Centro Universitário de Brasília – UniCEUB
Faculdade de Ciências Exatas e de Tecnologia – FAET
Curso de Engenharia da Computação
Disciplina: Projeto Final
Professor Orientador: Prof. M.Sc. Antonio José Gonçalves Pinto

Implementação de uma infra-estrutura IPv6 e IPv4, aplicando sobre estes o serviço de VoIP

Trabalho Final de Graduação

Por: Alessandro Catão Mito Kuramoto
R.A. 2001557/0

Brasília/DF, Junho/2006.

Centro Universitário de Brasília – UniCEUB
Faculdade de Ciências Exatas e de Tecnologia – FAET
Curso de Engenharia da Computação
Disciplina: Projeto Final

Por

Alessandro Catão Mito Kuramoto
R.A. 2001557/0

Implementação de uma infra-estrutura IPv6 e IPv4, aplicando sobre estes o serviço de VoIP

Trabalho Final de Graduação

Professor Orientador: Prof. M.Sc. Antonio José Gonçalves Pinto

Brasília/DF, Junho/2006.

ALESSANDRO CATÃO MITO KURAMOTO
RA: 2001557/0

Monografia aprovada como requisito para a obtenção do título de Bacharel em Engenharia da Computação – Centro Universitário de Brasília. Comissão Examinadora formada pelos professores:

Prof. M. Sc. Antonio J. Gonçalves Pinto
Orientador

Prof. Cláudio Penedo
Banca

Prof. Francisco Javier
Banca

Brasília, 28 de junho de 2006.

Agradecimentos

Tenho muito que agradecer aos meus pais que muito me ajudaram no desafio de adquirir novos conhecimentos e a desbravar novos horizontes com a arma do conhecimento. Eles apesar de estarem longe geograficamente, sempre estiveram perto em sentimentos e sempre acompanharam minha trajetória.

Ao meu filho Hugo e minha esposa Silvia, que acompanharam a angústia do desafio de escrever esta monografia e sempre tiveram a paciência em momentos que quiseram estar comigo e que não tive este tempo para estar com eles. Amo muito vocês.

Também aos meus amigos do trabalho e da faculdade que me ajudaram nesta caminhada.

Ao meu professor orientador, Antonio José Gonçalves Pinto, que sempre esteve dando todo o apoio necessário ao desenvolvimento desta monografia, até o último momento através de conselhos, críticas construtivas e dispondo de seu tempo para as nossas reuniões. Foi muito proveitoso e gratificante.

Aos professores Abiezer e Javier que participaram diretamente do meu aprendizado no UniCEUB. O professor Abiezer pela excelência que conduz o curso de Engenharia de Computação e pela determinação em erguer o nome deste curso junto às entidades como, por exemplo, o CREA. O professor Javier por ter sido um dos primeiros professores (1º Semestre) que tive no curso de Engenharia da Computação e pela dedicação com que este professor leva o aprendizado de seus alunos.

Ao professor Luiz Otávio, que deu uma alavancada na minha vida profissional e acadêmica. A nossa participação em projetos de redes abriu horizontes que culminaram na aquisição de novos conhecimentos e que me fez chegar à satisfação profissional. Os pães de queijo pago por ele na padaria, também ajudaram muito.

E principalmente a Deus por fazer que eu adquirisse novos conhecimentos e ter me dado saúde para esta árdua batalha.

Resumo

Dizer que atualmente está ocorrendo uma corrida por novas tecnologias e serviços, pode ser um tanto quanto comum, pois desde o advento dos computadores, a sociedade começou a passar por mudanças muito rápidas, nunca imaginadas antes. Estas mudanças sempre geram a criação de novas tecnologias e serviços. Acredita-se que sempre será assim, pois vários fatores como as demandas de infra-estruturas mais poderosas e robustas exigirão ambientes que as comportem.

Atualmente usa-se o protocolo de rede IPv4 (*Internet Protocol Version 4*), porém ele foi desenvolvido em uma época em que o grande interesse era o de interligar as redes. Foi muito bem desenvolvido e por isso é o protocolo utilizado na tão bem conhecida *Internet*. Já se passando o tempo em que este protocolo foi desenvolvido, as indústrias atuais sentem a necessidade por um novo protocolo. As exigências por mais espaço de endereçamento, o controle e o desígnio de um endereço mais simples na camada IP, melhor suporte à Qualidade de Serviço (QoS), maior segurança e um número crescente dispositivos com acesso a *Internet* têm contribuído para estudos de um novo protocolo, no caso o IPv6 (*Internet Protocol Version 6*).

Este trabalho visa mostrar que a transição do IPv4 para o IPv6 é possível, porém, por certo tempo os dois protocolos terão que conviver juntos, então, este trabalho implementa um infra-estrutura lógica de rede usando os protocolos IPv4, IPv6 e VoIP. A VoIP é uma ferramenta utilizada para testar esta transição.

Palavras-chaves: *IPv4, IPv6, VoIP, Linux, FreeBSD, Windows.*

Abstract

It is said that a research for new technologies and services, can be common nowadays, due to the advent of the computers the society has passed through many quick changes. Changes never before imagined and that have generate the creation of new technologies and services, It believe that it will always be like this, because several factors such as stronger and more powerful infrastructures will require environments that support them.

At present, the net protocol used is the IPv4 (Internet Protocol Version 4); however, this protocol was developed in days that the big interest was only to interconnect networks, it was so well developed that it is protocol used at the Internet. With the years that have passed since this protocol was developed, the industries now feel the need for a new protocol. The demands for more space for address, the control and the design of a simpler address in the layer IP, better support to QoS, higher security, and a growing number of devices with access to Internet has contributed for studies of a new protocol, in this case, the IPv6 (Internet Protocol Version 6).

This project is going to show that the transition from the IPv4 for the IPv6 is possible, however, for a certain time, the two protocols will have to live together, so this project implements a network logic infrastructure using IPv4, IPv6 and VoIP. The VoIP will be the tool used to test this transition.

Key-words: *IPv4, IPv6, VoIP, Linux, FreeBSD, Windows.*

SUMÁRIO

Lista de Abreviaturas

ARPA	*Advanced Research Projects Agency*
BSD	*Berkeley Software Distribution*
CREA	Conselho Regional de Engenharia, Arquitetura e Agronomia
DARPA	*Defense Advanced Research Projects Agency* (Agência de Pesquisa de Projetos Avançados de Defesa)
DHCP	*Dynamic Host Configuration Protocol*
DNS	*Domain Name System* (Sistema de Nomes de Domínio)
DOD	*Department Of Defense* (Departamento de Defesa)
EUI	*Extended Unique Interface*
FEC	*Forward Error Correction*
GNU	É um acrônimo recursivo para "GNU Não é *UNIX*" ou do inglês "*GNU is Not Unix*" a pronúncia é "guh-noo"
HTTP	*HyperText Transfer Protocol* (Protocolo de Transferência de Hipertexto)
IEEE	*Institute of Electrical and Electronic Engineers* (Instituto de Engenheiros Eletricistas e Eletrônicos)
IETF	*Internet Engineering Task Force*
IP	*Internet Protocol*
IPng	*IP next generation*
IPv4	*Internet Protocol Version 4*
IPv6	*Internet Protocol Version 6*
ISP	*Internet Service Provider* (Provedor de Acesso *Internet*)
ITU	*International Telecommunications Union*
ITU-T	*ITU Telecom Standardization Sector*
MAC	*Medium Access Control*
MOS	*Mean Opinion Score*
NLA	*Next Level Aggregator*
PABX	*Private Automatic Branch Exchange* (central de comunicação telefônica automática, de uso privado)
PCM	*Pulse Code Modulation* (Modulação por Codificação de Pulsos),
QoS	*Quality of Service* (Qualidade de Serviço)

RFC	*Request For Comment*
RPM	*Red Hat Package Manager* (Apesar do nome *Red Hat*, várias distribuições *GNU/Linux* utilizam os pacotes RPM para a instalação de programas, bibliotecas, componentes, etc)
RTCC	Rede Telefônica Comutada por Circuito
SLA	*Site-Level Aggregation*
TCP	*Transmission Control Protocol*
TCP/IP	*Transmission Control Protocol/Internet Protocol*
TLA	*Top Level Aggregator*
TOS	*Type Of Service* (Tipo de Serviço)
TTL	*Time To Live* (Tempo de Vida)
UDP	*User Datagram Protocol*
UniCEUB	Centro Universitário de Brasília
USB	*Universal Serial Bus*
USC	*University of Southern California*
VOIP	*Voice Over IP*
VPN	*Virtual Private Network*
WAN	Wide Area Network

Lista de Figuras

Lista de Tabelas

Capítulo 1 – Introdução

1.1 – MOTIVAÇÃO

O IPv6 (*Internet Protocol Version 6*) será importante, não neste ou no próximo ano, mas nos próximos cinco ou sete anos. Agora, o que norteia o desenvolvimento do IPv6 não é somente a possibilidade de mais endereços, mas também o fato do melhor gerenciamento de endereços, maior qualidade dos serviços e maior segurança. A estrutura lógica de rede que várias empresas utilizam atualmente é sobre o IPv4 (*Internet Protocol Version 4*) e em cima deste, vários serviços são empregados, entre eles a VoIP (*Voice over IP*) que é importante na convergência das tecnologias de telefonia para as redes IP.

A motivação da elaboração deste trabalho é apresentar as novas funcionalidades do IPv6, tendo em vista que logo o IPv4 terá que ser substituído devido ao fato de algumas de suas limitações e crescimento da *Internet*, então neste trabalho o desafio será implementar o ambiente, onde o IPv4 e o IPv6 possam funcionar juntos.

1.2 – OBJETIVOS DO TRABALHO

Este trabalho visa mostrar uma rede híbrida IPv4 e IPv6 onde aplicações como VoIP possam coexistir, pois estão sendo pensados mecanismos para a transição de IPv4 para IPv6. Assim, no trabalho será implementado o mecanismo de endereçamento IPv4 e IPv6 chamado de *Dual Stack*, o qual se atribui endereço IPv4 e IPv6 ao mesmo tempo para um *host*. Adicionalmente, a utilização da VoIP (*Voice over IP*) mostrará que esta infra-estrutura funciona de acordo com alguns critérios e necessidades, isto quer dizer, que certos fatores como: *software*, ambiente, etc. são necessários, isto é, devem ser empregados no ambiente *softwares* e *hardware*s que suportem o protocolo IPv4 e IPv6. Não é objetivo deste trabalho apresentar a estrutura de telefonia, porém para o bom entendimento do trabalho será referenciado algumas vezes no texto.

Com este trabalho a idéia sobre o IPv6 ficará mais clara e deixará um caminho para implementações de ambientes de testes em redes corporativas, o que contribuirá para migração para o IPv6.

1.3 – ESTRUTURA DO TRABALHO

O trabalho está organizado em cinco capítulos. Os dois primeiros capítulos fazem à apresentação do tema de pesquisa e fornecem o embasamento teórico do trabalho, nos dois capítulos seguintes são abordados a infra-estrutura proposta e os resultados alcançados, por último, a conclusão do trabalho. A organização detalhada é descrita a seguir:

Capítulo 1 – Apresenta o trabalho, com apresentação do tema e a estrutura do mesmo;

Capítulo 2 – Apresenta uma revisão bibliográfica onde são abordados os conceitos essenciais no entendimento do trabalho;

Capítulo 3 – Apresenta a descrição da infra-estrutura lógica proposta junto com as ferramentas utilizadas no trabalho.

Capítulo 4 – Explica detalhadamente como o trabalho foi implementado, as configurações realizadas, as características técnicas de cada ferramenta e os resultados obtidos.

Capítulo 5 – Apresenta a conclusão do trabalho juntamente com os benefícios alcançados e as sugestões de trabalhos futuros.

Capítulo 2 – Tecnologias IPv4, IPv6 e VoIP

2.1 – PROTOCOLO IP

O IP (*Internet Protocol*) é um acrônimo para a expressão inglesa "*Internet Protocol*", ou Protocolo da Internet. Este protocolo pertence à camada *Internet* da arquitetura TCP/IP (*Transmission Control Protocol/Internet Protocol*), que será explicado na seção "**2.2.1.3 – Camada Internet (IP)**". O IP é o protocolo que define o mecanismo de transmissão sem conexão, melhor esforço (*best-effort*) e não-confiável [COMER, 1998].

O serviço é conhecido como sem conexão porque cada pacote é independente dos outros, por exemplo, uma seqüência de pacotes enviados de um computador a outro pode percorrer caminhos diferentes [COMER, 1998]. O serviço é conhecido como não-confiável porque a entrega não é garantida, ou seja, o pacote pode ser perdido, reproduzido, atrasar-se ou ser entregue com problemas, porém o serviço não detectará tais condições, nem informará isso ao transmissor nem ao receptor [COMER, 1998]. A não-confiabilidade pode surgir quando os recursos esgotam-se ou as redes básicas falham [COMER, 1998]. E finalmente, o serviço utiliza uma transmissão *best-effort*, porque faz uma séria tentativa para entregar os pacotes, isso significa que a interligação em redes não rejeita pacotes por simples capricho.

O surgimento do IP veio com a necessidade de interligar as várias instituições do governo dos Estados Unidos e instituições de ensino e pesquisa na época da Guerra Fria. Em 1957 os russos colocaram em órbita o primeiro satélite artificial, ganhando assim uma corrida espacial contra os Estados Unidos [SMETANA, 2004]. Como resposta, em 1958, o Departamento de Defesa dos Estados Unidos (*Department of Defense – DoD*) decidiu criar a (*Defense*) *Advanced Research Projects Agency ((D)ARPA* – Agência de Pesquisa de Projetos Avançados de Defesa) [SMETANA, 2004]. A DARPA tinha como missão garantir que os Estados Unidos sempre estivessem na dianteira tecnológica militar e antecipar quais seriam os avanços tecnológicos dos "adversários". A DARPA mudou o nome algumas vezes para ARPA e novamente, algumas vezes, para DARPA [SMETANA, 2004].

A especificação do IPv4 foi publicada em setembro de 1981, sob o RFC 0791, com o auxílio do *Information Sciences Institute – University of Southern California* (Instituto de Ciências da Informação da Universidade do Sul da Califórnia) [SMETANA, 2004]. Em 1982 o TCP, *Transmission Control Protocol*, e o IP, *Internet Protocol*, foram adotados como os protocolos oficiais da ARPANET. Informações mais detalhadas sobre TCP podem ser obtidas em [COMER, 1998]. A popularização do IP veio quando ele passou a ser distribuído pelo *Berkeley Software Distribution UNIX* (*BSD UNIX*), versão 4.2c, em 1983 [SMETANA, 2004]. Chama-se de arquitetura TCP/IP o conjunto de protocolos que utilizam o TCP e o IP para estabelecer a comunicação entre redes [SMETANA, 2004].

2.2 – ARQUITETURA TCP/IP (*TRANSMISSION CONTROL PROTOCOL / INTERNET PROTOCOL*)

A arquitetura TCP/IP baseia-se principalmente em um serviço de transporte orientado à conexão, fornecido pelo TCP e em um serviço de rede não-orientado à conexão fornecido pelo IP [COMER, 1998]. A arquitetura *Internet* TCP/IP dá ênfase toda especial à interligação de diferentes tecnologias de redes [COMER, 1998] sendo formada por quatro camadas conceituais apresentadas em "**2.2.1 – Camadas TCP/IP**".

A Figura 2.1 mostra as camadas do modelo da arquitetura TCP/IP.

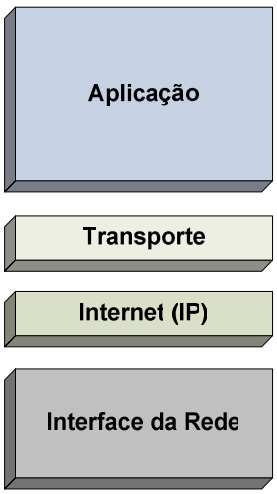

Figura 2.1 – Modelo da arquitetura TCP/IP. [COMER, 1998]

2.2.1 – Camadas TCP/IP

O modelo da arquitetura TCP/IP é organizado em quatro camadas descritas nas próximas seções.

2.2.1.1 – Camada de Aplicação

Na camada de Aplicação, os usuários rodam programas aplicativos que acessam serviços disponíveis através de uma interligação em redes TCP/IP. Um aplicativo interage com um dos protocolos do nível de transporte para enviar ou receber dados. Cada programa aplicativo escolhe o estilo de transporte necessário, que tanto pode ser uma seqüência de mensagens individuais ou um *Stream* contínuo de bytes. O programa aplicativo passa para o nível de transporte, os dados na forma adequada, para que possam, então, ser transmitidos. [COMER, 1998]

2.2.1.2 – Camada de Transporte

A camada de Transporte é fim-a-fim, o que significa dizer que uma entidade desta camada só se comunica com a sua entidade-par do *host* destinatário. É nesta camada que se faz o controle da conversação entre as aplicações intercomunicadas da rede. Dois protocolos aqui são usados: o TCP e o UDP. O TCP é orientado à conexão que fornece serviço confiável de transferência de dados fim-a-fim e o UDP fornece um serviço de transmissão sem conexão e não-confiável. O acesso das aplicações à camada de transporte é feito através de portas que recebem um número inteiro para cada tipo de aplicação. [COMER, 1998]

2.2.1.3 – Camada Internet (IP)

O IP é o protocolo que define o mecanismo de transmissão sem conexão e não-confiável [COMER, 1998]. Esta camada é responsável pelo endereçamento, roteamento e controle de envio e recepção dos pacotes [SANTOS, 2002].

O IP é o protocolo da camada *Internet*. Ele é encarregado da entrega de pacotes para todos os outros protocolos da família TCP/IP. O endereço IP é formado por um conjunto de 32 bits, explicados em "**2.3.2 –**

Endereçamento no IPv4", na atual versão 4 e 128 bits na versão 6, explicados em "**2.4.3 – Endereçamento no IPv6**" .

2.2.1.4 – Camada Interface da Rede

A camada da Interface de Rede é responsável pela aceitação de datagramas IP e por sua transmissão através de uma rede específica [COMER, 1998]. A Interface de Rede também é conhecida como camada de abstração de *hardware* e tem como principal função a interface do modelo TCP/IP com os diversos tipos de redes como, por exemplo, X.25 [COMER, 1998], ATM [SOARES et al, 1995], FDDI [SOARES et al, 1995], *Ethernet* [COMER, 1998], *Token Ring* [COMER, 1998] e *Frame Relay* [SOARES et al, 1995]. [SAKURAY, 2005] Por causa da grande variedade de tecnologias de rede, ela não é normatizada pelo modelo, o que provê a possibilidade de interconexão e interoperabilidade de redes heterogêneas [SANTOS, 2002].

A arquitetura *Internet* TCP/IP não faz nenhuma restrição às redes que são interligadas para formar a inter-rede. Portanto, qualquer tipo de rede pode ser ligada, bastando para isso que seja desenvolvida uma interface que compatibilize a tecnologia específica da rede com o protocolo IP. Essa compatibilização é a função da camada de Rede, que recebe os datagramas IP da camada *Internet* e os transmite através de uma rede específica. Para realizar essa tarefa, nesse nível, os endereços lógicos são traduzidos para os endereços físicos dos *hosts* ou *gateways* conectados à rede [SOARES et al, 1995].

2.3 – INTERNET PROTOCOL VERSÃO 4 (IPV4)

Os dados numa rede IP são enviados em blocos referidos como pacotes, ou datagramas (os termos são basicamente sinônimos no IP). Em particular, no IP nenhuma definição é necessária antes do *host* tentar enviar pacotes para um *host* com o qual não comunicou previamente.

O IP oferece um serviço de datagramas não confiável, melhor esforço (*best-effort*) e sem garantia, o que significa dizer que os datagramas podem chegar desordenados, ou podem chegar duplicados ou podem ser perdidos por inteiro. [SOARES et al, 1995]

Quando nos referimos a IP, estamos nos referindo a versão designada de IP versão 4, ou usando seu acrônimo: IPv4. São números com 32 bits, normalmente escritos como quatro octetos (em decimal), por exemplo: 10.7.3.1. [SOARES et al, 1995]

Na seção "**2.3.1 – Cabeçalho e Especificação do IPv4**" é explicado o formato do cabeçalho e especificação do IPv4.

2.3.1 – Cabeçalho e Especificação do IPv4

A Figura 2.2 ilustra o cabeçalho do datagrama do IPv4.

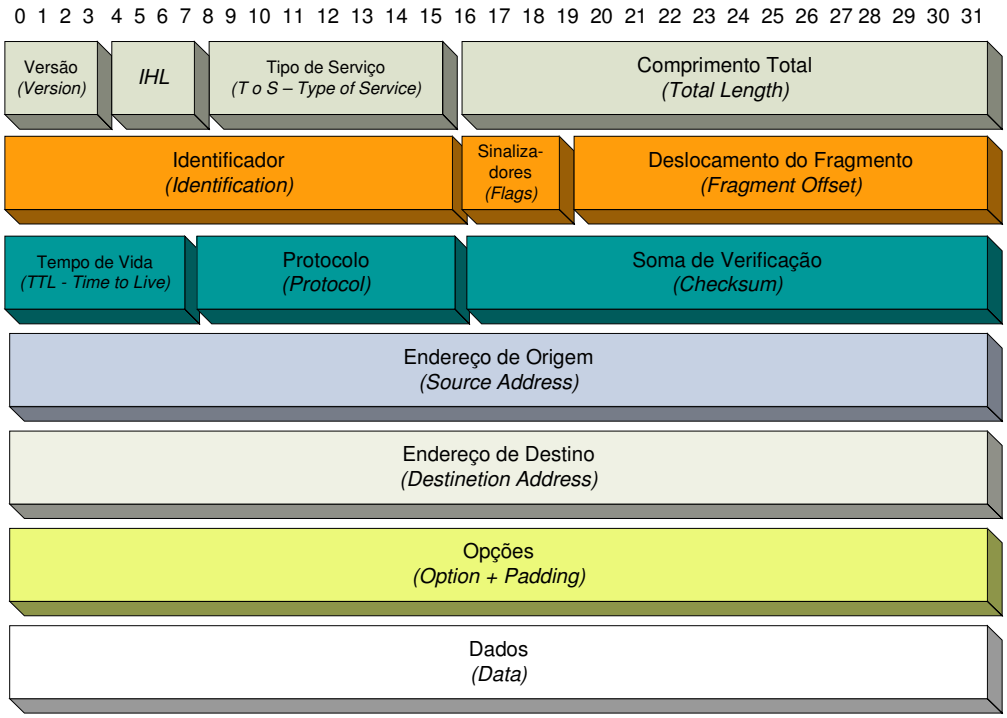

Figura 2.2 – Cabeçalho do datagrama IPv4 [RFC 0791, 1981]

O tamanho e a funcionalidade de cada campo podem ser vista nas próximas seções.

2.3.1.1 – Versão (*Version*)

O primeiro campo do cabeçalho (*Header*) do datagrama IPv4 é o campo da versão (*version*). A versão atual é a 4, motivo pelo qual chamamos de

IPv4 o protocolo IP. O campo versão tem o tamanho de quatro bits. [RFC 0791, 1981]

2.3.1.2 – IHL (Comprimento do Cabeçalho – *Internet Header Length*)

O comprimento de cabeçalho fornece o tamanho do cabeçalho do datagrama medido em palavras de 32 bits. O tamanho mínimo do cabeçalho é de 5 palavras de 32 bits. Todos os campos do cabeçalho contêm um comprimento fixo, exceto para "Opções" e os campos correspondentes "*Padding*". O campo comprimento do cabeçalho tem o tamanho de quatro bits. [RFC 0791, 1981]

2.3.1.3 – ToS (Tipo de Serviço – *Type of Service*)

O campo ToS é utilizado para indicar a QoS (Qualidade de Serviço – *Quality of Service*) [SOUZA, 2005] desejado. Seus bits caracterizam os serviços escolhidos para serem considerados pelos *gateways* para processar o pacote, como por exemplo, a precedência de um pacote. Um roteador (pode ser chamado de *gateway*) pode em situações de grande congestionamento, por exemplo, aceitar somente pacotes com um certo nível mínimo de precedência. Geralmente, deseja-se baixo atraso, alta confiabilidade e alta vazão (*throughput*). O campo ToS tem o tamanho de oito bits. A Figura 2.3 mostra os campos que compõe o campo ToS e a Tabela 2.1 mostra o significado de cada campo. [RFC 0791, 1981]

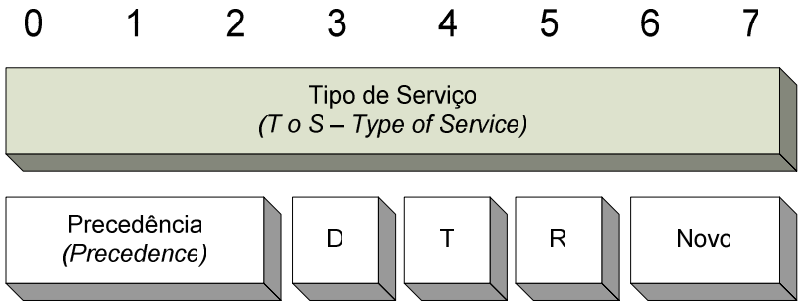

Figura 2.3 – Os cinco subcampos que compõem o campo Tipo de Serviço (Type of Service), de oito bits [RFC 0791, 1981]

Tabela 2.1 – Significado do Campo ToS [RFC 0791, 1981]

Bits	Descrição	Valores	
0 1 2	Precedência (*Precedence*)	000: Rotina (*Routine*) 001: Prioridade (*Priority*) 010: Imediato (Immediate) 011: "Relâmpago" (*"Flash"*)	100: "Relâmpago" Precedente (*Flash Override*) 101: Crítico (*Critic/ECP*) 110: Controle entre Redes (*Internetwork Control*) 111: Controle de Rede (*Network Control*)
3	D (Atraso – *Delay*)	0: Atraso normal. 1: Atraso baixo.	
4	T (Vazão – *Throughput*)	0: Vazão normal. 1: Alta vazão.	
5	R (Confiabilidade – *Relibility*)	0: Confiabilidade normal. 1: Alta confiabilidade.	
6 7	Reservados	Obrigatoriamente 00.	

O nível de precedência é crescente. [RFC 0791, 1981]

2.3.1.4 – Comprimento Total (*Total Length*)

O campo comprimento total informa o comprimento do datagrama, em octetos (bytes). O tamanho máximo do datagrama pode ser 65.535 octetos (64 KB). Esse tamanho de octeto é impraticável para a maior parte de *hosts* e redes. Todos os *hosts* devem ser capazes de no mínimo aceitar datagramas de até 576 octetos, fragmentados ou não. Esse número foi determinado partindo-se do pressuposto que 512 octetos seriam um número razoável de dados a ser enviado, considerando-se mais 64 bytes de cabeçalho, sendo que o tamanho máximo do cabeçalho *Internet* é de 60 octetos, mas o tamanho típico é de 20 octetos, dando-se margem para cabeçalhos de outras camadas. Recomenda-se que os *hosts* só enviem datagramas maiores que 576 bytes se houver a certeza que o endereço destino aceita receber a quantidade de dados enviados. O campo comprimento total tem o tamanho de 16 bits. [RFC 0791, 1981]

2.3.1.5 – Identificação (*Identification*)

O campo identificação contém um número inteiro único para a identificação do datagrama. Serve para permitir que o destino remonte os datagramas. O campo identificação tem o tamanho de 16 bits. [RFC 0791, 1981]

2.3.1.6 – Sinalizadores (*Flags*)

O campo referente aos sinalizadores contém os bits que identificam a transmissão de sinais de controle. Este campo tem o tamanho de três bits [RFC 0791, 1981]. A Figura 2.4 e a Tabela 2.2 mostram em detalhes o campo Sinalizador.

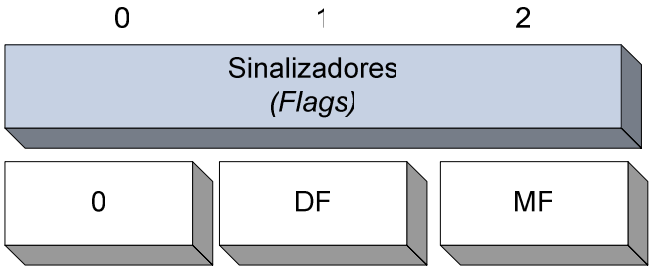

Figura 2.4 – Subdivisão do campo Sinalizador (*Flags*) [RFC 0791, 1981]

Tabela 2.2 – Significado dos bits do campo Sinalizador (*Flags*) [RFC 0791, 1981]

Bit	Descrição	Valores
0	Reservado	Obrigatoriamente 0.
1	DF (Não Fragmente – *Don't Fragment*)	0: Esse datagrama pode ser fragmentado. 1: Esse datagrama não pode ser fragmentado.
2	MF (Mais Fragmentos – *More Fragments*)	0: Esse datagrama é o último fragmento. 1: Há mais fragmentos.

2.3.1.7 – Deslocamento do Fragmento (*Fragment Offset*)

O campo deslocamento do fragmento indica a posição desse fragmento em relação ao datagrama original. O valor desse campo é expresso em unidades de oito octetos (64 bits), portanto o tamanho mínimo do campo de dados de um fragmento é de 64 bits. O primeiro fragmento tem valor 0 nesse campo. O campo deslocamento do fragmento tem o tamanho de 13 bits. [RFC 0791, 1981]

2.3.1.8 – TTL (Tempo de Vida – *Time to Live*)

O campo tempo de vida indica o tempo máximo que o datagrama pode permanecer na rede. Se o valor nesse campo for 0, o datagrama deve ser destruído. A intenção desse campo é limitar o tempo máximo de vida dos datagramas, evitando, por exemplo, que os mesmos sejam encaminhados em círculos. Inicialmente, a unidade do TTL era em segundos, mas como cada unidade processadora de datagramas (roteadores, *switches* de camada 3, etc.) deve diminuir o TTL de uma unidade e o tempo de processamento de pacotes é muito inferior a 1 segundo, o TTL passa a ser somente um limite superior da existência de cada datagrama. O campo tempo de vida tem tamanho de oito bits. [RFC 0791, 1981]

2.3.1.9 – Protocolo (*Protocol*)

O campo protocolo indica o protocolo da camada superior que está utilizando os serviços da camada IP. Esses valores estão definidos no RFC 0790 – *Assigned Network Numbers* (Números de Redes Designados) de 1981. Esse RFC foi substituído pelo RFC 1700 – *Assigned Numbers*. O número do TCP, por exemplo, é 6 (seis). Quando o IP estiver encapsulado em outra camada IP, como em uma Rede Privada Virtual – VPN (*Virtual Private Network*) [CHIN, 1998], por exemplo, o valor desse campo é quatro. O campo protocolo tem o tamanho de oito bits. [RFC 0791, 1981]

2.3.1.10 – Verificação (*Checksum*):

Um pacote em trânsito é alterado por cada roteador que atravesse, um desses roteadores pode comprometer o pacote, assim, o campo verificação tem a finalidade de fazer uma simples detecção da consistência do cabeçalho. O campo verificação tem o tamanho de 16 bits. [RFC 0791, 1981]

2.3.1.11 – Endereço de Origem (*Source Address*)

O campo endereço de origem informa o endereço de 32 bits do transmissor, embora o datagrama possa ser roteado através de muitos roteadores

intermediários, o campo da origem nunca muda. O campo endereço de origem tem o tamanho de 32 bits. [RFC 0791, 1981]

2.3.1.12 – Endereço de Destino (*Destination Address*)

O campo endereço de destino informa o endereço de destino. Essa informação é utilizada pelos roteadores para o encaminhamento (roteamento) do datagrama. Alguns equipamentos podem utilizar os campos IP de origem, de destino e até mesmo informações de protocolos de níveis superiores e o tipo de dado sendo transmitido para realizar o roteamento de pacotes e juntamente realizar algum tipo de priorização ou QoS. O campo endereço de destino tem o tamanho de 32 bits. [RFC 0791, 1981]

2.3.1.13 – Opções (*Options*)

O campo opções é opcional, podendo ou não ser transmitido. No entanto, todos os roteadores devem implementar meios de codificação/decodificação desse campo. Pode haver mais de uma opção nesse campo. As opções servem, entre outras coisas informar se o próprio campo *Option* deve ou não ser copiado para os fragmentos, caso o pacote venha a ser fragmentado, para embutir um *timestamp* da rede, adicionar informações relativas ao nível de segurança do pacote (confidencialidade) ou para especificar uma rota para um determinado destino. O campo opções tem o tamanho variável, entre 0 e 320 bits (40 octetos). [RFC 0791, 1981]

2.3.1.14 – Enchimento (*Padding*)

O campo *Padding* serve apenas para que o cabeçalho IP tenha um tamanho múltiplo de 32 bits. Só se faz o enchimento se o tamanho do campo opção não for múltiplo de 32 bits e este enchimento é feito obrigatoriamente com “0” (zero). O campo enchimento tem o tamanho variável, entre 0 e 31 bits. [RFC 0791, 1981]

2.3.2 – Endereçamento no IPv4

Os endereços IPv4 são números de 32 bits e são comumentes escritos como quatro octetos (em decimal), por exemplo: 10.7.3.1. No IPv6 o tamanho mudou e a forma de apresentação também.

- A identificação de rede (também conhecida como endereço de rede) identifica os sistemas que estão localizados no mesmo segmento físico de rede na abrangência de roteadores IP. Todos os sistemas na mesma rede física devem ter a mesma identificação de rede. A identificação de rede deve ser única na rede.

- A identificação de *host* (também conhecido como endereço de *host*) identifica uma estação de trabalho, servidor, roteador, ou outro *host* TCP/IP dentro de uma rede. O endereço para cada *host* deve ser único para a identificação de rede.

