVPNs é motivada principalmente pelos altos preços dos *links* dedicados, tal como o *Frame Relay* [Kurose03], que tem a distância como um fator de composição dos seus preços.

A Figura 11 apresenta dois possíveis cenários de utilização de SVPN. O primeiro é a utilização de VPN para conectar um nó, em trânsito, à rede privada local de uma empresa, nessa figura, representado por uma estação. O segundo cenário apresenta a utilização de VPN para interligar duas redes como, por exemplo, a de uma matriz e de uma filial de uma empresa.

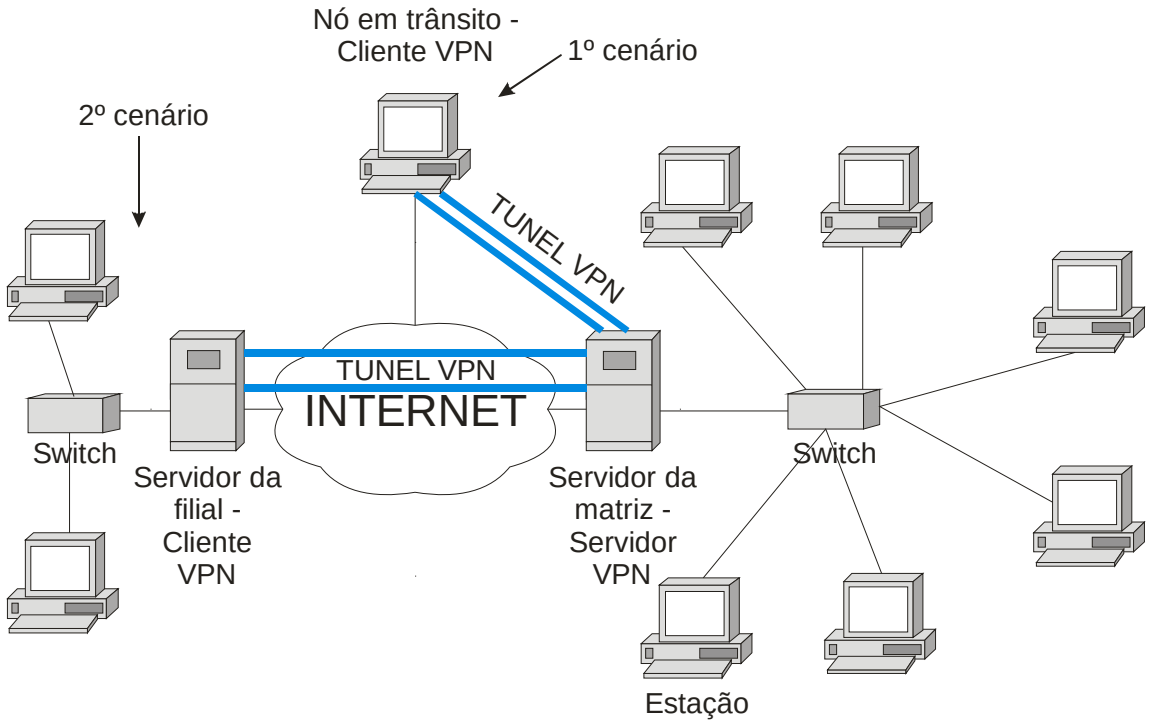

Figura 11. Exemplo de dois cenários de utilização de VPN.

Outro tipo de VPN é aquela oferecida pelos provedores de *links* de dados, como, por exemplo, a Embratel, Telemar, Intelig e outros. Ao invés de utilizar criptografia entre os pontos, um único provedor fica responsável pela segurança do tráfego dos dados pela sua rede privada de um nó ao outro. Esse tipo de VPN é denominado de TVPNs (*Trusted VPNs*).

Alguns protocolos que podem ser utilizados para a criação de VPNs são: IPSec (*IP Security*), SSL (*Secure Socket Layer*), TLS (*Transport Layer Security*) e PPTP (*Point-to-point tunneling protocol*) [Cobb04].

3.4 Tipos de Ataques

Existem diversos tipos de ataques que podem ser realizados com o objetivo de tentar comprometer alguns dos três fatores da segurança da informação: disponibilidade, integridade e confidencialidade.

Neste trabalho foram utilizados alguns tipos de ataque com o intuito de encontrar fragilidades no cenário proposto para o estudo.

Os ataques utilizados serão explanados nesta seção.

3.4.1 DoS (*Denial of Service*)

Ataques de negação de serviço têm o objetivo de sobrecarregar os limites de recursos disponíveis de um determinado serviço, fazendo com que o mesmo, temporariamente, pare de responder.

Por exemplo, caso um servidor seja capaz de processar 5 requisições por segundo, e sejam realizadas 20 requisições por segundo, o servidor descartará algumas requisições, ou, até mesmo, poderá parar de responder qualquer requisição por sobrecarga no sistema.

Normalmente o destino desses ataques são servidores conhecidos, tais como, servidores DNS, servidores Web e roteadores.

Esse tipo de ataque, quando utilizado, pode comprometer dois dos três fatores da segurança da informação: integridade e disponibilidade.

Um dos tipos de ataque de negação de serviços é o *Net Flood* [Pelaez02].

3.4.2 *Net Flood*

Esse tipo de ataque consiste em tentar saturar a capacidade de conexão de rede de um determinado sistema. Por exemplo, caso o *link* de dados de um servidor seja de 1 Mbps e sejam enviados pacotes à velocidade de 10 Mbps, esse *link* ficará saturado e haverá grandes perdas de informações.

As técnicas utilizadas nesse tipo de ataque se baseiam nos protocolos ICMP (*Internet Control Message Protocol*) e UDP (*User Datagram Protocol*) que não são orientados a conexão e permitem o envio de pacotes sem requisição prévia.

Esse tipo de ataque, quando utilizado, pode comprometer dois dos três fatores da segurança da informação: integridade e disponibilidade.

Dentre os diversos ataques do tipo *Net Flood*, os que foram utilizados neste estudo foram:

- *Ping Flood* [Pelaez02] - esse tipo de ataque consiste em enviar uma grande quantidade de mensagens ICMP de requisição de eco, com o objetivo de saturar o *link* de dados da vítima. Esse é um dos mais simples ataques do tipo *Net Flood*. Alguns sistemas operacionais, como, por exemplo, o Linux, já possui uma ferramente nativa que pode realizar esse ataque;
- UDP *Flood* [Pelaez02] – esse tipo de ataque consiste em enviar uma grande quantidade de pacotes UDP com o objetivo de saturar o *link* de dados da vítima.

3.4.3 MAC *Flooding*

MAC *Flooding* [Lockhart04] é uma técnica que tenta se aproveitar da limitação na quantidade de números diferentes de endereços MAC que um *switch* é capaz de armazenar. Essa técnica consiste em enviar uma série de pacotes com diferentes endereços MAC, fazendo com que essa capacidade chegue ao limite. Dependendo do *switch*, ele poderá entrar no modo de operação *failopen*, e, neste caso, funcionar como um *hub*. Como conseqüência, a rede fica mais lenta, já que todos os pacotes estarão sendo enviados para todos os equipamentos ligados ao *switch*.

Dessa forma, caso seja colocado um *sniffer* na rede, ele será capaz de capturar todo o tráefego, comprometendo a confidencialidade, caso a informação não esteja cifrada.

3.4.4 ARP *Spoofing*

ARP *Spoofing* [Lockhart04] também conhecido como ARP *Poisoning* é uma técnica que consiste em enviar mensagens ARP falsas para o *switch*. Essas mensagens contêm endereços MAC falsos que confundem os equipamentos de rede, como, por exemplo os *switchs*. Dessa forma, informações com destino a uma determinada interface do *switch* podem ser enviadas para outra interface que pode analisar as informações e reenviar os pacotes para o destino original, comprometendo a confidencialidade da informação, caso a mesma não esteja cifrada.

Capítulo 4

Estudo de caso

Para a realização dos estudos propostos neste trabalho, foi necessário preparar um ambiente conforme apresentado na Figura 12. Esse ambiente foi montado baseado em cenários de empresas que já utilizam VoIP. Essas informações foram colhidas junto a Ligu Consultores Associados, que é uma empresa de consultoria em tecnologia da informação que também presta serviços de VoIP para terceiros.

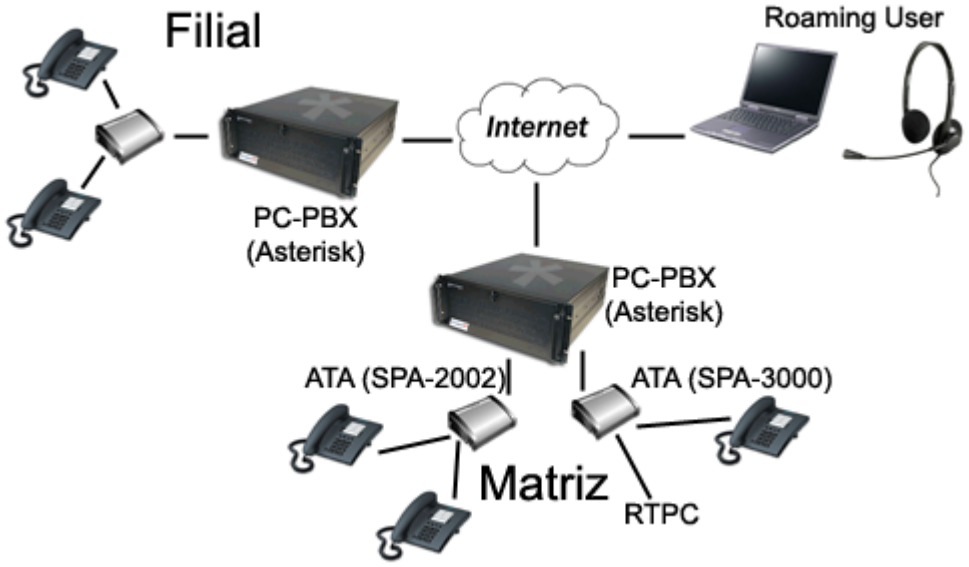

Figura 12. Ambiente de estudos.

O ambiente foi divido em 3 cenários para os estudos. No primeiro, foi instalado e configurado um servidor para funcionar como *firewall* da matriz de uma empresa e nele foi instalado e configurado o Asterisk. Nesse cenário foram realizados os estudos na rede local com ATAs e *softphone*.

O segundo e terceiro cenário abordaram a utilização da Internet. No segundo cenário, foi realizada uma instalação e configuração de um outro servidor Asterisk que utilizou um outro link de dados para simular a interligação de redes VoIPs entre matriz e filial.

O terceiro cenário abordou o estudo com uma estação que simula um *roaming user* (ramal virtual).

4.1 Preparação do ambiente para estudo na rede local (primeiro cenário)

Para montar o ambiente apresentado na Figura 12 foi necessário inicialmente escolher uma distribuição Linux para ser o sistema operacional do PC-PBX. A distribuição Linux escolhida foi o Fedora Core 4 [Negus04]. Os motivos principais relacionados com a escolha foram:

- a experiência prévia do proponente do estudo com essa distribuição;
- o Fedora é patrocinado pela RedHat Inc., que tem o maior número de servidores comerciais Linux instalado;
- o Ministério da Educação realizou um estudo comparativo entre as distribuições Mandrake, SuSE, Fedora Core, Conectiva, Slackware e Debian em relação a diversos requisitos, tais como: Estabilidade, Segurança, Facilidade de Uso, etc. O Fedora foi a distribuição escolhida por apresentar o melhor resultado [Mec06].

O CD de instalação do Fedora Core 4 (CD 1) tem um *bug* que provoca erro (*kernel panic*) ao carregar o *kernel* para a instalação em algumas placas mãe. Esse *bug* ocorreu no nosso servidor de testes. Para que o erro não acontecesse foi necessário inicialmente digitar uma opção não válida de instalação para que ele pudesse dar erro e só assim ser digitada a opção desejada.

A opção escolhida foi a *expert*, por apresentar mais customizações no momento da instalação, e o tipo de instalação escolhida foi a *custom*, para que fosse possível a escolha dos componentes a serem instalados. Foram adicionados os seguintes grupos de pacotes: *Editors*, *Server Configuration Tools*, *Web Server*, *Mail Server*, *Windows File Server*, *DNS Name Server*, *FTP Server*, *Network Servers* (dentro deste grupo foi selecionado o pacote dhcp), *Development Tools*, *Administration Tools* e *System Tools* e desmarcados os seguintes: *Office/Productivity* e *Printing Support*.

Logo após a instalação do sistema operacional as interfaces de redes foram configuradas. Uma interface para a rede local com IP 10.0.0.254/255.255.255.0 e a outra como cliente DHCP (*Dynamic Host Control Protocol*) [Kurose03] para ser configurado o acesso à Internet. O acesso à Internet foi realizado utilizando a tecnologia ADSL (*Asymmetric Digital Subscriber Line*) [Kurose03] através do serviço da concessionária telefônica local. A largura de banda foi de 1024 kbps para *download* e 328 kbps para *upload*.

O comando utilizado para a configuração desse acesso no Fedora Core 4 é o *adsl-setup*. As opções padrões foram utilizadas, com exceção: do usuário e senha, que foram especificados; da obtenção do servidor DNS (*Domain Name System*) [Kurose03], que seria automaticamente atribuído pelo servidor da concessionária telefônica local; da opção para implementação de regras de *firewall* automaticamente, que foi escolhida a opção para não implementar automaticamente nenhuma regra.

Após a configuração ADSL inicial foi necessário modificar o arquivo */etc/sysconfig/network-scripts/ifcfg-ppp0* pois o modem utilizado era um Home Connect da 3Com que requer um parâmetro adicional. A linha apresentada no Quadro 4 foi adicionada ao arquivo.

```
PPPOE_EXTRA="-f 3c12:3c13 -S VELOX"
```

Quadro 4. Linha adicionada ao */etc/sysconfig/network-scripts/ifcfg-ppp0*.

Para iniciar o acesso a Internet foi necessário executar o comando *adsl-start* e identificar-se na página do provedor do link ADSL com o usuário e senha fornecidos pelo provedor de acesso. Após a conexão foi iniciado um *ping* infinito em *background* já que, por experiência, caso

o *ping* não fosse executado a conexão iria cair após alguns segundos ou minutos sem utilização. Esse comportamento não deve ser atribuído ao Linux e sim ao serviço de banda larga da concessionária telefônica local. Existe uma opção para a desconexão automática após um período sem utilização, porém essa opção não foi utilizada.

Após a instalação do Fedora Core 4 foi realizada a atualização dos pacotes da distribuição através do comando do Quadro 5. A atualização é recomendada, pois corrige falhas nas versões mais antigas dos pacotes.

```
[root@server ~]# yum -y update
```

Quadro 5. Comando para a atualização do Fedora Core 4.

Para a instalação do Asterisk foi necessário fazer o *download* do código fonte, descompactá-lo e compilá-lo. Os pacotes instalados foram: *asterisk-1.2.5*, *zaptel-1.2.4* e *libpri-1.2.2*. Para a instalação desses pacotes foi seguido o procedimento encontrado no *site* e exibido no Quadro 6.

```
[root@server asterisk]# tar xvzf zaptel-1.2.4.tar.gz
[root@server asterisk]# tar xvzf libpri-1.2.2.tar.gz
[root@server asterisk]# tar xvzf asterisk-1.2.5.tar.gz
[root@server asterisk]# cd zaptel-1.2.4
[root@server zaptel-1.2.4]# make
[root@server zaptel-1.2.4]# make install
[root@server zaptel-1.2.4]# cd ../libpri-1.2.2
[root@server libpri-1.2.2]# make
[root@server libpri-1.2.2]# make install
[root@server libpri-1.2.2]# cd ../asterisk-1.2.5
[root@server asterisk-1.2.5]# make
[root@server asterisk-1.2.5]# make install
[root@server asterisk-1.2.5]# make samples
```

Quadro 6. Instalação e compilação do asterisk.

Após a instalação inicial do Asterisk, foi configurado o ATA Sipura SPA-3000 (Figura 13), que possui uma porta FXS e uma porta FXO.

FXS (*Foreign eXchange Subscriber*) [Goncalv05] é uma interface que deve receber um telefone comum, ou seja, é uma interface semelhante as que as companhias telefônicas provêem. A interface FXS provê tom de discagem, tensão para o toque e detecta DTMF (*Dual Tone MultiFrequential*). DTMF são os tons utilizados na discagem dos aparelhos telefônicos modernos. Normalmente uma interface FXS é um ramal da central telefônica.

FXO (*Foreign eXchange Office*) [Goncalv05] é uma interface que normalmente é conectada à RTPC ou a um ramal de uma central telefônica. Ela deve ser capaz de gerar DTMF, detectar o tom de discagem, a chamada e a desconexão.