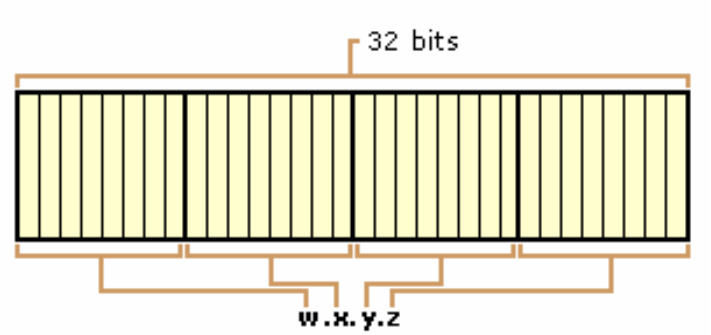

Figura 2.5 – Endereço IPv4. Exemplo: 10.7.3.1 [RFC 0820, 1983]

Originalmente, foram definidas três classes de endereços e posteriormente mais duas classes para atender as necessidades de novas tecnologias. As redes são identificadas pelo valor dos primeiros bits, após os bits de identificação de redes seguem os bits de identificação de *hosts*.

As cinco classes são:

- Classe A: 0.0.0.0 a 127.255.255.255

- Classe B: 128.0.0.0 a 191.255.255.255

- Classe C: 192.0.0.0 a 223.255.255.255
- Classe D: 224.0.0.0 a 239.255.255.255
- Classe E: 240.0.0.0 a 255.255.255.255

Os tópicos a seguir fazem uma breve descrição sobre as cinco classes.

2.3.2.1 – Classe A: 0.0.0.0 a 127.255.255.255

A classe A destina-se as organizações que possuem redes com número muito grande de *hosts*.

O bit de maior grau em uma classe A é sempre zero. Os próximos sete bits (preenchendo o primeiro octeto) completam a identificação de rede. Os 24 bits restantes (os últimos três octetos) representam a identificação do *host*. Um endereço classe A permite 126 redes e 16.777.214 *hosts* por rede. A Figura 2.6 ilustra a estrutura dos endereços classe A.

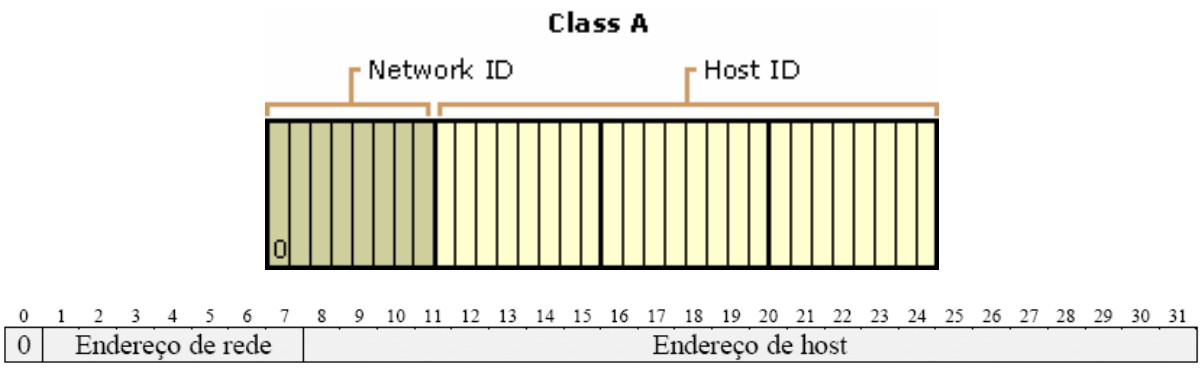

Figura 2.6 – Endereço IPv4 Classe A [RFC 0820, 1983].

2.3.2.2 – Classe B: 128.0.0.0 a 191.255.255.255

A classe B destina-se as organizações de tamanho médio, com número relativamente grande de *hosts*.

Os dois bits de maior grau em uma classe B são sempre os valores binários 10. Os próximos 14 bits (preenchendo primeiro e o segundo octeto) completam a identificação de rede. Os 16 bits restantes (os últimos dois octetos)

representam a identificação do *host*. Um endereço classe B permite 16.384 redes e 65.534 *hosts* por rede. A Figura 2.7 ilustra a estrutura dos endereços classe B

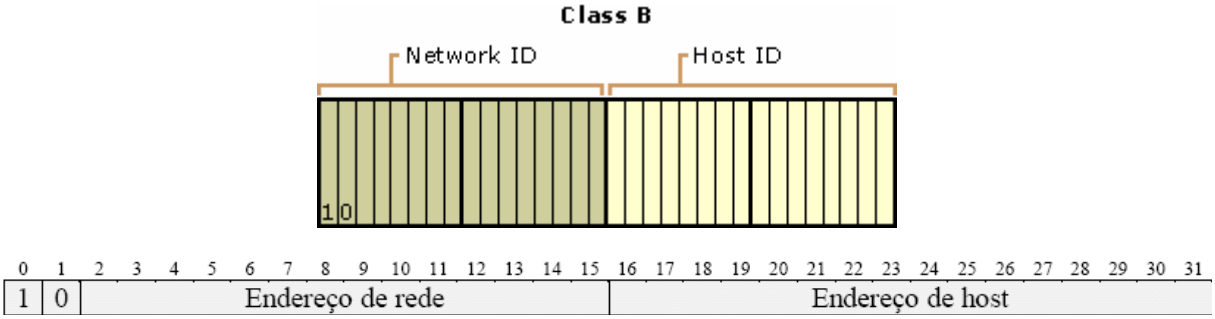

Figura 2.7 – Endereço IPv4 Classe B [RFC 0820, 1983]

2.3.2.3 – Classe C: 192.0.0.0 a 223.255.255.255

A classe C destina-se as organizações pequenas, com número pequeno de *hosts*.

Os três bits de maior grau em uma classe C são sempre os valores binários 110. Os próximos 21 bits (preenchendo os três primeiros octetos) completam a identificação de rede. Os oito bits restantes (o último octeto) representam a identificação do *host*. Um endereço classe C permite 2.097.152 redes e 254 *hosts* por rede. A Figura 2.8 ilustra a estrutura dos endereços classe C.

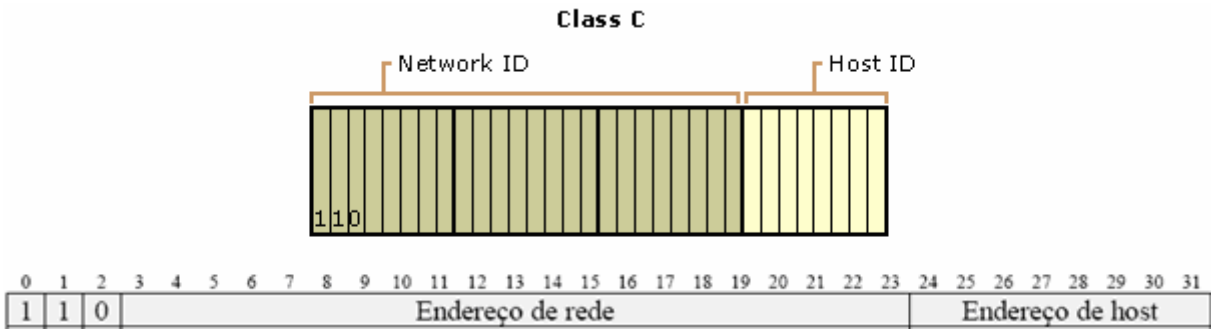

Figura 2.8 – Endereço IPv4 Classe C [RFC 0820, 1983]

2.3.2.4 – Classe D: 224.0.0.0 a 239.255.255.255

Endereços classe D são reservados para endereçamento IP de *Multicast*.

Os quatro bits de maior grau em uma classe D são sempre os valores binários 1110. Os bits restantes são utilizados para endereçamento dos *hosts* reconhecidos como interessados. A Figura 2.9 mostra os quatro bits de maior grau da classe D.

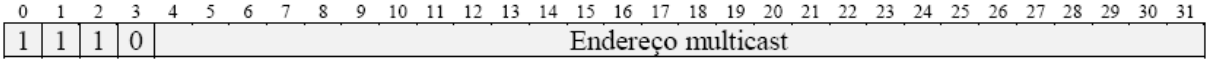

Figura 2.9 – Endereço IPv4 Classe D [RFC 1365, 1992]

2.3.2.5 – Classe E: 240.0.0.0 a 255.255.255.255

Classe E é um endereçamento experimental que está reservado para uso futuro. Os quatro bits de maior grau em uma classe E são sempre 1111 como mostra a Figura 2.10.

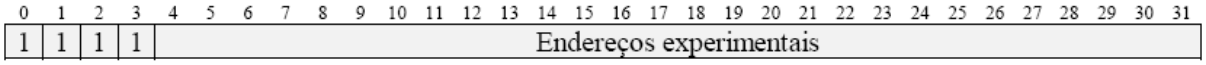

Figura 2.10 – Endereço IPv4 Classe E [RFC 1365, 1992]

A Tabela 2.3 mostra os endereços IPv4 reservados e as faixas de endereços utilizáveis.

Tabela 2.3 – Endereços IPv4 reservados e as faixas utilizáveis. [RFC 1918, 1996]

Classe	Faixa de endereços	Utilização
A	0.0.0.0 a 0.255.255.255	Não utilizável.
A	10.0.0.0 a 10.255.255.255	Endereço de rede reservado para uso em redes privadas.
A	127.0.0.0 a 127.255.255.255	Não utilizável. *Loopback* para teste de interfaces.
A	Demais faixas de endereços	Utilizáveis comercialmente.
B	172.16.0.0 a 172.31.255.255	Endereço de rede reservado para uso em redes privadas.
B	Demais faixas de endereços	Utilizáveis comercialmente.
C	192.168.0.0 a 192.168.255.255	Endereço de rede reservado para uso em redes privadas.
C	Demais faixas de endereços	Utilizáveis comercialmente.

2.4 – INTERNET PROTOCOL VERSÃO 6 (IPV6)

O IPv6 é a versão 6 do protocolo IP e também é conhecido como IPng (*IP next generation*). O IPv6 tem como objetivo substituir o padrão anterior, o IPv4.

Devido à principal proposta do IPv6, que é fornecer mais endereços para a *Internet*, o mesmo foi elaborado inicialmente para usar 160 bits em sua composição, logo sendo alterado para 128 bits, devido a uma convenção adotada entre IETF e o IEEE, o EUI-64 (*Extended Unique Interface*). O EUI-64 altera o endereço MAC [SOARES et al, 1995] dos novos dispositivos de rede fabricados de 48 bits para 64 bits, permitindo ao IPv6 utilizar 64 bits na identificação das redes e 64 bits na identificação dos *hosts*.

O IPv4 só suporta cerca de 4 bilhões (4×10^9) de endereços, enquanto que o IPv6 suporta $3,4 \times 10^{38}$ endereços. O número total de dispositivos conectados a *Internet* utilizando o IPv6 pode chegar a 340.282.366.920.938.463.463.374.607.431.768.211.456 [NED, 1998].

2.4.1 – Novidades nas especificações do IPv6

As principais mudanças com relação ao IPv4 são: espaço de endereçamento, endereçamento hierárquico, o formato do cabeçalho, cabeçalhos de extensão, suporte a qualidade diferenciada, capacidade de extensão e encriptação.

- **Espaço de Endereçamento** – os endereços IPv6 têm um tamanho de 128 bits. Foi previsto que a exaustão de todos os endereços IPv4 livres para atribuição a operadores é de janeiro de 2014 [RNP, 2006], o que significa que a transição da versão do IPv4 para o IPv6 é inevitável num futuro próximo. E o IPv6 suporta a quantidade de $2^{128} = 3,4028 \times 10^{38}$ contra os quatro bilhões (4×10^9) do IPv4.

- **Endereçamento Hierárquico** – simplifica as tabelas de encaminhamento dos roteadores da rede, diminuindo assim a carga de processamento dos mesmos.

- **Formato do Cabeçalho** – totalmente remodelados em relação ao IPv4. A Figura 2.11 mostra uma comparação entre o IPv4 e o IPv6.

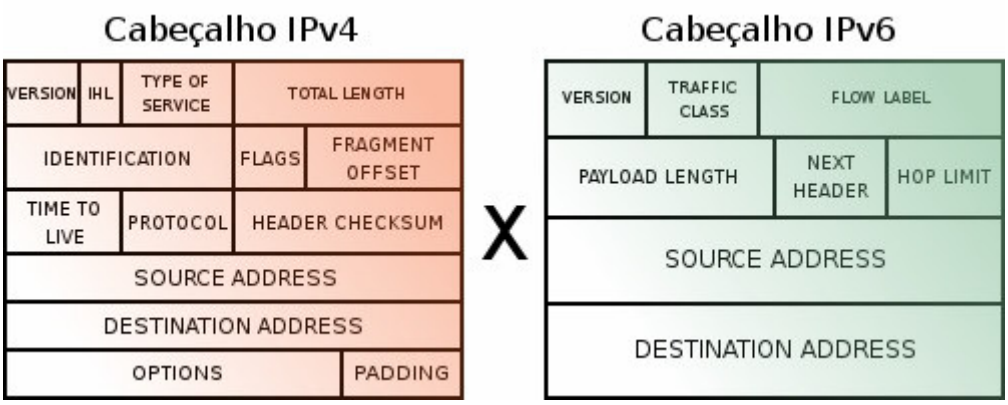

Figura 2.11 – Cabeçalho IPv4 X Cabeçalho IPv6 [IPV6 DO BRASIL, 2005]

Alguns campos e funções do protocolo IPv4 executavam tarefas que não eram necessárias para seu funcionamento, tornando o trabalho do protocolo lento. Alguns campos foram removidos, outros renomeados e movidos de lugar e um outro adicionado.

- **Cabeçalhos de Extensão** – opção para guardar informação adicional.

- **Suporte a qualidade diferenciada** – aplicações de áudio e vídeo passam a estabelecer conexões apropriadas tendo em conta as suas exigências em termos de Qualidade de Serviço (QoS).

- **Capacidade de Extensão** – permite adicionar novas especificações de forma simples.

- **Encriptação** – diversas extensões no IPv6 permitem, à partida, o suporte para opções de segurança como autenticação, integridade e confidencialidade dos dados.

2.4.2 – Cabeçalho e Especificação do IPv6

Um datagrama IPv6 é constituído por um cabeçalho base seguido de zero ou mais cabeçalhos de extensão, seguidos depois pelo bloco de dados.

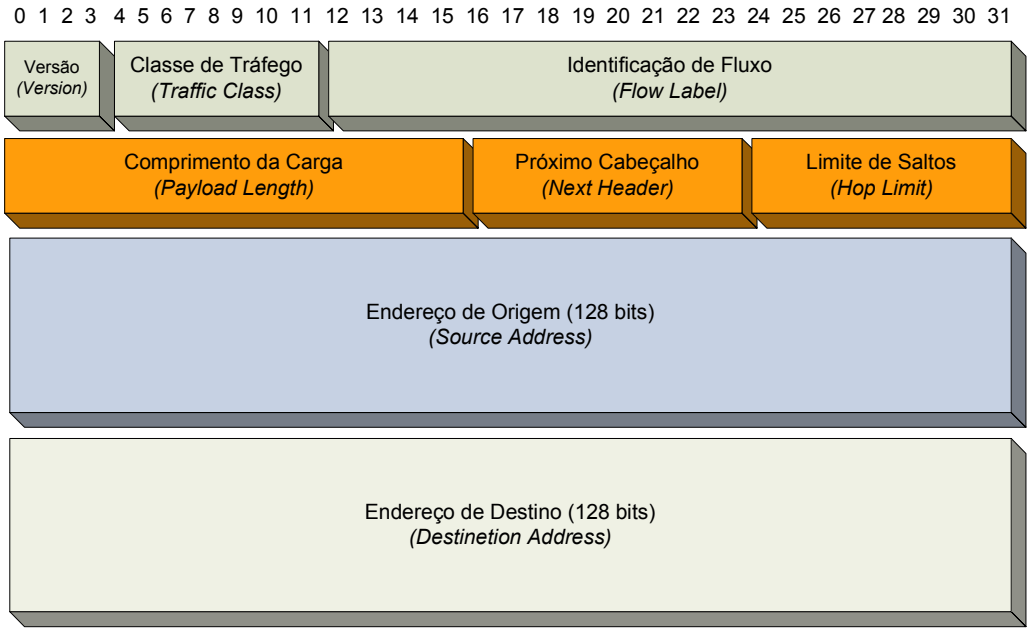

Figura 2.12 – Cabeçalho do datagrama (pacote) do IPv6 [RFC 2460]

O cabeçalho base do datagrama IPv6 tem menos informação que o cabeçalho do IPv4. A especificação de cada campo será explicada nas próximas seções.

2.4.2.1 – Versão (*Version*)

O primeiro campo do cabeçalho (*header*) do datagrama IPv6 é o campo da versão (*version*), com o tamanho de quatro bits. O valor da versão para o IPv6 é 6.

2.4.2.2 – Classe de Tráfego (*Traffic Class*)

O campo classe de tráfego (*traffic class*) é usado para assinalar a classe de serviço a que o pacote pertence, permitindo assim dar diferentes tratamentos a pacotes provenientes de aplicações com exigências distintas. Este campo serve de base para o funcionamento do mecanismo de Qualidade de Serviço (QoS) na rede. O tamanho do campo classe de tráfego é de oito bits.

2.4.2.3 – Identificação de Fluxo (*Flow Label*)

O campo identificação de fluxo (*flow label*) é usado com novas aplicações que necessitem de bom desempenho. Permite associar datagramas que

fazem parte da comunicação entre duas aplicações. Usados para enviar datagramas ao longo de um caminho pré-definido. O tamanho do campo identificação de fluxo é de 20 bits.

2.4.2.4 – Comprimento de Carga (*Payload Length*)

O campo comprimento de carga informa o comprimento dos dados, em octetos, encapsulados pela camada de rede, isto é, quantos bytes vêm depois do cabeçalho IPv6 (os campos de extensão são contabilizados). Caso esse campo seja 0, indica que o comprimento do *payload* é superior a 65.535 octetos e é informado em um *Extension Header*. O tamanho do campo comprimento de carga é de 16 bits.

2.4.2.5 – Próximo Cabeçalho (*Next Header*)

O campo próximo cabeçalho (*Next Header*) informa qual o protocolo da camada superior que está utilizando os serviços da camada IP. No IPv6, pode haver um campo opcional após o cabeçalho. Nesse caso, o valor de *Next Header* informa qual o tipo de extensão que vem após o cabeçalho IPv6. O tamanho do campo próximo cabeçalho é de oito bits.

2.4.2.6 – Limite de Saltos (*Hop Limit*)

O campo limite de saltos é semelhante ao TTL do IPv4, cada unidade processadora de pacotes (nó) decrementa esse valor de 1 unidade e quando esse valor chegar a 0, o pacote é descartado. O tamanho do campo limite de saltos é de oito bits.

2.4.2.7 – Endereço de Origem (*Source Address*)

O endereço de origem informa o endereço de 128 bits do transmissor, embora o datagrama possa ser roteado através de muitos roteadores intermediários, o campo da origem nunca muda. O tamanho do campo endereço de origem é 128 bits.

2.4.2.8 – Endereço de Destino (*Destination Address*)

O endereço de destino informa o endereço de 128 bits do destinatário. Essa informação é utilizada pelos roteadores para o encaminhamento (roteamento) do datagrama. O tamanho do campo endereço de destino é de 128 bits.

2.4.3 – Endereçamento no IPv6

2.4.3.1 – Formato de Representação do Endereço IPv6

Os endereços IPv6 são normalmente escritos como oito grupos de quatro dígitos hexadecimais, que são apresentados nas três formas para representar os endereços IPv6, conforme o RFC 2373.

1. A forma convencional é "**x:x:x:x:x:x:x:x**", onde os "**x**"s são números hexadecimais de 16 bits cada. Assim o endereço IPv6 é dividido em oito partes de 16 bits (8 x 16 bits = 128 bits). [RFC 2373, 1998]

   Exemplos:

   FEDC:BA98:7654:3210:FEDC:BA98:7654:3210

   1080:0:0:0:8:800:200C:417A

2. Para simplificar a forma de representar os endereços IPv6, pois em alguns casos pode haver seqüências de zeros, pode-se substituir estas seqüências de zeros pela agregação " :: ". No entanto, esta apenas poderá ser efetuada uma única vez em cada endereço. [RFC 2373, 1998]

Tabela 2.4 – Representação simplificada de endereços IPv6. [RFC 2373, 1998]

Forma convencional	Forma simplificada	Tipo de endereço
1080:0:0:0:8:800:200C:417A	1080::8:800:200C:417A	Endereço *Unicast*
FF01:0:0:0:0:0:0:101	FF01::101	Endereço *Multicast*
0:0:0:0:0:0:0:1	::1	Endereço *Loopback*
0:0:0:0:0:0:0:0	::	Endereço não especificado

3. A terceira forma é a representação de endereços compatíveis com IPv6-IPv4, é utilizado para a migração e coexistência de ambos os protocolos. A representação é feita da seguinte forma: **x:x:x:x:x:x:d.d.d.d**, onde, os "x"s indicam números hexadecimais (16 bits) e os "d"s são valores que representam os oito bits referentes ao endereço IPv4. [RFC 2373, 1998]

Tabela 2.5 – Representação IPv6-IPv4. [RFC 2373, 1998]

Forma convencional	Forma simplificada
0:0:0:0:0:0:192.168.3.3	::192.168.3.3
0:0:0:0:0:FFFF: 192.168.3.3	::FFFF: 192.168.3.3

2.4.4 – Tipos de Endereçamento e Hierarquia de Endereçamento

Foi criado para o IPv6 três tipos de endereçamentos. O *Unicast*, o *Anycast* e o *Multicast*. Os endereços de *broadcast* do IPv4 foram substituídos no IPv6 pelos endereços *Multicast*. [RFC 2373, 1998]

2.4.4.1 – Endereços Unicast

Endereços *Unicast* identificam uma única interface. Um pacote destinado a um endereço *Unicast* é enviado diretamente para a interface associada ao endereço. Foram definidos alguns tipos de endereços *Unicast*, que são:

a) ***Aggregatable Global Unicast Addresses*** - é o endereço *Unicast* que será globalmente utilizado na *Internet*. Seu novo formato possui seis campos: o prefixo **2000::/3 (001)** de três bits, um identificador TLA (*Top-Level Aggregation*), um campo RES (reservado), um identificador NLA (*Next-Level Aggregation*), um identificador SLA (*Site-Level Aggregation*) e o identificador da interface. Este formato quando utilizado em *links*, são agregados hierarquicamente, começando pelos clientes, em seguida por ISP (*Internet Service Provider*) intermédios e eventualmente por um ISP de topo. A Tabela 2.6 mostra o formato de endereço *Aggregatable Global Unicast Addresses*. [RFC 2374, 1998]

Tabela 2.6 – Formato de endereço *Aggregatable Global Unicast Addresses*. [RFC 2374, 1998]

3 bits	13 bits	8 bits	24 bits	16 bits	64 bits
FP (001)	TLA ID	RES	NLA ID	SLA ID	Interface ID
Topologia Pública (Provedor)				Topologia Site	Identificador de Interface (Host)

Onde o significado de cada campo é mostrado na Tabela 2.7.

Tabela 2.7 – Significado de cada campo no endereço *Aggregatable Global Unicast Addresses*. [RFC 2374, 1998]

Sigla	Descrição	Quantidade de Bits
FP	*Format Prefix (001)*	3 bits
TLA ID	Identificador *Top-Level Aggregation*	13 bits
RES	Reservado para uso futuro, todos os bits devem ser zerados	8 bits
NLA ID	Identificador Next-Level Aggregation	24 bits
SLA ID	Identificador Site-Level Aggregation	16 bits
Interface ID	Identificador de interface	64 bits

O TLA está no topo da hierarquia de roteamento e é utilizado para identificar ISP de topo. Este formato suporta 8.192 identificadores TLA, que podem ser aumentados ou através do aumento do tamanho do campo TLA, utilizando os bits reservados do campo RES, ou utilizando um prefixo de formato adicional. Os TLA são ligados numa zona livre e todos os roteadores existentes nessa zona devem possuir uma tabela de roteamento livre contemplando todas as identificações desses mesmos TLA. [RFC 2374, 1998]

O campo RES (Reservado) de 8 bits encontra-se reservado para suportar o crescimento de TLA e NLA. Este campo deverá ser sempre igual a "0". [RFC 2374, 1998]

Os identificadores NLA são utilizados pelas organizações que possuam um TLA ID para criar uma estrutura de endereçamento hierárquico e identificar sítios. [RFC 2374, 1998]

Os SLA são utilizados por organizações individualmente. Neste espaço, as organizações podem criar localmente sua própria estrutura de endereçamento hierárquico, num procedimento similar às divisões em sub-redes do IPv4, só que com um número muito maior de sub-redes. [RFC 2374, 1998]

O campo *Interface ID* é utilizado para identificar interfaces num *link* e devem ser únicos para esse *link*. Em muitos casos o identificador da interface poderá ser o mesmo ou baseado no endereço da interface da camada de enlace (parte física) e também devem ser únicos num escopo mais abrangente. Identificadores de Interface (*Interface ID*) utilizados em endereços do tipo "*Aggregable Global Unicast*" deverão ter o tamanho de 64bits e construídos utilizando o formato IEEE EUI-64. Este formato é explicado melhor no Anexo A – Identificadores de Interface. [RFC 2374, 1998]

b) ***Unspecified Address*** - representado como **0:0:0:0:0:0:0:0** ou "**::**", indica a ausência de um endereço e nunca deverá ser utilizado em nenhum nó. Um exemplo seria sua utilização como endereço de origem (*source address*) de estações ainda não inicializadas, ou seja, que ainda não tenham aprendido seus próprios endereços. Além disso, esse tipo de endereço não deve ser utilizado em endereço destino ou em cabeçalho de roteamento de pacotes IPv6. [RFC 2373, 1998]

c) ***Loopback Address*** - representado por **0:0:0:0:0:0:0:1** ou "**::1**" é chamado de endereço *loopback* e só pode ser utilizado quando um nó envia datagramas a si próprio. Não pode ser associado a nenhuma interface, porém pode ser imaginado como uma interface virtual. [RFC 2373, 1998]

d) ***IPv4 Compatible IPv6 Address*** ou ***Embedded IPv4 Addresses*** – este tipo é um endereço IPv6 com o endereço de 32 bits do IPv4 embutido. Assim anexando-se um prefixo nulo, 96 bits de zeros, a um endereço IPv4 obtém-se o formato de IPv6. Por exemplo **0:0:0:0:0:0:192.168.0.1** ou no seu formato abreviado **::192.168.0.1**. Este formato foi projetado como mecanismo de transição entre IPv6 e IPv4. [RFC 2373, 1998]

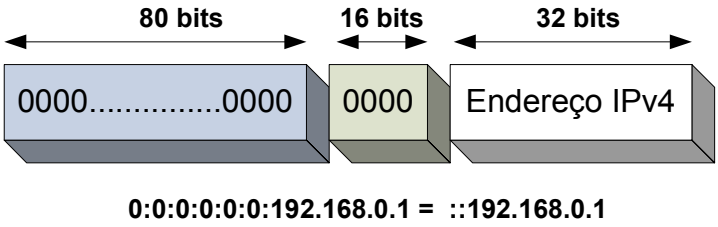

Figura 2.13 – Endereço *Unicast* – Endereço IPv6 do Tipo Compatível-IPv4 (*IPv4 Compatible IPv6*). [RFC 2373, 1998]

Para *hosts* sem suporte ao IPv6, quando da transição, foi definido um outro tipo de endereço chamado de "Endereço IPv6 do Tipo IPv4 Mapeado em IPv6" (*IPv4-mapped IPv6*): **::FFFF:192.168.0.1 .** [RFC 2373, 1998]

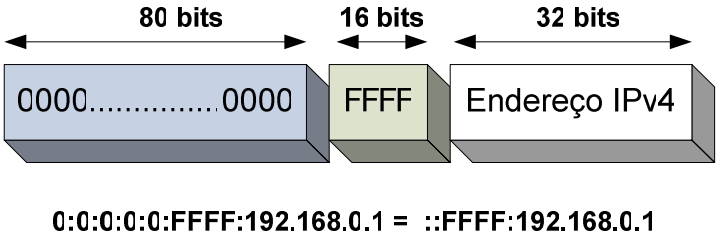

Figura 2.14 – Endereço *Unicast* – Endereço IPv6 do Tipo IPv4 Mapeado em IPv6 (*IPv4 mapped IPv6*). [RFC 2373, 1998]

2.4.4.2 – Endereços Anycast

São utilizados para identificar um grupo de interfaces pertencentes a nós diferentes. Um pacote destinado a um endereço *Anycast* é enviado para umas das interfaces identificadas pelo endereço. Especificamente, o pacote é enviado para a interface mais próxima de acordo com o protocolo de roteamento [RFC 2373, 1998].

Os endereços *Anycast* são alocados no mesmo espaço de endereçamento *Unicast*, utilizando qualquer um dos formatos dos endereços *Unicast*. Assim, ambos os tipos de endereços não são distinguíveis sintaticamente. Quando um endereço *Unicast* é configurado em mais de uma interface num mesmo nó, ele se torna num endereço *Anycast* e o nó deve ser explicitamente configurado para reconhecer este endereço. [RFC 2373, 1998]

Um possível uso deste tipo de endereço seria identificar o grupo de roteadores pertencentes a um provedor *Internet*. Ou então, identificar um conjunto de roteadores conectados a uma sub-rede, ou ainda identificar os roteadores provendo entrada para um domínio de roteamento específico [RFC 2373, 1998]. Na prática, a experiência com endereços *Anycast* na *Internet* ainda é muito incipiente e existem algumas complicações no uso generalizado desse endereço. Por isso, até que se adquira mais experiência e as soluções resolvam tais problemas, as seguintes restrições são impostas:

a) Um endereço *Anycast* não pode ser utilizado como endereço de origem (*Source Address*) de qualquer pacote IPv6;

b) Um endereço *Anycast* não pode ser configurado num *host* IPv6, ou seja, ele só pode ser associado a roteadores.

Foi predefinido um formato para os endereços *Anycast*, denominado *subnet-router Anycast address*, como mostra a Figura 2.15 o prefixo de sub-rede no endereço identifica um *link* específico. Este endereço *Anycast* é sintaticamente o mesmo endereço *Unicast*, só que com os bits do identificador da interface zerados, como mostrado. [RFC 2373, 1998]

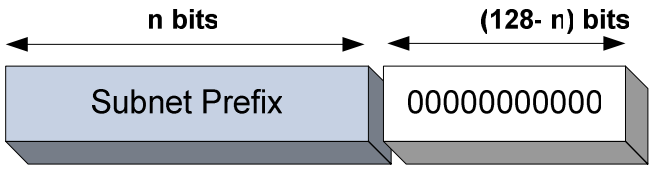

Figura 2.15 – Formato para endereços *Anycast*. [RFC 2373, 1998]

Pacotes enviados para um endereço *Subnet-router Anycast* serão entregues a um roteador na sub-rede. Todos os roteadores devem suportar endereços deste tipo para as sub-redes nas quais possuam interfaces. [RFC 2373, 1998]

Este tipo de endereçamento será útil na detecção rápida de um determinado servidor ou serviço. Por exemplo, poderá ser definido um grupo de servidores de DNS (*Domain Name System*) configurados com endereçamento

Anycast, assim, um *host* irá aceder ao servidor mais próximo utilizando este endereço. [RFC 2373, 1998]

2.4.4.3 – Endereços Multicast

Igualmente ao endereço *Anycast*, este endereço identifica um grupo de interfaces ou um grupo de nós, mas um pacote destinado a um endereço *Multicast* é enviado para todas as interfaces do grupo. Um nó pode pertencer a mais de um grupo *Multicast* [RFC 2373, 1998]

Os endereços *Multicast* têm o formato da Figura 2.16.

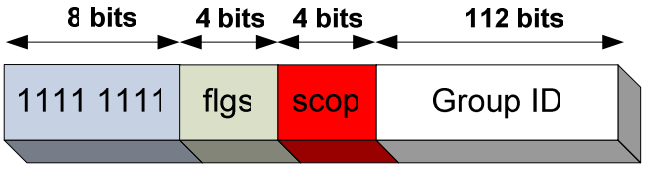

Figura 2.16 – Formato para endereços *Multicast* [RFC 2373, 1998]

Todo endereço iniciado por **1111 1111** (ou **FF**) é um endereço *Multicast*.

O campo **flgs** tem o formato 000T.

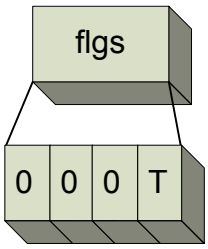

Figura 2.17 – Campo flgs [RFC 2373, 1998]

- T = 0, indica um endereço *Multicast* permanentemente (*Well-know*) alocado.
- T = 1, indica um endereço temporário.

Já o campo ***scop*** limita o escopo dos endereços *Multicast* e assume alguns valores representando endereços *Multicast, Node-local, Site-local, Link-local,*

Organization-local, Global, etc. Há vários endereços *Multicast* já alocados para algumas aplicações, outros reservados e algumas faixas ainda não alocadas. [RFC 2373, 1998]

2.4.5 – Motivação para mudar de IPv4 para IPv6

A motivação para alterar do IPv4 para o IPv6 considera o desejo por redes com arquiteturas mais escaláveis, maior segurança e integridade dos dados, extensões ao QoS, autoconfiguração, maior agregação no nível do *Backbone* global e outras necessidades.

2.4.5.1 – Espaço de endereçamento

Como o IP é o protocolo que viabiliza a *Internet* e esta vem crescendo a passos largos, sua capacidade de endereçamento está chegando ao seu limite. E o IPv6 como mostrado, melhora o espaço de endereços disponíveis.

2.4.5.2 – Qualidade de serviço

A convergência das redes de telecomunicações futuras para a camada de rede comum, o IPv6, prevê o aparecimento de novos serviços sobre IP (Ex.: VoIP, *Streaming* de vídeo em *Real-time*, etc.). O IPv6 suporta intrinsecamente classes de serviço diferenciadas, em função das exigências e prioridades do serviço em causa.

2.4.5.3 – Mobilidade

A mobilidade é um fator muito importante na sociedade de hoje em dia. O IPv6 suporta a mobilidade dos utilizadores, onde estes poderão ser conectados em qualquer rede através do seu endereço IPv6 de origem, ou seja, quando um computador sair de uma rede para outra, este assumirá um novo endereço automaticamente e informará a sua rede de origem seu novo endereço para que os pacotes enviados ao endereço anterior sejam roteados para o novo endereço. A autoconfiguração permitirá que a configuração seja automática e transparente ao usuário.

2.4.6 – Desvantagem do IPv6

Atualmente as desvantagens mais discutidas giram em torno da implementação e não do protocolo em si. Prováveis transtornos de migração podem ser a capacitação técnica dos profissionais de rede; muitos ainda não conhecem o IPv6. Então cabe a estes estarem preparados para enfrentar a situação. Logo, entra o gasto com aprendizado e treinamento para a área técnica. [AZEREDO, 2006]

O IPv6 tem a notação diferente, pois os números aparecem em hexadecimal. Porém se o DNS estiver implementado, fica mais fácil porque o nome do *host* é exibido no lugar dos caracteres em hexadecimal. [AZEREDO, 2006]

E será necessário, ainda que aos poucos, trocar equipamentos como roteadores e switches, pois nem todos suportam o novo protocolo.

2.5 – VOIP (*VOICE OVER IP*)

A principal finalidade da VoIP, que é a sigla em inglês de *Voice over IP*, Voz sobre IP, é trafegar a voz em redes de dados usando o protocolo IP (*Internet Protocol*), como na *Internet* e redes locais, o que quer dizer então que a VoIP utiliza a rede de computadores para trafegar voz.

VoIP geralmente é tratada em algumas ocasiões como sendo o mesmo que “Telefonia IP”, embora sejam definições totalmente distintas. VoIP é a tecnologia ou técnica de se transformar a voz no modo convencional em pacotes IP para ser transmitida por uma rede de dados, enquanto a Telefonia IP, que utiliza VoIP, traz consigo um conceito de serviços agregados muito mais amplo, já que carrega outras aplicações que não somente VoIP [WIKIPEDIA, 2005] . A Figura 2.18 apresenta a idéia do significado da VoIP**.**

Figura 2.18 – VoIP – tecnologia ou técnica de se transformar a voz no modo convencional em pacotes IP para ser transmitidas por redes de dados.

O que acontece na prática é a conversão dos pacotes de voz analógicos provenientes de aparelhos telefônicos, PABX (*Private Automatic Branch Exchange*) e microfones conectados ao computador em pacotes digitais, e fazê-los trafegar em uma rede de computadores usando o protocolo IP.

2.5.1 – Motivação para Utilizar VoIP

A maior motivação para o uso da tecnologia VoIP é a redução dos custos de utilização dos serviços de telefonia comum, principalmente em ambientes corporativos. As redes de dados já instaladas passam a também transmitir voz e os custos podem ser zero. O que significa que independente do dia da semana, da hora do dia e duração da conversa a ligação pode ser grátis [MORALLES, 2004].

2.5.2 – Princípios de Telefonia

Não serão apresentadas as características da telefonia em uma seção, mas algumas vezes será apresentada em referências no texto, por se tratar de um outro assunto que deve ser estudado com maior atenção em livros que tratam o tema com uma explicação mais detalhada. O livro “VoIP: Voz sobre IP” dos autores Sérgio Colcher, Antônio Tadeu A. Gomes, Anderson Oliveira da Silva, Guido L. Souza Filho e Luiz Fernando G. Soares, explica detalhadamente além do VoIP, a parte de telefonia. [COLCHER et al, 2005]

2.5.3 – Codificação do Sinal de Voz

A digitalização de um sinal de voz permite que seu armazenamento e transmissão sejam feito de forma mais eficiente. As redes telefônicas adotaram inicialmente a técnica de codificação PCM (*Pulse Code Modulation* – Modulação por Codificação de Pulsos), que consiste em 8.000 amostragens do sinal de voz contínuo, por segundo, representando o valor discreto amostrado em 8 bits. Isto implica na necessidade de um canal digital de 64 Kbps para transmissão de cada canal de voz. Este tipo de codificação procura reproduzir o sinal amostra por amostra. Possui baixo atraso para o processo e pequena complexidade, mas requer um taxa de transmissão elevada. [FERNANDO, 1999]

Ao longo dos anos, novas técnicas de codificação foram desenvolvidas, explorando-se os modelos de produção da voz. Estas técnicas fazem a segmentação do sinal analógico em intervalos periódicos, para formação de quadros após a digitalização. Os quadros são compostos por informações do sinal de voz deste período, mais as de uma parcela do quadro subseqüente. O tempo necessário para coletar as informações do próximo quadro é chamado de *lookahead.* A taxa requerida por esta técnica de codificação é baixa, mas o atraso e a complexidade são elevados, em comparação com a técnica descrita no parágrafo anterior [FERNANDO, 1999].

O ITU-T (*Telecommunication Standardization Sector of International Telecommunication Union*) padronizou várias codificações ao longo dos anos. A Figura 2.19 mostra alguns dos principais padrões de codificação.

Recomendação	Codificação	Taxa (Kbps)	Quadro / *lookahead* (ms)	Ano
G.711	PCM	64	0,125 / 0	1972
G.726	ADPCM	40, 32, 24, 16	0,125 / 0	1990
G.728	LD-CELP	16	0,625 / 0	1992
G.729	CS-ACELP	8	10 / 5	1996
G.723.1	MP-MLQ	6,3	30 / 7,5	1996
G.723.1	ACELP	5,3	30 / 7,5	1996

Figura 2.19 – Alguns padrões de codificação do sinal de voz [FERNANDO, 1999]

2.5.4 – Cenário para VoIP

Os cenários possíveis para VoIP são:

- "VoIP de terminal IP para terminal IP", neste cenário os interlocutores usam equipamentos dotados de Codecs (Codificadores/Decodificadores) [COLCHER et al, 2005] de áudio e interfaces ligadas a uma rede IP em suas conversações [COLCHER et al, 2005].

- "VoIP de terminal IP para telefone" neste cenário a integração entre RTCC (Rede Telefônica Comutada por Circuito) [COLCHER et al, 2005] e serviços conversacionais de VoIP envolve o uso de componentes adicionais chamados de *gateways* de voz e *gateways* de sinalização [COLCHER et al, 2005].

- E finalmente o cenário “VoIP de telefone para telefone” que é um cenário misto dos dois cenários anteriores. Os *gateways* de voz e de sinalização permitem que RTCC distintas utilizem redes IP para se interligarem [COLCHER et al, 2005].

O cenário “VoIP de terminal IP para terminal IP” da Figura 2.20 será o utilizado neste trabalho, onde são utilizados dois *hosts* para comunicação VoIP.

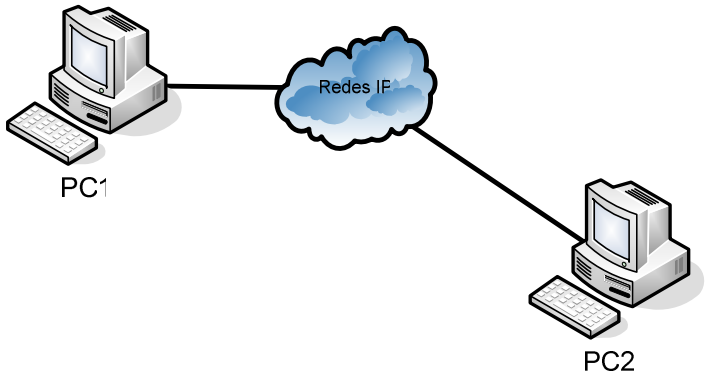

Figura 2.20 – Comunicação de voz de terminal IP para terminal IP

2.5.5 – Serviço de Voz Sobre Redes IP

A transmissão de voz codificada em uma rede com o protocolo IP, possui características peculiares a este ambiente e é importante a se considerar. Como o IP, por si só, não oferece nenhuma garantia de Qualidade de Serviço, categorizado como melhor esforço (*best-effort*), outros protocolos e soluções complementares devem ser agregados na formação da solução final, para permitirem um resultado comparável com o observado na rede de voz convencional [FERNANDO, 1999].

Define-se como Qualidade de Serviço (QoS) o conjunto de requisitos necessários a uma aplicação, para a qual exige-se que parâmetros como atrasos, banda, perdas, etc., estejam dentro de limites bem definidos [FERNANDO, 1999].

2.5.6 – Qualidade de Serviço

Os fatores que influenciam na QoS são vários e devem ser observados para que a transmissão de voz em IP seja viável e atinja a qualidade desejável pelos usuários.

2.5.6.1 – Banda

Cada tipo de codificador necessita de uma banda mínima para transmissão de um canal de voz, então se deve estar alerta ao consumo de banda em um projeto onde vários canais de voz compartilharão o mesmo canal digital, pois os valores de canais de alta velocidade são bastante onerosos. Também, é claro, existe um compromisso entre a qualidade desejada para o sinal de voz recebido e o custo da banda necessária. Podemos dividir a observação em dois ambientes: o corporativo e o residencial. O primeiro normalmente pode contar com meios de comunicação dedicados, sejam em ligações à *Internet* ou suas redes IP próprias, com velocidades de 64 Kbps à 2 Mbps. São meios de boa capacidade, contudo, compartilham aplicações de dados (emulação de terminal, transferências de arquivos, consulta a bases de dados, tráfego *web*, etc.) e vários canais de voz [FERNANDO, 1999].

O ambiente residencial é caracterizado por um acesso discado à *Internet*, que ao longo do tempo vem aumentando a velocidade de 28,8 Kbps, 33,6 Kbps, até os dias de hoje com a grande maioria a 56 Kbps [FERNANDO, 1999].

A fim de minimizar o requisito de banda, as técnicas mais empregadas são de supressão de silêncio na conversação e compressão de cabeçalhos dos pacotes IP. Ao longo de uma conversação existem vários períodos de silêncio, valendo-se disto, as implementações dos codificadores de voz podem reduzir a banda consumida por cada canal. O total de redução prático dependerá do tipo de codificação adotada. As soluções de áudio sobre IP utilizam, além deste protocolo, o UDP e o RTP (*Real Time Transport Protocol*) como protocolos de transporte. A soma dos cabeçalhos dos três protocolos resulta em 40 bytes (20 para o IP, 8 para o UDP e 12 para o RTP). Tomando como exemplo uma implementação de G.729 com um pacote formado por dois quadros de amostragem ou G.723.1 a 5,3 Kbps com um

pacote formado por um quadro de amostragem, em ambos os casos teremos 20 bytes de informação a ser transmitida. Fica evidente o despropósito na distribuição de bytes úteis e de controle. Assim, uma técnica adotada é descrita na RFC 2508, onde a maioria dos pacotes terão seus cabeçalhos comprimidos para 2 ou 4 bytes, dependendo do uso de *checksum* pelo UDP ou não. A idéia básica é que após a transmissão do primeiro pacote descomprimido, vários campos dos pacotes seguintes podem ser suprimidos ou variam de forma conhecida, sendo "remontados" no destino [FERNANDO, 1999].

2.5.6.2 – Atraso

O tempo necessário para que um pacote de áudio gerado numa origem chegue até seu destino não deve ultrapassar um patamar adequado, sob a pena de degradar a qualidade da aplicação. O patamar ideal depende fundamentalmente de três aspectos: o tipo de interatividade entre os usuários da aplicação, o nível de exigência dos usuários da aplicação e o quanto se está disposto em gastar para viabilizar uma solução que reflita pequenos atrasos. Dependendo da aplicação em questão, o grau de interação entre os usuários é grande ou não. Considerando uma conversa entre duas ou mais pessoas, o tempo entre a geração do pacote de voz e a entrega no destino deve estar entre 200 e 300 ms [FERNANDO, 1999]. A Norma G.114 do ITU-T, coloca que atrasos totais no sistema entre 0 e 150 ms são aceitável para a maioria das aplicações, entre 150 e 400 ms deve ser avaliado o impacto na qualidade da aplicação e superior a 400 ms geralmente é inaceitável [FERNANDO, 1999]. As principais parcelas que compõem o atraso total de sistema são: o processo de codificação, o tempo de transmissão e o tempo de propagação na rede [FERNANDO, 1999].

2.5.6.3 – Jitter

Outro fator importante para garantia de qualidade do sinal de voz recebido é a variação do tempo entre chegadas de pacotes consecutivos. Até determinados limiares, um tempo entre chegadas maiores, mas com uma variação menor, é observada como de melhor qualidade do que o contrário. A esta variação do tempo entre chegadas dá-se o nome de *jitter*. Na transmissão de voz sobre IP os datagramas podem tomar caminhos diferentes na rede, resultando em diferentes

tempos de propagação, ou podemos ter congestionamentos momentâneos que obriguem a maiores retardos. Para contornar este problema, são usados buffers nas entradas dos equipamentos decodificadores, a fim de guardar alguns pacotes como "reserva" em uma fila, que é servida de forma mais constante possível. Mesmo que alguns pacotes sofram uma demora maior que a normal para chegada (guardando certos limites), os pacotes no buffer são enviados para o decodificador com uma cadência homogênea. A quantidade ótima de pacotes armazenados nestes *buffers* depende do tamanho dos pacotes de voz, taxa de transmissão, atraso médio da rede e como não poderia deixar de ser, exigência da qualidade de voz requerida [FERNANDO, 1999].

2.5.6.4 – Erros na Transmissão

Algumas vezes os pacotes perdem a direção, ou são combinados juntos, ou corrompidos, enquanto estão sendo transmitidos em uma determinada rota. O receptor tem que detectar esta não conformidade e simplesmente descartar o pacote, solicitando ao transmissor que reenvie o pacote, porém a rede de telefonia IP, tem a transmissão em tempo real, não há como reenviar pacotes perdidos ou com erros para garantir uma boa qualidade na transmissão. Uma alternativa seria o uso de algoritmos "*Forward Error Correction*" (FEC), onde o mesmo pacote IP conteria vários quadros de voz implicando em uma redundância de quadro, sendo que só se aplica para codificação que geram pouco atraso, já que a formação de um pacote poderia tornar a solução inviável. [FERNANDES, 1999]

2.5.7 – Protocolos VoIP

Alguns dos protocolos utilizados em VoIP para sinalização de chamadas são: H.323, SIP, MGCP e outros [WIKIPEDIA, 2005]. Eles têm a função de converter a voz em dados digitais e compactá-la dentro do protocolo TCP/IP. Codificam a voz com G.711, G.723.1, G.729a e a enviam para o destino, chegando lá os dados são descompactados e convertidos para som digital, de modo que se pode estabelecer uma comunicação com outra pessoa em qualquer lugar do planeta [MORALLES, 2004].

2.5.7.1 – Protocolo H.323

O protocolo utilizado neste trabalho é o H.323, este protocolo foi definido pelo ITU-T com o objetivo principal de padronizar a transmissão de dados em sistemas de conferência audiovisual por meio de redes comutadas por pacotes [COLCHER et al, 2005].

O H.323 é uma recomendação extensa e flexível. Em seu perfil mais simples, estabelece procedimentos para a comunicação de áudio ponto a ponto em tempo real entre dois usuários em uma rede comutada por pacotes [COLCHER et al, 2005]. Pode ser usado, também, um *Gatekeeper* (veja como o *Gatekeeper* funciona na seção "**3.3.2 – Comunicação usando VoIP**") que de acordo com as recomendações H.323 deverá prover os seguintes serviços:

- Tradução de endereço;
- Controle de admissão;
- Controle de banda passante;
- Gerenciamento da zona administrativa;
- Sinalização de controle das chamadas
- Autorização de chamadas;
- Gerenciamento da banda passante; e
- Gerenciamento das chamadas.

A escolha deste protocolo se deve, além dos itens explicados nesta seção, ao fato de sua fácil implementação em redes IP e o suporte a IPv6 que o s*oftphone GnomeMeeting* provê juntamente com o H.323.

2.5.8 – Codec

Um codec (codificador/decodificador) é componente responsável por transformar a voz humana (um sinal analógico) em uma seqüência de bits (um sinal

digital) para transmissão numa rede de dados, fazendo amostragens periódicas no sinal de voz.

Cada codec provê certa qualidade de voz. A medida de qualidade da voz transmitida é uma resposta subjetiva de um ouvinte. Uma medida comum usada para determinar a qualidade do som produzido pelos codecs específicos é o MOS (*Mean Opinion Score*). Com o uso do MOS, um amplo range de ouvintes julgam a qualidade de uma amostra de voz (correspondendo a um codec particular) numa escala de 1 a 5. A partir desses resultados, é calculada a média dos *scores* para atribuir o MOS para aquela amostra [OLIVEIRA, 2005].

Capítulo 3 – Descrição da Infra-estrutura Lógica Proposta

3.1 – INTRODUÇÃO

Para implementar a infra-estrutura de rede com os protocolos IPv4, IPv6 e VoIP, é necessário a utilização de *software* e *hardware* que suportem estas tecnologias. Ainda existem poucos *softwares* que suportam o protocolo IPv6 quando comparado aos que suportam o IPv4, porém a maioria dos *softwares* existentes suporta o protocolo IPv4, por se tratar de um dos protocolos mais utilizados nas redes corporativas e na *Internet*. Há ainda sistemas que suportam ambos os protocolos. Já a VoIP trata-se de uma facilidade aplicada nas redes de computadores e existem várias aplicações VoIP com suporte ao IPv4 que rodam nos sistemas operacionais *Linux* e *Windows*.

Neste trabalho será utilizado o sistema operacional *Linux* para configurar o ambiente de redes com os protocolos IPv4, IPv6 e a aplicação *GnomeMeeting* para prover a comunicação usando VoIP entre os *hosts*. No *Windows* será configurado o DHCP (*Dynamic Host Configuration Protocol*) para automatizar as configurações do protocolo IPv4, o *NetMeeting* para prover a comunicação com o protocolo IPv4 ponto a ponto e utilizando o *Gatekeeper* para reconhecer os clientes VoIP. Adicionalmente serão configurados no *Red Hat 9* os serviços de DNS e HTTP (*Hypertext Transfer Protocol*), o primeiro para realizar o serviço de resolução de nomes e o segundo para servir a página *web*.

3.2 – FERRAMENTAS EMPREGADAS

As ferramentas utilizadas neste trabalho são os computadores, sistemas operacionais, *softwares, hub* e *headphones*.

Nos computadores serão instalados os sistemas operacionais *Windows 2003 Server* e *Windows XP Professional* que executarão os software necessário para o trabalho; como o *NetMeeting* para VoIP e *VMware Workstation* para rodar os sistemas operacionais *Red Hat 9*, *Conectiva Linux 10* e *Suse Linux 10* .

Os sistemas operacionais formam o ambiente computacional onde os *softwares* serão instalados ou compilados e configurado de modo que os protocolos de redes funcionem para os ambientes IPv4 e IPv6.

3.3 – CARACTERISTICAS

3.3.1 – Ambiente

O ambiente constitui-se dos *host* interligados em rede e comunicando-se sobre o protocolo IP. As distribuições *Linux* estarão com o protocolo IP versão 4 e 6, pois estes se comunicarão com ambas versões. O Windows usará somente o protocolo IP versão 4, pois não existe aplicação VoIP que dê suporte ao IPv6 do *Windows*.

3.3.2 – Comunicação usando VoIP

Para este trabalho foi utilizado cliente e servidor com suporte ao protocolo H.323 e configurado de duas formas de se conectar para realizar a comunicação VoIP.

A primeira e mais simples é a comunicação de cliente para cliente, onde foi configurado o IP de destino para realizar a chamada, e assim, promover a comunicação. A desvantagem deste método está no fato de que sempre que a pessoa que realiza a chamada deverá saber antecipadamente o local onde se encontra a pessoa com quem quer se falar. De modo geral não podemos determinar onde esta pessoa está o que constitui um problema.

A segunda forma é a utilização de um *Gatekeeper*. O *Gatekeeper* funciona como uma espécie de diretório para clientes H.323, evitando-se assim que um cliente H.323 precise saber o endereço IP da máquina da pessoa com quem quer se falar. Quando um cliente H.323 se "registra" em um *Gatekeeper* ele passa a ser conhecido pelos outros clientes conectados a esse *Gatekeeper* por um nome. Além dessa função de diretório o *Gatekeeper* é utilizado para controle de admissão (autorizando ou negando usuários) e gerenciamento de banda de acordo com recomendações H.323 explicadas anteriormente na seção “**2.5.7.1 – Protocolo H.323**”.

A maior dificuldade encontrada neste trabalho foi a de encontrar aplicações com suporte ao IPv6.

3.4 – VMWARE WORKSTATION (MÁQUINA VIRTUAL)

VMware Workstation, ou simplesmete VMware, é na realidade um produto da empresa *VMware* que fabrica *software* de virtualização de *desktops* e servidores. São as chamadas máquinas virtuais, ou seja, a emulação de um computador que rode um sistema operacional, este computador aloca os recursos de um computador físico existente. Para que se entenda melhor, imagine um computador com o "*Windows XP Professional*" instalado e sobre ele instalarmos o *VMware Workstation*. Ele possibilitará a criação de várias máquinas virtuais com seus próprios sistemas operacionais, inclusive *Linux*. É como se tivesse mais de um computador, porém tendo-se apenas um. A Figura 3.1 mostra uma máquina virtual em um computador físico e a Figura 3.2 mostra uma tela do *VMware Workstation*.

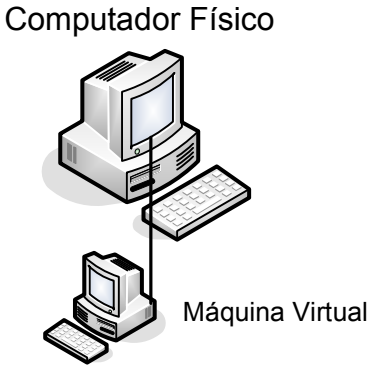

Figura 3.1 – Máquina Virtual em um Computador Físico

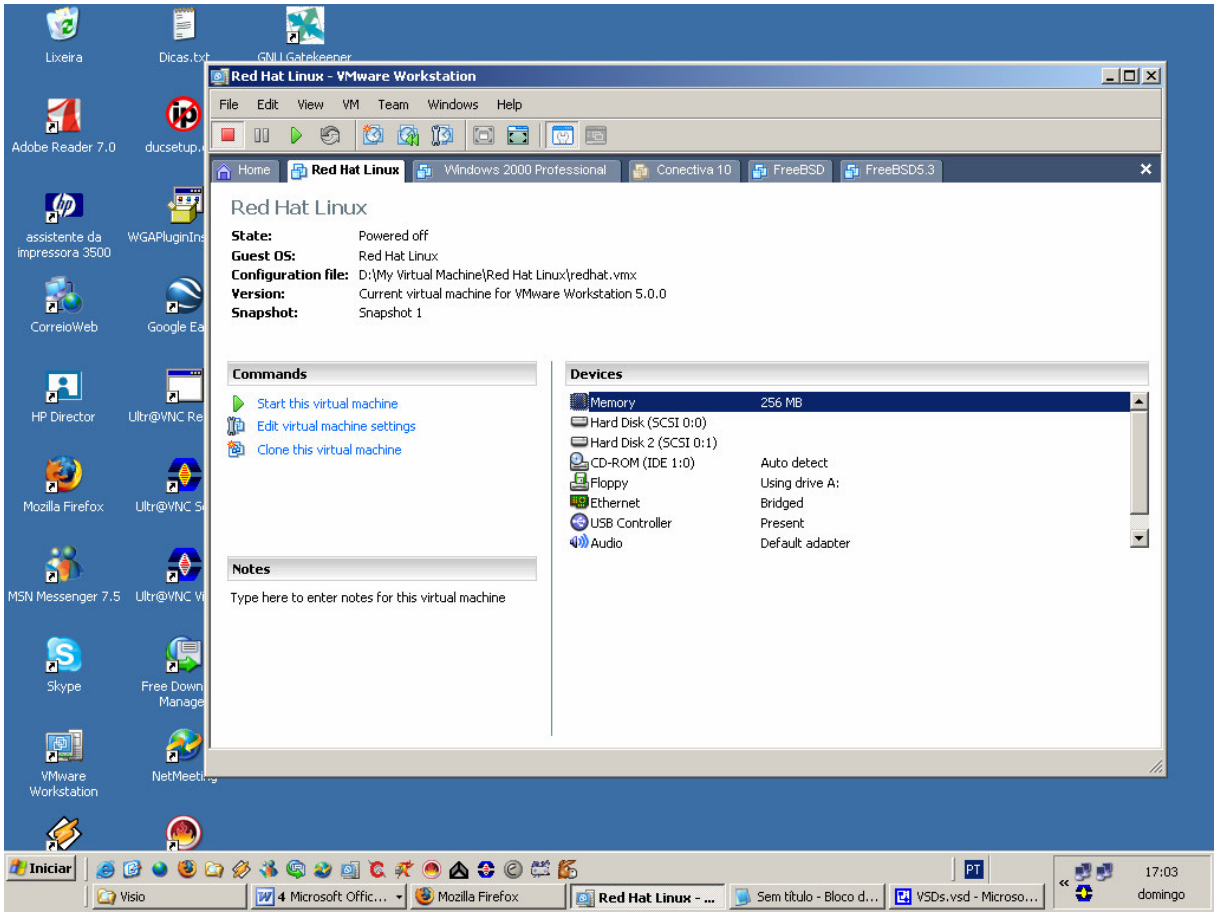

Figura 3.2 – VMware Workstation

A vantagem de utilizar a *VMware* no trabalho é a de não ter que comprar vários computadores, pois elevaria o custo da pesquisa. E a desvantagem nesta estrutura é o fato da *VMware* ter um retardo para receber e repassar o sinal de voz, quando o computador físico está com o processamento alto, isto pode ocasionar algumas interrupções na escuta.

3.5 – TOPOLOGIAS

3.5.1 – Topologia 1

Fisicamente um computador PC (*Personal Computer*) e um notebook estarão conectados a um *hub*. O PC usará o sistema operacional *Windows 2003 Server* e nele terão três máquinas virtuais, duas com o sistema operacional *Linux* e uma com o sistema operacional *FreeBSD*. O notebook usará o sistema operacional *Windows XP Professional* e uma máquina virtual com sistema operacional *Linux*.

Cada sistema estará com os serviços e aplicações necessárias instaladas. A Figura 3.3 ilustra a topologia 1.

Um dos problemas encontrado nesta estrutura é o fato da *VMware* ter um retardo para receber e repassar o sinal de voz quando o computador físico está com o processamento alto. A vantagem é não ter que comprar outros três computadores, pois elevaria o custo da pesquisa e dificultaria no transporte e espaço físico utilizado seria maior.

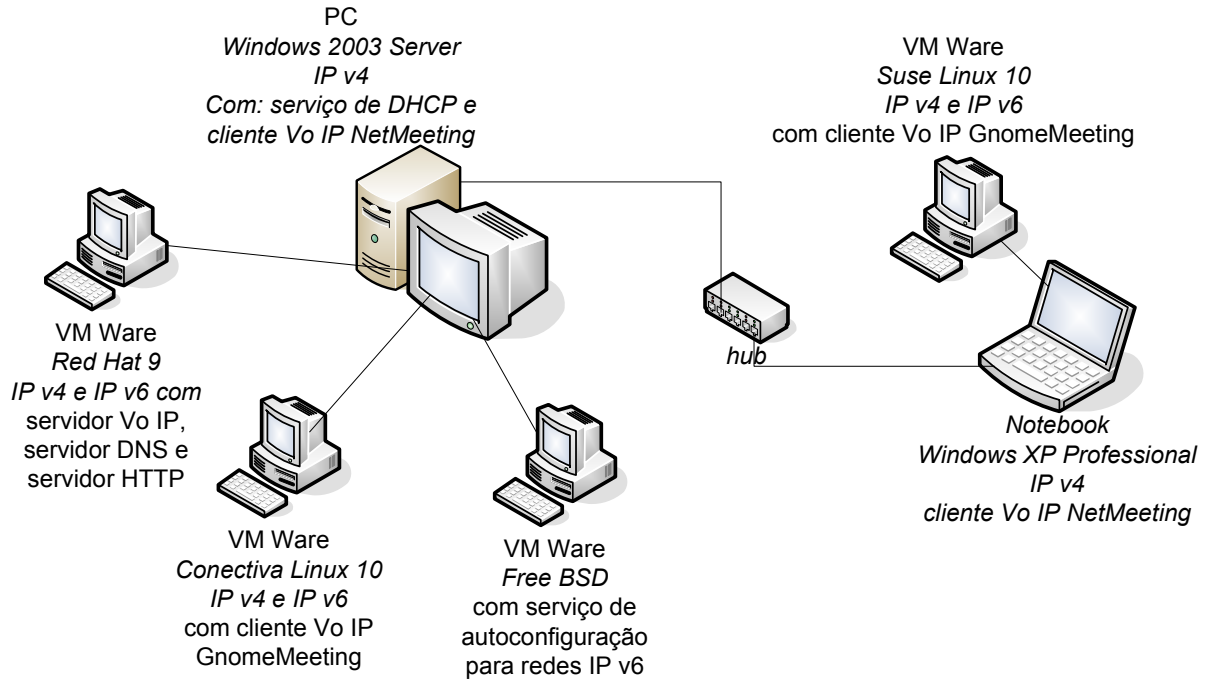

Figura 3.3 – Topologia da Estrutura 1

3.5.2 – Topologia 2

Na topologia 2, igualmente a topologia 1, fisicamente um computador PC e um notebook estarão conectados a um *hub*. O PC usará o sistema operacional *Windows 2003 Server* e nele terão três máquinas virtuais, duas com o sistema operacional *Linux* e uma com o sistema operacional *FreeBSD*. No notebook, porém, será utilizado o sistema operacional *Linux*. Cada sistema estará com os serviços e aplicações necessárias instaladas. A Figura 3.4 ilustra a topologia 2.

Um dos problemas encontrado ao usar a *VMware* é o fato de ter um retardo para receber e repassar o sinal de voz quando o computador físico está com o processamento alto, então para diminuir o processamento a *VMware* com *Suse Linux 10* foi subtraída.

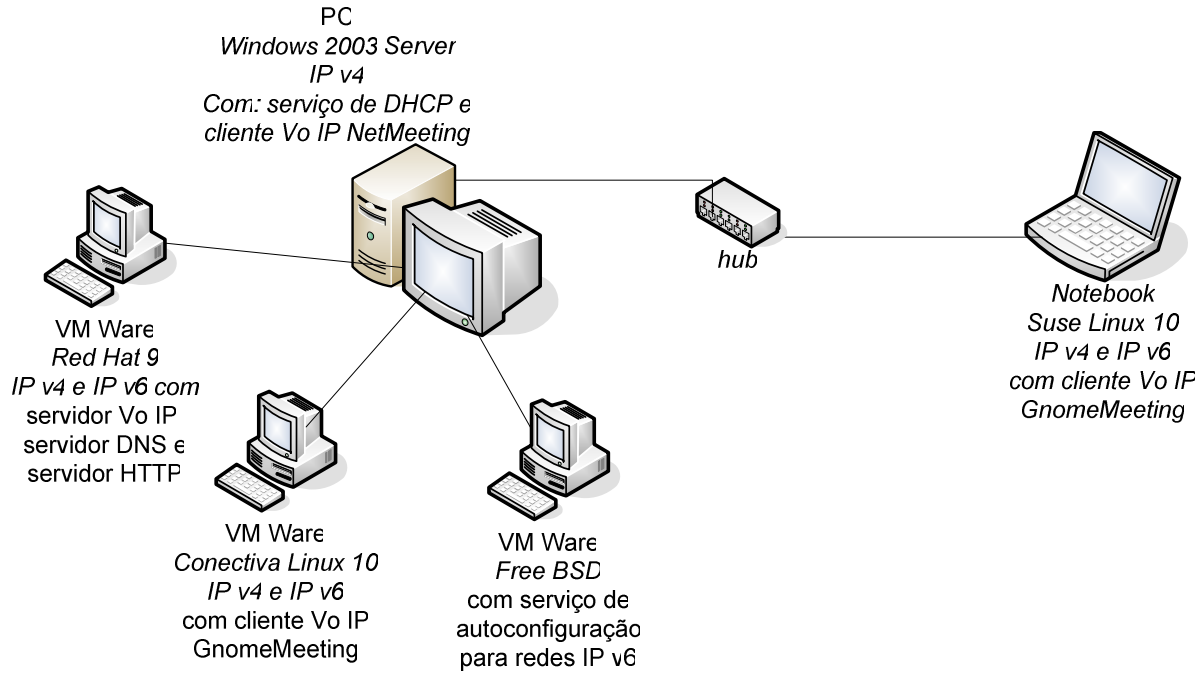

Figura 3.4 – Topologia da Estrutura 2

3.6 – LINUX

Linux é um sistema operacional criado por Linus Torvalds, estudante de Ciências da Computação da Universidade de Helsinki, na Finlândia. O que Linus Torvalds criou realmente foi somente o *kerne*l do sistema (*Linux*, na verdade, é o nome apenas deste *kernel*). Os aplicativos e outros recursos pertencem ao projeto *GNU*. *GNU* é um acrônimo recursivo para "*GNU* Não é *UNIX*" ou do inglês "*GNU is Not Unix*" a pronúncia é "*guh-noo*" [GNU, 1996]. Por isso, é considerado incorreto chamar o sistema operacional de apenas *Linux*. O correto é *GNU/Linux*. [ALECRIM, 2003]

Algumas distribuições *GNU/Linux* são gratuitas e distribuídas em vários sítios na *Internet*. Pelo fato do *GNU/Linux* ser originalmente um software livre, muitos programadores de todo o mundo se interessaram (e se interessam) em melhorá-lo.