A porta FXS do SPA-3000 foi configurada como ramal 10 e a porta FXO foi configurada para receber uma linha telefônica analógica de prefixo 3328.

Figura 13. Sipura SPA-3000, ATA com uma porta FXS e uma porta FXO.

Após a configuração do SPA-3000, foi configurado um ATA Sipura SPA-2002 (Figura 14), que possui duas portas FXS, como ramais 11 e 12.

Figura 14. Sipura SPA-2002, ATA com duas portas FXS.

Inicialmente os ATA Sipura SPA-2002 e SPA-3000 são configurados de fábrica como cliente DHCP, porém eles possuem uma interface IVR (*Interactive Voice Response*) de configuração que pode ser acessada ao teclar "****" através de um telefone convencional ligado a uma das portas FXS. O SPA-3000 foi configurado com o IP 10.0.0.201 e o SPA-2002 foi configurado com o IP 10.0.0.200. Após a configuração do endereço IP de ambos os equipamentos, pôde-se acessar a interface *web* de configuração dos aparelhos através de um navegador conforme exemplificado na Figura 15.

Na Tabela 4 seguem os parâmetros da opção *advanced* que foram modificados no Sipura SPA-3000 e na Tabela 5 os que foram modificados no Sipura SPA-2002.

Figura 15. Página de configuração do Sipura SPA-2002.

Tabela 4. Parâmetros modificados na configuração do SPA-3000.

Parâmetros na opção *advanced*	**Valor**
System -> User Password	testevoip
System -> NetMask	255.255.255.0
System -> Gateway	10.0.0.254
System -> Hostname	SPA3000
Regional -> Set Local Date (mm/dd)	mm/dd/aa (dia da configuração)
Regional -> Set Local Time (HH/mm)	HH:MM (hora da configuração)
Regional -> Time Zone	GMT-03:00
Regional -> FXS Port Input Gain	0
Regional -> FXS Port Output Gain	0
Line 1 -> Proxy	10.0.0.254
Line 1 -> Display Name	Ramal 10
Line 1 -> User ID	ramal10
Line 1 -> Password	testevoip
PSTN Line -> Proxy	10.0.0.254
PSTN Line -> Display Name	Linha 3328xxxx
PSTN Line -> User ID	linha3328xxxx
PSTN Line -> Password	testevoip
PSTN Line -> Dial Plan 8	S0<:s@10.0.0.254>
PSTN Line -> PSTN Ring Thru Line 1	no
PSTN Line -> PSTN Caller Default DP	8
PSTN Line -> Detect PSTN Long Silence	yes
PSTN Line -> Detect VoIP Long Silence	yes
PSTN Line -> PSTN Long Silence Duration	180
PSTN Line -> VoIP Long Silence Duration	180
PSTN Line -> Disconnect Tone	425@-30,425@-30;2(.25/.25/.25/.25)

Tabela 5. Parâmetros modificados na configuração do SPA-2002.

Parâmetros na opção *advanced*	**Valor**
System -> User Password	testevoip
System -> NetMask	255.255.255.0
System -> Gateway	10.0.0.254
System -> Hostname	SPA2002
Regional -> Set Local Date (mm/dd)	mm/dd/aa (dia da configuração)
Regional -> Set Local Time (HH/mm)	HH:MM (hora da configuração)
Regional -> Time Zone	GMT-03:00
Regional -> FXS Port Input Gain	0
Regional -> FXS Port Output Gain	0
Line 1 -> Proxy	10.0.0.254
Line 1 -> Display Name	Ramal 11
Line 1 -> User ID	ramal11
Line 1 -> Password	testevoip
Line 2 -> Proxy	10.0.0.254
Line 2 -> Display Name	Ramal 12
Line 2 -> User ID	ramal12
Line 2 -> Password	testevoip

Após a configuração dos ATAs foi realizada a configuração do *softphone* X-lite (Figura 16) da CounterPath que também utiliza o protocolo SIP. A Figura 17 apresenta a tela de configuração desse *softphone* com os parâmetros que foram modificados para configurá-lo como ramal 13.

Figura 16. *Softphone* X-lite.

Figura 17. Configuração X-lite como ramal 13.

Após configurar os clientes SIP (ATAs e *softphone*) foi necessário configurar o Asterisk para que fosse o Servidor SIP e também foi necessário configurar o plano de discagem. O plano de discagem é responsável por determinar as ações que serão tomadas quando for discado um determinado número em um dos ATAs ou *softphones* e também quando uma ligação for recebida através da linha convencional.

O diretório que contém os arquivos de configuração do Asterisk é o */etc/asterisk*. O arquivo de configuração do protocolo SIP é o *sip.conf* e o do plano de discagem é o *extensions.conf*. O Quadro 7 apresenta a configuração do ramal 10 no arquivo *sip.conf* (a configuração dos outros ramais é semelhante), o Quadro 8, a configuração da linha telefônica (porta FXO do SPA-3000) do mesmo arquivo e, o Quadro 9, as configurações adicionadas ao arquivo *extensions.conf*.

```
[ramal10]
username=ramal10
secret=testevoip
type=friend
host=dynamic
nat=no
canreinvite=no
context=ramais
callerid=Ramal 10
```

Quadro 7. Configuração do ramal 10 no *sip.conf*.

```
[linha3328XXXX]
username=linha3328XXXX
secret=testevoip
type=friend
host=dynamic
nat=no
canreinvite=no
context=linhas
callerid=Linha 3328XXXX
dtmfmode=rfc2833
```

Quadro 8. Configuração da linha analógica no *sip.conf*.

```
[ramais]
exten => 10,1,Dial(SIP/ramal10)
exten => 11,1,Dial(SIP/ramal11)
exten => 12,1,Dial(SIP/ramal12)
exten => 13,1,Dial(SIP/ramal13)
exten => 600,1,MixMonitor(echo.wav)
exten => 600,n,Playback(demo-echotest)
exten => 600,n,Echo
exten => 600,n,Playback(demo-echodone)
exten => _0.,1,Dial(SIP/linha3328XXXX/${EXTEN:1})
[linhas]
exten => s,1,Dial(SIP/ramal10)
```

Quadro 9. Plano de discagem (*extensions.conf*).

Essas configurações foram realizadas com o intuito de simular o ambiente local de uma empresa que adotou o Asterisk como PC-PBX. Dessa forma, a central foi configurada para que:

- Os ramais liguem entre si quando os números 10, 11, 12 e 13 fossem discados;
- Os ramais liguem para um telefone qualquer utilizando a linha telefônica convencional, como, por exemplo, 33333333 quando fosse discado 033333333;
- As ligações recebidas através da linha telefônica convencional tocassem no ramal 10;
- Entrasse em um teste de eco ao se discar 600 e gerasse um arquivo *echo.wav* no diretório */var/spool/asterisk/monitor* com o áudio. Esse teste faz com que tudo que for dito seja repetido, numa aparente semelhança ao *ping*. Dessa forma pode-se verificar o atraso na transmissão da voz e também verificar problemas como voz picotada e a não transmissão ou recepção da voz.

Após essas configurações o Asterisk foi executado em modo console com nível cinco de depuração através do comando apresentado no Quadro 10 e os planos de discagem foram testados para verificar se tudo estava funcionando como pretendido. Após essas verificações o próximo passo foi a configuração dos parâmetros básicos das regras do *firewall*.

```
[root@server ~]# asterisk -vvvvvc
```

Quadro 10. Comando para executar o Asterisk em modo console.

O *iptables* é o utilitário responsável por modificar as regras de filtragem de pacotes no Linux. Para o início da configuração das regras de filtragem de pacotes é comum negar a entrada e passagem de todos os pacotes e ir habilitando conforme necessário.

As regras do *firewall* que foram adicionadas utilizando o *iptables* e suas respectivas explicações estão apresentadas na Tabela 6.

Tabela 6. Regras iniciais adicionadas ao *firewall*.

Parâmetros do comando *iptables*	Função
-P INPUT DROP	Descarta os pacotes que tem o *firewall* como destino
-P FORWARD DROP	Descarta os pacotes que passam de uma interface para a outra do *firewall* e que não tem como origem e destino o próprio *firewall*
-A INPUT -i lo -j ACCEPT	Permite a entrada de pacotes pela interface local (lo)
-A INPUT -s 10.0.0.200 -p udp -m udp --dport 5060 -j ACCEPT Obs.: Idem para o IP 10.0.0.201 e 10.0.0.1 (*softphone*)	Permite as conexões com origem no IP 10.0.0.200 e destino na porta UDP/5060 (SIP)
-A INPUT -s 10.0.0.200 -p udp -m udp --dport 10000:20000 -j ACCEPT Obs.: Idem para o IP 10.0.0.201 e 10.0.0.1	Permite as conexões com origem no IP 10.0.0.200 e destino nas portas UDP/10000 a 20000 (Fluxo de mídia)
-I INPUT -p tcp -m tcp --dport 22 –j ACCEPT	Permite o acesso remoto através de SSH (*Secure Shell*) [Negus04]
-I INPUT –p icmp --icmp-type echo-reply –j ACCEPT	Permite receber pacotes "*ICMP (Internet Control Message Protocol*) *Echo Reply*" para a resposta dos *pings*.

Após a configuração das regras do *firewall*, que nesse caso não estão, por enquanto, contemplando o acesso a outros serviços tais como: acesso a *web* e *email*, foi necessário salvar as alterações para que sejam automaticamente utilizadas após a reinicialização do servidor ou do serviço de *firewall*. Para isso foi necessário executar o comando do Quadro 11. Após a execução, o ambiente estava preparado para o início do estudo.

```
iptables-save > /etc/sysconfig/iptables
```

Quadro 11. Comando para salvar as regras do *firewall*.

4.2 Estudos na rede local (primeiro cenário)

No primeiro cenário, foi instalado e configurado um servidor para funcionar como *firewall* da matriz de uma empresa e nele foi instalado e configurado o Asterisk. Neste cenário foram realizados os estudos na rede local com ATAs e *softphone*.

Este cenário foi montado para que se pudessem estudar algumas das suas vulnerabilidades, tentar implementar medidas de segurança e verificar as implicações destas implementações.

Os testes que foram realizados tentaram comprometer, de alguma forma, os três fatores da segurança da informação: disponibilidade, integridade e confidencialidade em relação à comunicação VoIP.

Alguns ataques utilizados para comprometer a disponibilidade e a integridade dos serviços disponibilizados em rede são os ataques de DoS (*Denial of Service*) [Pelaez02]. Esse ataque

consiste em tentar fazer com que o serviço ou o servidor pare de responder devido a um excesso de solicitações enviadas a eles.

A técnica que foi utilizada para tentar comprometer a disponibilidade e integridade da comunicação de voz foi o *ping flood*. O objetivo foi tentar comprometer as chamadas VoIP e verificar o comportamento dos ATAs e do *softphone*.

Inicialmente, dados foram colhidos, através do comando *ping*, para identificar os tempos mínimos, médios e máximos que um pacote leva para ir de um nó para outro e se houve perda na transferência de pacotes de 3072 *bytes*, que é aproximadamente a largura de banda consumida por um canal de voz totalmente ocupado utilizando o *codec* G.729. Os testes foram realizados a partir de uma estação Linux ligada à rede local para os ATAs e para o micro que possui o *softphone*. Para se ter uma base considerável, 3600 coletas foram realizadas para cada destino diferente. Os tempos estão apresentados na Tabela 7. Para que fosse possível usar o *ping* para o servidor, foi necessário temporariamente permitir a entrada de pacotes *icmp* no *firewall*.

Tabela 7. Dados da coleta do *ping*.

IP	Mínimo (ms)	Médio (ms)	Máximo (ms)	Perda
10.0.0.1	1,020	1,144	5,631	0%
10.0.0.200	9,000	9,269	13,386	0%
10.0.0.201	9,009	9,238	13,633	0%

Inicialmente, foram realizados os testes com o Sipura SPA-2002 com IP 10.0.0.200. A partir de uma estação Linux local foi iniciado um *ping flood*, em *background*, com o tamanho de pacote de 1024 *bytes*. Após o primeiro *ping flood* foi realizado 60 *pings* do servidor para o ATA para verificar diferenças nos tempos de resposta. O tempo médio dos *pings* teve uma pequena variação de 0.494 ms, o que não deveria implicar diferença perceptível numa chamada. Depois foram realizados testes de eco para verificar se havia alguma diferença em relação ao atraso e à qualidade da voz. Não foi encontrada nenhuma diferença que fosse perceptível. Uma gravação foi reproduzida no microfone do telefone para que fosse depois analisada por um *software* de análise de áudio, já que as ligações estariam sendo gravadas através da aplicação *MixMonitor*.

Foi descoberto, então, um problema na aplicação *MixMonitor* do Asterisk. Essa é a aplicação que permite gravar uma ligação para um ramal em um arquivo de áudio. Alguns testes funcionavam, ou seja, todo o áudio era gravado no arquivo, porém várias vezes esses arquivos não continham toda a gravação. Esse problema estava descrito no *site* de problemas da Digium sob o ID 6457 [Bugs06]. O problema ainda estava com a situação *open*, porém estavam disponibilizados dois arquivos de atualização na tentativa de corrigir o problema. No *site* era sugerido aplicar a atualização *patch.mixmonitor1*. A atualização foi aplicada ao código fonte do Asterisk e depois ele foi recompilado. Os passos seguidos encontram-se no Quadro 12.

```
[root@server ~]# cd asterisk/asterisk-1.2.5
[root@server asterisk-1.2.5]# make clean
[root@server asterisk-1.2.5]# patch -p0 < ../../patch.mixmonitor1
[root@server asterisk-1.2.5]# make
[root@server asterisk-1.2.5]# make install
```

Quadro 12. Passos para aplicação da atualização no Asterisk.

Testes foram realizados e o problema não se repetiu.

Contudo, outro problema, surgiu. Após a primeira gravação descobriu-se que o *MixMonitor* não gravava as respostas do teste de eco, ou seja, ele somente gravava o que era

falado e não a sua resposta. Através de tentativas descobriu-se que era possível fazer a gravação do teste de eco fazendo o seguinte procedimento:

- o arquivo *extensions.conf* foi modificado conforme o Quadro 13 para que todos os ramais pudessem ser gravados e foi retirada a gravação do teste de eco já que não funcionava;

```
[ramais]
exten => 10,1,MixMonitor(ramal10.wav)
exten => 10,n,Dial(SIP/ramal10)
exten => 11,1,MixMonitor(ramal11.wav)
exten => 11,n,Dial(SIP/ramal11)
exten => 12,1,MixMonitor(ramal12.wav)
exten => 12,n,Dial(SIP/ramal12)
exten => 13,1,MixMonitor(ramal13.wav)
exten => 13,n,Dial(SIP/ramal13)
exten => 600,1,Playback(demo-echotest)
exten => 600,n,Echo
exten => 600,n,Playback(demo-echodone)
exten => _0.,1,Dial(SIP/linha3328XXXX/${EXTEN:1})
[linhas]
exten => s,1,Dial(SIP/ramal10)
```

Quadro 13. Novo arquivo contendo as modificações no plano de discagem.

- para que o teste de eco funcionasse realizava-se a ligação para o número 600 e, enquanto isso, colocava-se a chamada em espera, e então se discava para outro ramal colocando as chamadas em conferência. Dessa forma, o que era dito no outro ramal, passava pelo teste de eco e era gravado.

Após a solução do problema, separou-se o arquivo de áudio com apenas um *ping flood* para análise posterior.

Mais *ping floods* foram sendo colocados em *background* com o mesmo tamanho de pacote (1024 *bytes*) e realizando os mesmos testes. Em relação aos *pings* do servidor para o ATA houve crescimentos muito pequenos nos tempos médios de resposta e nenhuma perda de pacote.

No sexto *ping flood* percebeu-se uma diferença mais acentuada nos testes de eco, ou seja, era facilmente perceptível que a resposta demorava algum tempo para retornar. Nessa etapa foi separado mais um arquivo de áudio, reproduzindo a mesma gravação, para analise posterior. O tempo médio de *ping* continuou tendo uma pequena variação para maior.

No sétimo *ping flood* o ATA parou de funcionar (não era mais detectado o tom de ramal no aparelho telefônico). Percebeu-se que ele parou de funcionar exatamente no momento que houve perda de pacotes do *ping* do servidor para o ATA. Apesar de continuar sem fornecer o tom de ramal, os *pings* começaram novamente a ter retorno em um tempo bem superior (aproximadamente 400 a 500% maior) em relação ao *ping* com apenas 6 *ping floods*. A largura de banda ocupada pelos 7 ping floods era de, aproximadamente, 4,9 Mbps, ou seja, não chegava a comprometer o limite da largura de banda dos ATAs que era de 10 Mbps.

Não foi mais necessário continuar aumentando a quantidade de *ping floods* pois o objetivo dos testes foi alcançado. Conseguiu-se comprometer a disponibilidade e a integridade das chamadas que utilizavam o ATA.

Para a análise dos dois arquivos de áudio gerados e separados anteriormente nos testes, foi utilizado o *software* SIGVIEW [Sigview06], que é uma ferramenta de análise de sinais. Ele

possui várias funcionalidades para análise de espectro via método de Fourier (FFT - *Fast Fourier Transform*), funções estatísticas e um sistema de visualização de fácil compreensão [Sigview06].

A Figura 18 apresenta o primeiro arquivo de áudio e a Figura 19 mostra o segundo arquivo de áudio. Nas figuras, os tempos são indicados na parte superior dos retângulos. O tempo de resposta é calculado subtraindo-se o primeiro tempo do segundo. Cada figura possui duas amostras de áudio e seus respectivos ecos. Através delas é possível perceber que a maior quantidade de *ping floods* aumenta o atraso na resposta do eco, conseqüentemente dificultando a qualidade da conversação. No primeiro arquivo a média da resposta foi de 120,6125 ms e no segundo a média foi de 301,085 ms. Como foi utilizado um recurso de conferência para poder gravar os testes esses tempos precisam ser divididos por dois para que seja obtido o tempo real aproximado, ou seja, eles ficaram respectivamente: 60,30625 ms e 150,5425 ms. Para manter uma boa qualidade de voz, semelhante a telefônica convencional, o *round-trip delay* deve ficar aproximadamente em 120 ms [Fong02].

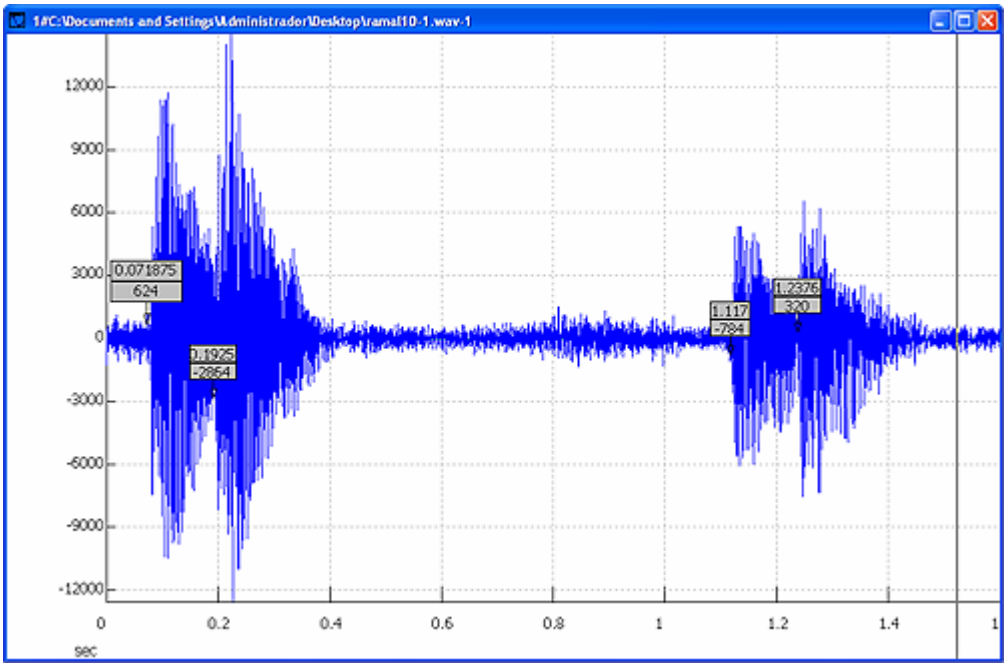

Figura 18. Primeiro áudio gravado (1 *ping flood* de 1024 *bytes*).

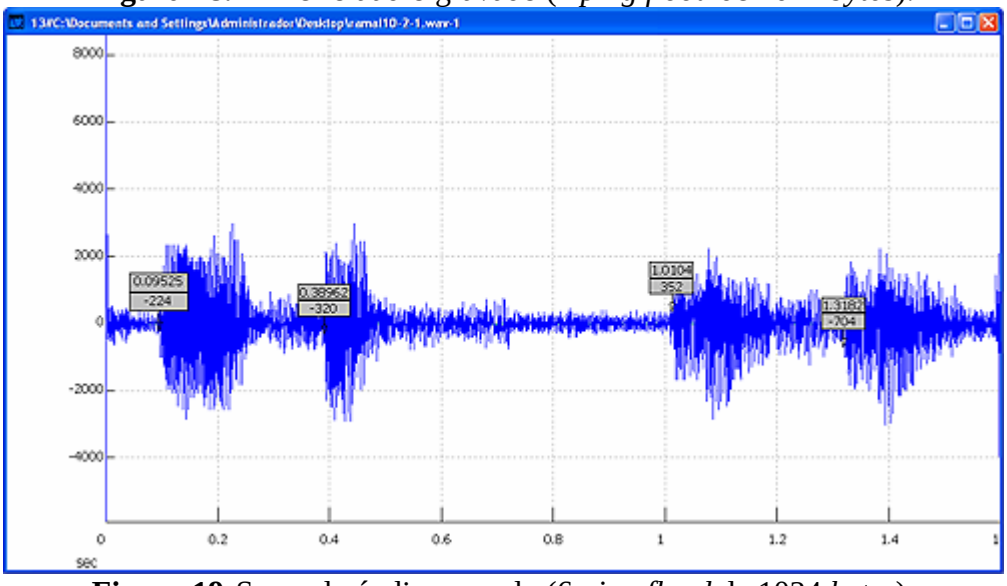

Figura 19. Segundo áudio gravado (6 *ping flood* de 1024 *bytes*).

Foram realizados os mesmos testes aplicados ao Sipura SPA-2002 com o Sipura SPA-3000 e os mesmos comportamentos foram verificados.

Foram iniciados testes semelhantes com o *softphone*. A diferença em relação aos testes realizados anteriormente é que foi aumentado o tamanho dos pacotes do *ping flood* para 10240 *bytes*, já que a interface de rede do computador era de 100 Mbps enquanto a dos ATAs era de 10 Mbps.

Não se percebeu diferença razoável no tempo dos *pings* do servidor para a estação Windows com o *softphone*, nem atrasos perceptíveis no teste de eco até o décimo quinto *ping flood* em *background*.

No décimo sexto *ping flood* a comunicação tornou-se impraticável havendo cortes prolongados no teste de eco e nas comunicações. O teste do *ping* do servidor para a estação apresentou perdas de 93% dos pacotes.

O objetivo desse teste de comprometer a comunicação VoIP também foi alcançado no caso da estação executando o *softphone*.

Por fim, tentou-se comprometer o próprio servidor através dos *ping floods*. Neste caso, como as regras do *firewall* não estão permitindo a entrada de ICMP *Echo Request*, os *pings* não recebem respostas. Independente dessa restrição, conseguiu-se comprometer a conversação depois da execução de 16 *ping floods* em *background*. A partir deste momento toda a conversação teve grandes tempos de cortes.

Através do utilitário iptraf [Iptraf06] foi verificado que os 16 *pings flood* para o servidor estavam gerando um tráfego entre aproximadamente 70 e 90 Mbps, chegando próximo ao limite de largura de banda disponível (100 Mbps), o que explica o comprometimento das chamadas.

O concentrador de rede utilizado foi um *switch* simples de 8 portas, não gerenciável e que não suporta QoS. Caso o *switch* suportasse QoS, poderia ser dada prioridade a certos tráfegos e esses *pings* poderiam não interferir na conversação se, por exemplo, os tráfegos na porta UDP 5060 (SIP) e nas portas UDP de 10000 a 20000 (fluxo de mídia) fossem priorizados.

O próximo objetivo foi tentar comprometer a chamada através de ataques DoS nas portas que possivelmente poderiam estar priorizadas, tais como, a porta UDP 5060.

O pacote packETH [Packeth06], uma ferramenta que permite gerar tráfego na rede apresentada na Figura 20, foi instalado na estação Linux. Ele permite criar e enviar qualquer seqüência de pacotes na rede.

Para simular uma comunicação SIP no PackETH, foi necessário configurar os endereços MAC (*Media Access Control*) das placas de rede da origem e do destino, os IPs de origem e destino, o protocolo e as portas de origem e destino, que neste caso foi escolhido o UDP na porta 5060 tanto para origem como destino. Além disso foi necessário escolher os dados para serem enviados. Um padrão de 1024 *bytes* com o caractere 00x foi inserido já que o conteúdo não era importante para o tipo de teste pretendido.

A aba “Gen-b” apresentada na Figura 21 foi selecionada para que permitisse enviar um número infinito de pacotes indeterminadamente sem atraso entre eles.

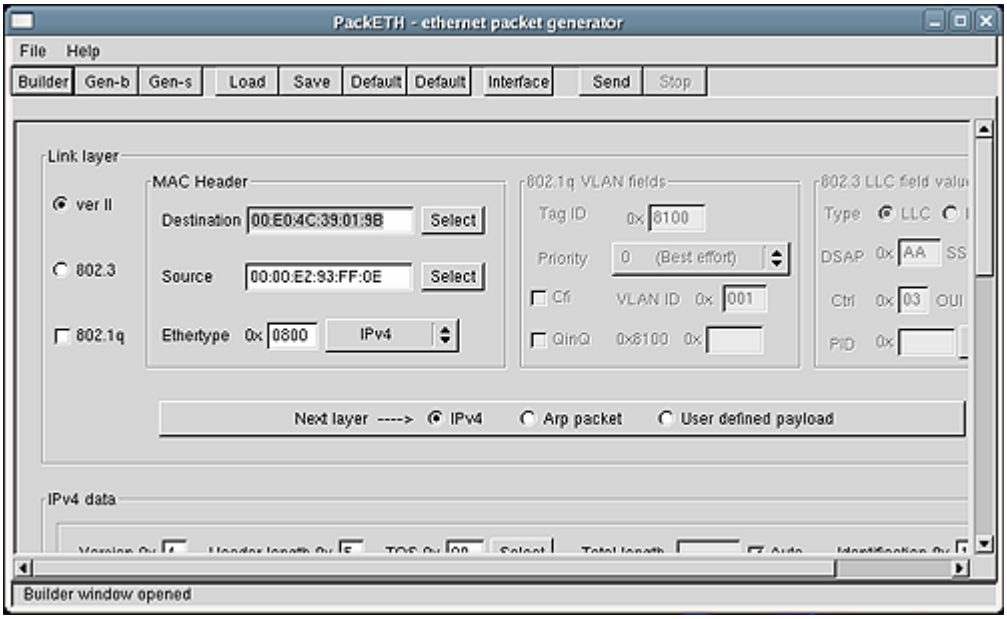

Figura 20. PackETH configurado para gerar tráfego UDP para o servidor.

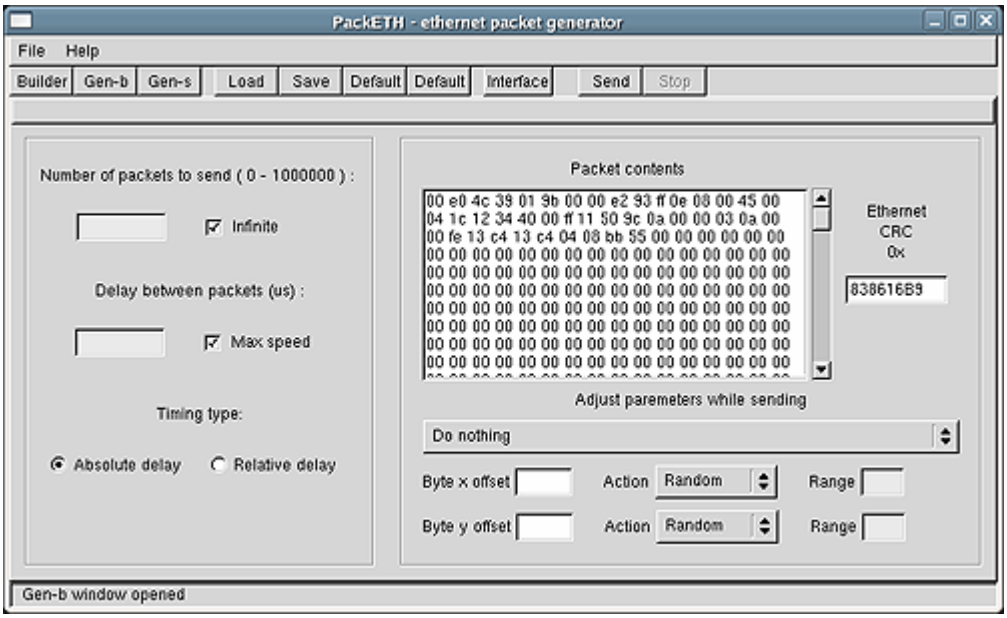

Figura 21. PackETH aba “Gen-b”.

Depois de iniciado o envio dos pacotes, foram realizados testes de eco e de conversação. Ambos os testes tiveram grandes perdas nos pacotes de voz, além dos telefones ligados aos ATAs apresentarem ruídos estranhos. Esses ruídos poderiam estar vinculados à perda de pacotes na comunicação entre o servidor e o ATA, porém isto não foi certificado por não ser o objetivo do estudo.

Os testes novamente foram concluídos pois o objetivo pretendido, que era comprometer a conversação, foi atingido.

Após a realização dos testes relativos à disponibilidade e à integridade, testes para tentar comprometer a confidencialidade da informação foram realizados.

Em redes que utilizam *hubs*, toda a informação transmitida de um computador para o outro é enviada a todos os computadores, ou seja, caso seja utilizado um *sniffer*, que é um

programa que pode capturar os pacotes de rede, você pode obter toda a informação que está sendo trafegada independente de ter sido enviada para a sua estação.

Em redes que utilizam *switchs*, caso uma informação seja enviada de um computador A para um computador B, apenas o computador B receberá a informação. Isso ocorre porque os *switches* armazenam o endereço MAC da interface de rede e associa o endereço a uma determinada porta. Com isso ele consegue enviar a informação apenas para a interface de destino. Existem algumas exceções como, por exemplo, os pacotes de *broadcast* que devem ser enviados a todos os computadores.

Existem algumas técnicas que permitem enganar essa característica dos *switches*, como, por exemplo, o MAC *Flooding* e o ARP (*Address Resolution Protocol*) *Spoofing*.

A técnica utilizada para a tentativa da captura dos pacotes de voz na rede foi o ARP *Spoofing*. Para isso foi necessário instalar o pacote dsniff [Dsniff06] na estação linux local. O dsniff é um pacote com ferramentas para auditoria em rede e testes de invasão. O pacote binário (RPM-*RedHat Package Manager* [Negus04]) do dsniff foi transferido para o servidor e instalado com a opção *--no deps* (sem verificação de dependências), já que ele possuía dependência de versões antigas de bibliotecas. Foi necessário fazer *links* com nomes antigos para as bibliotecas novas para que alguns aplicativos do pacote pudessem funcionar.

Para que os pacotes desviados pudessem chegar ao seu destino inicial foi necessário habilitar na estação linux o IP *forwarding*. Para isso foi necessário executar o comando apresentado no Quadro 14.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Quadro 14. Comando para habilitar IP *Forwarding*.

Inicialmente foi necessário executar o *arpspoof* para os nós cujos pacotes deveriam ser interceptados. Nesse caso os IPs escolhidos foram os dos ATAs: 10.0.0.200 e 10.0.0.201. Os comandos foram colocados em *background* e executados conforme apresentado no Quadro 15.

```
arpspoof 10.0.0.200 &
arpspoof 10.0.0.201 &
```

Quadro 15. Comando executado para o ARP *spoofing*.

No linux existe uma ferramenta chamada *tcpdump* [Tcpdump05] [Kretchmar03] que gera uma descarga do tráfego de rede, por exemplo, em um arquivo. Ao utilizar o *arpspoof* todo o tráfego gerado para os IPs 10.0.0.200 e 10.0.0.201 foi redirecionado para a estação Linux. Através do uso do utilitário *tcpdump* foi possível armazenar o tráfego desses nós em um arquivo para análise posterior. O comando *tcpdump* foi executado conforme apresentado no Quadro 16 gerando, então, o arquivo *voip.file*.

```
tcpdump -s 1500 host 10.0.0.200 or host 10.0.0.201 -n -w voip.file
```

Quadro 16. Comando executado para armazenar os pacotes.

Para que os canais de voz pudessem ser separados do arquivo de *dump* foi necessário utilizar o *software* ethereal [Ethereal05] [Orebaugh04] que é um pacote analisador de protocolos de rede. Através do YUM (*Yellowdog Update Modified*) [Negus04] instalou-se o pacote *ethereal* e o *ethereal-gnome*. A Figura 22 apresenta o arquivo de *dump voip.file* aberto no *ethereal*.