Uns contribuem programando *drivers* (arquivos que ensinam o sistema operacional a lidar com um *hardware* específico), outros adicionam funções extras, outros melhoram o acesso à *Internet*, enfim. Com isso, o *GNU/Linux* vem ganhando cada vez mais espaço. Prova disso é que companhias como a *Sun* e a *IBM* investem cada vez mais em produtos com o *GNU/Linux*, então, atualmente existem várias distribuições *GNU/Linux* como, por exemplo: *Red Hat*, *Suse*, *Mandriva*, *Slackware*, entre outras. [ALECRIM, 2003]

Uma característica muito importante do *GNU/Linux* é que não precisa ter um computador poderoso para rodá-lo. O requerimento mínimo é de um processador 486, 16 MB de memória RAM e 200 MB de espaço no HD. Mas, neste caso, somente será possível utilizá-lo em modo texto, ou seja, sem recursos gráficos. Para isso, ele exige a mesma capacidade de *hardware* que o *Windows*. [ALECRIM, 2003]

Neste trabalho, a grande característica do *GNU/Linux* inclui, a gama extensa de *softwares* livres que podem ser baixados gratuitamente na *Internet* para o uso em redes e juntamente a isto, as funcionalidades dos *softwares* que têm várias opções que podem ser configuradas pelo utilizador. Foram utilizadas as distribuições *Red Hat 9*, *Suse Linux 10.0* e *Conectiva Linux 10.0*.

3.6.1 – IPv4 no GNU/Linux

Como nos sistemas operacionais da família "*Windows*" o *GNU/Linux* vem com o IPv4, basta para isto que haja pelo menos uma placa de rede e que se configure esta para então realizar as configurações de endereçamento IPv4 no sistema. Será realizada a configuração do IPv4 no *GNU/Linux* para este trabalhar em um ambiente misto IPv6/IPv4.

3.6.2 – IPv6 no GNU/Linux

O IPv6 no *GNU/Linux* vem nas distribuições mais atuais com o *kernel* habilitado para que se configure o IPv6 em uma ou mais placas de redes. Nas distribuições mais antigas o IPv6 não vem habilitado e é necessário a recompilação do *kernel* ou habilitação de módulos para que o IPv6 funcione. Será configurado o IPv6 no *GNU/Linux* para se realizar a comunicação IPv6 -> IPv6.

3.6.3 – VoIP no *GNU/Linux*

Para fazer a VoIP funcionar no *GNU/Linux* é necessário a utilização de *softphones* e bibliotecas de protocolos da VoIP. Existem vários *softphones* que rodam no *GNU/Linux* e que permitem realizar esta comunicação. Os *softphones* são *softwares* que simulam telefones e transformam o computador em um verdadeiro centro de comunicações. Os *softphones* podem ser utilizados com *headsets* ou com telefones USB para obter mais conforto e qualidade na conversação. Os headsets e telefones USB devem ser conectados ao computador. A Figura 3.5 mostra um *headset* e a Figura 3.6 mostra um telefone USB

Figura 3.5 – *Headset* utilizado com o *Softphone*

Figura 3.6 – Telefone USB

Neste trabalho serão utilizados os *headsets* nos dois computadores para realizar a conversa e escuta. A utilização do *headset* foi escolhida por ser mais fácil de se adquirir e mais barato do que o telefone USB. Os *headsets* utilizados custaram em torno de R$ 18,00 cada. O telefone USB custa em média R$ 80,00.

3.6.4 – O que é o GnomeMeeting?

GnomeMeeting é um *software* VoIP (*softphone*), que permite a conversa entre pessoas e a transmissão de vídeo sobre a rede IP. O *GnomeMeeting* suporta o protocolo H.323 provido pela biblioteca OpenH323 . Ele pode se conectar com uma variedade de aplicações que suportam o protocolo H.323, inclusive com o *Windows NetMeeting*.

3.6.5 – O que é o GnuGK?

GnuGK é o acrônimo de "*GNU Gatekeeper*", ele é um projeto de código-fonte aberto que implementa um *Gatekeeper* H.323. O nome formal do projeto é *OpenH323 Gatekeeper - The GNU Gatekeeper*, porém por questões de conveniência e também porque ele realmente é chamado por seu apelido *GnuGK*, é assim que será referenciado aqui.

O *GnuGK* implementa muitas funções da recomendação H.323 através do uso da biblioteca OpenH323.

3.6.6 – DNS

Será configurado no *Red Hat 9* um servidor de DNS para resolver as consultas realizadas por nomes na rede IPv4 e IPv6.

O DNS é a sigla de *Domain Name System*. Ele é o tradutor de nomes para números em uma rede ou na *Internet*. Quando se quer acessar um computador é muito mais fácil lembrar o nome dele do que o endereço IP, por isso é utilizado o DNS, para resolver nomes em endereço IP. Então, quando se procura o nome de um computador, por exemplo, tigreredhat, o servidor de DNS irá resolver o nome deste computador para o solicitante, por exemplo, o IPv4 dele 10.7.3.30 ou o IPv6 dele FEC0:2006:6::30.

3.6.7 – HTTP

Será utilizado no *Red Hat 9* um sevidor web (servidor HTTP) para testar a comunicação sobre o IPv4 e IPv6.

HTTP significa *HyperText Transfer Protocol*, ou seja, Protocolo de Transferência de Hipertexto, e é utilizado para transferência de dados na *Internet* e também em redes locais. O mesmo transfere dados de hiper-midia (imagens,sons e textos). O protocolo HTTP surgiu da necessidade de distribuir informações pela *Internet*. Para que essa distribuição fosse possível, foi necessário criar uma forma padronizada de comunicação entre os clientes e os servidores da *web*. Com isso, o protocolo HTTP passou a ser utilizado para a comunicação entre computadores na *Internet* e a especificar como seriam realizadas as transações entre clientes e servidores, através do uso de regras básicas.

3.7 – MICROSOFT WINDOWS

Microsoft Windows é um sistema operacional utilizado em muitos computadores ao redor do mundo. É um produto da empresa *Microsoft*, criada por Bill Gates e Paul Allen, e que possui várias versões. Para o trabalho foi utilizado a versão *Windows Server 2003 Standard Edition* e o *Windows XP Professional*, no qual as "Máquinas Virtuais" estarão rodando. Também será utilizado o "*NetMeeting*" para promover a comunicação.

3.7.1 – IPv4 no Windows

O IPv4 é o protocolo padrão para as versões do *Windows* utilizadas neste trabalho. A configuração do IPv4 é relativamente simples e estará sendo realizada através de um servidor DHCP instalado no *Windows 2003 Server*.

3.7.2 – IPv6 no Windows

Existe um projeto chamado *Microsoft Research IPv6*, desenvolvido pela *Microsoft* em conjunto com a USC (*University of Southern California*), porém as pesquisas estão em andamento e ainda são poucos os *software* que rodam sobre o IPv6 para *Windows*, e também, não há um *software* para a comunicação em VoIP, então não será configurado IPv6 no *Windows*.

As próximas versões dos *Windows*, *Microsoft Windows Vista* como o *Windows Longhorn Server*, atualmente em teste beta, incluem a próxima geração da pilha TCP/IP, uma pilha redesenhada com versões integradas do IPv4 e IPv6.

3.7.3 – O que é o Windows NetMeeting?

O *Windows NetMeeting* é um *software* da empresa *Microsoft* de conferência via rede. Possibilita que diversas pessoas interajam juntas de diferentes lugares, via *chat*, comunicação de voz usando o protocolo H.323 e vídeo. Possui, além disto, um recurso que permite que os usuários dividam a mesma tela de um software.

O *Windows NetMeeting* vem nas versões de *Windows* utilizado neste trabalho (*Windows XP Professional* e *Windows 2003 Server*), então foram feitas as configurações para a utilização dele para redes IPv4.

O *Windows NetMeeting* ainda não dá suporte para redes IPv6, para este fim foi utilizado o *GnomeMeeting*.

A Figura 3.7 mostra a tela inicial do *NetMeeting*.

Figura 3.7 – NetMeeting

A utilização do *Windows NetMeeting* no trabalho tem o objetivo de mostrar a grande variedade de ferramentas mistas que podem ser utilizadas, juntando-se com a facilidade com que ele pode ser configurado.

3.7.4 – DHCP

O DHCP é um serviço utilizado para automatizar as configurações do protocolo TCP/IP nos componentes de uma rede, por exemplo, os computadores. Sem o uso deste serviço os administradores da rede teriam de configurar, manualmente, as propriedades do protocolo TCP/IP em cada componente da rede.

3.8 – FREEBSD

3.8.1 – O que é o FreeBSD?

O *FreeBSD* é um sistema operacional livre do tipo *Unix* descendente do BSD (*Berkeley Software Distribution*) desenvolvido pela Universidade de *Berkeley*. O *FreeBSD* é um sistema operacional multiusuário, capaz de executar em multitarefa. Ainda que o *FreeBSD* não possa ser chamado apropriadamente de *Unix*, por não estar sob a licença do "*The Open Group*", ele foi desenvolvido para ser compatível com a norma *POSIX*, assim como outros clones do *Unix*.

O *FreeBSD* fornece compatibilidade binária com muitos outros clones do *Unix*, incluindo o *Linux*. A razão por trás disso está em poder utilizar programas desenvolvidos para Linux, geralmente comerciais, que só são distribuídos em forma binária e que por isso não podem ser portados para o *FreeBSD*.

O *FreeBSD* começou a ser desenvolvido em 1993, a partir do *386BSD*. Contudo, devido a problemas legais em relação às fontes do *386BSD*, ele precisou ser totalmente reconstruído para a versão 2.0 com base no *4.4BSD-Lite* em 1995.

Será utilizado para o trabalho a versão *FreeBSD 5.4 Release*.

3.8.2 – IPv4 no FreeBSD

Como nos sistemas operacionais da família "*Windows*" e *Linux* o IPv4 no *FreeBSD* também é utilizado como protocolo para interligação em redes. É necessário, porém, que se tenha pelo menos uma placa de rede e que se configure esta. O IPv4 será configurado no *FreeBSD* para prover a comunicação entre os nós da rede.

3.8.3 – IPv6 no FreeBSD

Para se utilizar o IPv6 no *FreeBSD* é necessário habilitar esta funcionalidade e compilar o *kernel*. E para prover endereços IPv6 automaticamente para computadores na rede, será utilizada a característica de "*Stateless Address Autoconfiguration*" do IPv6 definida no RFC 2462, porém para aumentar a gama de conhecimento que este trabalho pode proporcionar, em cada distribuição *Gnu/Linux* será configurado o IPv6 manualmente.

Capítulo 4 – Implementação do Trabalho e Resultados Obtidos

Neste capítulo são apresentados a implementação do trabalho juntamente com os resultados das configurações obtidos. Serão explicadas as configurações realizadas e mostrado o teste de funcionamento da estrutura. Assim, na seção "**4.8 – Resultados Obtidos**" será mostrado que a comunicação de rede e a comunicação utilizando a VoIP, foi realizada com sucesso na estrutura que foi proposta e como foi também proposto no objetivo deste trabalho.

Nas configurações não será apresentada a instalação dos sistemas operacionais, por se tratar de um assunto que pode deixar o texto extenso e cansativo. A forma de instalação para cada sistema operacional pode ser encontrada na documentação própria que vem junto com o CD de instalação ou no sítio do desenvolvedor.

4.1 – RED HAT 9

O *Red Hat 9* é o *GNU/Linux* da empresa *Red Hat, Inc*. Esta distribuição é uma das mais famosas entre as distribuições existentes. A *Red Hat, Inc* fornece o *GNU/Linux Red Hat 9* gratuitamente para *download* em seu sítio.

4.1.1 – Configurando o Endereçamento IPv4

Para realizar as configurações necessárias de IPv4 no *Red Hat 9*, foi configurado o arquivo "**ifcfg-eth0**" do diretório "/**etc**/**sysconfig**/**network-scripts**", o arquivo "**network**" do diretório "/**etc**/**sysconfig**" e o arquivo "**resolv.conf**" do diretório "/**etc**".

Segue os arquivos com os comentários e configurações realizadas:

Arquivo: **ifcfg-eth0**

```
# /etc/sysconfig/network-scripts

DEVICE=eth0                 # Nome da interface, eth0
BOOTPROTO=static            # static, configuração de rede feito
```

```
                                    # manualmente
IPADDR=10.7.3.30                    # Endereço IPv4
NETMASK=255.255.255.0               # Máscara de sub-rede
NETWORK=10.7.3.0                    # Endereço de rede
BROADCAST=10.7.3.255                # Endereço broadcast da rede
ONBOOT=yes                          # Iniciar ao ligar o sistema
USERCTL=yes                         # Determina se os usuários podem
                                    # ligar/desligar ("yes") ou não ("no") a
                                    # interface.
PEERDNS=no                          # Em "no" não permite a modificação
                                    dinâmica do /etc/resolv.conf
TYPE=Ethernet                       # Tipo de interface
```

Arquivo: **network**

```
# /etc/sysconfig/network

NETWORKING=yes                      #
HOSTNAME=tigreredhat                # Nome do host
GATEWAY=10.7.3.1                    # Gateway padrão
```

Arquivo: **resolv.conf**

```
# /etc/resolv.conf

search csnet.rede                   #  Domínio a ser utilizado quando
                                    # nenhum domínio for especificado
nameserver 10.7.3.30                # Endereço IP do servidor de DNS a
                                    # ser utilizado
```

Após configurar o endereçamento de rede é necessário levantar o serviço de rede digitando o comando "**service network start**". Pode-se usar também o comando "**ifup eth0**" para levantar a interface eth0 especificamente. E para checar as configurações da interface, executa-se o comando "**ifconfig**".

4.1.2 – Configurando o Endereçamento IPv6

No *Red Hat 9* foi feita a configuração do endereçamento IPv6 semelhante ao que foi configurado para o IPv4 editando os arquivos de configuração de rede.

O primeiro passo é adicionar a linha "**NETWORKING_IPV6=yes**" no arquivo "**network**", este é o mesmo arquivo utilizado para configurar o IPv4. Veja como ficou:

Arquivo: **network**

```
# /etc/sysconfig

NETWORKING=yes          # Habilita a network IPv4
HOSTNAME=tigreredhat    # Nome do host
GATEWAY=10.7.3.1        # Gateway padrão

NETWORKING_IPV6=yes     # Habilita a network IPv6
```

Em seguida deve-se configurar a interface com um endereço IPv6. No caso foi utilizado arquivo "**ifcfg-eth0**" e adicionados as seguintes linhas:

```
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6ADDR=[endereço ipv6 do host]
```

Veja o arquivo "**ifcfg-eth0**" configurado:

Arquivo: **ifcfg-eth0**

```
# /etc/sysconfig/network-scripts

DEVICE=eth0             # Nome da interface: eth0
BOOTPROTO=static        # Static, configuração de rede feito
                        # manualmente
IPADDR=10.7.3.30        # Endereço IPv4
NETMASK=255.255.255.0   # Máscara de sub-rede
NETWORK=10.7.3.0        # Endereço de rede
BROADCAST=10.7.3.255    # Endereço broadcast da rede
ONBOOT=yes              # Iniciar ao ligar o sistema
USERCTL=yes             # Determina se os usuários podem
                        # ligar/desligar ("yes") ou não ("no") a
                        # interface.
PEERDNS=no              # Em "no" não permite a modificação
                        # dinâmica do /etc/resolv.conf
TYPE=Ethernet           # Tipo de interface

IPV6INIT=yes
IPV6_AUTOCONF=yes
```

IPV6ADDR=FEC0:2006:6::30/64 # Endereço IPv6

Arquivo: **resolv.conf**

```
# /etc/resolv.conf

search csnet.rede              # Domínio a ser utilizado quando
                               # nenhum domínio for especificado
nameserver 10.7.3.30           # Endereço IP do servidor de DNS a
                               # ser utilizado
```

4.1.3 – Configurando o DNS

No momento da instalação do *Red Hat 9* é possível escolher a opção de instalação de um servidor de DNS, como a instalação dos sistemas operacionais não será abordada neste trabalho, será então considerado que o *Red Hat 9* foi instalado com a opção de servidor de DNS também.

Para verificar se o serviço de DNS foi instalado executa-se o seguinte comando:

rpm -q bind

Se o serviço de DNS tiver sido instalado, aparecerá o nome do pacote do "**bind**", que é o pacote de instalação do serviço de DNS conforme mostra a Figura 4.1.

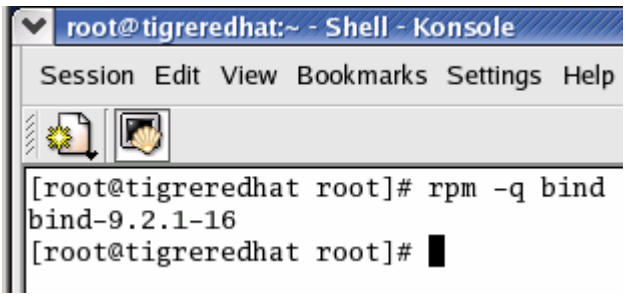

Figura 4.1 – Verificando se o serviço de DNS está instalado.

Como o serviço de DNS está instalado, será então configurado nos passos seguintes:

Os arquivos necessários para configuração do servidor de DNS são:

- **/etc/resolv.conf** – guarda o nome do domínio e os servidores DNS em cadeia hierárquica, este arquivo foi configurado nas seções de configuração do IPv6 de cada distribuição *GNU/Linux*.

- **/etc/named.conf** – guarda todos os parâmetros usados pelo named, daemon que controla o serviço DNS.

E além destes acima, será necessário criar dentro do diretório "**/var/named**" os dados do domínio. O arquivo "**/var/named/csnet.rede.zone**" criado contém estes dados.

Configurando:

Segue a parte do arquivo "**named.conf**", onde foi configurado o domínio, o arquivo completo está no Anexo C – Arquivo "named.conf":

```
zone  "csnet.rede" {
      type master;
      file  "csnet.rede.zone";
};
```

O arquivo "**csnet.rede.zone**" é apresentado no Anexo D – Arquivo "csnet.rede.zone".

Agora basta iniciar o serviço com o comando:

service named start

Veja o resultado na Figura 4.2.

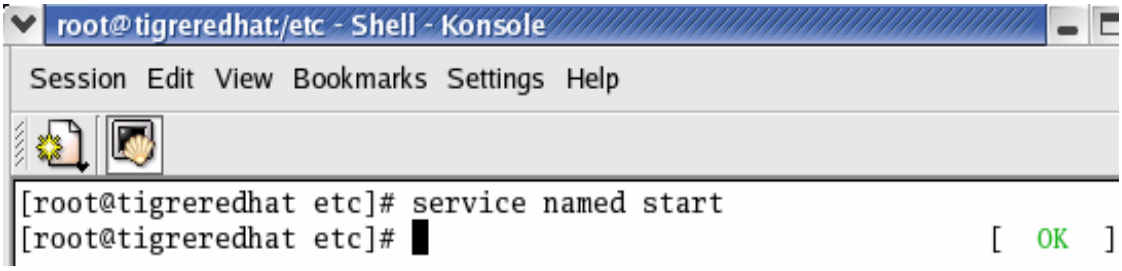

Figura 4.2 – Iniciando o serviço de DNS.

4.1.4 – Servidor HTTP

Foi instalado o servidor de páginas HTTP no *Red Hat 9* no momento de sua instalação, para verificar o funcionamento da rede IPv4 e IPv6. Não foi realizada a configuração do servidor HTTP, pois este se limita a realizar o teste de conectividade na rede IPv4 e IPv6, então basta iniciar o serviço HTTP com o comando "**service httpd start**". Nas próximas seções serão realizadas conexões no servidor HTTP usando os endereços IPv4 e IPv6.

4.1.5 – Resultados das Configurações

O resultado da configuração realizada é que o sistema operacional tem endereços IPv4 e IPv6, e provê o serviço de DNS resolvendo nomes para redes IPv4 e IPv6. O serviço HTTP será testado no momento da configuração dos outros *hosts*.

Pode-se comprovar e testar estas configurações fazendo-se uso dos comandos "**ifconfig**", "**ping**" e "**ping6**".

Executando o comando "**ifconfig**" pode-se verificar as configurações de redes para IPv4 e IPv6, como mostra a Figura 4.3.

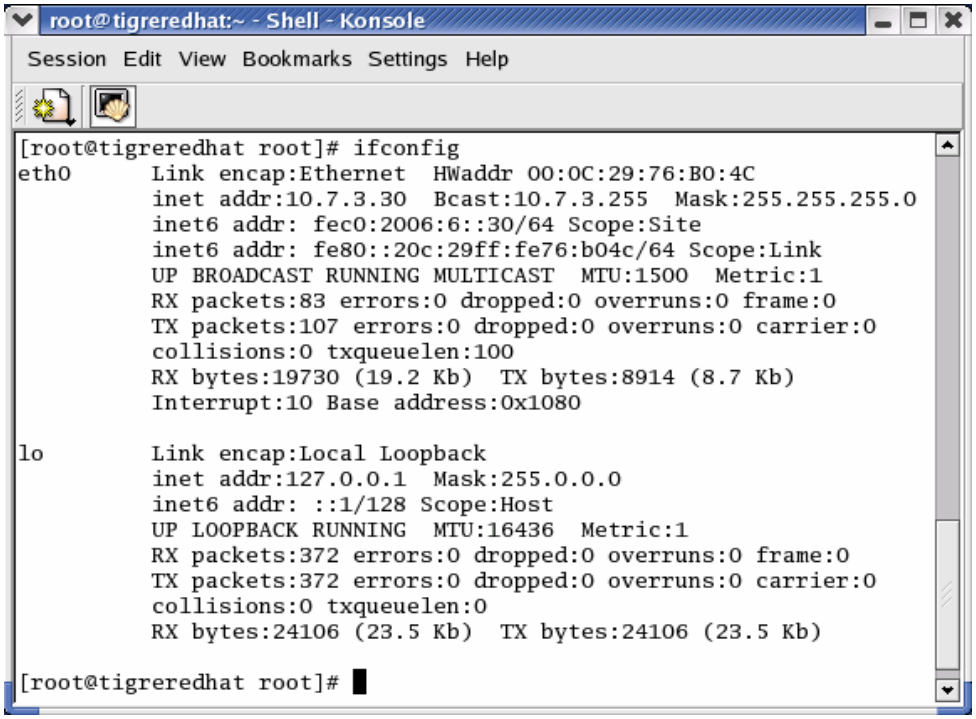

Figura 4.3 – Configurações de rede do *Red Hat 9*

Explicando resumidamente, na Figura 4.3 é possível verificar que o endereço 10.7.3.30 é um endereço IPv4 e o endereço FEC0:2006:6::30 é um endereço IPv6. Assim, usando o comando "**ping**" e "**ping6**" para testar se o DNS está resolvendo o nome do servidor em seu endereço IPv4 e IPv6 respectivamente, obtemos o resultado da Figura 4.4.

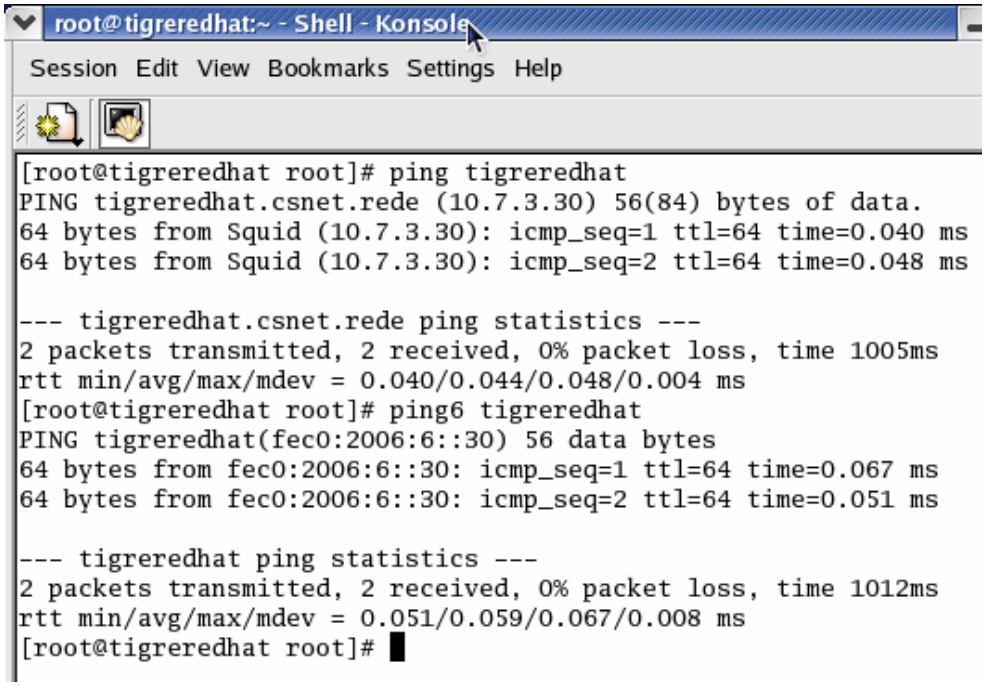

Figura 4.4 – Teste usando os comandos: "ping" e "ping6"

Observa-se que o servidor de DNS resolveu o nome do servidor, *tigreredhat*, tanto em um endereço IPv4 como em um endereço IPv6. O comando normalmente usado para testar um servidor de DNS é "**nslookup**" ou o "**dig**", porém, nenhum dos dois comandos dá suporte ao IPv6, por isso foi usado o comando "**ping6**", que é a ferramenta padrão para testar a conectividade em redes IPv6.

Assim, pode-se dizer que o servidor de DNS está funcionando para resolver nomes para redes IPv4 e IPv6.

4.2 – CONECTIVA LINUX 10

O *Conectiva Linux 10* é *GNU/Linux* da empresa *Conectiva*, atual *Mandriva Conectiva*, empresa brasileira com sede em Curitiba. A *Mandriva Conectiva* é a operação brasileira da *Mandriva*, desenvolvedora e distribuidora do

sistema operacional *Mandriva Linux*, resultado da fusão ocorrida em fevereiro deste ano entre a *Mandrakesoft*, uma das principais distribuições *Linux* da Europa, com atuação mundial em mais de 120 países, e a Conectiva, pioneira na distribuição *GNU/Linux* e código aberto em português, espanhol e inglês para toda a América Latina. O *Conectiva Linux 10* pode ser baixado do sítio da empresa gratuitamente.

4.2.1 – Configurando o Endereçamento IPv4

As configurações de endereçamento IPv4 no *Conectiva Linux 10* segue as mesmas configurações realizadas para o *Red Hat 9*, então mostrarei somente os arquivos de configuração.

Segue o arquivo "**network**" configurado:

Arquivo: **network**

```
# /etc/sysconfig

NETWORKING=yes          # Habilita a network IPv4
GATEWAY=10.7.3.1        # Gateway padrão
```

Segue o arquivo "**ifcfg-eth0**" com os comentários e configurações realizadas:

Arquivo: **ifcfg-eth0**

```
# /etc/sysconfig/network-scripts

DEVICE=eth0             # Nome do dispositivo: eth0
ONBOOT=yes              # Iniciar ao ligar o sistema
BOOTPROTO=static        # static, configuração manual de rede
IPADDR=10.7.3.21        # Endereço IPv4
NETMASK=255.255.255.0   # Máscara de rede
NETWORK=10.7.3.0        # Endereço de rede
USERCTL=yes             # Determina se os usuários podem
                        # ligar/desligar ("yes") ou não ("no") a
                        # interface.
PEERDNS=no              # Em "no" não permite a modificação
                        # dinâmica do /etc/resolv.conf
GATEWAY=10.7.3.1        # Gateway padrão
TYPE=Ethernet           # Tipo de interface
```

Arquivo: **resolv.conf**

```
# /etc/resolv.conf

search csnet.rede
nameserver 10.7.3.30
```

Como no *Red Hat 9*, após configurar o endereçamento de rede é necessário levantar o serviço de rede digitando o comando "**service network start**". Pode-se usar também o comando "**ifup eth0**" para levantar a interface eth0 especificamente.

4.2.2 – Configurando o Endereçamento IPv6

Vamos habilitar o IPv6 no *Conectiva Linux 10* usando o módulo para IPv6, então, primeiro vamos testar se o módulo IPv6 foi carregado ou não:

test -f /proc/net/if_inet6 && echo "Ok"

Se aparecer a mensagem "**Ok**" é porque o módulo foi carregado e o sistema está pronto para configurar o endereçamento IPv6. Se caso falhar, é porque o módulo não foi carregado.

Se não foi carregado, será necessário digitar o comando "**modprobe ipv6**" e o comando "**ifconfig <interface> inet6 add <endereço ipv6>/<prefixo>**" para configurar a interface com o endereço IPv6.

Veja o exemplo de configuração de uma interface:

ifconfig eth0 inet6 add FEC0:2006:6::21/64

Para testar o sucesso do carregamento do módulo IPv6 digita-se o comando "**lsmod | grep -w 'ipv6' && echo "O módulo IPv6 foi carregado"**".

Pode-se, também, carregar o módulo IPv6 no momento em que o sistema é iniciado editando-se o arquivo "**rc.local**" que fica no diretório "/**etc**". O arquivo "**rc.local**" é um arquivo de "*script*" e é executado após todos os outros

arquivos de "*scripts*" de *init* [RIBEIRO, 2004]. O diretório "/**etc**" é composto por arquivos necessários à configuração do sistema.

Segue o arquivo "**rc.local**" configurado para carregar o módulo IPv6 e atribuindo o endereço "**FEC0:2006:6::21/64**" à interface de rede "**eth0**".

Arquivo: **rc.local**

```
#!/bin/sh
#Habilita o IPv6 no Conectiva Linux 10
modprobe ipv6
#Configura os enderecos IPv6 na placa de rede eth0
ifconfig eth0 inet6 add FEC0:2006:6::21/64
touch /var/lock/subsys/local
```

4.2.3 – Resultados das Configurações

O resultado da configuração realizada no *Conectiva Linux 10* é o sistema operacional com o endereço IPv4 e endereço IPv6 configurado conforme se vê na Figura 4.5.

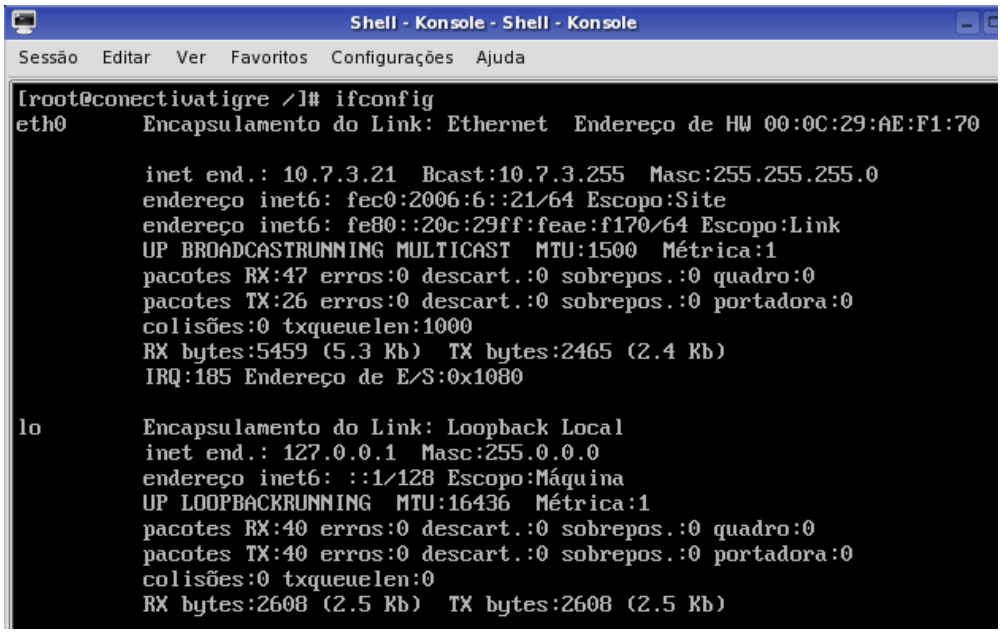

Figura 4.5 – Configuração IPv4 e IPv6 no *Conectiva Linux 10*.

Utilizando do comando "**ifconfig**" pode-se ver as configurações da Figura 4.5.

Com o *Red Hat 9* e o *Conectiva Linux 10* configurados com os protocolos IPv4 e IPv6, pode-se realizar a comunicação entre os dois *hosts*. Então, podem-se ver os resultados fazendo os testes de “ping”, “ping6” e acessar o servidor HTTP usando o navegador *web*.