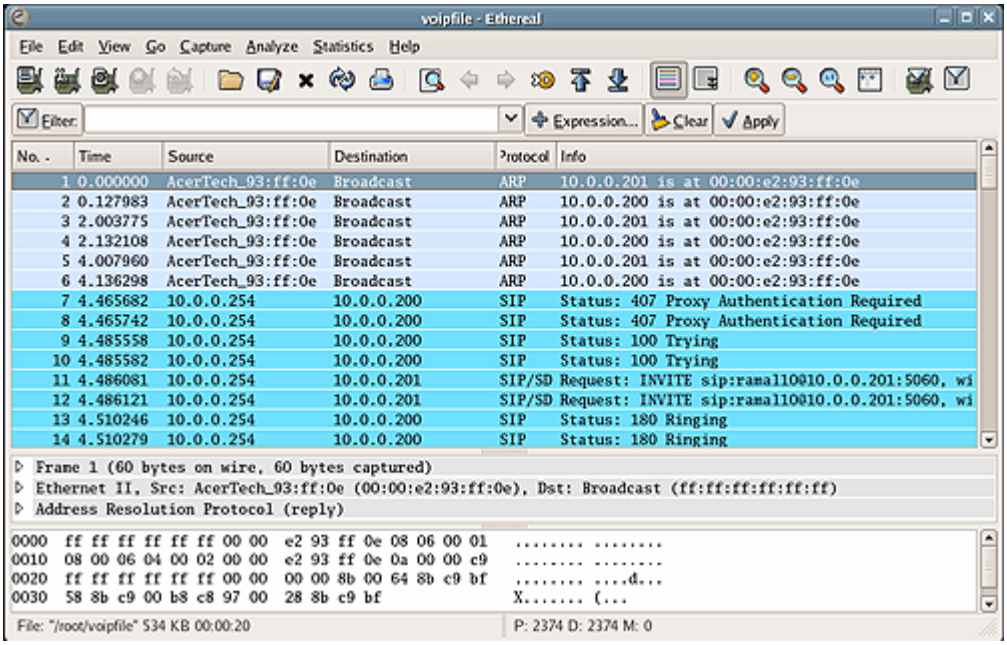

Figura 22. Ethereal com *voip.file* aberto.

No *ethereal* através da opção "*Statistics->RTP->Show All Streams*", foi possível visualizar os dois canais de voz como apresentado na Figura 23.

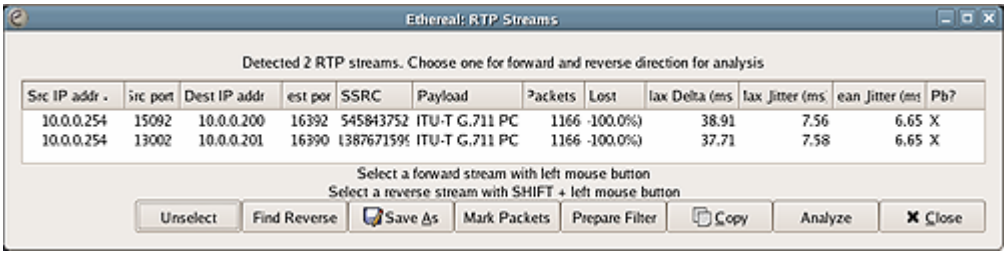

Figura 23. Streams de voz.

Ao ser selecionado um dos canais de voz e clicado no botão *Analyze* foi aberta uma nova janela com um botão "*Save payload...*". Através dessa opção, é possível salvar um arquivo de áudio no formato *.raw* e *.au* com um dos lados da conversa.

Através desses testes foi possível confirmar a fragilidade do ambiente preparado em relação à confidencialidade da informação.

Após a conclusão dos testes nesse primeiro ambiente, verificou-se que todos os aspectos de segurança da informação: disponibilidade, integridade e confidencialidade, em relação a comunicação VoIP, foram comprometidos. A disponibilidade e integridade foram comprometidas através dos ataques de DoS que conseguiram interromper o serviço ou inviabilizar a comunicação. Foi confirmado, que utilizando técnicas simples, a confidencialidade também pôde ser comprometida, já que foi obtido acesso ao conteúdo da chamada através de uma estação ligada à rede.

Em segurança da informação, trabalha-se com uma assertiva: um ambiente nunca está 100% seguro [Rezende98]. Apesar de se ter essa certeza, os administradores de segurança da informação trabalham para tentar chegar perto desse percentual minimizando os riscos e conseqüentemente a exposição a falhas.

Nesse primeiro ambiente, algumas ações podem ser realizadas para minimizar os riscos. Dentre as possibilidades sugere-se que:

- em relação aos ATAs, eles deverão ser colocados em uma rede diferente das demais estações. Ou seja, deverá haver uma rede específica, de acesso exclusivo aos equipamentos apenas de VoIP, como os ATAs. Isso pode ser facilmente realizado através da adição de uma placa de rede ao *firewall*. Essa interface deverá possuir um endereço de rede diferente e os ATAs deverão estar ligados a *switches* diferentes ou estar ligados a *switches* que suportem VLANs (*Virtual LANs*) [Pelaez02]. *Switches* que suportam VLANs, conseguem separar determinados computadores em redes diferentes, fazendo com que, num mesmo *switch*, diferentes redes possam existir. Além disso, esses ATAs precisarão estar fisicamente isolados para que ninguém tenha acesso aos pontos de rede deles. Sugere-se que eles estejam junto do servidor e que se apliquem as mesmas restrições de acesso físico do servidor aos ATAs. Essas ações visam garantir a segurança da informação nos seus três aspectos. Em relação à disponibilidade e à integridade, nenhum equipamento que pudesse executar ataques, por exemplo, do tipo DoS, estaria na mesma rede. A confidencialidade também seria garantida pelo mesmo motivo, ou seja, a inexistência de equipamentos que pudessem funcionar como *sniffers*. O ponto de convergência que necessita de extrema atenção é o *firewall*, já que, caso ele seja comprometido, a telefonia IP também seria comprometida. A preocupação com a segurança do *firewall* normalmente já deve ser uma premissa da empresa, uma vez que o *firewall* já é normalmente utilizado, independentemente da telefonia IP;
- em relação a estações com *softphone*, não adiantaria isolá-las em uma rede diferente, já que as mesmas são ameaças potenciais entre si. Em relação à disponibilidade e integridade sugere-se a utilização de IDS (*Intrusion Detection Systems*) [Pelaez02] que são *softwares* que, ligados à porta espelho do *switch* (porta que recebe todo o tráfego), tentam identificar possíveis tentativas de ataque, como, por exemplo, ataques DoS. Caso seja detectado, ele pode rapidamente comunicar o ocorrido ao administrador de rede para que tome as devidas ações, ou, em alguns casos, pode até desabilitar a porta do *switch* que possui a estação que está realizando os ataques. Em relação à confidencialidade, sugere-se a adoção de uma VPN da estação ao servidor. Necessita-se, porém, conhecer os impactos que essa adoção poderia trazer a chamada, já que é sabido que a utilização de VPN provoca um acréscimo de tempo para a transmissão devido ao tempo de cifragem. Com essa dúvida, o próximo passo do estudo foi a implementação de uma VPN entre a estação com *softphone* e o servidor para que as conseqüências dessa adoção fossem estudadas.

O pacote escolhido para a implementação de VPN foi o *OpenVPN* [Lockhart04]. O OpenVPN é uma solução de VPN baseada em SSL. Através do YUM, instalou-se o pacote OpenVPN no servidor. Os arquivos de configuração do *OpenVPN* estão no diretório */etc/openvpn*. O tipo de autenticação escolhida para ser realizada entre o cliente e o servidor foi a que utiliza chaves públicas e privadas. O pacote *OpenVPN* traz arquivos com exemplos de configurações no diretório */usr/share/doc/openvpn-<versão>/sample-config-files*. O arquivo *server.conf* foi copiado para o diretório */etc/openvpn* para que fossem realizadas as alterações necessárias. As alterações realizadas estão apresentadas na Tabela 8.

Tabela 8. Alterações no arquivo de configuração do *OpenVPN*.

Linha alterada	Função
local 10.0.0.254	Identifica o IP que receberá as requisições de VPN
server 10.0.1.0 255.255.255.0	Identifica os IPs que serão dados aos clientes, o servidor fica com o primeiro IP da classe

Para a criação das chaves públicas e privadas do servidor, inicialmente, foi necessário copiar o diretório */usr/share/openvpn/easy-rsa* para o diretório */etc/openvpn*. Após a cópia, os passos do Quadro 17 foram seguidos utilizando os dados da Tabela 9 para preencher os campos.

```
[root@server ~]# cd /etc/openvpn
[root@server openvpn]# cp -pr /usr/share/openvpn/easy-rsa .
[root@server openvpn]# cd easy-rsa/
[root@server easy-rsa]# . ./vars
[root@server easy-rsa]# ./clean-all
[root@server easy-rsa]# ./build-ca
[root@server easy-rsa]# ./build-key-server server
[root@server easy-rsa]# ./build-dh
[root@server easy-rsa]# cp keys/ca.crt ../.
[root@server easy-rsa]# cp keys/server.key ../.
[root@server easy-rsa]# cp keys/server.crt ../.
[root@server easy-rsa]# cp keys/dh1024.pem ../.
```

Quadro 17. Passos para criação das chaves do servidor.

Tabela 9. Parâmetros preenchidos na criação das chaves do servidor.

Certification Authority (./build-ca)	
Country	BR
State or Province Name	Pernambuco
Locality Name	Recife
Organization Name	VoIP
Common Name	server-CA
Server Key (./build-key-server server)	
Country	BR
State or Province Name	Pernambuco
Locality Name	Recife
Organization Name	VoIP
Common Name	Server

Após a configuração do servidor é necessário permitir o tráfego de algumas determinadas portas, no *firewall*, para o funcionamento da VPN. As regras da Tabela 10 foram adicionadas.

Tabela 10. Regras adicionadas ao *firewall* para acesso a VPN.

Parâmetros do comando *iptables*	Função
-A INPUT –p udp --dport 1194 –j ACCEPT	Permite acesso ao OpenVPN (porta UDP/1194)
-A INPUT –s 10.0.1.0/24 –p udp –m udp --dport 5060 –j ACCEPT	Permite acesso da rede VPN ao servidor na porta UDP/5060 (SIP)
-A INPUT –s 10.0.1.0/24 –p udp –m udp --dport 10000:20000 –j ACCEPT	Permite acesso da rede VPN ao servidor nas portas de UDP/10000 a 20000 (fluxo de *media*)

Após a preparação do servidor foi necessário preparar as chaves do cliente. Os passos do Quadro 18 foram seguidos utilizando os parâmetros da Tabela 11 para criar as chaves do cliente OpenVPN para Windows.

```
[root@server easy-rsa]# . ./vars
[root@server easy-rsa]# ./build-key-pass client1
```

Quadro 18. Criação das chaves do cliente.

Tabela 11. Parâmetros para a criação das chaves do cliente.

Client Key (./build-key-pass client1)	
PEM pass phrase	testevoip
Country	BR
State or Province Name	Pernambuco
Locality Name	Recife
Organization Name	VoIP
Common Name	client1

Após a criação das chaves do cliente, o serviço OpenVPN foi iniciado no servidor.

Na estação Windows, o cliente OpenVPN-GUI foi instalado. Os arquivos do servidor no diretório */etc/openvpn/easy-rsa/keys*, *ca.crt*, *client1.crt* e *client1.key*, foram copiados para a estação no diretório "*\Arquivos de programas\OpenVPN\config*". Apenas o arquivo *ca.crt* mudou de nome para *client1-ca.crt*. Copiou-se o arquivo

\Arquivos de programas\OpenVPN\sample-config\client.ovpn

para o diretório

\Arquivos de programas\OpenVPN\config

com o nome de *client1.ovpn* e as alterações apresentadas na Tabela 12 foram realizadas.

Tabela 12. Modificações do arquivo *client1.ovpn*.

Linhas Alteradas	Função
remote 10.0.0.254 1194	Identifica o IP e porta do servidor VPN
ca client1-ca.crt	Especifica o nome do arquivo da Certification Authority
cert client1.crt	Especifica o nome do arquivo do certificado do cliente
key client1.key	Especifica o nome do arquivo da chave privada do cliente

Após a configuração do cliente OpenVPN, o estabelecimento da VPN foi testado ao clicar no ícone da barra de tarefas com o botão direito e selecionar client1->Connect. Após a inserção da senha da chave privada, a VPN foi estabelecida com sucesso.

Foram realizadas coletas de áudio de forma semelhante às realizadas anteriormente nos testes do *ping flood*. Do ramal 13 (*softphone* X-pro, similar ao X-lite, porém com mais recursos) foi realizada uma ligação para o número 600 (teste de eco) e depois para o ramal 10 (ATA). Inicialmente a configuração do *softphone* estava por fora da VPN, ou seja, o servidor SIP configurado era o IP 10.0.0.254. Após colocá-los em conferência, um pedaço de áudio foi armazenado e separado para análise posterior.

Depois da primeira coleta, estabeleceu-se novamente a VPN e a configuração do *softphone* foi modificada para contemplar o servidor SIP no IP 10.0.1.1 que é o IP do servidor através da VPN. O mesmo procedimento foi adotado, ou seja, foi realizada a ligação do ramal 13 para o número 600 e depois feita a conferência com o ramal 10. O áudio foi coletado e separado para análise futura.

A Figura 24 apresenta o sinal de áudio da primeira coleta. O tempo de resposta do teste de eco está representado na parte superior do retângulo inserido na figura. A parte inferior do retângulo contém a freqüência do áudio naquele momento, porém esse dado não é importante para as análises. O tempo de resposta foi de 151,88 ms, porém como foi utilizado o recurso de conferência, ele deverá ser dividido por dois para se ter o tempo real aproximado. Esse tempo divido por dois resulta em 75,94 ms.

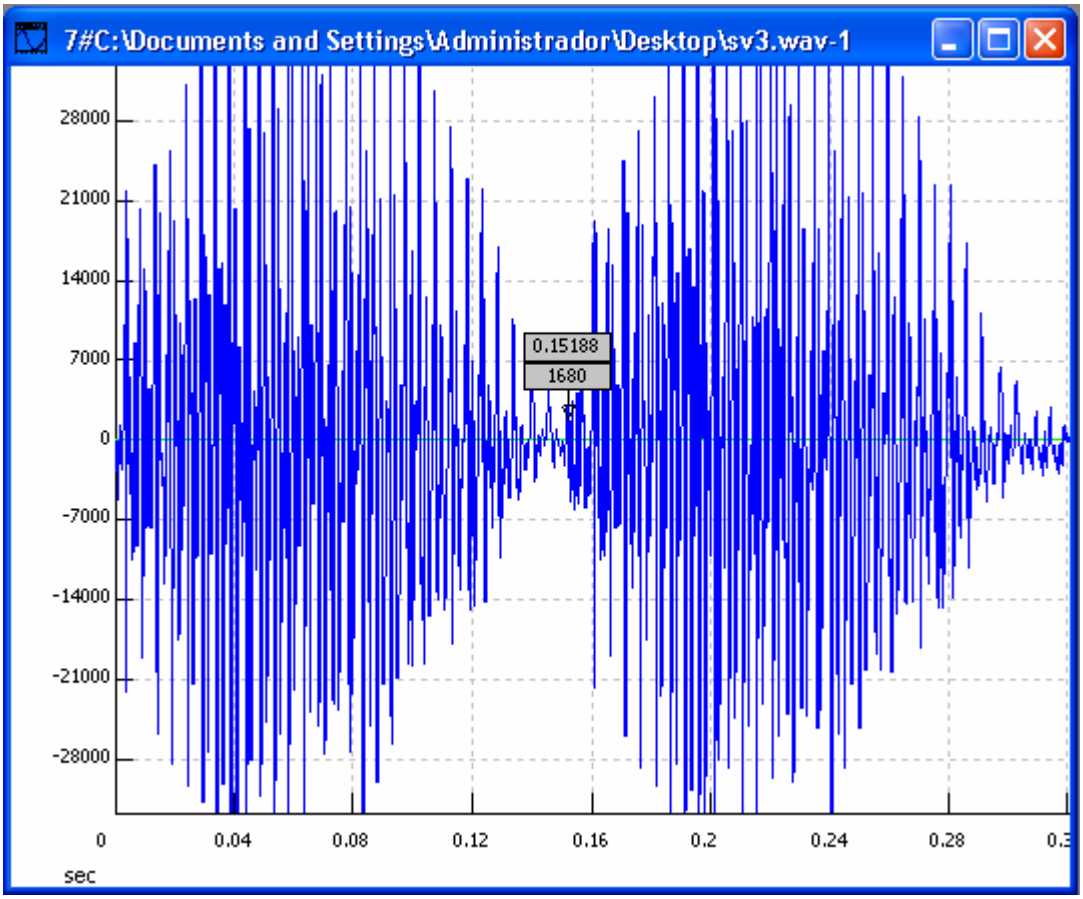

Figura 24. Coleta de áudio sem a utilização da VPN.

A Figura 25 apresenta o sinal de áudio da segunda coleta. Nesse caso o tempo de resposta do teste de eco foi de 258,38 ms. O resultado da divisão por dois, do tempo encontrado, é 129,19 ms.

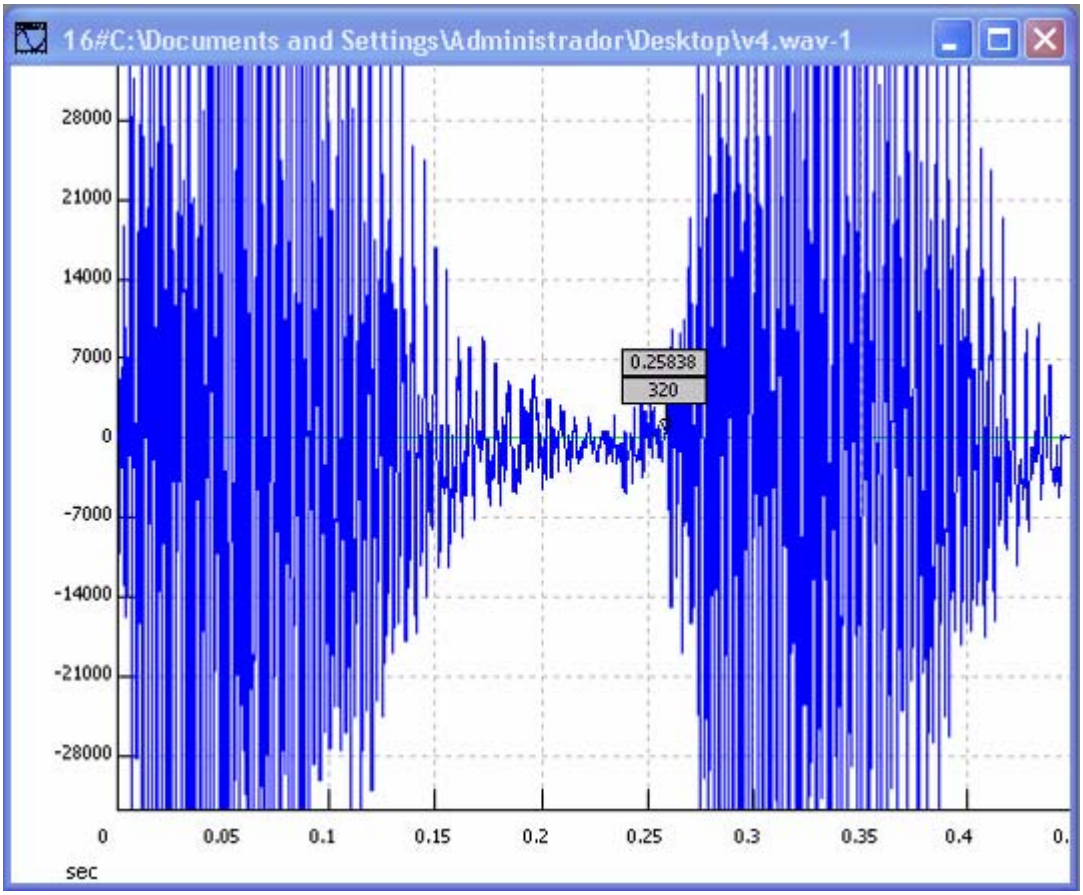

Figura 25. Coleta de áudio com a utilização da VPN.

Conforme já esperado foi adicionado um atraso adicional no áudio. Esse atraso é causado pelo tempo de cifragem do conteúdo dos pacotes. A diferença foi de apenas 53,25 ms. O que é completamente aceitável. Em testes de conversação esse tempo passou imperceptível.

Após os testes chegou-se à conclusão que essa solução sugerida e testada pode ser aplicada com sucesso nos ambientes semelhantes ao estudado. As estações com *softphone* podem estabelecer uma VPN para o servidor garantindo assim a confidencialidade da conversa. Deve-se lembrar que o tempo de cifragem é diretamente proporcional ao poder de processamento do servidor e da sua disponibilidade.

4.3 Resultados dos estudos na rede local (primeiro cenário)

Inicialmente foram demonstradas algumas fragilidades do cenário objeto do estudo. Através do ataque de *ping flood* com destino aos ATAs, à estação com softphone e ao servidor, e do ataque de UDP *flood* com destino ao servidor, foram comprometidos dois fatores da segurança da informação: a disponibilidade e a integridade.

Como recomendação, sugere-se o isolamento dos ATAs em uma rede exclusiva e que sejam aplicadas as mesmas restrições de acesso físico dos servidores a eles.

Em relação ao *softphone* recomenda-se utilizar duas placas de rede, uma para a rede de dados convencional e outra exclusiva para VoIP para minimizar os riscos de ataque.

Através da utilização da técnica de ARP *Spoofing* foi obtido sucesso no comprometimento da confidencialidade em uma chamada VoIP. Foi, então, aplicada uma VPN entre a estação com o softphone e o servidor, e a diferença 53,25 ms entre o tempo de resposta sem a VPN e com a aplicação de VPN não prejudica a chamada, conforme afirmado por James Kurose e Keith Ross [Kurose03]: “Para aplicações de áudio altamente interativas, como o telefone por Internet, atrasos fim a fim menores do que 150 milissegundos não são percebidos pelo ouvido humano;”.

4.4 Preparação do ambiente para estudos com a Internet (segundo cenário)

A preparação do ambiente para o estudo da parte do cenário que utiliza a Internet foi iniciada após o estudo do cenário com foco na rede local ter sido finalizado.

As regras do *firewall* que foram adicionadas e suas respectivas explicações estão apresentadas na Tabela 13.

Tabela 13. Regras adicionadas ao *firewall* para o cenário com a *Internet*.

Parâmetros do comando *iptables*	**Função**
-t nat –A POSTROUTING -s 10.0.0.0/24 -j MASQUERADE	Permite a utilização de NAT através da rede local.
-A INPUT -p tcp -m tcp ! --tcp-flags SYN,RST,ACK SYN -j ACCEPT	Permite a entrada, no servidor, do pacote TCP que não seja tentativa de nova conexão, ou seja, permite as respostas às requisições previamente efetuadas.
-A INPUT -p udp -m udp --dport 4569 -j ACCEPT	Permite o acesso ao servidor na porta UDP/4569 (IAX2)
-A INPUT -p udp -m udp --sport 53 -j ACCEPT	Permite receber as respostas dos servidores DNS (porta UDP/53). Não foram especificados os servidores DNS de origem já que este tipo de segurança não está no escopo deste projeto.
-A INPUT -s 10.0.0.0/24 -p udp -m udp --dport 53 -j ACCEPT	Permite receber as requisições dos clientes DNS dos nós da rede interna com destino ao serviço DNS local (porta UDP/53).
-A FORWARD -s 10.0.0.0/24 -p tcp -m tcp --dport 80 –j ACCEPT	Permite a passagem de tráfego com origem na rede local e destino na porta TCP/80.
-A FORWARD -d 10.0.0.0/24 -p tcp -m tcp ! --tcp-flags SYN,RST,ACK SYN -j ACCEPT	Permite a passagem do pacote TCP, que não seja tentativa de nova conexão, para os nós da rede local.

Foi necessário modificar o valor do parâmetro *net.ipv4.ip_forward* para 1 (um) no arquivo */etc/sysctl.conf* para habilitar o *IP Forwarding*. Essa modificação permitirá o encaminhamento de pacotes entre as diferentes interfaces de rede. Para que os parâmetros do arquivo */etc/sysctl.conf* sejam lidos sem a necessidade de reiniciar o servidor, é necessário executar o comando apresentado no Quadro 19.

```
[root@server ~]# sysctl -p
```

Quadro 19. Comando executado para que o arquivo */etc/sysctl.conf* seja lido.

Após a configuração das regras do *firewall* para o acesso à Internet no servidor da matriz foi necessário instalar e configurar o servidor que simularia o *firewall* e servidor de telefonia da filial. Os mesmos procedimentos de instalação do servidor da matriz foram utilizados.

A diferença inicial em relação a instalação dos servidores foi o endereço IP da interface de rede local que, neste caso, foi escolhido o 192.168.0.254 de máscara de rede 255.255.255.0. O acesso à Internet também foi realizado através da tecnologia ADSL. As larguras de banda foram as mesmas: 1024 kbps para *download* e 328 kbps para *upload*. O mesmo procedimento de configuração da Internet do servidor da matriz foi seguido, porém não foi necessário modificar o arquivo */etc/sysconfig/network-scripts/ifcfg-ppp0*, pois o modem ADSL utilizado era um SpeedStream da Efficient Networks que não necessita do parâmetro PPPOE_EXTRA.

Os mesmos procedimentos utilizados para a instalação dos pacotes: *asterisk-1.2.5*, *zaptel-1.2.4* e *libpri-1.2.2*, no servidor da matriz foram realizados. O arquivo */etc/sysctl.conf* também foi modificado conforme instruções anteriores.

As mesmas regras do *iptables* do servidor da matriz apresentadas na 0 e na 0 foram adicionadas ao servidor da filial, porém as três primeiras posições do endereço de rede foram modificadas de 10.0.0 para 192.168.0 para que pudesse ser contemplado a rede do servidor da filial.

Foi necessário habilitar o serviço de DNS em ambos os servidores para que as estações pudessem utilizar o serviço. O pacote BIND (*Berkeley Internet Name Domain*), que implementa o DNS no FC4 já estava instalado e já vem pré-configurado como servidor de *cache*, ou seja, foi apenas necessário iniciá-lo. Os comandos para o início do serviço e para que ele fosse iniciado automaticamente na reinicialização do servidor estão respectivamente apresentados no Quadro 20.

```
[root@server ~]# service named start
[root@server ~]# chkconfig named on
```

Quadro 20. Comandos executados para habilitar o serviço DNS.

O próximo passo foi a configuração do Asterisk para interligar a matriz e a filial. O protocolo escolhido para ser utilizado nessa conexão foi o IAX2. Essa escolha foi motivada principalmente pela característica de *trunking* que o IAX2 é capaz de fazer por utilizar uma mesma porta para trafegar a sinalização e o fluxo de mídia, caso exista mais de um canal de voz entre os mesmos nós. Essa característica é extremamente importante por economizar largura de banda e permitir que mais canais trafeguem através do mesmo link. Através do *trunking*, por exemplo, seria possível passar pelo menos 103 chamadas utilizando o codec G.729 em um link de 1 Mbps [Segrest05].

O serviço ADSL da concessionária telefônica local utiliza DHCP para fornecer o IP válido de acesso à Internet. Por esse motivo, foi utilizado um recurso fornecido pela Ligu Consultores Associados que permitia que, ao se conectar à Internet, o servidor atualizasse o seu IP em uma base de dados para que fosse consultado posteriormente. Esse IP consultado foi utilizado nos arquivos de configuração do Asterisk. Esse recurso não foi automatizado, pois não faz parte do escopo deste projeto.

A utilização de DNS dinâmico foi descartada devido à demora no tempo de atualização do DNS.

O arquivo de configuração do asterisk para o protocolo IAX2 é o */etc/asterisk/iax.conf*. As linhas de configuração inicialmente adicionadas ao *iax.conf* em ambos os servidores estão apresentadas no Quadro 21. Foram adicionadas as linhas iniciadas por *disallow* e *allow* para permitir apenas o uso do codec iLBC devido ao pouco consumo de largura de banda com boa

qualidade e por possuir melhor qualidade quando há perda de pacotes [Aguru06] [Ilbcfree06]. Além disso, foi habilitada a opção *jitterbuffer* e escolhido o valor 2 para a opção *dropcount* nas opções gerais do arquivo de configuração. Essas opções habilitam o *buffer* para minimizar problemas com o *jitter* e faz com que o tamanho do *buffer* se ajuste automaticamente para que no máximo 2 (valor do *dropcount*) pacotes sejam descartados em 2 segundos.

```
[testevoipiax]
type=friend
host=dynamic
secret=testevoip
qualify=yes
disallow=all
allow=ilbc
context=internetcall
```

Quadro 21. Linhas adicionadas ao *iax.conf* para configuração do servidor.

Essas linhas adicionadas especificam a configuração de cada servidor remoto, porém ainda foi necessário adicionar uma linha para que cada servidor pudesse se registrar no outro. A linha adicionada em cada servidor nos parâmetros [general] do arquivo */etc/asterisk/iax.conf* está apresentada no Quadro 22.

```
register => testevoipiax:testevoip@ip_do_outro_servidor
```

Quadro 22. Linha adicionada ao *iax.conf* para registro no outro servidor.

No servidor da matriz, no contexto [ramais] do *extensions.conf*, foi adicionada a linha apresentada no Quadro 23 que permitia que, ao se discar 2 (dois) mais qualquer ramal, o ramal no servidor da filial fosse chamado.

```
exten => _2.,1,Dial(IAX2/testevoipiax@testevoipiax/${EXTEN:1})
```

Quadro 23. Linha adicionada ao *extensions.conf* para discar ramal no outro servidor.

No servidor da filial foram adicionadas as linhas apresentadas no Quadro 24 no arquivo *extensions.conf*.

```
[internetcall]
exten => 600,1,Playback(demo-echotest)
exten => 600,n,Echo
exten => 600,n,Playback(demo-echodone)
```

Quadro 24. Linhas adicionadas ao *extensions.conf* no servidor da filial.

Os servidores ficaram configurados de forma que, ao discar 2600 a partir do servidor da matriz, fosse executado o teste de eco no servidor da filial.

Os testes neste ambiente foram iniciados após a finalização da configuração dos servidores.

4.5 Estudos com a *Internet* (segundo cenário)

Os testes foram realizados para tentar, inicialmente, demonstrar algumas fragilidades deste cenário em relação aos três fatores de segurança da informação. Após o término desses testes, e a

conseqüente identificação de algumas fragilidades do ambiente, foram implementados alguns recursos de segurança em rede, para que fosse possível estudar as implicações destas implementações.

O primeiro fato verificado foi que, a comunicação VoIP era prejudicada quando o tráfego de acesso à Internet, a partir das estações, era mais intenso. Através dos testes de eco remoto puderam ser verificados atrasos e cortes na voz quando, por exemplo, alguns arquivos estavam sendo transferidos, da *Internet* e para a *Internet*, pelas estações.

Sabe-se que só é possível controlar a qualidade de serviço (QoS) em um *link* de um ponto a outro, caso se tenha o controle de todos os nós por onde o pacote trafega. Esse não é o caso da Internet [Kurose03]. Porém, neste cenário, através desse comportamento apresentado, já se verificou a necessidade de implementar QoS nos servidores da matriz e filial, para que os problemas encontrados fossem minimizados.

Uma das alternativas utilizada para a resolução desse problema foi a implementação de reserva de largura de banda para determinados tráfegos. Para que fosse possível essa implementação nos servidores Linux, através do uso de *qdiscs* (escalonadores), foi escolhida a alternativa de utilização do módulo IMQ (*Intermediate Queueing Device*) [Linuximq06] no *kernel* do sistema operacional. Para essa implementação, foi necessário baixar o *patch* do módulo IMQ e aplicá-lo ao código fonte do *kernel*.

Para que o módulo pudesse ser compilado, foi necessário baixar o código fonte do *kernel* do Fedora Core 4, cuja versão foi a 2.6.15-1.833_FC4, os *patches* do IMQ para esse *kernel* e também para o *iptables*, já que seria necessário também baixar e compilar este pacote.

Os passos realizados para a aplicação do *patch* IMQ no *kernel* e para sua compilação estão apresentados no Quadro 25.

```
[root@server ~]# rpm -ivh kernel-2.6.15-1.1833_FC4.src.rpm
[root@server ~]# cd /usr/src/redhat
[root@server redhat]# rpm-build -bp --target=i686 SPECS/kernel-2.6.spec
[root@server redhat]# cd /usr/src
[root@server src]# ln -s redhat/BUILD/kernel-2.6.15/linux-2.6.15/ .
[root@server src]# ln -s linux-2.6.15/ linux
[root@server src]# cd linux
[root@server linux]# patch -p1 < /root/linux-2.6.14-imq6.diff
[root@server linux]# cp configs/kernel-2.6.15-i686.config .config
[root@server linux]# make oldconfig
[root@server linux]# make menuconfig
[root@server linux]# make bzImage && make modules && make modules_install
[root@server linux]# cp arch/i386/boot/bzImage /boot/vmlinuz-2.6.15-prep
[root@server linux]# mkinitrd -v /boot/initrd-2.6.15-prep.img 2.6.15-prep
```

Quadro 25. Procedimento realizado para compilar o *kernel*.