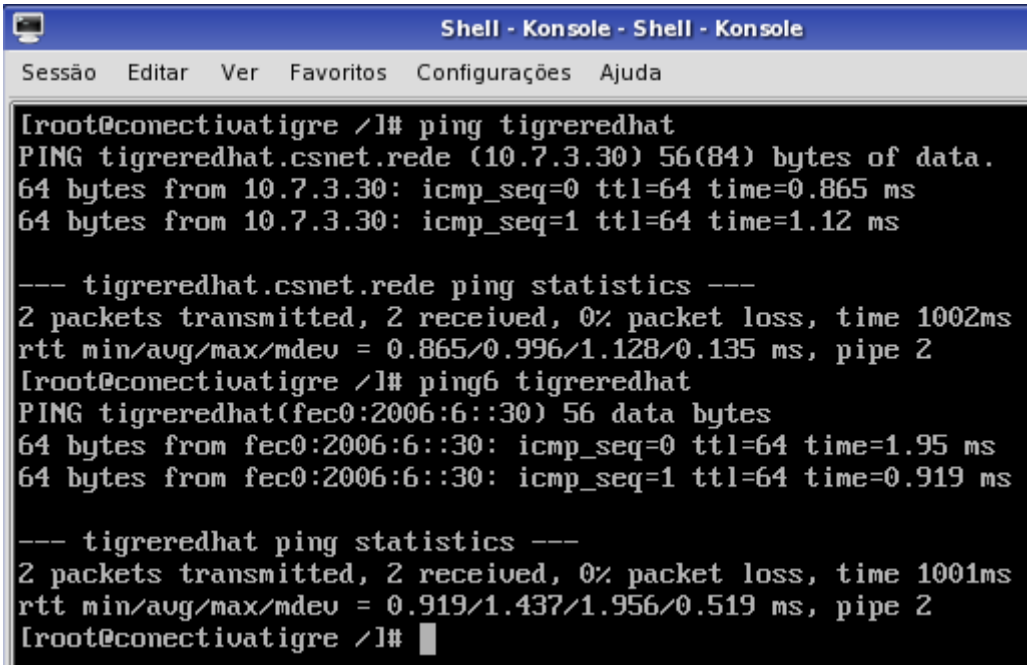

Figura 4.6 – Teste de “ping” e ”ping6”

A Figura 4.6 mostra um teste de “**ping**” e “**ping6**” realizado do *Conectiva Linux 10* para o *Red Hat 9* e como se vê na figura, o resultado é a resposta do *Red Hat 9* (10.7.3.30 e FEC0:2006:6::30) ao *Conectiva Linux 10*.

A Figura 4.7 mostra um acesso feito ao servidor de HTTP usando o protocolo IPv6. Para evidenciar mais que o acesso foi feito usando o protocolo IPv6, ao invés de acessar pelo nome do servidor, o acesso foi realizado pelo endereço IPv6. Este acesso também pode ser realizado digitando o endereço IPv4 ou pelo nome do servidor.

Obs: note que o endereço IPv6 tem que ser digitado entre colchetes, por exemplo, http://[FEC0:2006:6::30].

Figura 4.7 – Acessando a página usando o IPv6

Assim, fica evidente que a infra-estrutura de sistemas operacionais, DNS e HTTP funcionam junto com os protocolos IPv4 e IPv6 simultaneamente. Nas próximas seções será mostrada a VoIP funcionando nesta infra-estrutura.

4.3 – SUSE LINUX 10

O *Suse Linux 10* é um sistema operacional da empresa *Novell*. A *Novell* apresentou o *Suse Linux 10* como a primeira distribuição de software livre baseado no recém-lançado projeto *Open Suse*, que congrega desenvolvedores de código aberto a participarem da construção do sistema aberto com base no *Suse Professional Edition*, esta é a versão comercial do *Suse Linux*. O *Suse Linux 10* pode ser baixado no sítio do projeto *Open Suse*.

4.3.1 – Configurando o Endereçamento IPv4

Para a configuração do IPv4 no *Suse Linux 10* foi utilizada a facilidade do DHCP (*Dynamic Host Configuration Protocol*), usando como servidor de DHCP o "*Window 2003 Server*".

Para que o *Suse Linux 10* se conecte a rede é necessário configurar o arquivo "**ifcfg-eth-id-00:08:02:9e:36:9d**", este nome de arquivo foge um pouco da nomeclatura usual das distribuições *GNU/Linux*, que é "**ifcfg-ethx**" onde "**x**" é o número da interface de rede no sistema. Já no *Suse Linux 10* é utilizado um número de identificação, que é o endereço MAC da placa de rede.

Arquivo: **ifcfg-eth-id-00:08:02:9e:36:9d**

```
BOOTPROTO='dhcp'              # Configurações de rede usando um
                              # DHCP Server
BROADCAST=''
IPADDR=''
MTU=''
NAME='Compaq RTL-8139/8139C/8139C+'    # Nome da placa de
                              # rede
NETMASK=''
NETWORK=''
REMOTE_IPADDR=''
STARTMODE='auto'
UNIQUE='rBUF.ZR5JW9ZgVz7'
USERCONTROL='no'
_nm_name='bus-pci-0000:00:0b.0'
```

4.3.2 – Configurando o Endereçamento IPv6

No *Suse Linux 10* não há o arquivo "**rc.local**", este foi alterado para o arquivo "**boot.local**" que fica na pasta "**/etc/rc.d**", foi realizada a tentativa de configurar este arquivo para que no momento da inicialização do sistema, fosse realizada a configuração de endereçamento IPv6 automática, porém não foi possível, pois o *Suse Linux 10* instalado não estava lendo este arquivo, assim a configuração do IPv6 foi realizada manualmente após a inicialização do sistema e por "*stateless address autoconfiguration*" [RFC2462].

A configuração manual consiste em executar o comando "**ifconfig**".

ifconfig eth0 inet6 add FEC0:2006:6::12/64

E a configuração por "*stateless address autoconfiguration*" consiste no *Gateway* da rede divulgar seu prefixo de rede IPv6 e os *hosts* que ingressarem nesta rede se autoconfigurarem com o prefixo mais seu endereço MAC, formando assim um endereço IPv6 válido para rede.

4.3.3 – Resultados das Configurações

Os resultados das configurações realizadas no *Suse Linux 10* dependem das configurações realizadas no DHCP configurado no *Windows 2003 Server*, seção "**4.6.3 – Configurando um Servidor DHCP no Windows 2003**

Server", e nas configurações realizadas no *FreeBSD*, seção "**4.7 – FreeBSD**". No entanto, há também configuração realizada manualmente do IPv6, o que faz o *Suse Linux 10* poder acessar a rede IPv6.

Como foram criadas duas topologias, mostrado na seção "3.5 – Topologias", usando o *Suse Linux 10*, será mostrado aqui o resultado de uma das topologias que equivale as duas, o que muda são os endereços.

A Figura 4.8 mostra as configurações realizadas. O comando "**ifconfig eth0 add FEC0:2006:6::15/64**", como mencionado anteriormente acrescenta o endereço IPv6.

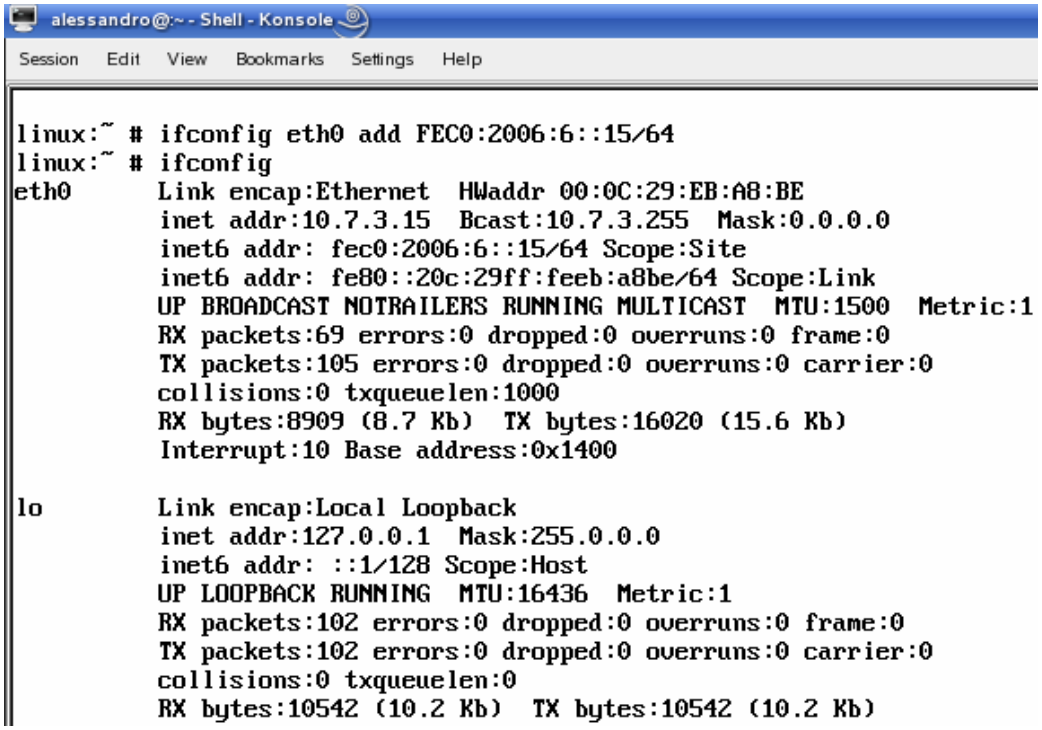

Figura 4.8 – Configuração do *Suse Linux 10*

Com esta configuração o *Suse Linux 10* pode acessar o serviço de DNS e HTTP, configurados no *Red Hat 9*.

4.4 – USANDO O GATEKEEPER GNUGK

O *Gatekeeper GnuGK* foi configurado no *Red Hat 9*, pois este *GNU/Linux* é estável e bastante funcional quando se trata de prover serviços de redes.

4.4.1 – Instalando Bibliotecas e Configurando o GnuGK

Para configurar o *GnuGK* é necessário baixá-lo antes no sítio do projeto, "*Gnu Gatekeeper*" (http://www.gnugk.org/h323download.html). Após baixá-lo, deve-se seguir os passos descritos nas seções seguintes para configurá-lo.

4.4.1.1 – Instalando as Bibliotecas

Antes de configurar o *GnuGK* é necessário a instalação das bibliotecas PWLib e OpenH323, estas bibliotecas podem ser baixadas no sítio http://rpmfind.net. Foram usados os pacotes "RPM" (*Red Hat Package Manager*) para instalação das bibliotecas necessárias. Os passo seguintes mostram a instalação da PWLib e da OpenH323.

1. Deve-se efetuar login no "*Red Hat 9*" como "*root*" e abrir um *terminal* no *Gnome* (*terminal* - onde pode ser executados comandos em modo texto no *Linux* e *Gnome* - interface gráfica do *Red Hat 9* utilizado) e ir em "*Run Program...*", aparecerá uma janela onde se deve digitar "**gnome-terminal**" conforme mostra a Figura 4.9**.** Depois clique em "***Run*****".**

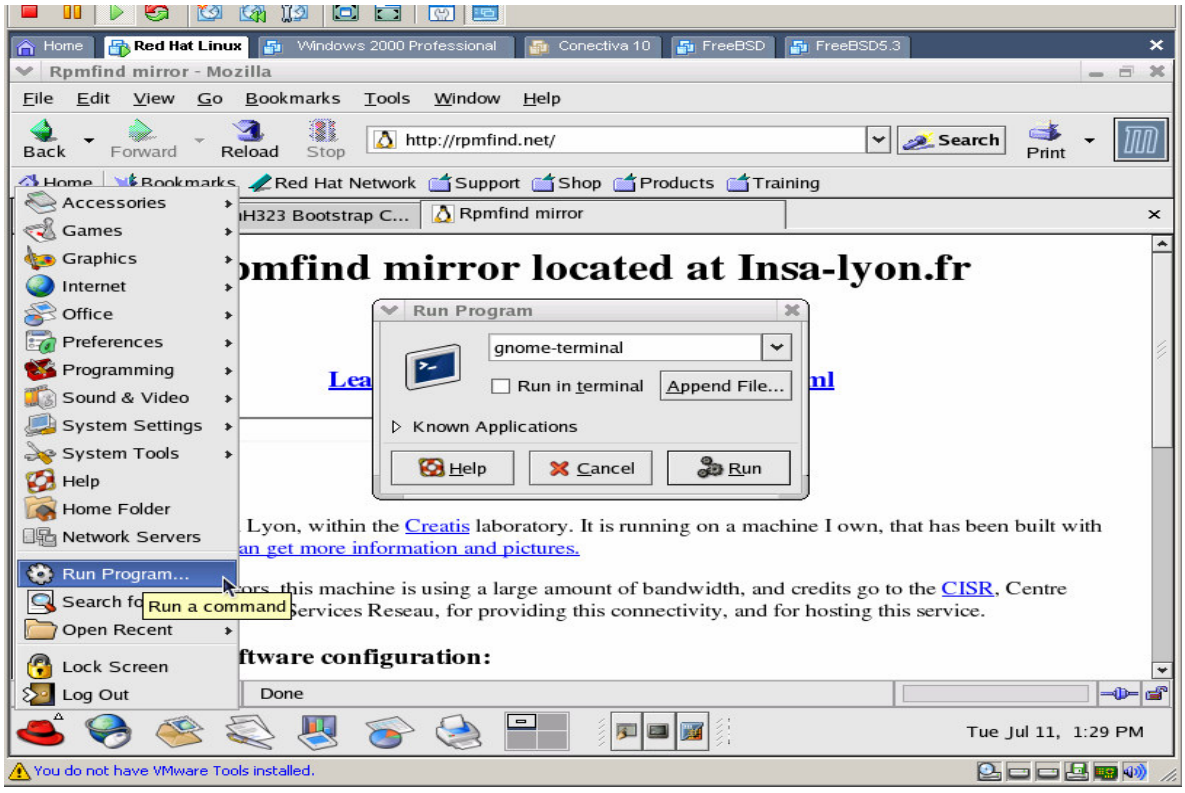

Figura 4.9 – Rodando o gnome-terminal

2. Aparecerá a tela de terminal (**Figura 4.10**)

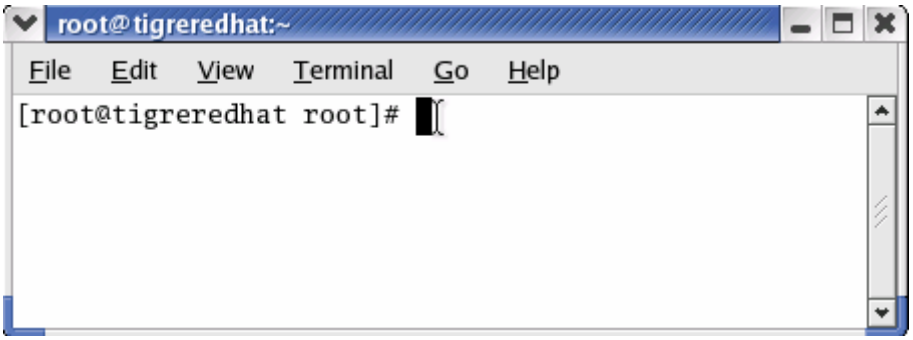

Figura 4.10 – Tela do gnome-terminal

3. Então se executa o comando "**ls**" para checar se os pacotes necessários estão no diretório (**Figura 4.11**)

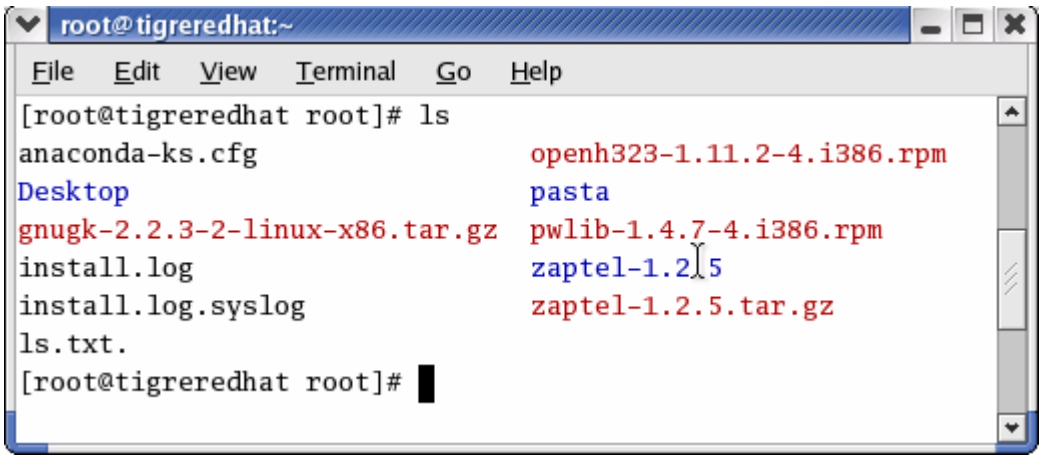

Figura 4.11 – Executando o "ls" e verificando se os pacotes necessários para instalação estão no diretório

4. Para instalar os pacotes se executa o comando "**rpm -ivh nome_do_pacote**" que no caso serão (**Figura 4.12**):

rpm -ivh pwlib-1.4.7-4.i386.rpm; e

rpm -ivh openh323-1.11.2-4.i386.rpm.

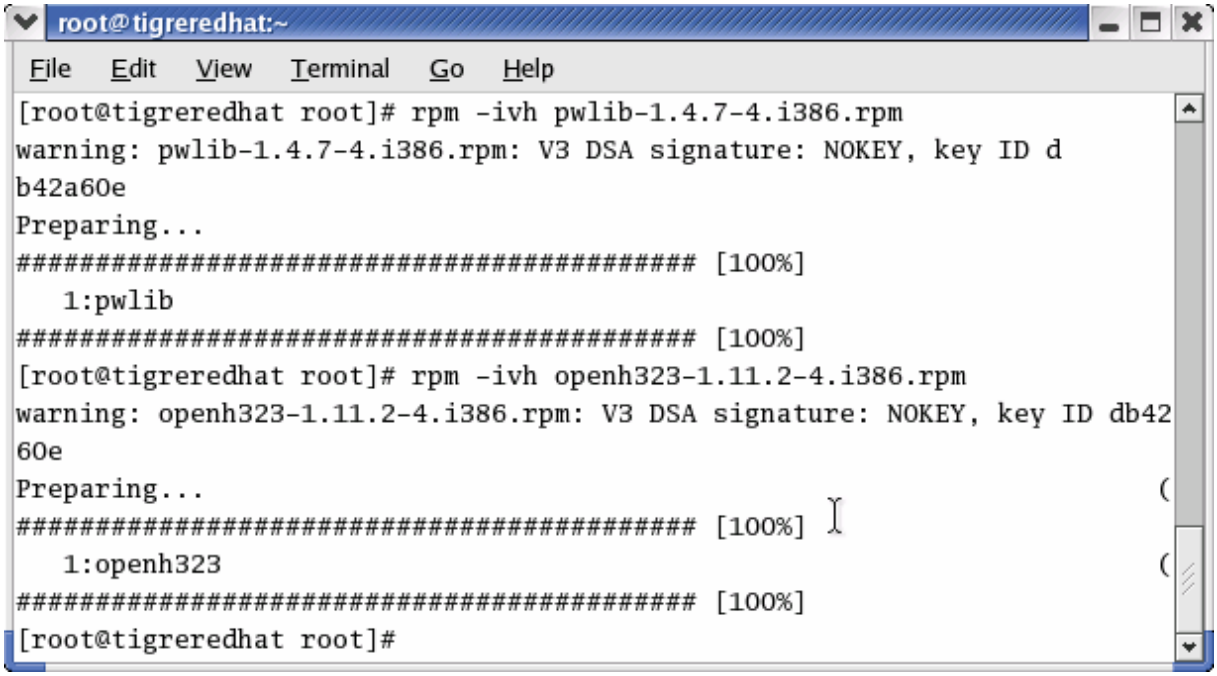

Figura 4.12 – Executando o comando "rpm"

Na próxima seção é mostrado como instalar e configurar o *GnuGK*.

4.4.1.2 – Instalando e Configurando o GnuGK

Não existe nenhum procedimento especial para a instalação do *GnuGK*, simplesmente é necessário descompactar o executável no diretório desejado e criar um arquivo de configuração para ele.

Foi baixado o arquivo "***gnugk-2.2.3-2-linux-x86.tar.gz***" e descompactado com o comando "**tar -xzvf gnugk-2.2.3-2-linux-x86.tar.gz**" no diretório "/**opt**". No *Linux* o diretório "/**opt**" é onde ficam instalados os aplicativos que não vêm junto com a distribuição.

Para configurar o *GnuGK* é preciso criar e editar o arquivo "*gnugk.ini*". Para criar o arquivo "gnugk.ini" pode-se executar o comando "**addpasswd gnugk.ini GkStatus::Auth gkadmin senha**" e para criar os usuários executa-se o comando "**addpasswd gnugk.ini Password usuario senha**". Estes comandos devem ser executados dentro do diretório base do *GnuGK*. O arquivo de configuração completo do *GnuGK* está no Anexo E – Arquivo de configuração GnuGK.

Para iniciar o *GnuGK* é necessário executar o comando "**gnugk -c gnugk.ini -ttt**"

4.4.2 – Resultados das Configurações

Com as configurações do *GnuGK* e posteriormente as configurações do *NetMeeting* realizadas, seção "**4.6.6.2 – Iniciando uma Chamada com NetMeeting usando um Gatekeeper**", o resultado desta configuração foi a comunicação entre os clientes *NetMeeting* usando como servidor o *GnuGK*

A comunicação sobre o IPv6 neste cenário não foi possível pelo fato do *GnuGK* e o *NetMeeting* não darem suporte ao IPv6.

4.5 – CONFIGURANDO O GNOMEMEETING

4.5.1 – Iniciando uma Chamada com o GnomeMeeting de Cliente para Cliente

Como o *GnomeMeeting* suporta tanto o IPv6 como o IPv4, serão mostradas as duas maneiras de realizar as chamadas, pois o que muda é somente o endereçamento IP.

Para iniciar uma chamada usando o IPv6 com *GnomeMeeting* deve-se colocar o endereço IPv6 do *host* de destino no campo de endereço conforme mostra a Figura 4.13. Observe que o endereço IPv6 deve estar entre colchetes.

Figura 4.13 – Realizando chamada usando o IPv6 com o *GnomeMeeting*

E para iniciar uma chamada usando o IPv4 com *GnomeMeeting* deve-se colocar no lugar do endereço IPv6, o endereço IPv4 do *host* de destino no campo de endereço conforme mostra a Figura 4.14.

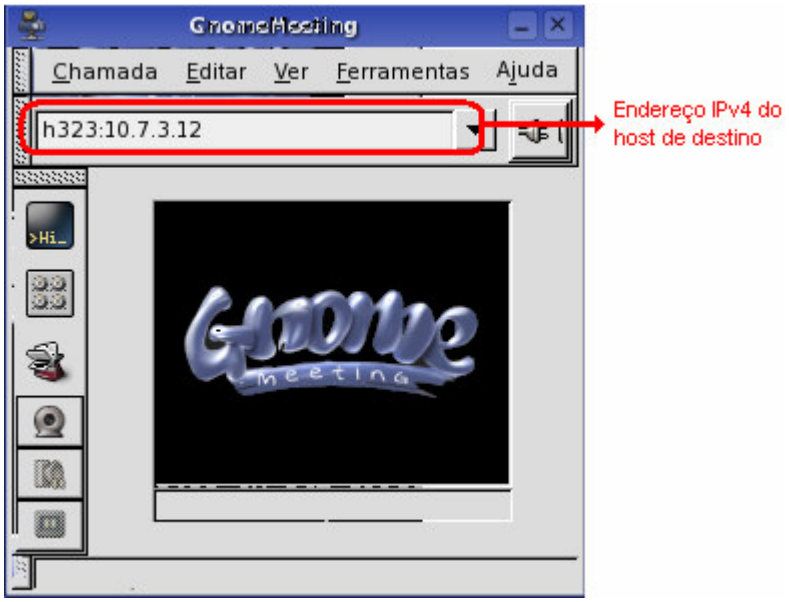

Figura 4.14 – Realizando chamada usando o IPv4 com o *GnomeMeeting*

Desta forma pode-se realizar a comunicação de uma origem com outro destino usando as arquiteturas IPv4 e IPv6.

4.5.2 – Resultados das Configurações

Neste cenário foi possível a comunicação entre os clientes *GnomeMeeting* usando os protocolos IPv4 e IPv6, o que comprova que independente do protocolo utilizado a VoIP irá funcionar se as aplicações derem suporte ao dois protocolos.

4.6 – MICROSOFT WINDOWS

4.6.1 – Configurando o Endereçamento IPv4 no Windows 2003 Server

Pode-se configurar de duas formas o endereço IPv4 no *Windows*, por DHCP e manualmente (endereço IP fixo). Por DHCP é necessário um servidor DHCP, como o *host* (*Windows 2003 Server*) em questão é o próprio servidor DHCP,

este será configurado manualmente, o que consiste em digitar as configurações com as informações necessárias.

Para configurar o endereço IPv4 no *Windows Server 2003* clique em: “Iniciar –> Painel de Controle”, aparecerá a janela do “Painel de Controle” onde se deve dar um clique duplo no ícone “Conexões de Rede”, e então seleciona-se a conexão desejada, no caso a “Rede Interna” (nome dado a conexão), aparecerá a janela “Propriedades da Rede Interna”, onde deve-se selecionar “Protocolo TCP/IP” e clicar no botão “Propriedades”, aparecerá a janela “Propriedades de Protocolo TCP/IP” onde será feita a configuração conforme a Figura 4.15. Todas as telas dos passos descritos aqui estão no “Anexo F – Telas com os passos de configuração do endereço IPv4 *no Windows 2003 Server*.”

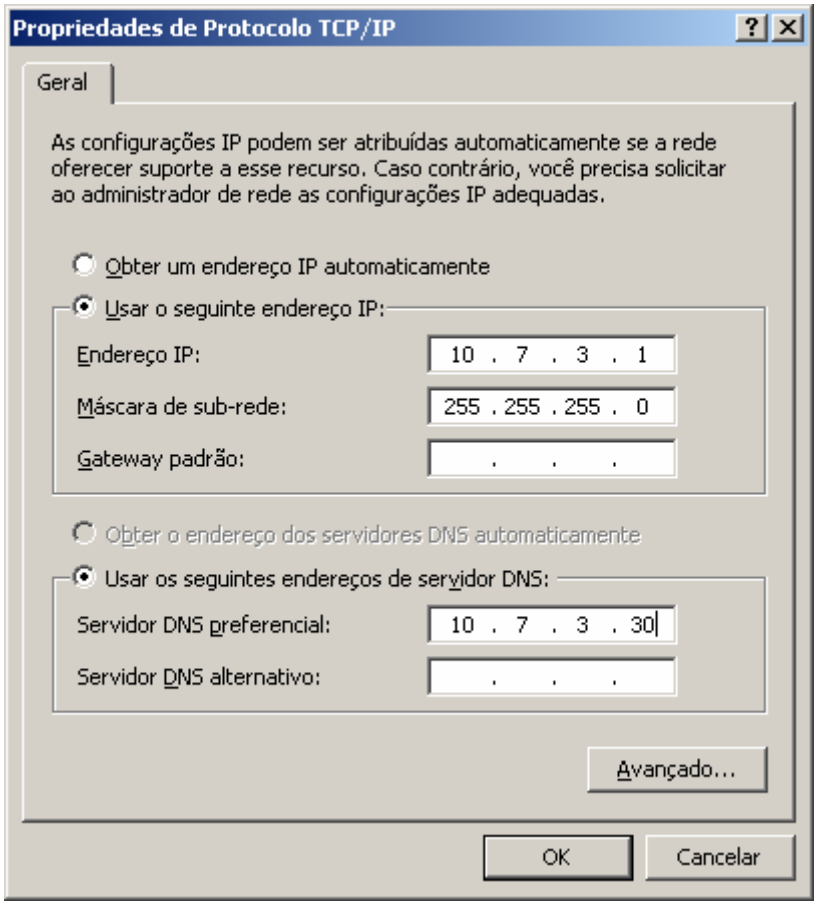

Figura 4.15 – Propriedades de Protocolo TCP/IP

Endereço IP: 10.7.3.1

Máscara de sub-rede: 255.255.255.0
Gateway padrão: o próprio servidor.
Servidor de DNS preferencial: 10.7.3.30

4.6.2 – Configurando o Endereçamento IPv6 no Windows 2003 Server

Não será configurado o endereçamento IPv6 para o *Windows 2003 Server* conforme explicado anteriormente na seção "**3.7.2 – IPv6 no Windows**".

4.6.3 – Configurando um Servidor DHCP no Windows 2003 Server

O *Windows 2003 Server* provê vários serviços de redes, entre eles o DHCP. Para instalar o DHCP é necessário o CD (*Compact Disk*) de instalação do *Windows 2003 Server*.

Os passos seguintes mostram como instalar o DHCP.

1. Abrir a janela de "Adicionar ou remover programas" (Figura 4.16).

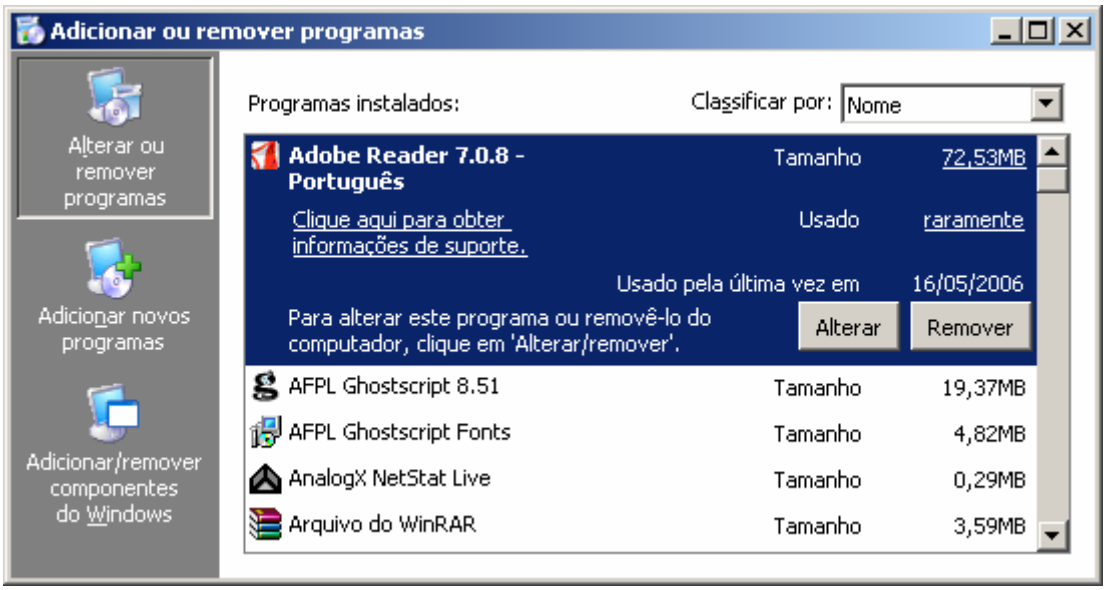

Figura 4.16 – Janela: Adicionar ou remover programas

2. Clique em "Adicionar/remover componentes do Windows"

3. E aparecerá a janela da Figura 4.17

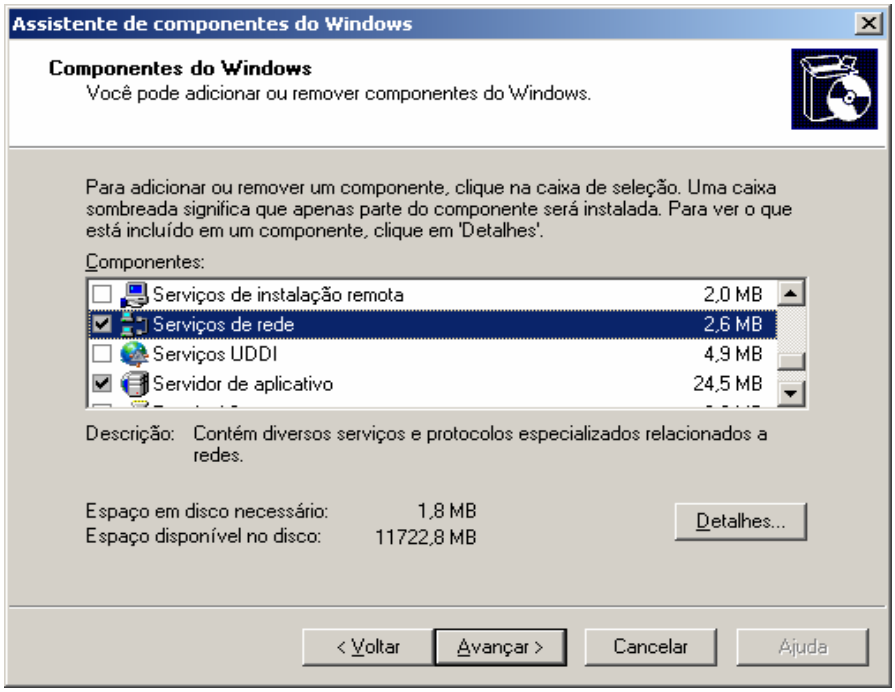

Figura 4.17 – Janela: Assistente de componentes do Windows

4. Deve-se, então, selecionar “Serviços de rede” e clicar em detalhes.

5. Selecione na janela “Serviços de rede” o “Protocolo de configuração dinâmica de *hosts* (DHCP)” e clique em “Ok”

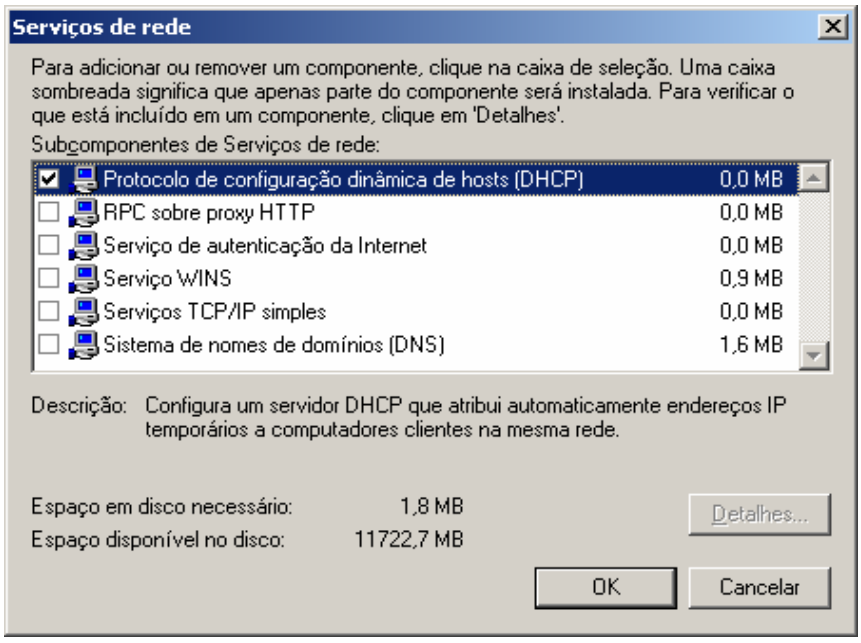

Figura 4.18 – Janela: Serviços de rede

6. Na janela “Assistente de componentes do Windows” clique em “Avançar” e estará terminada a instalação do servidor de DHCP.