Após esse procedimento, foi necessário adicionar as linhas apresentadas no Quadro 26 no arquivo */etc/grub.conf*. Essa modificação era necessária para que o novo *kernel* pudesse ser utilizado na próxima inicialização. Após essa modificação o servidor foi reiniciado.

```
title Fedora Core (2.6.15-prep)
        root (hd0,1)
        kernel /vmlinuz-2.6.15-prep ro root=LABEL=/ rhgb quiet
        initrd /initrd-2.6.15-prep.img
```

Quadro 26. Linhas adicionadas ao */etc/grub.conf*.

Para a aplicação do *patch* no *iptables* foi necessário baixar o seu código fonte, cuja versão foi a 1.3.5.

O procedimento realizado para a aplicação do *patch* no *iptables* e sua recompilação está apresentado no Quadro 27. Na edição do arquivo *extensions/Makefile* que consta no procedimento foi adicionado ao final das linhas que começam com PF_EXT_SLIB e PF6_EXT_SLIB a palavra IMQ.

```
[root@server ~]# tar xvjf iptables-1.3.5.tar.bz2
[root@server ~]# cd iptables-1.3.5
[root@server iptables-1.3.5]# patch -p1 < ../iptables-1.3.0-imq1.diff
[root@server iptables-1.3.5]# chmod 755 extensions/.IMQ*
[root@server iptables-1.3.5]# vi extensions/Makefile
[root@server iptables-1.3.5]# cd /usr/src/linux/include/linux/netfilter_ipv4
[root@server netfilter_ipv4]# cp ipt_IMQ.h /usr/include/linux/netfilter_ipv4/.
[root@server netfilter_ipv4]# cd ../netfilter_ipv6
[root@server netfilter_ipv6]# cp ip6t_IMQ.h /usr/include/linux/netfilter_ipv6/.
[root@server netfilter_ipv6]# cd /root/iptables-1.3.5/
[root@server iptables-1.3.5]# make KERNEL_DIR=/usr/include BINDIR=/sbin LIBDIR=/
lib MANDIR=/usr/share/man
[root@server iptables-1.3.5]# make KERNEL_DIR=/usr/include BINDIR=/sbin LIBDIR=/
lib MANDIR=/usr/share/man install
```

Quadro 27. Procedimento realizado para compilar o *iptables*.

A aplicação dos *patches* e a recompilação do *kernel* e do *iptables* foram realizadas nos dois servidores. O objetivo principal foi separar uma largura de banda de 128 kbps de *upload* e *download* para o tráfego dos canais de voz e liberar o restante para o acesso geral à Internet. Essa alocação de largura de banda é feita dinamicamente, ou seja, caso os canais de voz não estejam sendo utilizados a largura de banda fica disponível para outras requisições.

Através do que foi exposto por Dan Singletary [Singlet06], verificou-se a necessidade de um cuidado adicional em relação ao controle de largura de banda utilizada nos *links* ADSL. Como já descrito anteriormente, o *link* disponibilizado para os dois servidores foi de 1 Mbps para *download* e 328 kbps para *upload*.

A interface que conectava o modem ao servidor era de 10 Mbps. Dessa forma, os dados para *upload* eram enviados ao modem à velocidade de 10 Mbps, porém a velocidade do *link* ADSL era de apenas 328 kbps, o que ocasionava o enfileiramento dos pacotes para serem enviados do modem ADSL.

Ocasionalmente, a fila de pacotes do modem fica cheia e os pacotes que chegam ao modem vão sendo descartados até que algum pacote seja transmitido e seja liberado um espaço na fila. O protocolo TCP foi desenvolvido para, automaticamente, ajustar o tamanho da janela de transmissão para aproveitar o máximo da largura de banda disponível. Porém, enquanto o enfileiramento de pacotes é um recurso que melhora o aproveitamento da largura de banda, esse mesmo enfileiramento pode causar maior latência para aplicações interativas em tempo real.

O tráfego de *download* também é enfileirado da mesma forma que os pacotes são enfileirados para o envio. Porém, essa fila é formada no provedor do *link* de dados. Dessa forma, é impossível ter o controle direto do enfileiramento dos pacotes, mas é possível diminuir a latência tentando fazer com que as conexões não enviem os dados muito rápido, já que a maioria do tráfego da rede é através do protocolo TCP. Existem duas maneiras de atingir esse objetivo: através do descarte de pacotes entrantes e através da manipulação do tamanho da janela de entrada que é disponibilizada.

Para tentar evitar que os pacotes não fossem enfileirados nos *modems* ADSL, foi necessário limitar a largura de banda de entrada e saída para 75% do total da capacidade do *link*. Chegou-se a esse valor através dos seguintes testes:

- um *ping* foi executado com destino a um servidor com IP 64.34.45.100 e o tempo médio do *ping* após 120 coletas foi de 181 ms com 0% de perda de pacotes. Nessa primeira coleta a Internet não estava sendo utilizada;
- uma largura de banda de 128 kbps foi reservada para os pacotes que entravam e saíam do servidor com o IP 64.34.45.100 (a mesma largura de banda que será reservada para os canais de voz);
- de uma estação Windows e de uma estação Linux foram colocados vários arquivos para serem baixados e para serem enviados;
- foi ativado o controle de tráfego inicialmente com 100% de utilização do link e verificado o tempo médio do *ping* para o servidor e a perda de pacotes após 120 coletas;
- aumenta-se a limitação da largura de banda de entrada e saída de 5 em 5% e o procedimento foi repetido até que o tempo médio dos *pings* estivesse próximo da primeira marcação e que o tempo dessa marcação fosse parecido com a da marcação anterior (5% a menos).

A Tabela 14 apresenta uma tabela com os tempos médios e o percentual de perda de pacotes para cada limitação da largura de banda.

Tabela 14. Tabela com tempos médios e perda de pacotes para limitação de banda.

Limitação	100%	95%	90%	85%	80%	**75%**	70%
Tempo médio (ms)	636	646	643	580	395	**246**	238
Perda de pacotes	20%	12%	15%	8%	0%	**0%**	0%

Já que o procedimento para a habilitação do controle de tráfego é bastante trabalhoso, foi criado um script Shell [Beebe05] [Newham05], que é apresentado no Apêndice A, para facilitar a habilitação e desabilitação desse controle. Esse script pode ser facilmente alterado para se adequar a diferentes velocidades de *links* e também para diferentes reservas de largura de banda.

Após essa implementação, verificou-se que não havia mais problemas de lentidão causados pelo uso da Internet através das estações.

O próximo passo foi realizar testes relativos à fragilidade do ambiente nos quesitos de disponibilidade e integridade. Foram realizados novamente testes de DoS só que desta vez através da Internet.

Inicialmente tentou-se comprometer o servidor da filial com o uso do UDP *Flood* através do PackETH em uma estação Linux conectada à rede local do servidor da matriz. Foi necessário liberar no *firewall* o tráfego de entrada e saída a partir dessa estação. Os comandos apresentados no Quadro 28 foram executados para que isso fosse possível.

```
[root@server ~]# iptables -I FORWARD -s 10.0.0.3 -j ACCEPT
[root@server ~]# iptables -I FORWARD -d 10.0.0.3 -j ACCEPT
```

Quadro 28. Comandos para liberação do acesso da estação.

Após essas modificações e o início do UDP *flood*, foram realizados chamadas para testes com o eco remoto para que fosse identificado possíveis problemas. As chamadas foram realizadas com sucesso.

Foi verificado que não houve problemas, pois a velocidade com que o UDP *Flood* era enviado, foi bem inferior à velocidade de *download* do *link* do servidor da filial. Como não se dispunha de um *link* com velocidade de *upload* superior, optou-se por tentar comprometer um outro servidor interligado à Internet que dispunha de um *link* ADSL de 256 kbps de *download* e 128 kbps de *upload*.

Neste terceiro servidor, chamado de servidor de testes, configurou-se o controle de tráfego para limitar a 70% da largura de banda total do *link* ADSL e foi reservado 64 kbps para o tráfego entre ele e o servidor da filial.

Foi colocado um *ping* com 120 coletas para se obter o tempo médio de resposta entre o servidor de testes e o servidor da filial. O tempo médio apresentado foi de 59,278 ms.

Através da mesma estação Linux da rede local do servidor da matriz foi disparado o UDP *flood* com destino ao servidor de testes. E no servidor de testes foi colocado o *ping* para o servidor da filial. Mesmo com o controle de tráfego, houve perda de pacotes e respostas do *ping* de até 3212 ms. Através desse teste, foi confirmada a eficácia desse tipo de ataque, ou seja, caso a largura de banda de *upload* do originador do ataque seja maior ou equivalente à largura de banda de *download* do alvo o ataque obterá sucesso.

Esse tipo de ataque será eficaz independentemente das ações que possam ser tomadas. Uma alternativa que pode minimizar o tempo de exposição do servidor a esse tipo de ataque é a utilização de um IDS para que o mesmo possa identificar o *flood* e reiniciar automaticamente a conexão ADSL. Como o serviço ADSL da concessionária telefônica local utiliza IP dinâmico, na reinicialização o servidor possuiria um novo endereço IP que não mais estaria sendo alvo de ataques. É de extrema importância também que o *log* do IDS seja armazenado e enviado para o provedor de Internet para que ele possa tomar as medidas necessárias para tentar identificar e punir os responsáveis. Nos Estados Unidos, por exemplo, esse tipo de ataque é considerado crime com punição de 5 a 10 anos de prisão [Pelaez02].

Já é sabido, através dos testes realizados com cenário local, a fragilidade da implementação de VoIP sem cifragem do conteúdo. Para o cenário que utiliza a Internet, foram estudadas as dificuldades de implementação e quais as implicações de aplicar dois métodos diferentes de VPN para a comunicação pela Internet.

Um das formas de implementação de VPN, que é a que utiliza o *OpenVPN*, já foi estudado na rede local em testes anteriores, porém desejava-se saber quais as diferenças que poderiam ser encontradas na aplicação dessa implementação através da Internet, entre os dois servidores.

A outra forma de implementação de VPN escolhida para este estudo, foi a que utiliza o SSH, que diferentemente do *OpenVPN*, utiliza o protocolo TCP para o transporte dos dados.

Inicialmente foi gerado um arquivo de áudio de uma chamada, semelhante aos gerados nos testes anteriores com o teste de eco remoto, para que fosse comparado posteriormente com os gerados com a utilização das VPNs.

A primeira VPN foi implementada utilizando o SSH. Para isso, foi necessário seguir o seguinte procedimento:

- instalar o pacote *pty-redir98-1.0-1.i386.rpm* no servidor da filial;
- editar o arquivo */etc/ppp/options* no servidor da matriz e deixá-lo conforme apresentado no Quadro 29.

```
lock
ipcp-accept-local
ipcp-accept-remote
proxyarp
noauth
```

Quadro 29. Arquivo */etc/ppp/options*.

- criar um usuário vpn-user no servidor da matriz com o shell: */usr/sbin/pppd*;
- modificar a permissão do arquivo */usr/sbin/pppd* no servidor da matriz para 4755 (*setuid*, ou seja, ele será executado com as permissões do dono do arquivo que é o root) utilizando o chmod;
- gerar chaves assimétricas no servidor da filial com o comando apresentado no Quadro 30. A chave pública foi colocada no servidor da matriz para o usuário vpn-user, para que o método de autenticação fosse realizado através das chaves assimétricas.

```
[root@server ~]# ssh-keygen -t rsa
```

Quadro 30. Comando para geração das chaves assimétricas.

Para iniciar a VPN foi necessário digitar os comandos apresentados no Quadro 31 no servidor da filial.

```
[root@server ~]# /usr/bin/pty-redir98 /usr/bin/ssh -l vpn-user -t -e none -o 'Ba
tchmode yes' -o 'StrictHostKeyChecking no' -c blowfish 200.165.251.98 > /tmp/vpn
-device
[root@server ~]# /usr/sbin/pppd `cat /tmp/vpn-device` 10.0.1.2:10.0.1.1
```

Quadro 31. Comandos para iniciar a VPN entre os servidores.

Também foi necessário habilitar, no *firewall*, o tráfego entre os nós. Para habilitar esse tráfego no *firewall*, as seguintes regras, apresentadas na Tabela 15, foram incluídas através do *iptables*.

Tabela 15. Regras adicionadas ao *firewall* para habilitar o tráfego das VPN.

Parâmetros do comando *iptables*	**Função**
-A INPUT -s 10.0.1.1 -j ACCEPT	Regra para permitir o acesso a partir do host 10.0.1.1 no servidor da filial.
-A INPUT -s 10.0.1.2 -j ACCEPT	Regra para permitir o acesso a partir do host 10.0.1.2 no servidor da matriz.

Após a VPN ter sido estabelecida, para que o tráfego passasse por ela, foi necessário modificar os arquivos */etc/asterisk/iax.conf* nos servidores da matriz e da filial para contemplar os novos IPs do servidor. O IP do servidor da matriz pela VPN era o 10.0.1.1 e o da filial era o 10.0.1.2.

Após essas modificações o Asterisk foi reiniciado e mais uma coleta de arquivo de áudio, semelhante à anterior, foi feita para comparação e análise posterior.

Para a criação da VPN, através do *OpenVPN*, do servidor da filial para o servidor da matriz, foi necessário instalar o pacote *OpenVPN* no servidor da filial. O pacote foi instalado da mesma maneira que no servidor da matriz.

Os arquivos *client1-ca.crt*, *client1.ovpn*, *client1.crt* e *client1.key*, do diretório "*C:\Arquivos de Programas\OpenVPN\config*", colocados na estação Windows de onde se

realizou os testes de VPN da rede local, foram copiados para o servidor da filial no diretório */etc/openvpn*. O arquivo *client1.ovpn* foi renomeado para *client1.conf*.

Para que fosse possível especificar o IP que o servidor da filial receberia após o estabelecimento da VPN, foi necessário editar o arquivo */etc/openvpn/server.conf* do servidor da matriz e remover o comentário da linha apresentada no Quadro 32, criar um diretório */etc/openvpn/ccd* e criar o arquivo *client1* dentro deste diretório com o conteúdo apresentado no Quadro 33.

```
client-config-dir ccd
```

Quadro 32. Linha descomentada do arquivo */etc/openvpn/server.conf*.

```
ifconfig-push 10.0.1.2 10.0.1.1
```

Quadro 33. Conteúdo do arquivo */etc/openvpn/ccd/client1*.

Foi necessário incluir nas regras do *firewall* do servidor da filial a entrada apresentada na Tabela 16.

Tabela 16. Regras do *firewall* adicionadas ao servidor da filial.

Parâmetros do comando *iptables*	**Função**
-A INPUT -p udp -m udp --sport 1194 -j ACCEPT	Permite que o cliente VPN no servidor da filial receba as respostas do servidor da matriz.

Não foi necessário modificar as configurações do Asterisk, pois as últimas modificações já contemplavam os novos IPs 10.0.1.1 e 10.0.1.2 do servidor da matriz e da filial, respectivamente.

Foi realizada mais uma coleta de áudio para comparação e análise posterior através da VPN com o *OpenVPN*.

A Figura 26 apresenta os sinais de áudio das seguintes coletas do teste de eco remoto, respectivamente: a) sem a utilização de VPN; b) com a utilização de VPN (*OpenVPN*); c) com a utilização de VPN (SSH). O tempo representado na parte superior dos retângulos representa o tempo de resposta dos testes de eco.

Obteve-se sucesso na implementação das VPNs e na realização dos testes das chamadas. Através da observação de uma pequena diferença nos tempos entre os testes de eco e da não percepção de prejuízo na qualidade das chamadas, pôde-se concluir que a implementação de VPN não ocasionou problemas nas chamadas.