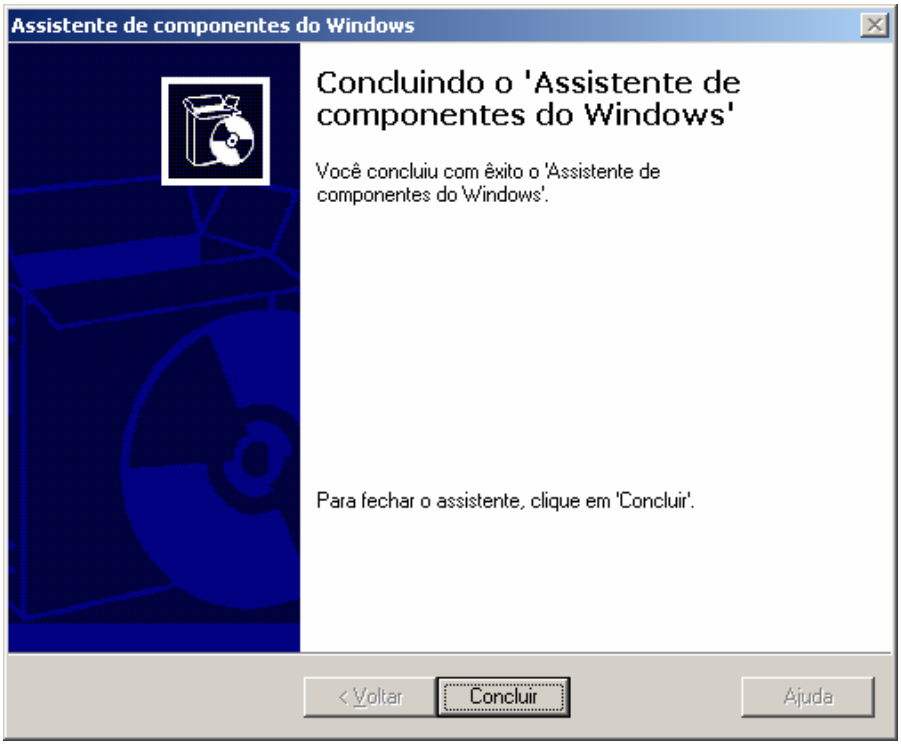

Figura 4.19 – Janela: finalizando a instalação

Após realizar a instalação é necessário configurar o servidor.

Para configurar o servidor DHCP é necessário seguir os seguintes passos:

1. Ir em “Iniciar –> Executar... e digitar “**dhcpmgmt.msc**” e pressionar “***Enter***” (Figura 4.20)

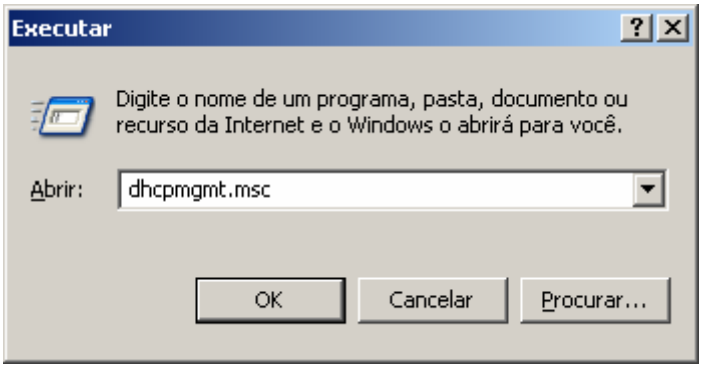

Figura 4.20 – Janela: Executar

2. Aparecerá a janela de configuração do servidor de DHCP (Figura 4.21)

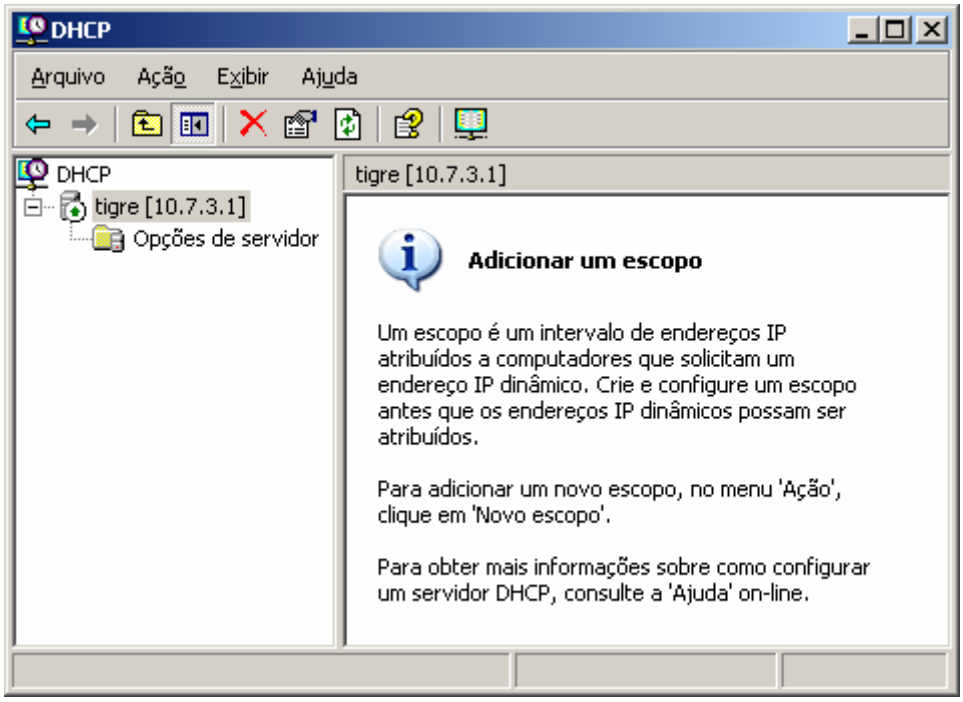

Figura 4.21 – Janela: DHCP

3. Então, clique no servidor com o botão direito e depois em Novo escopo... (Figura 4.22)

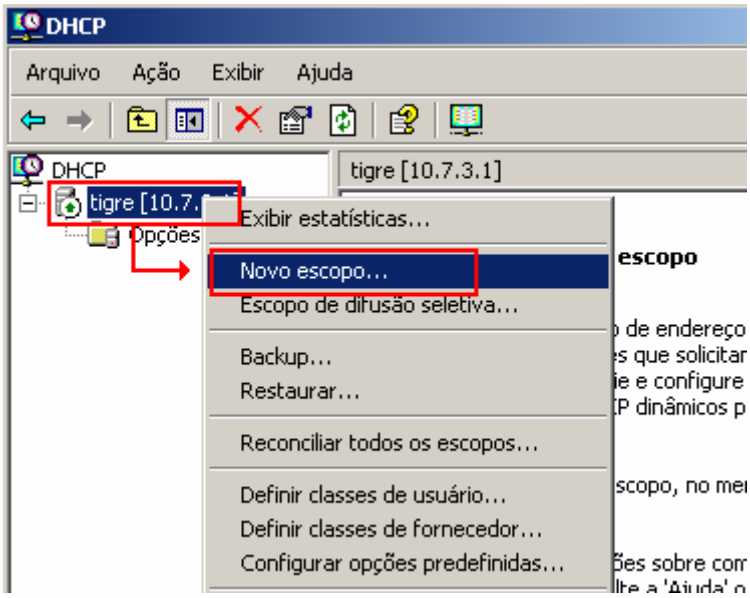

Figura 4.22 – Janela: Novo escopo...

Após ter selecionado a opção “Novo escopo...”, as telas dos próximos passos da configuração estarão no Anexo I deste trabalho, pois são várias telas e que se colocadas aqui podem deixar o texto sobrecarregado, então será descrito aqui as opções a serem utilizadas. As opções são mostradas na Tabela 4.1.

Tabela 4.1 – Opções do servidor DHCP

Nome	csnet
Descrição	Escopo DHCP
Endereço IP inicial	10.7.3.1
Endereço IP final	10.7.3.254
Comprimento	24
Máscara de sub-rede	255.255.255.0
Intervalo de endereços Excluídos	10.7.3.1 a 10.7.3.10
Duração da concessão	90 dias
Roteador (*gateway* padrão)	10.7.3.1
Domínio pai	csnet.rede
Servidor DNS	10.7.3.30
Servidor WINS	-
Ativar escopo	Sim

4.6.4 – Configurando o Endereçamento IPv4 no Windows XP Professional

No *Windows XP Professional* a configuração será realizada por um servidor DHCP, então seguindo os mesmos passos para a configuração do *Windows Server 2003*, no *Windows XP Professional* a configuração deverá ficar conforme a Figura 4.23, ou seja, marcar a opção “Obter um endereço IP automaticamente”.

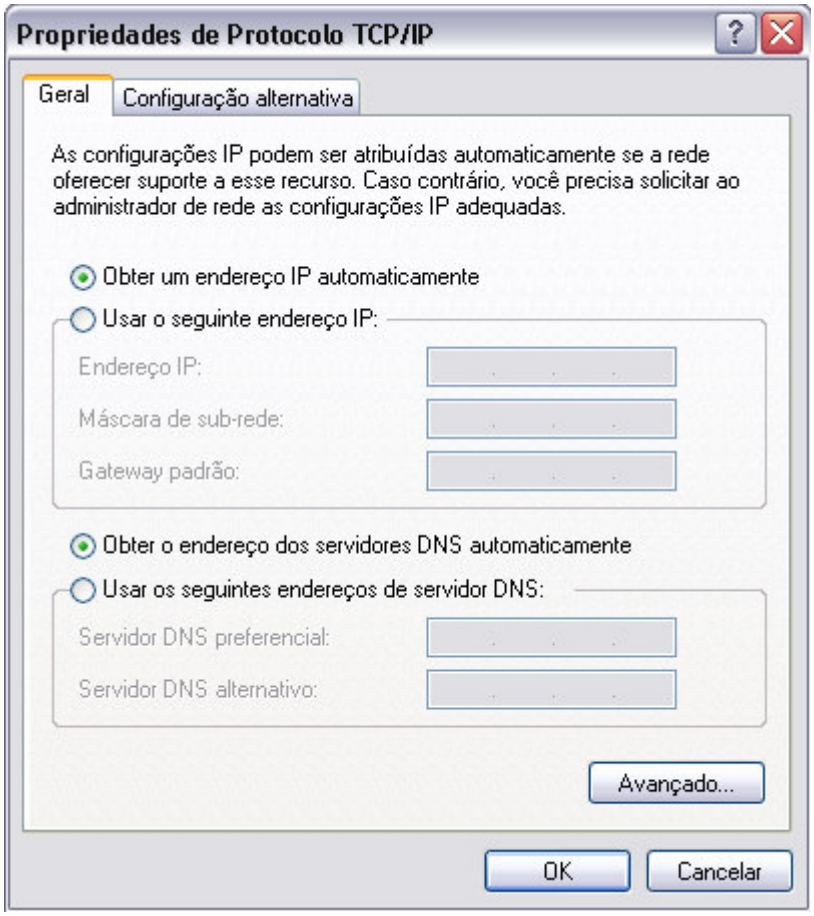

Figura 4.23 – Propriedades de Protocolo TCP/IP

4.6.5 – Configurando o Endereçamento IPv6 no Windows XP Professional

Não será configurado o endereçamento IPv6 para o *Windows XP Professional* conforme explicado na seção "**3.7.2 – IPv6 no Windows**".

4.6.6 – NetMeeting

A configuração do *NetMeeting* explicada aqui é válida tanto para o *Windows 2003 Server* como para o *Windows XP Professional*.

Para configurar o *NetMeeting* é preciso iniciá-lo em "**C:\Arquivos de programas\NetMeeting\conf.exe**", onde será iniciado um assistente de configuração, as telas de configuração podem ser vistas no "Anexo G – Telas de configuração do NetMeeting". Após a configuração, para iniciar o *NetMeeting* dê um clique duplo no ícone criado na área de trabalho como mostra a Figura 4.24.

Figura 4.24 – Iniciando o NetMeeting após a configuração inicial.

4.6.6.1 – Iniciando uma Chamada com NetMeeting de Cliente para Cliente

Para realizar chamadas com o *NetMeeting* de um cliente para um cliente é preciso apenas digitar o endereço IP ou nome do *host* da pessoa com quem quer se falar como mostra a Figura 4.25.

Figura 4.25 – Iniciando uma chamada

4.6.6.2 – Iniciando uma Chamada com NetMeeting usando um Gatekeeper

Para realizar uma chamada usando um *Gatekeeper* é preciso primeiramente realizar as configurações de *Gatekeeper* e nome de conta:

- *Gatekeeper* – IP ou nome do servidor *Gatekeeper*.
- Nome da conta – nome da conta cadastrada no *Gatekeeper*.

Veja a janela de configuração na Figura 4.26.

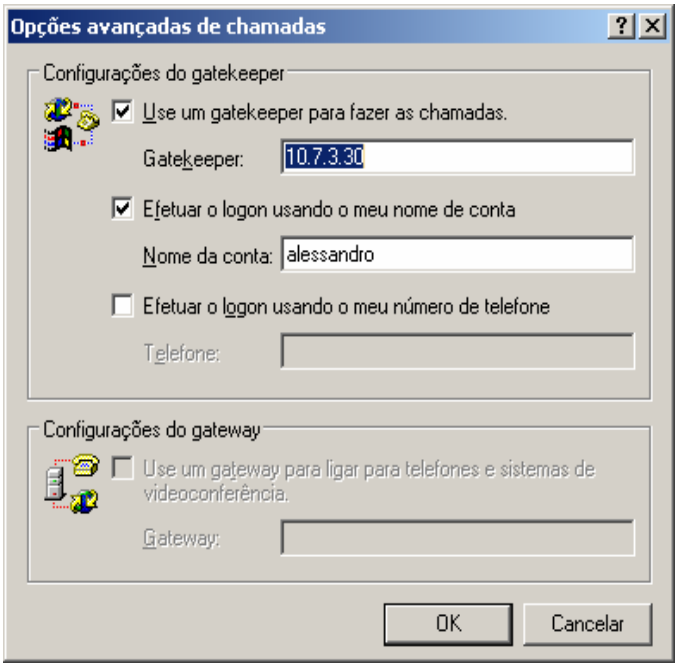

Figura 4.26 – Configuração do *Gatekeeper*

Para realizar as chamadas agora é preciso simplesmente saber o nome de conta cadastrada no *Gatekeeper* da pessoa com quem quer se falar e realizar a chamada digitando o nome desta pessoa. Veja a Figura 4.27.

Figura 4.27 – Realizando chamada usando o *Gatekeeper*

4.6.7 – Resultados das Configurações

Com a configuração de endereçamento no *Windows 2003 Server* e *Windows XP Professional*, ambos ficaram utilizando o IPv4, o que permite a comunicação com os sistemas *GNU/Linux*, pois como estes trabalham sobre as duas pilhas de protocolos (IPv4 e IPv6) a comunicação é possível. Então, como resultado se tem que os *Windows* podem acessar os serviços de DNS, HTTP e o *Gatekeeper*, provido pelo servidor *Red Hat 9* usando o protocolo IPv4. Com a configuração do DHCP no *Windows 2003 Server*, foi possível que o *Windows XP Professional* e o *Suse Linux 10* utilizassem as configurações de rede feita dinamicamente através do serviço de DHCP.

4.7 – FREEBSD

O *FreeBSD* pode ser baixado gratuitamente no sítio de seu projeto denominado "*The FreeBSD Project*". Existem várias versões, e a que foi utilizada neste trabalho foi a versão *5.4-RELEASE* com as ferramentas do projeto "*KAME*" [KAME, 1998].

4.7.1 – Configurando o Endereçamento IPv4

A configuração do endereçamento IPv4 no *FreeBSD* é relativamente fácil, pois por ser um sistema variante do *Unix*, este tem as configurações semelhantes às feitas no *Gnu/Linux*. Para configurar endereçamento IPv4 é necessário editar o arquivo "**rc.conf**" que fica no diretório "/**etc**".

Segue as configurações realizadas no arquivo "**rc.conf**" com os comentários:

Arquivo: **rc.conf**

```
hostname="free.csnet.rede"        # Nome do host
ifconfig_lnc0="inet 10.7.3.61 netmask 0xffffff00"  # Endereço IPv4
defaultrouter="10.7.3.1"          # Gateway padrão
```

4.7.2 – Configurando o Endereçamento IPv6

Para configurar o endereçamento IPv6 do *FreeBSD* foi compilado o *Kernel* do sistema operacional com as opções relativas ao IPv6 e ferramentas de redes necessárias ao trabalho do IPv6 em redes IPv6.

Segue os passos necessário para compilar o *Kernel* no *FreeBSD*:

1. Baixar o arquivo "**kame-20060522-freebsd54-snap.tgz**" do sítio ftp://ftp.kame.net/pub/kame/snap

2. Descompactar o arquivo "**kame-20060522-freebsd54-snap.tgz**" no diretório "/**usr**/**src**"

   cd /usr/src

   tar -xzvf kame-20060522-freebsd54-snap.tgz

3. Ir para o diretório "/**usr**/**src**/**kame**".

   cd /usr/src/kame

4. Criar os atalhos necessários para compilar o Kernel

make TARGET=freebsd5 prepare

5. Ir para o diretório "**/usr/src/kame/freebsd5/sys/i386/conf**"

   cd /usr/src/kame/freebsd5/sys/i386/conf

   Neste diretório fica o arquivo de configuração das opções do *Kernel* a ser compilado.

6. Editar o arquivo "**NOVOKERNEL**"

7. Criar os binários do *Kernel*

   /usr/sbin/config NOVOKERNEL

8. Mudar para o diretório dos binários do novo *Kernel*

   cd ../compile/NOVOKERNEL

9. Compilar o novo *Kernel*

   make depend
   make

10. Instalar o novo *Kernel*

    make install

Passos para instalar as ferramentas necessárias, como por exemplo, a ferramenta **ping6**:

1. Ir para o diretório "**/usr/src/kame/freebsd5**"

   cd /usr/src/kame/freebsd5

2. Executar os seguintes comandos para instalar as ferramentas

   make includes
   make install-includes

make

make install

3. Agora basta reiniciar o *FreeBSD*

fastboot

Após compilar e instalar o *Kernel* e ferramentas, será necessário configurar o arquivo "**rc.conf**" para que o IPv6 funcione.

Segue as configurações realizadas no arquivo "**rc.conf**".

Arquivo: **rc.conf**

```
# /etc/rc.conf
hostname="free.csnet.rede"      # Nome do host

ifconfig_lnc0="inet 10.7.3.61 netmask 0xffffff00" # Endereço IPv4

defaultrouter="10.7.3.1"        # Gateway padrão

ifconfig lnc1 inet6 FEC0:2006:6::61/64  # Endereço IPv6

ipv6_enable="YES"               # Habilita o IPv6
ipv6_network_interfaces="lnc1" # Interface configurada com IPv6
ipv6_gateway_enable="YES"   # "YES" o host é um gateway, "NO" o
                            # host não é gateway
ipv6_prefix_lnc1="FEC0:2006:6" # Prefixo da autoconfiguração
rtadvd_enable="YES"             # "YES" habilita um router IPv6
rtadvd_interfaces="lnc1"

gateway_enable="YES"
inetd_enable="NO"
keymap="br275.iso.acc"
linux_enable="YES"
moused_enable="YES"
sshd_enable="YES"
usbd_enable="YES"               # Habilita a porta USB

# Adicionar novos caminhos à variável PATH
PATH=/usr/local/v6/sbin:/usr/local/v6/bin:${PATH}
```

4.7.3 – Resultados das configurações

O resultado da configuração do *FreeBSD* pode ser visto na Figura 4.28, onde foi executado o comando "**ifconfig**" no *Red Hat 9*, pode se ver no retângulo vermelho o endereço IPv6 "**FEC0:2006:6:0:20C:29FF:FE76:B04C**", este endereço foi atribuído dinamicamente juntando-se o prefixo "**FEC0:2006:6**" configurado no *FreeBSD* e o endereço MAC da placa de rede, neste caso o MAC virtual "**00:0C:29:76:B0:4C**" do *Red Hat 9*, nota-se no entanto, que não é o endereço MAC literalmente dizendo, pois existe um padrão convencionado pelo IEEE (Anexo A – Identificadores de Interface) onde é definido o formato do endereço IPv6 para placas de redes com o MAC de 48 bits.

```
[root@tigreredhat root]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:76:B0:4C
          inet addr:10.7.3.30  Bcast:10.7.3.255  Mask:255.255.255.0
          inet6 addr: fec0:2006:6:0:20c:29ff:fe76:b04c/64 Scope:Site
          inet6 addr: fec0:2006:6::30/64 Scope:Site
          inet6 addr: fe80::20c:29ff:fe76:b04c/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:81 errors:0 dropped:0 overruns:0 frame:0
          TX packets:43 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:11443 (11.1 Kb)  TX bytes:3332 (3.2 Kb)
          Interrupt:10 Base address:0x1080
```

Figura 4.28 – Endereço IPv6 atribuído dinamicamente pelo *FreeBSD*

4.8 – RESULTADOS OBTIDOS

O resultado obtido neste trabalho foi a implementação de uma infra-estrutura IPv4 e IPv6, onde os *hosts* se comunicaram tanto através de um servidor HTTP, DNS e DHCP, como em uma tecnologia mais complexa que é a VoIP. A configuração realizada em cada sistema tornou viável esta comunicação. Os clientes VoIP conseguiram transmitir o tráfego de voz usando os protocolos IPv4 e IPv6, a QoS ficou um tanto quanto comprometida, devido ao fato da utilização das Máquinas Virtuais, que causou certo retardo na comunicação VoIP, porém, o uso das Máquinas Virtuais foi de extremo valor, pois estas facilitaram a configuração em cada sistema, permitindo até mesmo a cópia de segurança da Máquina Virtual e no

caso de um erro de configuração, sendo possível recuperar a configuração anterior. Além do pouco espaço utilizado e no transporte dos equipamentos físicos.

Capítulo 5 – Conclusão

Este trabalho procurou evidenciar a importância do novo protocolo IPv6 com a implementação junto ao protocolo IPv4, mostrando as duas estruturas e as vantagens de utilizar o IPv6.

Foi mostrado que apesar de apresentar os conceitos estruturais do IPv4, o que é de certa forma lógico, mudar totalmente para um novo protocolo sem dar continuidade ao que já é largamente usado pode ser traumático. O IPv6 é a continuação do IPv4, porém, também é um protocolo renovado, contendo características e funcionalidades novas. Num futuro próximo o IPv4 não será capaz de cobrir com eficiência as dimensões alcançadas pela *Internet*, nem garantir novos serviços que utilizem transmissão multimídia, como vídeo sob demanda, vídeo-conferência, telefonia IP e transmissões de TV, então, outra preocupação deste trabalho foi mostrar o serviço de VoIP funcionando com os dois protocolos, e que ainda é preciso bastante esforço por parte dos fabricantes de software na fabricação de softwares que dêem suporte para o IPv4 e IPv6 simultaneamente, pois continuar com o IPv4 é preciso devido a alta gama de equipamentos que foram construídos para este protocolo, mas também, é necessário a adoção do IPv6, pois como foi mostrado, este dispõe de uma quantidade maior de endereços, maior segurança, maior performance e QoS.

Com a implementação do trabalho e os passos descritos, foi mostrado que é possível os dois protocolos trabalharem simultaneamente, desde que haja o ambiente para isto e que as aplicações dêem suporte para os dois protocolos. A falta de aplicações com suporte ao IPv6 causou a dificuldade de implementar este trabalho, mas com a pesquisa intensa na *Internet* pela procura por tecnologias, foram obtidos sistemas operacionais e o *softphone* que suportassem o IPv4 e IPv6.

Pode-se dizer que o objetivo do trabalho foi alcançado, pois foi mostrado que a comunicação em redes dos *hosts* usando o IPv4 e IPv6 é possível e que o serviço de VoIP funciona na estrutura proposta.

5.1 – TRABALHOS FUTUROS

Como trabalho futuro é recomendado à criação de mais *softwares* com suporte ao IPv6, e como os ambientes de IPv4 e IPv6 terão que conviverem juntos por um bom tempo, a recomendação é que se desenvolva *software* com o suporte para os dois protocolos.

Existe também a possibilidade da ampliação deste trabalho com a implementação de um firewall com suporte ao IPv4 e ao IPv6 e a criação de outros ambientes com implementação de um roteador com suporte ao NAT-PT (*Network Address Translator – Protocol Translator*) o que pode fazer com que uma ambiente puramente IPv4 se comunique com um ambiente puramente IPv6.

Referência Bibliográfica

[ALECRIM, 2003] ALECRIM, Emerson. **O que é Linux**. Disponível em: http://www.infowester.com/linux1.php, 30 de junho de 2003, último acesso em 22/05/2006.

[AZEREDO, 2006] – AZEREDO, Patrícia. **IPv6: mudanças na Internet e no seu trabalho**. Disponível em: http://www.timaster.com.br/revista/materias/main_materia.asp?codigo=1116&pag=2, 18 de abril de 2006, último acesso em 01/06/2006.

[CHIN, 1998] – CHIN, Liou Kuo. **Rede Privada Virtual – VPN**: RNP – Rede Nacional de Ensino e Pesquisa. Disponível em: http://www.rnp.br/newsgen/9811/vpn.html, 13 de novembro de 1998, último acesso em 20/05/2006.

[COLCHER et al, 2005] – COLCHER, Sérgio; GOMES, Antônio Tadeu Azevedo; SILVA, Anderson Oliveira da; FILHO, Guido Lemos de Souza; SOARES, Luiz Fernando Gomes. **VoIP: Voz sobre IP**. Ed. Campus, Rio de Janeiro, 2005.

[COMER, 1998] – COMER, Douglas E. **Interligação em Rede com TCP/IP - Volume I**. 1º Edição. Ed. Campus, Rio de Janeiro, 1998.

[FERNANDO, 1999] – FERNANDO, Nelson Luiz Leal Fernandes. **Voz Sobre IP: Uma Visão Geral**. Disponível em: http://www.ravel.ufrj.br/arquivosPublicacoes/nelson_voip.pdf, último acesso em 02/05/2006.

[GNU, 1996] – GNU. **O Sistema Operacional GNU**. Disponível em: http://www.gnu.org/home.pt.html, 1996, último acesso em 22/05/2006.

[HUSTON, 2006] – HUSTON, Geoff. **IPv4 Address Report**. Disponível em: http://bgp.potaroo.net/ipv4/, 24 de maio de 2006, último acesso em 26/05/2006.

[IEEE, 2005] – IEEE - Institute of Electrical and Electronic Engineers. **GUIDELINES FOR 64-BIT GLOBAL IDENTIFIER (EUI-64) REGISTRATION AUTHORITY**: IEEE - Institute of Electrical and Electronic Engineers. Disponível em:

http://standards.ieee.org/regauth/oui/tutorials/EUI64.html, último acesso em 05/06/2006.

[IPV6 DO BRASIL, 2005] – IPv6 do Brasil. **IPv6 – A Próxima Geração da Internet**: IPv6 do Brasil. Disponível em: http://www.ipv6dobrasil.com.br/index.php?id_pagina=39, 2005, último acesso em 20/05/2006.

[MORALLES, 2004] – MORALLES, Diego. **VoIP. Um dia você vai ter um telefone assim**: UOL - Web Insider. Disponível em: http://webinsider.uol.com.br/vernoticia.php/id/2291, 06 de dezembro de 2004, último acesso em 07/06/2006.

[NED, 1998] – NED , Frank. **A Nova Geração de Protocolos IP**: RNP – Rede Nacional de Ensino e Pesquisa. Disponível em: http://www.rnp.br/newsgen/9811/intr-ipv6.html#ng-enderecamento, último acesso em 18/05/2006.

[OLIVEIRA, 2005] OLIVEIRA, Sérgio Ricardo de Freitas. **Solução Corporativa VoIP: Caso Prático**. Disponível em: http://www.teleco.com.br/tutoriais/tutorialcorpvoip2/pagina_3.asp, 22 de agosto de 2005, último acesso em 11/05/2006.

[KAME, 1998] The KAME project. Disponível em: http://www.kame.net, último acesso em 01/07/2006.

[RFC 0791, 1981] – DARPA - Defense Advanced Research Projects Agency. **Internet Protocol**. Disponível em: http://www.ietf.org/rfc/rfc0791.txt, último acesso em 01/07/2006.

[RFC 0820, 1983] – POSTEL, J. Assigned Numbers. Disponível em: http://www.faqs.org/rfcs/rfc820.html, 1983, último acesso em 01/06/2006.

[RFC 1365, 1992] – SIYAN, K. **An IP Address Extension Proposal**. Disponível em: http://www.isi.edu/in-notes/rfc1365.txt, setembro de 1992, último acesso em 01/06/2006.

[RFC 1918, 1996] – REKHTER, Y.; MOSKOWITZ, B.; KARRENBERG, D.; GROOT, G. J. de; LEAR, E. **Address Allocation for Private Internets**. Disponível em: http://www.faqs.org/rfcs/rfc1918.html, fevereiro de 1996, último acesso em 01/07/2006.

[RFC 2373, 1998] – HIDEN, Robert M.; DEERING, Stephen E. **IP Version 6 Addressing Architecture**. Disponível em: http://www.faqs.org/rfcs/rfc2373.html, 1998, último acesso em 07/05/2006.

[RFC 2374, 1998] – HIDEN, Robert M.; O'DELL, M.; DEERING, Stephen E. **An IPv6 Aggregatable Global Unicast Address Format**. Disponível em: http://www.rfc-archive.org/getrfc.php?rfc=2374, julho de 1998, último acesso em01/06/2006.

[RFC 2460, 1998] – HIDEN, Robert M.; DEERING, Stephen E. **Internet Protocol, Version 6 (IPv6) Specification**. Disponível em: http://www.faqs.org/rfcs/rfc2460.html, 1998, último acesso em 07/05/2006.

[RFC 2462, 1998] – THOMSON, S.; NARTEN, T. **IPv6 Stateless Address Autoconfiguration**. Disponível em: http://www.faqs.org/rfcs/rfc2462.html, dezembro de 1998, último acesso em 07/05/2006.

[RIBEIRO, 2004] – RIBEIRO, Uirá. **Certificação Linux**. 1º Edição. Ed. Axcel Books, Rio de Janeiro, 2004.

[RNP, 2006] – RNP - Rede Nacional de Ensino e Pesquisa. **Notícias RNP**. http://www.rnp.br/noticias/2006/not-060424a.html, último acesso em 07/05/2006.

[SAKURAY, 2005] – SAKURAY, Fábio. **Camadas do Modelo OSI e TCP/IP.**: UEL – Universidade Estadual de Londrina. Disponível em: http://www.dc.uel.br/~sakuray, 12 de março de 2005, último acesso em 30/04/2005.

[SANTOS, 2002] – SANTOS, Luiz Carlos dos. Disponível em: http://www.abusar.org/como_funciona_o_tcp.html, último acesso em 30/04/2006.

[SILVA; FARIA, 2001] – SILVA, Adailton J. S.; FARIA, Marcel R. **Hierarquia de Endereços IPv6**: RNP – Rede Nacional de Ensino e Pesquisa. Disponível em: http://www.rnp.br/newsgen/0103/end_ipv6.html, último acesso em 07/05/2006.

[SMETANA, 2004] – SMETANA, George Marcel M. A. **IPv4 e IPv6**. Disponível em: www.redes.usp.br/conteudo%5C%5Cdocumentos%5CArtigoIP.pdf, último acesso em 04/04/2006.

[SOARES et al, 1995] – SOARES, Luiz Fernando Gomes; LEMOS, Guido; COLCHER, Sérgio. **Redes de Computadores: das LANs, MANs e WANs às redes ATM.** 2º edição revisada e ampliada. Ed. Campus, Rio de Janeiro, 1995.

[SOUZA, 2005] – SOUZA, Jorge Moreira de. **Qualidade de Serviço (QoS) I: Dependabilidade**: Teleco – Informação em Telecomunicações. Disponível em: http://www.teleco.com.br/tutoriais/tutorialqos/default.asp, 12 de dezembro de 2005, último acesso em 20/05/2006.

[VMWARE, 2006] – VMware Inc, VMware Workstation, Disponível em: http://www.vmware.com, último acesso em 04/06/2006.

[WIKIPEDIA, 2005] – Wikipédia. **Voz sobre IP**: Wikipédia, a enciclopédia livre. http://pt.wikipedia.org/wiki/VoIP, último acesso em 21/05/2006.

Glossário

Biblioteca – coleção de rotinas que executam operações que são requisitadas por diversos programas.

Bit – abreviação de **Bi**nary digi**t** (dígito binário), é a menor unidade de informação em uma máquina. Um Bit tem um único valor binário: 0 ou 1. Informações mais úteis são obtidas pela combinação de bits consecutivos em escala maior, como Bytes, por exemplo.

Bytes – é um conjunto de 8 bits, como por exemplo 10010011.

DNS – *Domain Name System* (Sistema de Nomes de Domínio) é responsável por informar o nome ou número IP dos hosts do domínio.

Frame Relay – é uma técnica de comutação de pacotes baseada em um conjunto de protocolos especificados pelo ITU-T, sendo a técnica mais recomendada para implementação de redes WAN para conectividade entre *hosts* e redes locais.

FreeBSD – sistema operacional baseado no *Unix*.

Gateway – computador que faz a ligação entre duas redes, mesmo com uma única rede deve-se ser configurado um "gateway padrão" nos host.

GNU/Linux – o GNU/Linux é o verdadeiro nome do sistema operacional Linux, então o nome é sistema operacional GNU/Linux com suas várias distribuições: Conectiva, Red Hat, Suse, Debian, Slackware, etc.

Host – estação de trabalho com uma placa de rede física ou virtual, pode ser referenciada com mais de uma placa de rede, caso esta referência não exista o host fará referência somente a uma placa de rede.

IP – Protocolo *Internet* ou *Internet* Protocol. É um protocolo da camada *Internet* da arquitetura TCP/IP.

IPv4 – Protocolo *Internet* Versão 4 ou *Internet Protocol Version 4*. Veja também: IP.

IPv6 – Protocolo *Internet* Versão 6 ou Internet *Protocol Version 6*. Veja também: IP.

Kernel – núcleo do sistema operacional. A parte central de um sistema operacional, sobre o qual o restante do sistema está baseado.

Linux – Sistema operacional parecido com o *UNIX*, originalmente iniciado por Linus Torvalds. "*Linux*" realmente se refere somente o kernel do sistema operacional. Veja *GNU/Linux*.

Network – Rede.

Nó – dispositivo que implementa IPv4 ou IPv6.

Octeto – conjunto de oito Bits que juntos formam um Byte.

Roteador – um nó que encaminha pacotes IPv4 e IPv6.

Router – Roteador.

TCP/IP – *Transmission Control Protocol/Internet Protocol*.

VMware – Empresa que fabrica *software* de virtualização de *desktops* e servidores como a *VMware Workstation*.

Anexo A – Identificadores de Interface

Este anexo contém uma explicação sobre o identificador de interface IPv6. Os últimos 64 bits de um endereço IPv6 são o identificador de interface exclusivo ao prefixo de 64 bits do endereço IPv6. O endereço EUI-64 de 64 bits é definido pelo Instituto de Engenheiros Elétricistas e Eletrônicos (IEEE). Os endereços EUI-64 são atribuídos a um adaptador de rede ou derivados de endereços IEEE 802. [IEEE, 2005]

Os identificadores de interface tradicionais para adaptadores de rede usam um endereço de 48 bits chamado endereço IEEE 802. Esse endereço é composto por uma identificação de empresa de 24 bits (também chamada de identificação do fabricante) e uma identificação de extensão de 24 bits (também chamada de identificação da placa). A combinação da identificação de empresa, que é exclusivamente atribuída a cada fabricante de adaptadores de rede, e a identificação de placa, que é exclusivamente atribuída a cada adaptador de rede durante a montagem, produz um endereço globalmente exclusivo de 48 bits. Esse endereço de 48 bits também é denominado endereço físico, de hardware ou de controle de acesso à mídia (MAC). [IEEE, 2005] A Figura 6.1 mostra o formato do endereço MAC de 48 bits.

Figura 6.1 – Formato do endereço MAC de 48 bits [IEEE, 2005]

Os bits definidos no endereço IEEE 802 são os seguintes:

- Universal/Local (U/L)

O bit U/L é o sétimo bit do primeiro byte e é usado para determinar se o endereço é administrado universal ou localmente. Se o bit U/L for definido para 0, é sinal de que o IEEE, através da designação de uma identificação de empresa exclusiva, administrou o endereço. Mas se o bit U/L for definido para 1, o endereço

será administrado localmente. O administrador da rede substituiu o endereço fabricado e especificou um outro endereço. [IEEE, 2005]

- Individual/Group (I/G)

O bit I/G é o bit inferior do primeiro byte, que é usado para determinar se o endereço é individual (de difusão ponto a ponto) ou de grupo (de difusão seletiva). Quando definido para 0, o endereço é de difusão ponto a ponto. Quando definido para 1, o endereço é de difusão seletiva. [IEEE, 2005]

No caso de um endereço de adaptador de rede 802.x típico, os bits U/L e I/G são definidos para 0, que correspondem a um endereço MAC de difusão ponto a ponto administrado universalmente. [IEEE, 2005]

O endereço IEEE EUI-64 representa um novo padrão para o endereçamento de interface de rede. A identificação de empresa ainda é de 24 bits, mas a identificação de extensão é de 40 bits, criando um espaço de endereço muito maior para um fabricante de adaptador de rede. O endereço EUI-64 usa os bits U/L e I/G da mesma maneira que o endereço IEEE 802. [IEEE, 2005] A Figura 6.2 mostra o formato do endereço MAC de 64 bits.

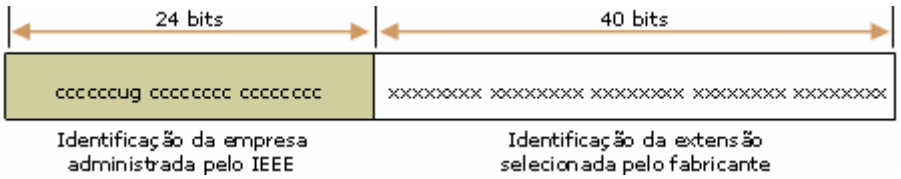

Figura 6.2 – Formato do endereço MAC de 64 bits [IEEE, 2005]

Para criar um endereço EUI-64 a partir de um endereço IEEE 802, os 16 bits 11111111 11111110 (0xFFFE) são inseridos no endereço IEEE 802 entre a identificação de empresa e a identificação de extensão. A Figura 6.3 mostra a conversão de um endereço IEEE 802 em endereço EUI-64. [IEEE, 2005]

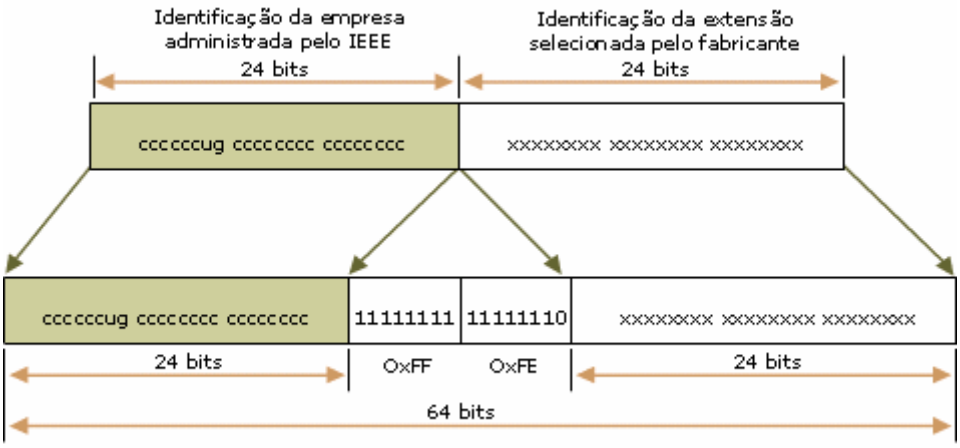

Figura 6.3 – Conversão de um endereço IEEE 802 em endereço EUI-64 [IEEE, 2005]

Para obter o identificador de interface de 64 bits para endereços IPv6 de difusão ponto a ponto, o bit U/L do endereço EUI-64 é complementado (se for 1, será definido para 0; se for 0, será definido para 1). A Figura 6.4 mostra a conversão de um endereço EUI-64 de difusão ponto a ponto administrado universalmente. [IEEE, 2005]

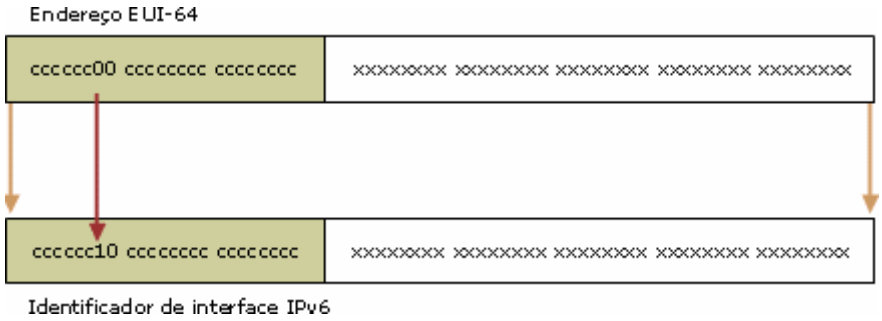

Figura 6.4 – Conversão de um endereço EUI-64 de difusão ponto a ponto administrado universalmente [IEEE, 2005]

Para obter um identificador de interface IPv6 a partir de um endereço IEEE 802, primeiro mapeia o endereço IEEE 802 para um endereço EUI-64 e, depois, complemente o bit U/L. A Figura 6.5 mostra o processo de conversão de um endereço IEEE 802 de difusão ponto a ponto administrado universalmente. [IEEE, 2005]

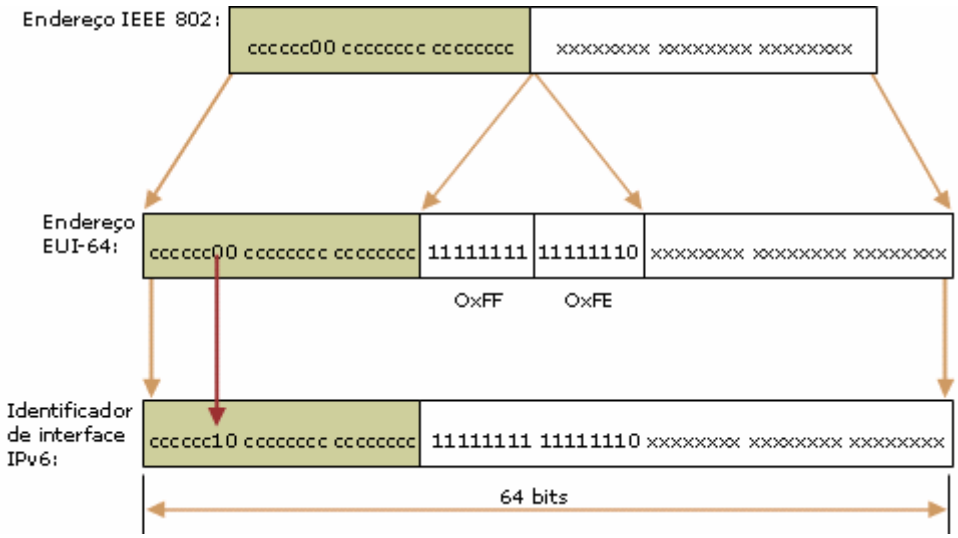

Figura 6.5 – Processo de conversão de um endereço IEEE 802 de difusão ponto a ponto administrado universalmente [IEEE, 2005]

Exemplo de conversão de endereço IEEE 802:

O *host* A tem o endereço Ethernet MAC 00-AA-00-3F-2A-1C. Primeiro, ele é convertido em formato EUI-64 através da inserção de FF-FE entre o terceiro e quarto bytes, o que produzirá o resultado 00-AA-00-FF-FE-3F-2A-1C. Depois, o bit U/L, que é o sétimo bit do primeiro byte, é complementado. O primeiro byte do formato binário é 00000000. Quando o sétimo bit for complementado, ele se tornará 00000010 (0x02). O resultado final será 02-AA-00-FF-FE-3F-2A-1C que, após a conversão em notação hexadecimal com dois-pontos, se tornará o identificador de interface 2AA:FF:FE3F:2A1C. Conseqüentemente, o endereço de conexão local correspondente ao adaptador de rede que possui o endereço MAC 00-AA-00-3F-2A-1C será FE80::2AA:FF:FE3F:2A1C.

Observação: Ao complementar o bit U/L, adicione 0x2 ao primeiro byte se o endereço EUI-64 for administrado universalmente e subtraia 0x2 do primeiro byte se o endereço EUI-64 for administrado localmente.

Anexo B – Comandos do *GNU/Linux* utilizados

cd – muda de diretório. Ex.: cd /home/usuario

gnome-terminal – entra no modo de comando no *Gnome.*

ifconfig – serve para configurar a rede e ver como ela está configurada.

konsole – entra no modo de comando no KDE e *Gnome.*

ls – lista o conteúdo de um diretório.

make – comando lê um arquivo Makefile, onde está determinado o "roteiro" necessário para a compilação do programa.

tar – resumidamente falando, ele descompacta arquivos no formato *tar* e *tar.gz*. Os parâmetros xzvf significam, x - extrai arquivos de um arquivo *tar*, z - descomprime o arquivo *tar* do gzip, v - exibe detalhes da operação e o f - especifica o arquivo *tar* a ser usado.

Anexo C – Arquivo “named.conf”

```
## named.conf - configuration for bind
#
 key "rndc-key" {
        algorithm hmac-md5;
        secret "MW1imZ07+GVzun9OSBj9Og==";
 };

 controls {
        inet 127.0.0.1 port 953
                        allow { 127.0.0.1; } keys { "rndc-key"; };
 };

#include "/etc/named.custom";

#include "/etc/rndc.key";

options {
            directory "/var/named";
            pid-file "/var/run/named/named.pid";
};
zone  "0.0.127.in-addr.arpa" {
            type master;
            file  "0.0.127.in-addr.arpa.zone";
};


zone  "localhost" {
            type master;
            file  "localhost.zone";
};
zone  "csnet.rede" {
            type master;
            file  "csnet.rede.zone";
};
```

Anexo D – Arquivo “csnet.rede.zone”

```
$TTL 86400
@       IN      SOA     csnet.rede.  root.localhost (
                        3 ; serial
                        28800 ; refresh
                        7200 ; retry
                        604800 ; expire
                        86400 ; ttl
                        )


        IN      NS      10.7.3.30


;Nome para IPv6
linux           IN      AAAA    FEC0:2006:6::15
tigreredhat     IN      AAAA    FEC0:2006:6::30
conectivatigre  IN      AAAA    FEC0:2006:6::21
presario900us   IN      AAAA    FEC0:2006:6::12
free            IN      AAAA    FEC0:2006:6::61

;Nome para IPv4
tigre           IN      A       10.7.3.1
presario900us   IN      A       10.7.3.12
linux           IN      A       10.7.3.15
conectivatigre  IN      A       10.7.3.21
tigreredhat     IN      A       10.7.3.30
free            IN      A       10.7.3.61
```

Anexo E – Arquivo de configuração GnuGK

```
[Gatekeeper::Main]
FourtyTwo=42
Name=GnuGk
EndpointSuffix=_gnugk
home=10.7.3.30
TimeToLive=60
StatusTraceLevel=0
UseBroadcastListener=0
UseMulticastListener=0

[GkStatus::Auth]
rule=explicit
10.7.3.1=allow
10.7.3.12=allow
10.7.3.30=allow
10.7.3.161=allow
default=forbid
Shutdown=allow
gkadmin=P5l3xBNhsGE=

[RoutedMode]
GKRouted=1
H245Routed=0
CallSignalPort=1721
AcceptNeighborCalls=1
AcceptUnregisteredCalls=0
RemoveH245AddressOnTunneling=1
RemoveCallOnDRQ=0
DropCallsByReleaseComplete=1
SendReleaseCompleteOnDRQ=0
SupportNATedEndpoints=1
TranslateFacility=1

[Proxy]
Enable=0
ProxyForNAT=1
ProxyForSameNAT=0

[FileIPAuth]

[RasSrv::RRQFeatures]
AcceptEndpointIdentifier=1
AcceptGatewayPrefixes=1

[RasSrv::ARQFeatures]
CallUnregisteredEndpoints=1

[CallTable]
GenerateNBCDR=0
GenerateUCCDR=1

[Password]
alessandro=wFvtvSfCAdA=
silvia=K5pneHTEeNc=
```

Anexo F – Telas com os passos de configuração do endereço IPv4 no *Windows 2003 Server*

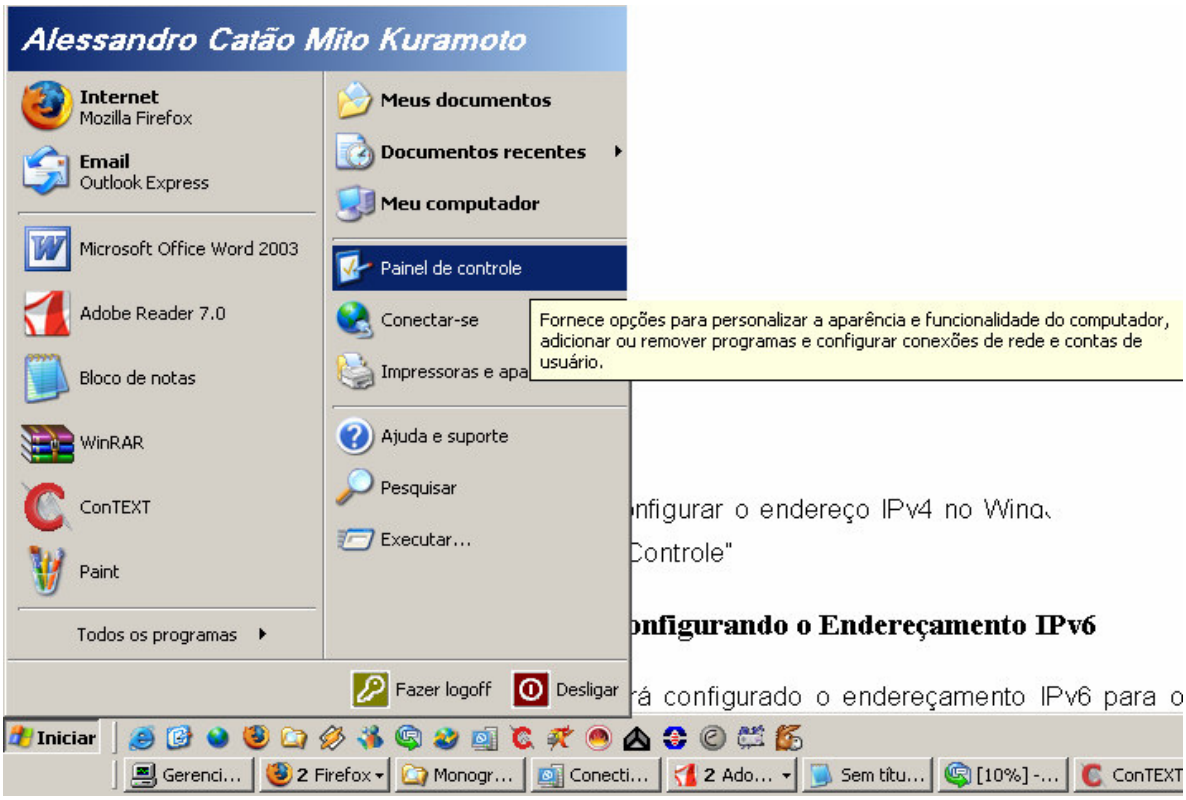

Figura 7.1 – Iniciar – Painel de Controle

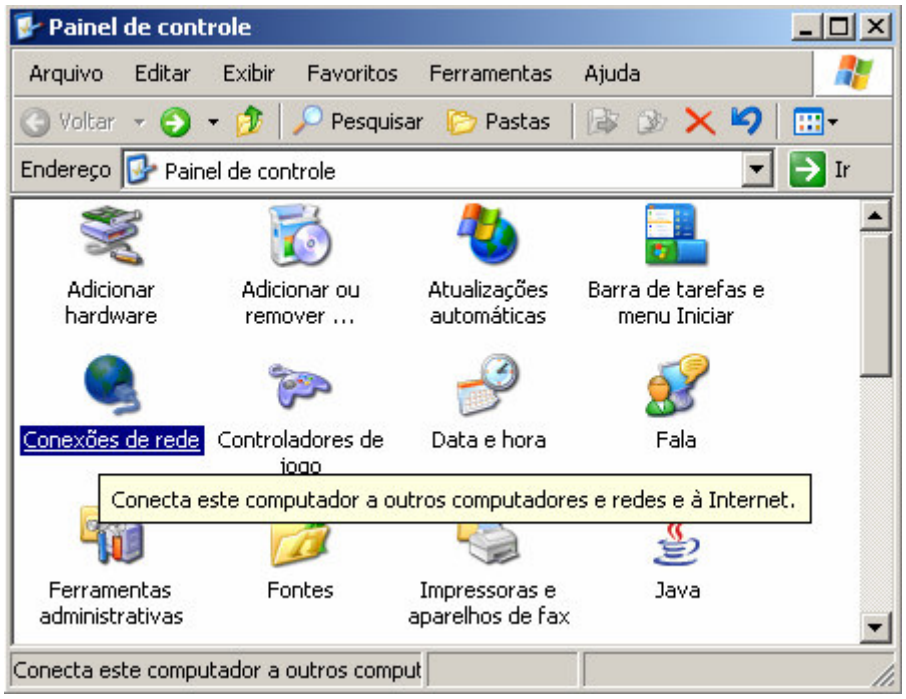

Figura 7.2 – Janela: Painel de Controle

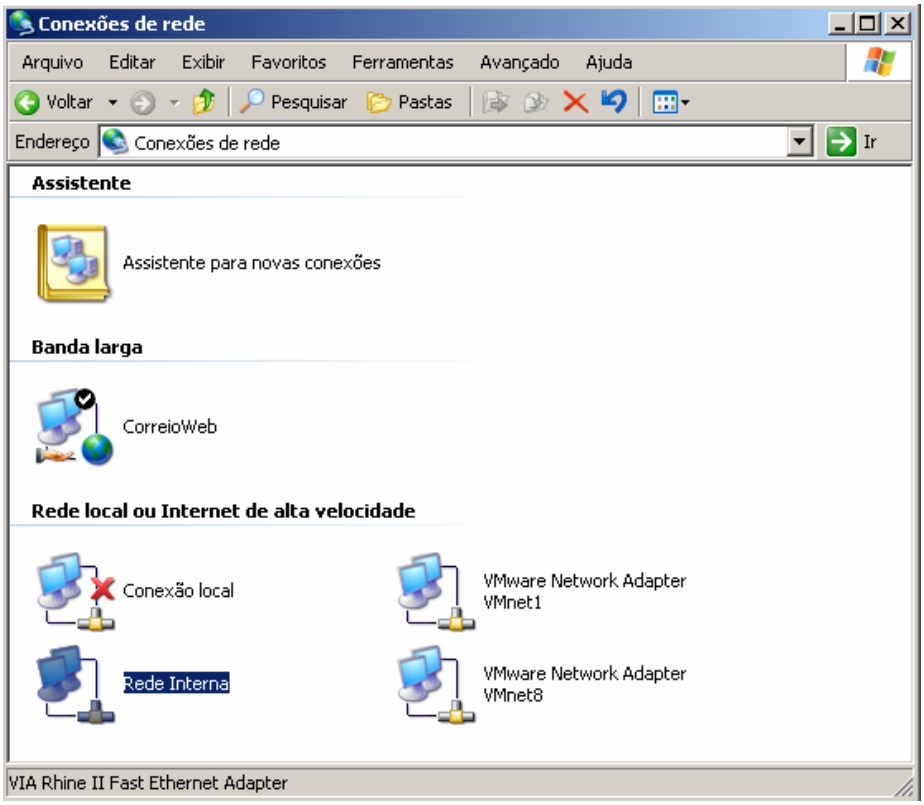

Figura 7.3 – Janela: Conexão de Rede

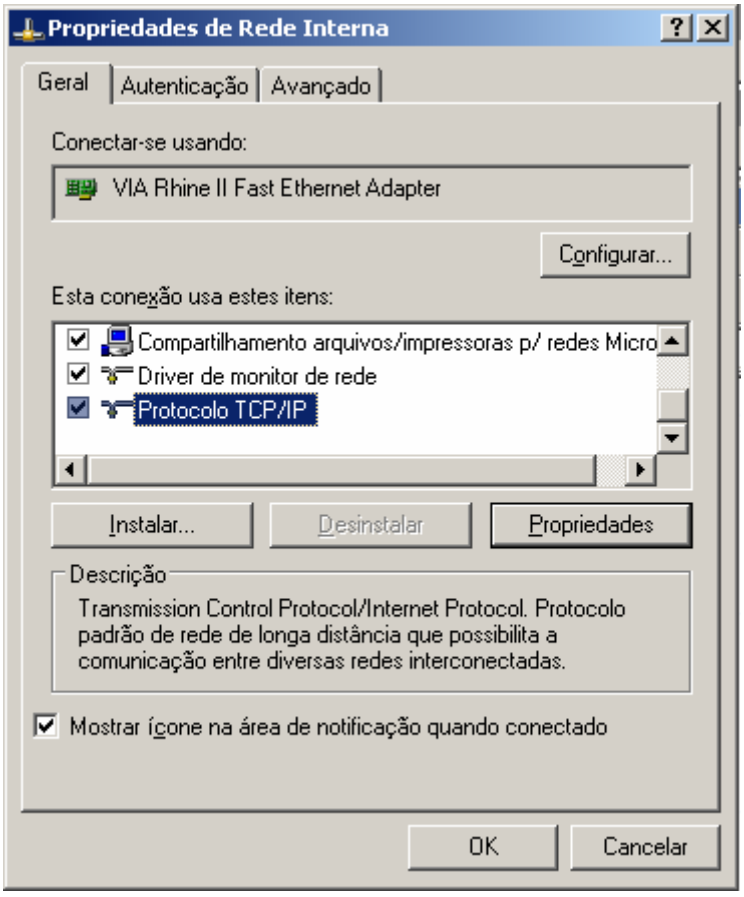

Figura 7.4 – Janela: Propriedade de "Rede Interna"

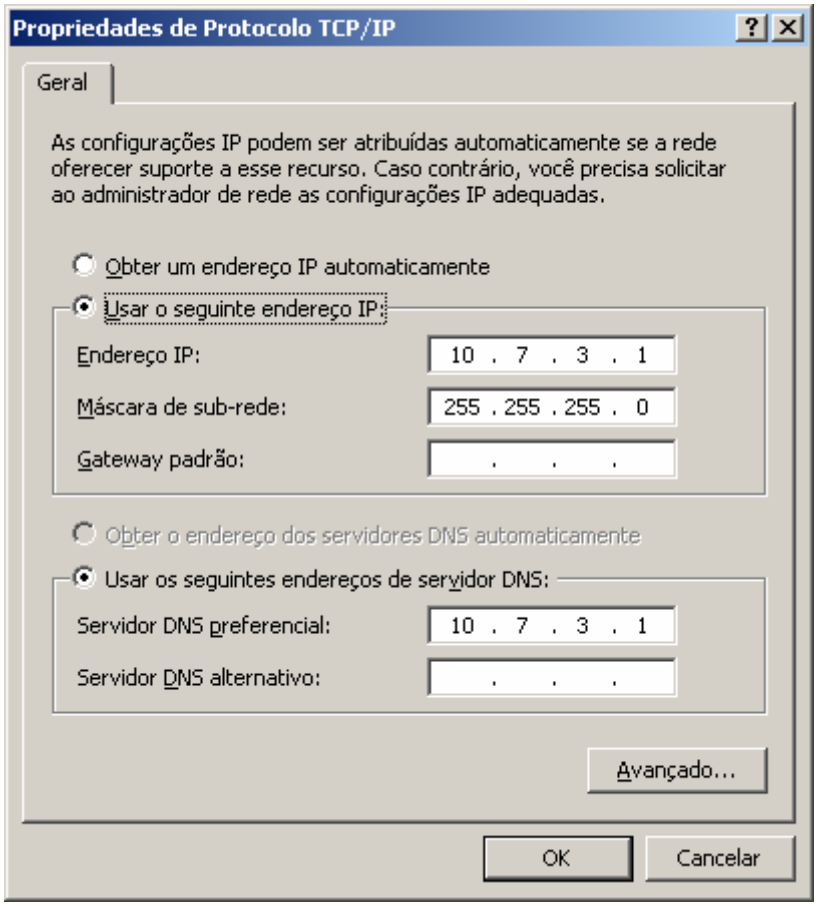

Figura 7.5 – Janela: Propriedade de Protocolo TCP/IP

Anexo G – Telas de configuração do NetMeeting

Figura 8.1 – NetMeeting: Tela inicial

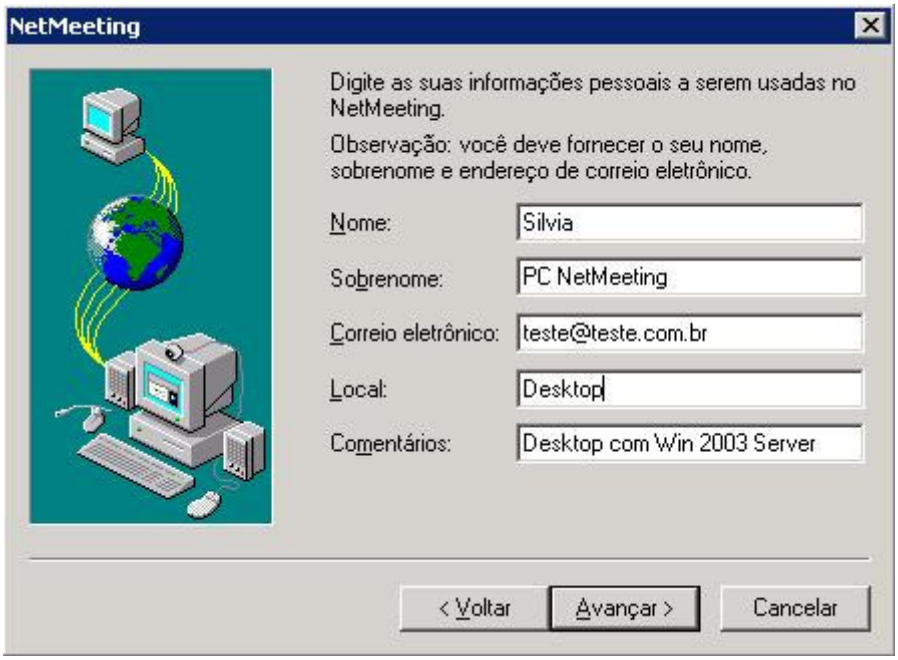

Figura 8.2 – NetMeeting: Digitando as informações

Figura 8.3 – NetMeeting: Tela para selecionar um servidor de doretório. Não será configurado aqui.

Figura 8.4 – NetMeeting: Velocidade da conexão. Selecionar a rede local.

Figura 8.5 – NetMeeting: Criar atalhos na área de trabalha e na barra do *Quick Launch*

Figura 8.6 – NetMeeting: Assistente de ajuste de áudio

Figura 8.7 – NetMeeting: Tela de ajuste de áudio. Após esta tela, a configuração padrão estará terminada.