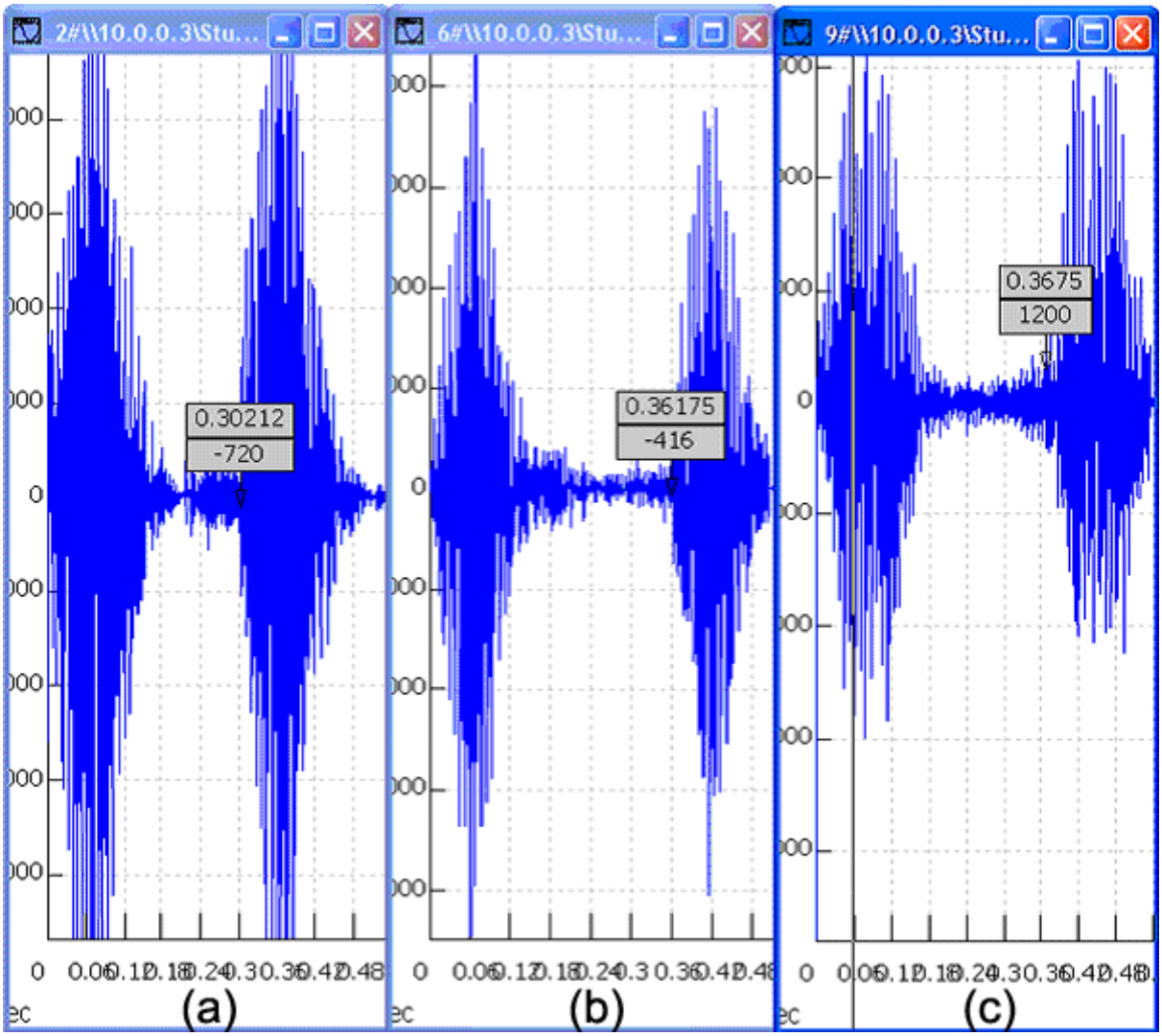

Figura 26. Coletas de áudio: a) sem a utilização de VPN; b) com a utilização através do *OpenVPN*; c) com a utilização através de SSH.

4.6 Resultados dos estudos com a *Internet* (segundo cenário)

Inicialmente foi descoberta a necessidade de limitar o tráfego da Internet para evitar a criação de fila de pacotes nos modems ADSL. Após a limitação foi necessário reservar uma largura de banda dinâmica específica para as chamadas VoIP para que elas não concorresem com o tráfego normal de dados.

Mesmo após a limitação e reserva de banda para VoIP, foi obtido sucesso no ataque de UDP *Flood* ao servidor de testes. Com isso, verificou-se que, caso a velocidade de *upload* do atacante seja maior que a velocidade de *download* da vítima, o ataque obterá sucesso. Essa é uma fragilidade da arquitetura dos protocolos utilizados nas redes TCP/IP.

Como já era conhecida a fragilidade do cenário quanto ao quesito confidencialidade, uma VPN entre o servidor da matriz e o servidor da filial foi estabelecida utilizando o OpenVPN e, posteriormente, outra utilizando o SSH com PPP. Foi verificado pela pequena diferença de aproximadamente 30 ms no tempo de entrega dos pacotes, que as chamadas não foram

prejudicadas, conforme afirmado por James Kurose e Keith Ross [Kurose03]: “Para aplicações de áudio altamente interativas, como o telefone por Internet, atrasos fim a fim menores do que 150 milissegundos não são percebidos pelo ouvido humano;”.

Recomenda-se, então, o uso de VPN entre os servidores da matriz e da filial para as chamadas VoIP.

Caso os servidores já possuam uma VPN estabalecida para o tráfego de dados, esta não deverá ser utilizada para as chamadas VoIP. Deverá ser criada uma VPN exclusiva para o tráfego de voz para que possa ser feita uma reserva de largura de banda específica e para que não haja concorrência entre os pacotes de voz e os de dados na fila de cifragem.

4.7 Preparação do ambiente para estudo com o *Roaming User* (terceiro cenário)

O terceiro cenário de testes é composto por uma estação Windows que simula um ramal remoto através da Internet. Como não se tinha disponível um terceiro *link* de Internet, foi utilizado o *link* do servidor da matriz para a realização dos testes.

O servidor da filial serviu como o servidor SIP do ramal remoto. O objetivo dos testes nesse cenário era verificar as dificuldades e implicações da aplicação de uma VPN para o tráfego das chamadas da estação para o servidor. Esses testes abordaram o quesito da confidencialidade da informação para este cenário.

Como não se conseguiu ferramentas gratuitas de controle de banda para a implementação no Windows, não foi possível realizar os testes de priorização para a estação. Conforme testes anteriores, porém, já é sabido que tentativas de ataques DoS, do tipo *Net Flood*, obtêm sucesso, caso a largura de banda de *upload* do atacante seja superior ou equivalente a de *download* do alvo.

Para preparar o ambiente de estudo, foram seguidos os seguintes passos:

- copiar os arquivos do */etc/openvpn* do servidor da matriz para o servidor da filial;
- renomear o arquivo do */etc/openvpn/client1.conf* do servidor da filial para *client1.ovpn*, para que este arquivo não fosse usado na inicialização do serviço;
- editar o arquivo *client1.ovpn* da estação Windows com o *OpenVPN* instalado, para contemplar o IP do servidor da filial. Para isso, foi modificada a diretriz: remote;
- copiar a configuração do ramal13 do servidor da matriz para o servidor da filial, modificando o contexto para “internetcall” e adicionando as linhas para forçar a utilização do *codec* iLBC;
- habilitar a entrada, no *firewall*, dos pacotes com origem na rede 10.0.1.0/24.

As coletas de áudio para o estudo das diferenças de tempo, sem a utilização e com a utilização de VPN, iriam ser realizadas da mesma forma que foram realizados da estação local para o servidor da matriz, porém, descobriu-se que o *softphone* X-lite não possuía o recurso de conferência para que isso fosse possível.

Foi instalado, então, o Eyebeam da X-tem, mostrado na Figura 27, que possuía o recurso de conferência, e descobriu-se que o produto também já possuía recurso de gravação de áudio. Dessa forma não foi necessário realizar a conferência para que a gravação do teste de eco obtivesse sucesso.

Figura 27. *Softphone* Eyebeam.

4.8 Estudos com o *Roaming User*

O Eyebeam foi configurado inicialmente com o IP público do servidor da filial para que a coleta de áudio fosse realizada sem a utilização da VPN. Os dados da configuração do *softphone* foram os mesmos do ramal13 configurados no X-lite. A diferença foi, apenas, o IP público. A pasta de gravação dos arquivos de áudio também foi escolhida na configuração do *softphone*. Após a configuração, foi realizada uma chamada para o ramal 600 (teste de eco) do servidor da filial e foi apertado o botão de gravação para que o áudio fosse gerado.

Após a coleta realizada sem a VPN, foi modificada a configuração do Eyebeam para contemplar o IP da VPN (10.0.1.1). O cliente *OpenVPN* da estação Windows foi iniciado, a VPN foi estabelecida e, só a partir daí, foi realizada a coleta do áudio através do botão de gravação do *softphone*.

Esse *softphone* gera o arquivo no formato *avi* e o SIGVIEW não possui suporte a esse formato. Como não se conseguiu um conversor para formato wav, utilizou-se um recurso que permite gravar o que é reproduzido nas caixas de som. Ou seja, o arquivo foi colocado para tocar e foi salvo, através da utilização da aplicação “Gravador de Som” do Windows. Dessa forma, foram gerados os arquivos no formato wav para análise no SIGVIEW.

A Figura 28 apresenta o sinal de áudio gravado de duas chamadas realizadas respectivamente: a) sem a utilização de VPN; b) com a utilização da VPN. O tempo de resposta do eco está representado nessa figura na parte superior de cada retângulo.

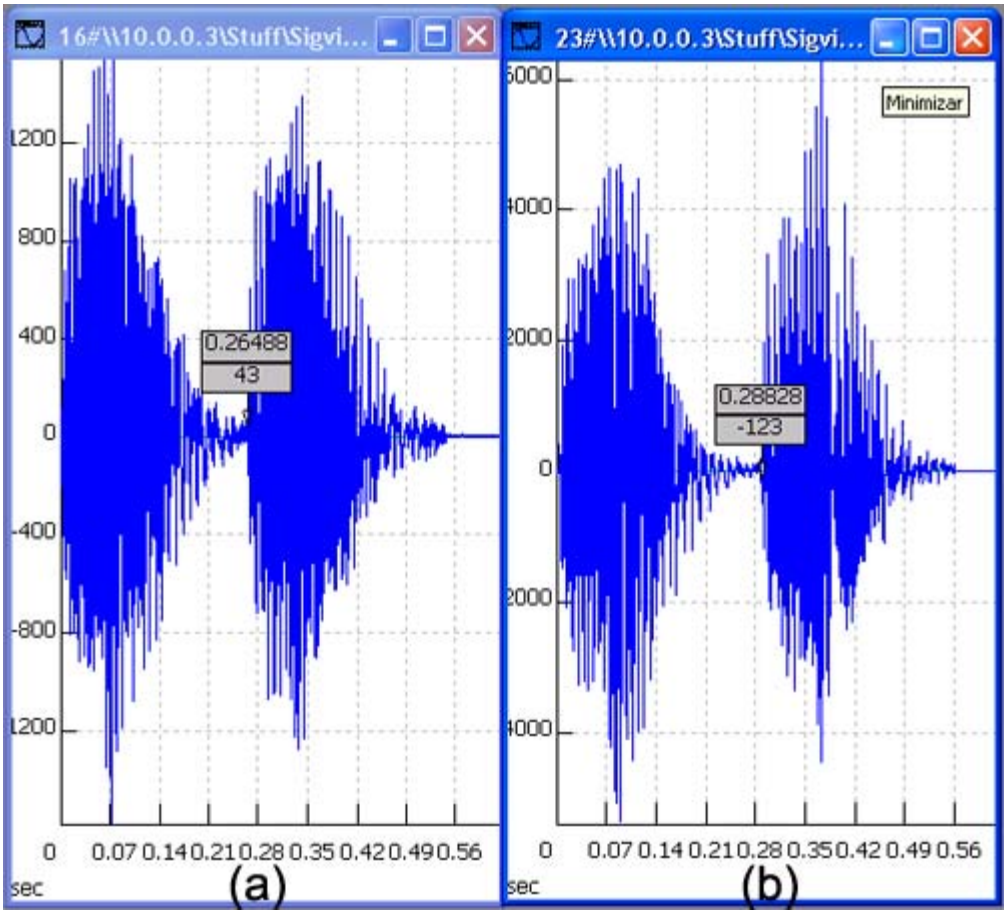

Figura 28. Coletas de áudio do *Roaming User*: a) sem a utilização de VPN; b) com a utilização de VPN.

Foi obtido sucesso na implementação das VPNs e na realização dos testes das chamadas. Através da pequena diferença nos tempos dos diferentes testes de eco e da não percepção de prejuízo na qualidade das chamadas, pôde-se concluir que a implementação de VPN não ocasionou problemas.

4.9 Resultados dos estudos com o *Roaming User*

Como não se conseguiu ferramentas gratuitas de controle de banda para a implementação no Windows, não foi possível realizar os testes de priorização para a estação. Conforme testes anteriores, porém, já é sabido que tentativas de ataques DoS, do tipo *Net Flood*, obtêm sucesso, caso a largura de banda de *upload* do atacante seja superior ou equivalente à de *download* do alvo.

Foi aplicada uma VPN entre a estação com o *softphone* (*Roaming User*) e o servidor, já que a fragilidade quanto ao quesito confidencialidade já era conhecida. A diferença no tempo de entrega dos pacotes, de aproximadamente 12 ms, entre o teste de eco sem a VPN e com a utilização de VPN, demonstra que as chamadas não serão prejudicadas, conforme afirmado por James Kurose e Keith Ross [Kurose03]: “Para aplicações de áudio altamente interativas, como o telefone por Internet, atrasos fim a fim menores do que 150 milissegundos não são percebidos pelo ouvido humano;”.

Recomenda-se, então, o uso de VPN entre os *Roaming Users* e o servidor VoIP.

Capítulo 5

Conclusões e Trabalhos Futuros

5.1 Conclusões

Neste trabalho, inicialmente, foi realizada uma revisão bibliográfica sobre a recomendação H.323 e os protocolos SIP e IAX, além de outros conceitos e tecnologias que seriam importantes para o seu desenvolvimento.

A recomendação H.323 consiste em uma série de orientações a serem seguidas por sistemas de conferência de áudio e vídeo. A sua primeira versão aprovada no ITU-T é datada de fevereiro de 1996 [Colcher05]. É uma recomendação bastante extensa e complexa, verticalmente integrada, que não só engloba a comunicação de áudio entre dois pontos, mas também está muito presente nos antigos equipamentos de telefonia IP. A implementação em código aberto mais completa da recomendação H.323 é a OpenH323 [Openh32306]. Não foram realizados testes com a recomendação H.323 já que os equipamentos disponíveis para o estudo suportavam apenas SIP e porque a inclusão de suporte a SIP é uma tendência dos fabricantes de equipamentos VoIP.

Não se pode afirmar que um protocolo é completamente superior ao outro, porém através da revisão bibliográfica verificou-se que, ao comparar o SIP com a H.323, as funcionalidades de ambas as implementações são semelhantes, mas as implementações SIP são mais simples de serem realizadas e possuem maior escalabilidade [Schulz03].

Por sua vez, as vantagens do protocolo IAX sobre o SIP são:

- a facilidade da sua implementação em redes que utilizam NAT, já que o tráfego de sinalização e transporte de dados é feito através da mesma porta UDP;
- a utilização de *trunking* que permite uma maior economia de largura de banda quando mais de um canal é trafegado entre os mesmos nós.

Porém, deve-se lembrar que o IAX é otimizado para implementações VoIP e, que, por enquanto, ainda não são encontrados com facilidade equipamentos VoIP com suporte a este protocolo. Recomenda-se fortemente o seu uso para interligar servidores Asterisk e também quando os equipamentos VoIP tiverem suporte a esse protocolo.

A biblioteca de implementação do IAX no código fonte do Asterisk já possui referências à possibilidade de cifragem da comunicação (sinalização e mídia), porém isto ainda está em estágio *alpha* de desenvolvimento. Caso se obtenha sucesso nessa implementação, isto representará um grande diferencial se comparado ao SIP e H.323.

Existem alguns protocolos que fazem a cifragem do seu conteúdo, como, por exemplo, o SIPS relacionado ao SIP e o SRTP relacionado ao RTP. Esses protocolos não foram objeto de estudo deste projeto já que não existe implementação desses protocolos para a comunicação no cenário que foi proposto.

Verificou-se que a adoção de VoIP é uma tendência, mas que existem diversos problemas relacionados à segurança da informação, dos quais a maioria dos administradores de segurança em redes ainda não estão cientes.

O Asterisk veio como um marco na telefonia IP, já que provê as funcionalidades de uma grande central telefônica inteligente com suporte a VoIP, além de código aberto e baixo custo.

Para cada um dos três cenários preparados simulando uma implementação real de telefonia IP, verificou-se inicialmente algumas das suas fragilidades em relação à segurança da informação nos seus três aspectos: disponibilidade, integridade e confidencialidade.

Após a confirmação da falta de segurança, foram implementados alguns recursos de segurança em rede previamente estudados e, posteriormente, verificadas as dificuldades e implicações dessas implementações. Diferentemente do que foi apresentado por Thomas Wlash e Richard Kuhm [Walsh05], não foram encontradas grandes dificuldades na implementação de segurança nesses ambientes, principalmente, quanto às aplicações de VPN.

Algumas das falhas de segurança encontradas relativas à disponibilidade e integridade, são inerentes da arquitetura dos protocolos usados na Internet como, por exemplo, a suscetibilidade a ataques do tipo DoS. Caso a implementação fosse realizada com *links* dedicados, a aplicação de recursos para a garantia da qualidade de serviço (QoS) resolveria grande parte desses problemas, porém o estudo em *links* dedicados não foi objeto de estudo deste trabalho, já que a tendência é a migração da comunicação para *links* com a Internet devido ao baixo custo se comparado àquele.

Do resultado dos testes realizados neste trabalho, surgiram recomendações que podem ser seguidas para que as implementações de VoIP, em cenários parecidos com os apresentados aqui, possam ter sucesso em relação a alguns requisitos de segurança. Seguem as recomendações com as devidas explicações:

- colocar os equipamentos exclusivos de VoIP, como, por exemplo, os ATAs, em uma rede separada e aplicar as restrições de acesso físico dos servidores também a esses equipamentos. Essa recomendação visa anular os riscos da tentativa de uso de *sniffers* e também dos ataques DoS a esses equipamentos;
- para as estações que utilizam *softphones*, utilizar duas interfaces de rede. Uma será utilizada para acesso a rede de dados usual e a outra para acesso a uma rede VoIP diferente, utilizada apenas por usuários de *softphone*. Essas estações, apesar de estarem em redes distintas, deverão estabelecer VPNs para o servidor. Se possível, utilizar *switches* que permitam fixar o MAC para cada interface dessa rede VoIP de *softphones*;
- no caso de uso de telefones IP, colocá-los em uma outra rede de dados e, se possível, interligá-los em *switches* que possam fixar o MAC para cada interface;
- no caso de disponibilizar servidores VoIP para uso através da Internet, o *firewall* deverá possuir restrições no limite da largura de banda total para que o enfileiramento dos pacotes não ocorra nem no *modem*, nem no roteador. Esse controle da fila deverá ficar no *firewall* para que os problemas de disponibilidade e integridade sejam minimizados;
- as comunicações das estações com os *softphones*, dos *roaming users* e entre servidores de telefonia deverão ser feitas através do uso de VPNs. Dois tipos de implementação de VPN foram aplicados com sucesso neste estudo, através da

utilização do *OpenVPN* e do SSH. É importante salientar que o poder de processamento da máquina estará diretamente relacionado ao número de canais que poderão ser cifrados sem que haja maiores atrasos;

- caso os pontos já se conectem através de VPN para a comunicação de dados, deverá ser implementada uma VPN exclusiva para a comunicação de voz para que se possa fazer uma reserva de banda e também para que não haja problemas no enfileiramento de dados para a cifragem;
- evitar o uso de *softphones*, principalmente se a segurança for um fator primordial. É importante lembrar que as estações com *softphones* estão também suscetíveis a outros tipos de ameaças como, por exemplo, *worms* e vírus;
- verificar, sempre, a disponibilidade de atualização de *firmwares* relacionados à segurança para os equipamentos VoIP.

Através deste estudo espera-se que algumas implementações de segurança em VoIP nos cenários semelhantes ao estudado sejam de fácil realização. Este estudo possui procedimentos para a realização dos passos necessários, além das recomendações que podem, também, ser seguidas em outros cenários.

5.2 Trabalhos Futuros

Como sugestão para a continuidade deste trabalho está o estudo:

- das dificuldades e implicações da implementação de outros tipos de VPN para a comunicação VoIP. Algumas sugestões de protocolos que podem ser estudados para o estabelecimento das VPNs são o IPSEC e o PPTP;
- da viabilidade da implementação e sua implementação propriamente dita, do uso do SIPS, SRTP e outros protocolos no Asterisk;
- mais aprofundado da arquitetura dos protocolos VoIP e suas fragilidades. Para essa proposta sugere-se a referência [Voipsa06] como fonte de consulta para o início dos estudos;
- sobre segurança em outros cenários como, por exemplo, a implicação, nos aspectos de segurança, da distribuição de um contato através de uma SIP URI. Já que dessa forma o servidor deverá ficar exposto à Internet sem cifragem do seu conteúdo e com maior visibilidade;
- da implementação de cifragem do IAX, que está em estado inicial (*alpha*) de desenvolvimento;
- da segurança dos recursos disponibilizados através do uso da tecnologia VoIP no Asterisk como, por exemplo, o uso de caixa postal de voz, integração com Banco de Dados e outros;
- das implicações das diferentes características da implementação do IPv6 se comparado ao IPv4 nas soluções VoIP;
- da implementação de recursos de *fail-over* e balanceamento de carga no Asterisk.

Bibliografia

[Aguru06] ASTERISK GURU. **Bandwidth Calculator**. Disponível em: <http://www.asteriskguru.com/bandwidth_calculator.php>. Acesso em: 8 abr. 2006.

[Bantel05] BANTEL, B. **A revolução da telefonia IP**. 2005. Disponível em: <http://idgnow.uol.com.br/AdPortalv5/A_revolucao_da_telefonia_IP.htm> Acesso em: 9 dez. 2005.

[Beebe05] BEEBE, N. H. F., ROBBINS, A. **Classic Shell Scripting**. USA: O'Reilly, 2005.

[Bugs06] BUGS DIGIUM. **0006457: Mixmonitor stopped storing**. Disponível em: <http://bugs.digium.com/view.php?id=6457>. Acesso em: 15 fev. 2006.

[Cobb04] COBB, C. **Cryptography For Dummies**. USA: For Dummies, 2004.

[Colcher05] COLCHER, S. et al. **VoIP: Voz Sobre IP**. Rio de Janeiro: Editora Campus, 2005.

[Cunvin05] CUNVIN, A. **The Rise of Security Threats**. Disponível em: <http://www.net-security.org/article.php?id=740>. Acesso em: 29 de out. 2005.

[Davidson00] DAVIDSON, J.; PETERS, J. **Voice Over IP Fundamentals**. USA: Cisco Press, 2000.

[Donaldson02] DONALDSON, M. E. **Inside the Buffer Overflow Attack: Mechanism, Method, & Prevention**. Disponível em: < http://www.sans.org/reading_room/papers/download.php?id=386&c=80c9dec08d4b581d738caceb89082798&portal=8254b2201f0410275ac4cfed5a05ea4c>. Acesso em: 25 jun. 2006.

[Dsniff06] DSNIFF. **Site da ferramenta**. Disponível em: <http://monkey.org/~dugsong/dsniff/>. Acesso em: 18 mar. 2006.

[Ethereal05] ETHEREAL. **Site da ferramenta**. Disponível em: <www.ethereal.com> Acesso em: 9 dez. 2005.

[Federal06] FEDERAL TRADE COMISSION FOR THE CONSUMER. **ChoicePoint Settles Data Security Breach Charges; to Pay $10 Million in Civil Penalties, $5 Million for Consumer Redress**. Disponível em: < http://www.ftc.gov/opa/2006/01/choicepoint.htm>. Acesso em: 18 mar. 2006.

[Fong02] FONG, P.J. et al. **Configuring Cisco Voice Over IP, Second Edition**. USA: Syngress Publishing, 2002.

[Goncalv05] GONÇALVES, F. E. A. **Asterisk PBX: Guia de configuração**. Brasil: Voffice, 2005.

[IEC05] IEC. **SS7 over IP Signaling Transport & SCTP**. Disponível em: <http://www.iec.org/online/tutorials/ss7_over/>. Acesso em: 18 mar. 2006.

[Ihets06] IHETS. **Technology fundamentals**. Disponível em: <http://www.ihets.org/news/fundamentals.html>. Acesso em: 24 jun. 2006.

[Ilbcfree06] ILBCFREEWARE. **Site do projeto**. Disponível em: <http://www.ilbcfreeware.org/>. Acesso em: 9 abr. 2006.

[Iptraf06] IPTRAF. **Site da ferramenta**. Disponível em: <http://freshmeat.net/projects/iptraf/>. Acesso em: 15 fev. 2006.

[Kretchmar03] KRETCHMAR, J. **Open Source Network Administration**. USA: Prentice Hall PTR, 2003.

[Kurose03] KUROSE, J. F.;ROSS,K. W.. **Redes de Computadores e a Internet: uma nova abordagem**. São Paulo: Addison Wesley, 2003.

[Leyden04] LEYDEN, J. **MS UK 0wn3d by hackers. Again**. Disponível em: <http://www.theregister.co.uk/2004/05/25/ms_uk_defaced/>. Acesso em: 18 mar. 2006.

[Lexisnex05] LEXISNEXIS MEDIA RELATIONS. **LexisNexis Concludes Review of Data Search Activity, Identifying Additional Instances of Illegal Data Access**. Disponível em: <http://www.lexisnexis.com/about/releases/0789.asp>. Acesso em: 18 mar. 2006.

[Linuximq06] LINUXIMQ. **Site da ferramenta**. Disponível em: <http://www.linuximq.net/>. Acesso em: 19 fev. 2006.

[Lockhart04] LOCKHART, A. **Network Security Hacks – 100 Industrial-Strength Tips & Tools**. USA: O'Reilly, 2004.

[Marcink03] MARCINKOWSKI, S. J.; STANTON, J. M. **Motivational Aspects of Information Security Policies**. In: IEEE International Conference on Systems, Man and Cybernetics, 2003. **Proceedings**… Washigton, DC, USA: IEEE Press, 2003, Volume 3, p. 2527-2532.

[Mec06] MEC. **Ministério da Educação adota Fedora em suas estações de trabalho**. Disponível em: <http://www.softwarelivre.gov.br/noticias/fedoranomec>. Acesso em: 29 out. 2005.

[Messenger06] MSN Messenger. **Site da ferramenta**. Disponível em: <http://messenger.msn.com>. Acesso em: 19 mar. 2006.

[Negus04] NEGUS, C. **Red Hat Linux Bible – Fedora and Enterprise Edition**. USA: Wiley Publishing, 2004.

[Newham05] NEWHAM, C. **Learning the bash Shell**, 3rd Edition. USA: O'Reilly, 2005.

[Openh32306] OPENH323. **Site da ferramenta**. Disponível em: <http://www.openh323.org/>. Acesso em: 16 abr. 2006.

[Orebaugh04] OREBAUGH, A. **Ethereal Packet Sniffing**. USA: Syngress Publishing, 2004.

[Packeth06] PACKETH. **Site da ferramenta**. Disponível em: <http://packeth.sourceforge.net/>. Acesso em: 27 jan. 2006.

[Pelaez02] PELÁEZ, R. S. **Análisis de seguridad de la familia de protocolos TCP/IP y sus servicios asociados**. 2002.

[Rezende98] REZENDE, P. A. D. **Criptografia e Segurança na Informática**. Brasília: UNB. Copymarket.com, 1998.

[Schulz03] SCHULZRINNE, H., ROSENBERG, J. **A Comparison of SIP and H.323 for Internet Telephony**. USA: Columbia University, 2003.

[Schwartau05] SCHWARTAU, W. **With VoIP, it's déja vu all over again**. Disponível em: < http://www.networkworld.com/columnists/2005/111405schwartau.html?page=2 >. Acesso em: 18 mar. 2006.

[Segrest05] SEGREST, R. **IAX protocol simplifies VoIP**. Network World. Disponível em: <http://www.networkworld.com/news/tech/2005/041105techupdate.html>. Acesso em: 27 jan. 2006.

[Siever00] SIEVER, E. et al. **Linux in a Nutshell**. Third Edition. USA: O'Reilly & Associates, 2000.

[Sigview06] SIGVIEW. **Site da ferramenta**. Disponível em: <http://www.sigview.com/>. Acesso em: 27 jan. 2006.

[Singlet06] SINGLETARY, D. **ADSL Bandwidth Management HOWTO**. Disponível em: <http://www.faqs.org/docs/Linux-HOWTO/ADSL-Bandwidth-Management-HOWTO.html>. Acesso em: 23 jan. 2006.

[Skype06] SKYPE. **Site da ferramenta**. Disponível em: <http://www.skype.com>. Acesso em: 19 mar. 2006.

[Spencer05] SPENCER, M.; MILLER, F.W. **Inter-Asterisk Exchange (IAX) Version 2**. 2005. Disponível em: <http://www.cornfed.com/iax.pdf> Acesso em: 18 out. 2005.

[TCPdump05] TCPDUMP. **Site da ferramenta**. Disponível em: <www.tcpdump.org> Acesso em: 9 dez. 2005.

[Vijayan05] VIJAYAN, J. **Novell Server Hacked**. Disponível em: < http://www.linuxworld.com.au/index.php/id;2128628770;fp;16;fpid;0>. Acesso em: 18 mar. 2006.

[Voipsa06] VOIPSA. **Voice Over IP Security Alliance**. Disponível em: < http://www.voipsa.com/>. Acesso em: 12 fev. 2006.

[Walsh05] WALSH, T.J.; KUHM, D.R. **Challenges in securing voice over IP**. Security & Privacy Magazine, IEEE. Volume 3, Issue 3. 44-49, 2005.

Apêndice A

Script para o controle de tráfego nos servidores

Este apêndice contém o script inicial que foi criado para realizar o controle de tráfego nos servidores.

```
#!/bin/bash
if [ -z $1 ]
then
  echo "Digite a interface."
  exit
fi
if [ "$1" = "status" ]
then
  echo ---$2
  /sbin/tc -s -d qdisc show dev $2
  echo ---IMQ0
  /sbin/tc -s -d qdisc show dev imq0
  exit
fi
if [ "$1" = "stop" ]
then
  /sbin/tc qdisc del dev $2 root
  /sbin/tc qdisc del dev imq0 root
  /sbin/iptables -t mangle -F
  /sbin/iptables -t mangle -X MYSHAPER-OUT
  /sbin/iptables -t mangle -X MYSHAPER-IN
  /sbin/rmmod imq
  /sbin/rmmod ipt_IMQ
  exit
fi
DEV=$1
```

```
DOWNLOAD=1024
DPERCENTUAL=70
UPLOAD=328
UPERCENTUAL=70
VOIP=128

/sbin/tc qdisc add dev $DEV root handle 1:0 htb default 15

/sbin/tc class add dev $DEV parent 1:0 classid 1:1 htb rate
$[$UPERCENTUAL*$UPLOAD/100]kbit

/sbin/tc class add dev $DEV parent 1:1 classid 1:3 htb rate ${VOIP}kbit ceil
$[$UPERCENTUAL*$UPLOAD/100]kbit prio 0
/sbin/tc class add dev $DEV parent 1:1 classid 1:15 htb rate
$[$UPERCENTUAL*$UPLOAD/100-$VOIP]kbit ceil
$[$UPERCENTUAL*$UPLOAD/100]kbit prio 1

/sbin/tc qdisc add dev $DEV parent 1:3 handle 3:0 sfq perturb 10
/sbin/tc qdisc add dev $DEV parent 1:15 handle 15:0 sfq perturb 10

/sbin/tc filter add dev $DEV protocol ip parent 1:0 prio 0 handle 1 fw flowid 1:3

/sbin/iptables -t mangle -N MYSHAPER-OUT
/sbin/iptables -t mangle -I POSTROUTING -o $DEV -j MYSHAPER-OUT

# Inicio das regras de saida
/sbin/iptables -t mangle -I MYSHAPER-OUT -p udp -m udp –sport 4569 -j MARK --set-mark 1
/sbin/iptables -t mangle -I MYSHAPER-OUT -p udp -m udp –dport 4569 -j MARK --set-mark 1
# Fim das regras de saída

/sbin/modprobe imq numdevs=1
/sbin/modprobe ipt_IMQ

/sbin/ip link set imq0 up

/sbin/tc qdisc add dev imq0 handle 1:0 root htb default 15

/sbin/tc class add dev imq0 parent 1:0 classid 1:1 htb rate
$[$DPERCENTUAL*$DOWNLOAD/100]kbit

/sbin/tc class add dev imq0 parent 1:1 classid 1:3 htb rate ${VOIP}kbit ceil
$[$DPERCENTUAL*$DOWNLOAD/100]kbit prio 0
/sbin/tc class add dev imq0 parent 1:1 classid 1:15 htb rate
$[$DPERCENTUAL*$DOWNLOAD/100-$VOIP]kbit ceil
$[$DPERCENTUAL*$DOWNLOAD/100]kbit prio 1

/sbin/tc qdisc add dev imq0 parent 1:3 handle 3:0 sfq perturb 10
/sbin/tc qdisc add dev imq0 parent 1:15 handle 15:0 sfq perturb 10
```

```
/sbin/tc filter add dev imq0 protocol ip parent 1:0 prio 0 handle 2 fw flowid 1:3

/sbin/iptables -t mangle -N MYSHAPER-IN
/sbin/iptables -t mangle -I PREROUTING -i $DEV -j MYSHAPER-IN

# Inicio das regras de entrada
/sbin/iptables -t mangle -I MYSHAPER-IN -p udp -m udp –sport 4569 -j MARK --set-mark 2
/sbin/iptables -t mangle -I MYSHAPER-IN -p udp -m udp –dport 4569 -j MARK --set-mark 2
# Fim das regras de entrada

/sbin/iptables -t mangle -A MYSHAPER-IN -j IMQ
```