Anexo H – Arquivo NOVOKERNEL

```
#
# GENERIC -- Generic kernel configuration file for FreeBSD/i386
#
# For more information on this file, please read the handbook section on
# Kernel Configuration Files:
#
#    http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig-config.html
#
# The handbook is also available locally in /usr/share/doc/handbook
# if you've installed the doc distribution, otherwise always see the
# FreeBSD World Wide Web server (http://www.FreeBSD.org/) for the
# latest information.
#
# An exhaustive list of options and more detailed explanations of the
# device lines is also present in the ../../conf/NOTES and NOTES files.
# If you are in doubt as to the purpose or necessity of a line, check first
# in NOTES.
#
# $FreeBSD: src/sys/i386/conf/GENERIC,v 1.413.2.6.2.2 2004/10/24 18:02:52 scottl Exp $

machine         i386
cpu             I486_CPU
cpu             I586_CPU
cpu             I686_CPU
ident           GENERIC

# To statically compile in device wiring instead of /boot/device.hints
#hints          "GENERIC.hints"         # Default places to look for devices.

options SCHED_4BSD              # 4BSD scheduler
options INET                    # InterNETworking
options INET6                   # IPv6 communications protocols
options FFS                     # Berkeley Fast Filesystem
options SOFTUPDATES             # Enable FFS soft updates support
options UFS_ACL                 # Support for access control lists
options UFS_DIRHASH             # Improve performance on big directories
options MD_ROOT                 # MD is a potential root device
options NFSCLIENT               # Network Filesystem Client
options NFSSERVER               # Network Filesystem Server
options NFS_ROOT                # NFS usable as /, requires NFSCLIENT
options MSDOSFS                 # MSDOS Filesystem
options CD9660                  # ISO 9660 Filesystem
options PROCFS                  # Process filesystem (requires PSEUDOFS)
options PSEUDOFS                # Pseudo-filesystem framework
options GEOM_GPT                # GUID Partition Tables.
options COMPAT_43               # Compatible with BSD 4.3 [KEEP THIS!]
options COMPAT_FREEBSD4         # Compatible with FreeBSD4
options SCSI_DELAY=15000        # Delay (in ms) before probing SCSI
options KTRACE                  # ktrace(1) support
options SYSVSHM                 # SYSV-style shared memory
options SYSVMSG                 # SYSV-style message queues
options SYSVSEM                 # SYSV-style semaphores
options _KPOSIX_PRIORITY_SCHEDULING # POSIX P1003_1B real-time extensions
options KBD_INSTALL_CDEV        # install a CDEV entry in /dev
options AHC_REG_PRETTY_PRINT    # Print register bitfields in debug
```

```
                                        # output.  Adds ~128k to driver.
options AHD_REG_PRETTY_PRINT    # Print register bitfields in debug
                                        # output.  Adds ~215k to driver.
options ADAPTIVE_GIANT          # Giant mutex is adaptive.

device          apic            # I/O APIC

# Bus support.  Do not remove isa, even if you have no isa slots
device          isa
device          eisa
device          pci

# Floppy drives
device          fdc

# ATA and ATAPI devices
device          ata
device          atadisk         # ATA disk drives
device          ataraid         # ATA RAID drives
device          atapicd         # ATAPI CDROM drives
device          atapifd         # ATAPI floppy drives
device          atapist         # ATAPI tape drives
options ATA_STATIC_ID           # Static device numbering

# SCSI Controllers
device          ahb             # EISA AHA1742 family
device          ahc             # AHA2940 and onboard AIC7xxx devices
device          ahd             # AHA39320/29320 and onboard AIC79xx devices
device          amd             # AMD 53C974 (Tekram DC-390(T))
device          isp             # Qlogic family
device          mpt             # LSI-Logic MPT-Fusion
#device         ncr             # NCR/Symbios Logic
device          sym             # NCR/Symbios Logic (newer chipsets + those of `ncr')
device          trm             # Tekram DC395U/UW/F DC315U adapters

device          adv             # Advansys SCSI adapters
device          adw             # Advansys wide SCSI adapters
device          aha             # Adaptec 154x SCSI adapters
device          aic             # Adaptec 15[012]x SCSI adapters, AIC-6[23]60.
device          bt              # Buslogic/Mylex MultiMaster SCSI adapters

device          ncv             # NCR 53C500
device          nsp             # Workbit Ninja SCSI-3
device          stg             # TMC 18C30/18C50

# SCSI peripherals
device          scbus           # SCSI bus (required for SCSI)
device          ch              # SCSI media changers
device          da              # Direct Access (disks)
device          sa              # Sequential Access (tape etc)
device          cd              # CD
device          pass            # Passthrough device (direct SCSI access)
device          ses             # SCSI Environmental Services (and SAF-TE)

# RAID controllers interfaced to the SCSI subsystem
device          amr             # AMI MegaRAID
device          asr             # DPT SmartRAID V, VI and Adaptec SCSI RAID
device          ciss            # Compaq Smart RAID 5*
device          dpt             # DPT Smartcache III, IV - See NOTES for options
device          hptmv           # Highpoint RocketRAID 182x
```

```
device          iir             # Intel Integrated RAID
device          ips             # IBM (Adaptec) ServeRAID
device          mly             # Mylex AcceleRAID/eXtremeRAID
device          twa             # 3ware 9000 series PATA/SATA RAID

# RAID controllers
device          aac             # Adaptec FSA RAID
device          aacp            # SCSI passthrough for aac (requires CAM)
device          ida             # Compaq Smart RAID
device          mlx             # Mylex DAC960 family
device          pst             # Promise Supertrak SX6000
device          twe             # 3ware ATA RAID

# atkbdc0 controls both the keyboard and the PS/2 mouse
device          atkbdc          # AT keyboard controller
device          atkbd           # AT keyboard
device          psm             # PS/2 mouse

device          vga             # VGA video card driver

device          splash          # Splash screen and screen saver support

# syscons is the default console driver, resembling an SCO console
device          sc

# Enable this for the pcvt (VT220 compatible) console driver
#device         vt
#options        XSERVER                 # support for X server on a vt console
#options        FAT_CURSOR # start with block cursor

device          agp             # support several AGP chipsets

# Floating point support - do not disable.
device          npx

# Power management support (see NOTES for more options)
#device         apm
# Add suspend/resume support for the i8254.
device          pmtimer

# PCCARD (PCMCIA) support
# PCMCIA and cardbus bridge support
device          cbb             # cardbus (yenta) bridge
device          pccard          # PC Card (16-bit) bus
device          cardbus                 # CardBus (32-bit) bus

# Serial (COM) ports
device          sio             # 8250, 16[45]50 based serial ports

# Parallel port
device          ppc
device          ppbus           # Parallel port bus (required)
device          lpt             # Printer
device          plip            # TCP/IP over parallel
device          ppi             # Parallel port interface device
#device         vpo             # Requires scbus and da

# If you've got a "dumb" serial or parallel PCI card that is
# supported by the puc(4) glue driver, uncomment the following
# line to enable it (connects to the sio and/or ppc drivers):
```

```
#device        puc

# PCI Ethernet NICs.
device          de              # DEC/Intel DC21x4x (``Tulip'')
device          em              # Intel PRO/1000 adapter Gigabit Ethernet Card
device          ixgb            # Intel PRO/10GbE Ethernet Card
device          txp             # 3Com 3cR990 (``Typhoon'')
device          vx              # 3Com 3c590, 3c595 (``Vortex'')

# PCI Ethernet NICs that use the common MII bus controller code.
# NOTE: Be sure to keep the 'device miibus' line in order to use these NICs!
device          miibus          # MII bus support
device          bfe             # Broadcom BCM440x 10/100 Ethernet
device          bge             # Broadcom BCM570xx Gigabit Ethernet
device          dc              # DEC/Intel 21143 and various workalikes
device          fxp             # Intel EtherExpress PRO/100B (82557, 82558)
device          lge             # Level 1 LXT1001 gigabit ethernet
device          nge             # NatSemi DP83820 gigabit ethernet
device          pcn             # AMD Am79C97x PCI 10/100 (precedence over 'lnc')
device          re              # RealTek 8139C+/8169/8169S/8110S
device          rl              # RealTek 8129/8139
device          sf              # Adaptec AIC-6915 (``Starfire'')
device          sis             # Silicon Integrated Systems SiS 900/SiS 7016
device          sk              # SysKonnect SK-984x & SK-982x gigabit Ethernet
device          ste             # Sundance ST201 (D-Link DFE-550TX)
device          ti              # Alteon Networks Tigon I/II gigabit Ethernet
device          tl              # Texas Instruments ThunderLAN
device          tx              # SMC EtherPower II (83c170 ``EPIC'')
device          vge             # VIA VT612x gigabit ethernet
device          vr              # VIA Rhine, Rhine II
device          wb              # Winbond W89C840F
device          xl              # 3Com 3c90x (``Boomerang'', ``Cyclone'')

# ISA Ethernet NICs.  pccard NICs included.
device          cs              # Crystal Semiconductor CS89x0 NIC
# 'device ed' requires 'device miibus'
device          ed              # NE[12]000, SMC Ultra, 3c503, DS8390 cards
device          ex              # Intel EtherExpress Pro/10 and Pro/10+
device          ep              # Etherlink III based cards
device          fe              # Fujitsu MB8696x based cards
device          ie              # EtherExpress 8/16, 3C507, StarLAN 10 etc.
device          lnc             # NE2100, NE32-VL Lance Ethernet cards
device          sn              # SMC's 9000 series of Ethernet chips
device          xe              # Xircom pccard Ethernet

# ISA devices that use the old ISA shims
#device         le

# Wireless NIC cards
device          wlan            # 802.11 support
device          an              # Aironet 4500/4800 802.11 wireless NICs.
device          awi             # BayStack 660 and others
device          wi              # WaveLAN/Intersil/Symbol 802.11 wireless NICs.
#device         wl              # Older non 802.11 Wavelan wireless NIC.

# Pseudo devices.
device          loop    1       # Network loopback
device          mem             # Memory and kernel memory devices
device          io              # I/O device
device          random          # Entropy device
```

```
device          ether           # Ethernet support
device          sl              # Kernel SLIP
device          ppp             # Kernel PPP
device          tun             # Packet tunnel.
device          pty             # Pseudo-ttys (telnet etc)
device          md              # Memory "disks"
device          gif     4       # IPv6 and IPv4 tunneling
device          faith   1       # IPv6-to-IPv4 relaying (translation)

# The `bpf' device enables the Berkeley Packet Filter.
# Be aware of the administrative consequences of enabling this!
device          bpf     4       # Berkeley packet filter

# USB support
device          uhci            # UHCI PCI->USB interface
device          ohci            # OHCI PCI->USB interface
device          usb             # USB Bus (required)
#device         udbp            # USB Double Bulk Pipe devices
device          ugen            # Generic
device          uhid            # "Human Interface Devices"
device          ukbd            # Keyboard
device          ulpt            # Printer
device          umass           # Disks/Mass storage - Requires scbus and da
device          ums             # Mouse
device          urio            # Diamond Rio 500 MP3 player
device          uscanner        # Scanners
# USB Ethernet, requires mii
device          aue             # ADMtek USB Ethernet
device          axe             # ASIX Electronics USB Ethernet
device          cue             # CATC USB Ethernet
device          kue             # Kawasaki LSI USB Ethernet
device          rue             # RealTek RTL8150 USB Ethernet

# FireWire support
device          firewire # FireWire bus code
device          sbp             # SCSI over FireWire (Requires scbus and da)
device          fwe             # Ethernet over FireWire (non-standard!)

# KAME extensions
#
# IPSEC does not work due to FAST_IPSEC changes
#options        IPSEC                   #IP security
#options        IPSEC_ESP               #IP security (crypto; define w/ IPSEC)
#options        IPSEC_DEBUG                     #debug for IP security
#options        NATM                    #native mode ATM
#options        ENABLE_DEFAULT_SCOPE

#options        ND6_DEBUG   # net.inet6.icmp6.nd6_debug=1 by default

options NEW_STRUCT_ROUTE        # mandatory

#options        RADIX_MPATH         # equal cost multipath

#options        SCTP            # adds SCTP stack to kernel - requires INET6
#options        SCTP_DEBUG # adds debugging support for SCTP
#options        SCTP_TCP_MODEL_SUPPORT          # adds TCP model support
#options        SCTP_USE_ADLER32 # use obsolete Adler32 checksum,

# Router Preference on host side
#options        RTPREF
```

```
# Habilita o firewall IPv6, não foi habilitado
#options        "IPV6FIREWALL”
#options        "IPV6FIREWALL_VERBOSE"
#options        "IPV6FIREWALL_DEFAULT_TO_ACCEPT"

# ALTQ
#options        ALTQ            #alternate queueing
#options        ALTQ_CBQ        #class based queueing
#options        ALTQ_WFQ        #weighted fair queueing
#options        ALTQ_FIFOQ      #fifo queueing
#options        ALTQ_RED        #random early detection
#options        ALTQ_FLOWVALVE  #flowvalve for RED (needs RED)
#options        ALTQ_RIO        #triple red for diffserv (needs RED)
#options        ALTQ_LOCALQ     #local use
#options        ALTQ_HFSC       #hierarchical fair service curve
#options        ALTQ_JOBS       #joint buffer management and scheduling
#options        ALTQ_IPSEC      #check ipsec in IPv4
#options        ALTQ_CDNR       #diffserv traffic conditioner
#options        ALTQ_BLUE       #blue by wu-chang feng
#options        ALTQ_PRIQ       #priority queue
#options        ALTQ_NOPCC      #don't use processor cycle counter
#options        ALTQ_DEBUG      #for debugging
# you might want to set kernel timer to 1kHz if you use CBQ,
# especially with 100baseT
#options        HZ=1000

options TCP_ECN                 # ECN support in TCP

# Network Address Translation - Protocol Translation (NAT-PT)
#options                        # IPv6 -> IPv4 translation.
#options        NATPT_NAT       # IPv4 -> IPv4 NAT.
                                # Valid only when "options NATPT" is defined.

# Source-Specific Multicast (SSM)
#options        IGMPV3                  # IPv4
#options        MLDV2           # IPv6

#device         atm     1
device  dummy 1
device          stf     1

# Mobile IPv6
#options        MIP6
#options        MIP6_HOME_AGENT
#options        MIP6_MOBILE_NODE
#device         hif     1
#options        MIP6_DEBUG

device          pf
device          pflog
device          pfsync

# Datagram Congestion Control Protocol
#options        DCCP
#options        DCCP_TFRC
```

Anexo I – Telas de configuração do DHCP no Windows 2003 Server

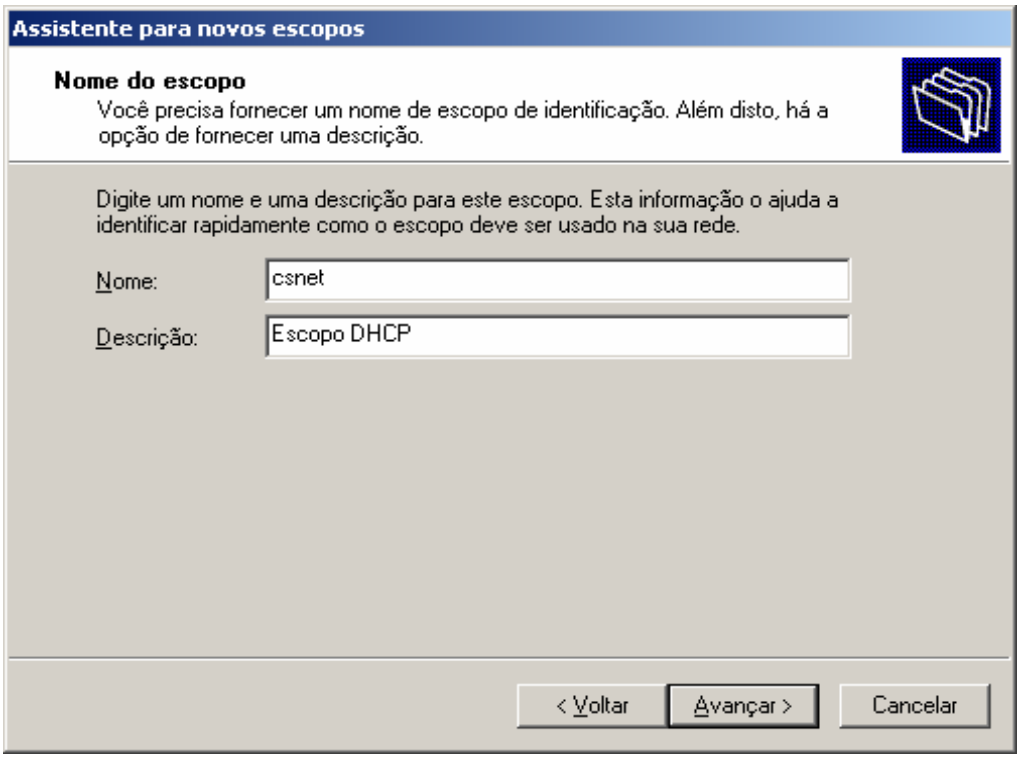

Figura 9.1 – DHCP no Windows 2003 Server: Atribuindo um nome ao escopo.

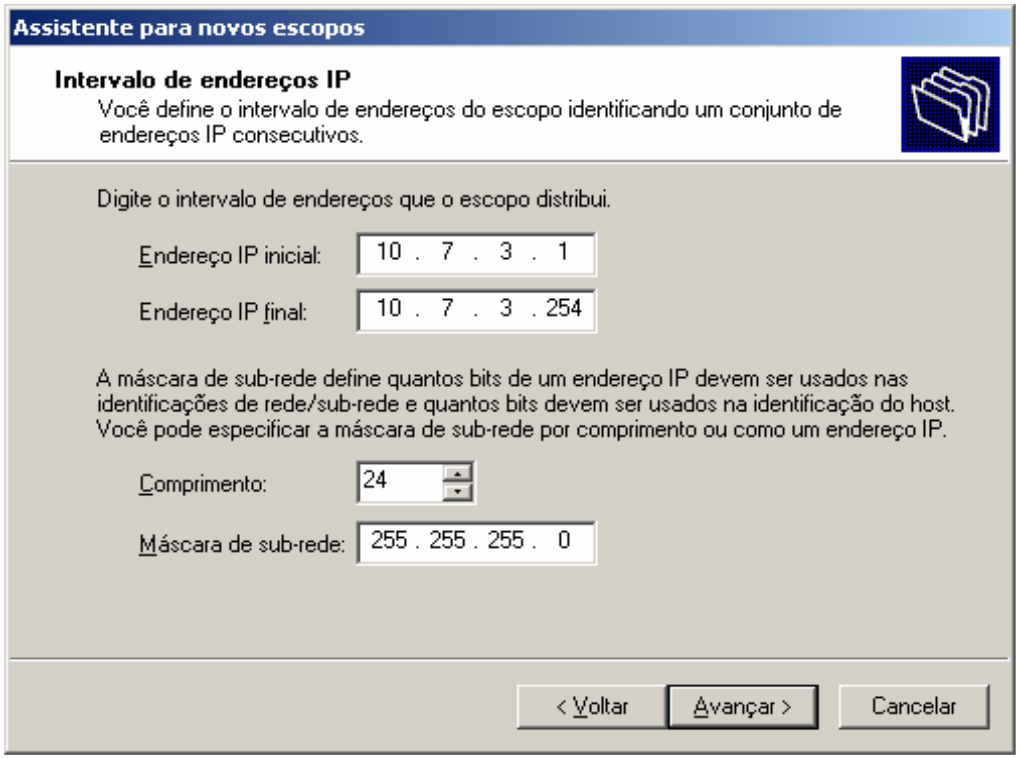

Figura 9.2 – DHCP no Windows 2003 Server: Atribuindo o intervalo de endereços.

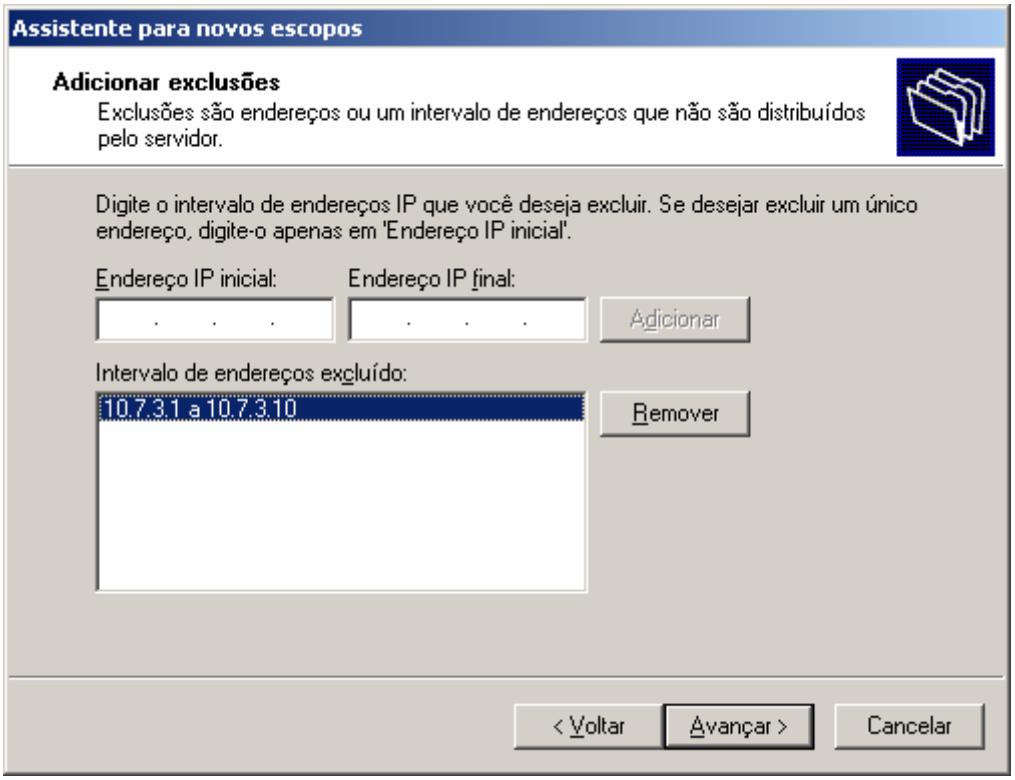

Figura 9.3 – DHCP no Windows 2003 Server: Definindo as exclusões (intervalo de endereços que não serão distribuídos)

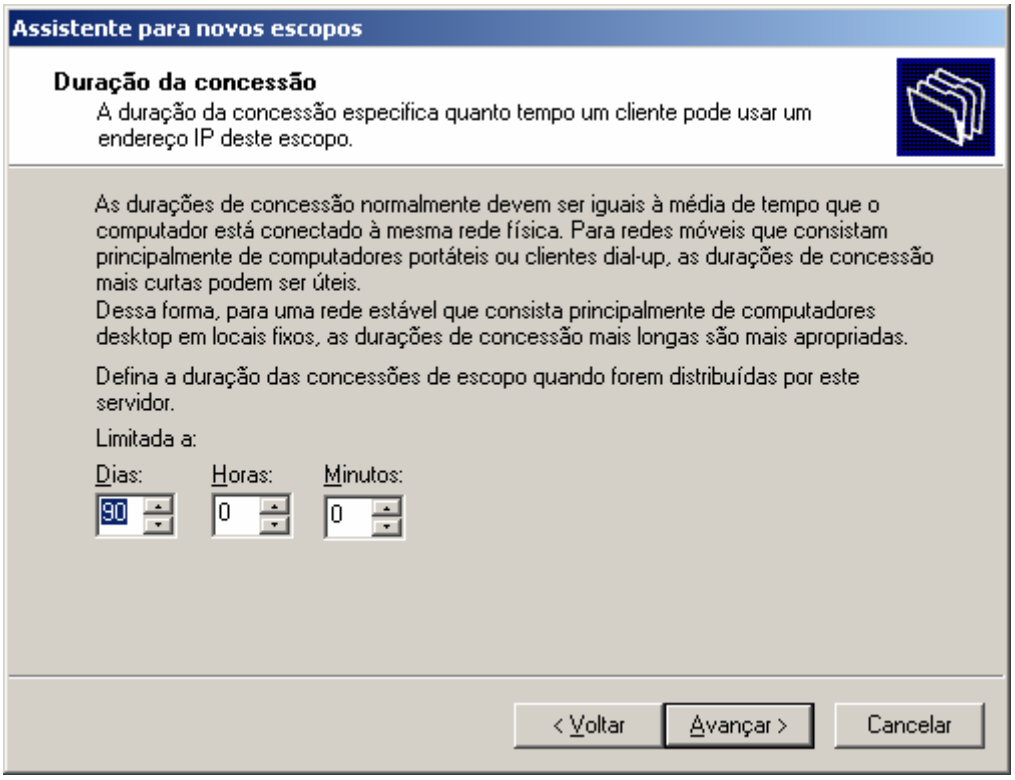

Figura 9.4 – DHCP no Windows 2003 Server: Definindo o período de concessão do endereço IP.

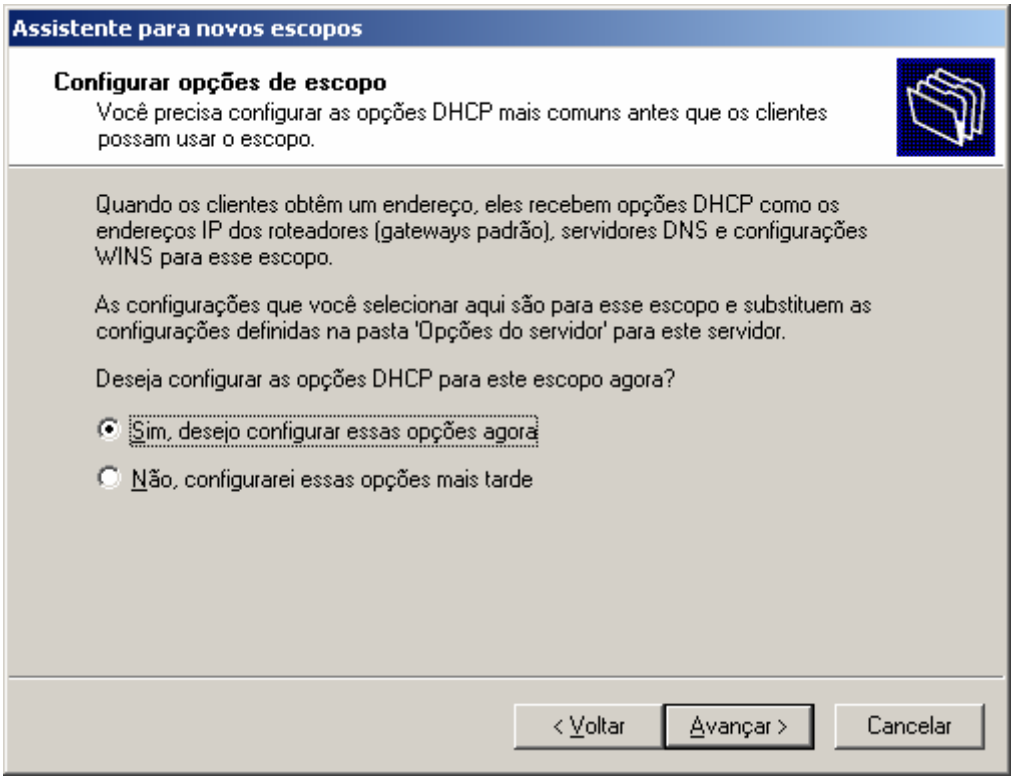

Figura 9.5 – DHCP no Windows 2003 Server: Opções de escopo.

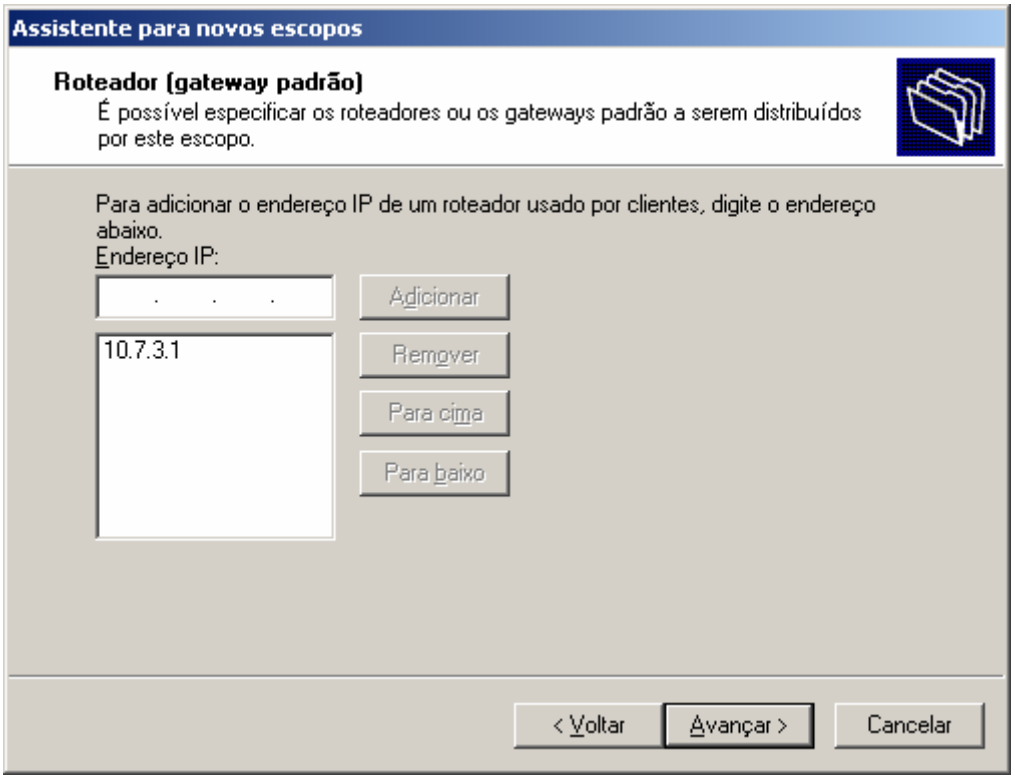

Figura 9.6 – DHCP no Windows 2003 Server: Definindo o *gateway* padrão.

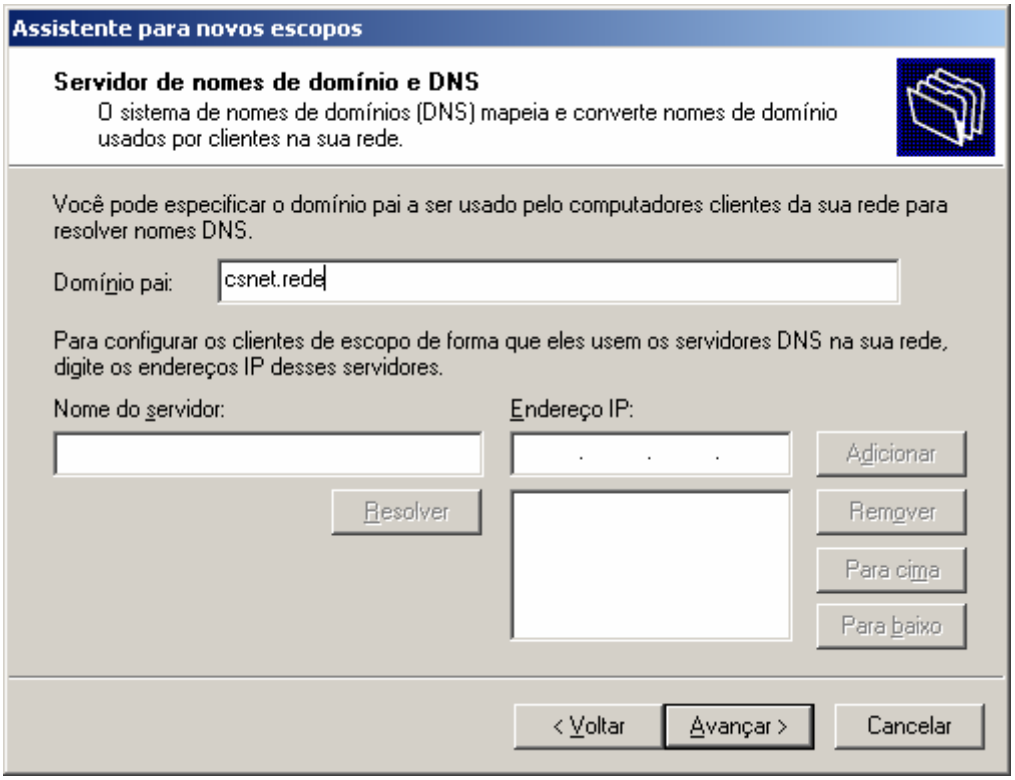

Figura 9.7 – DHCP no Windows 2003 Server: Definindo opções do servidor de DNS.

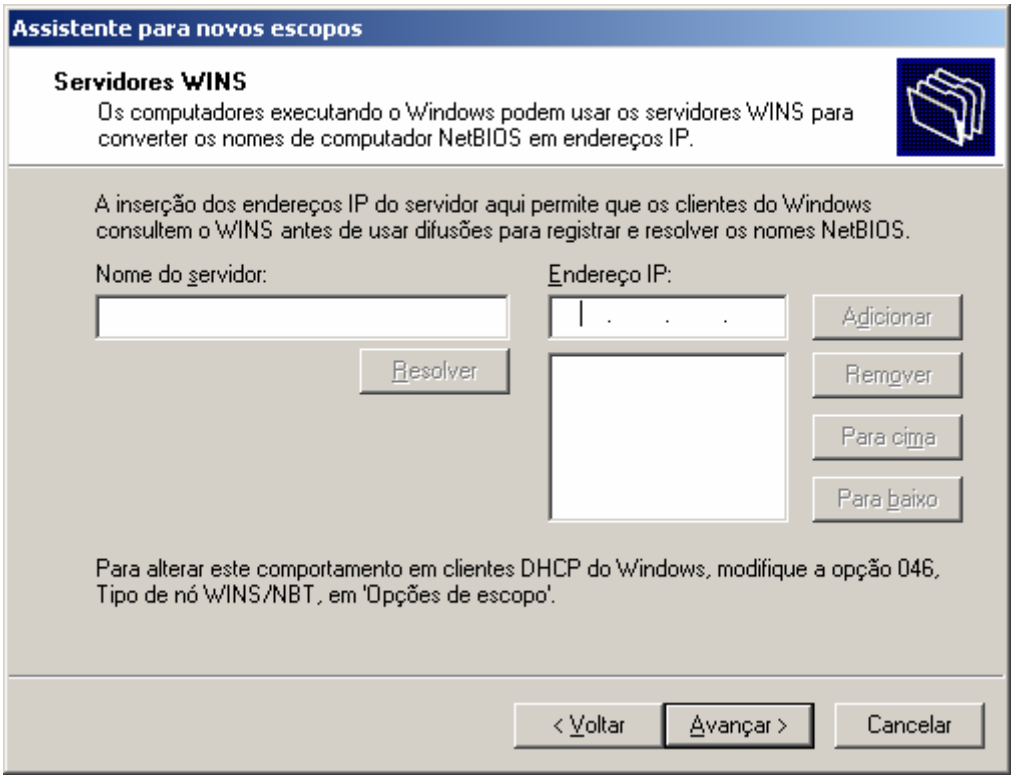

Figura 9.8 – DHCP no Windows 2003 Server: Definindo configurações de WINS (não precisa configurar).

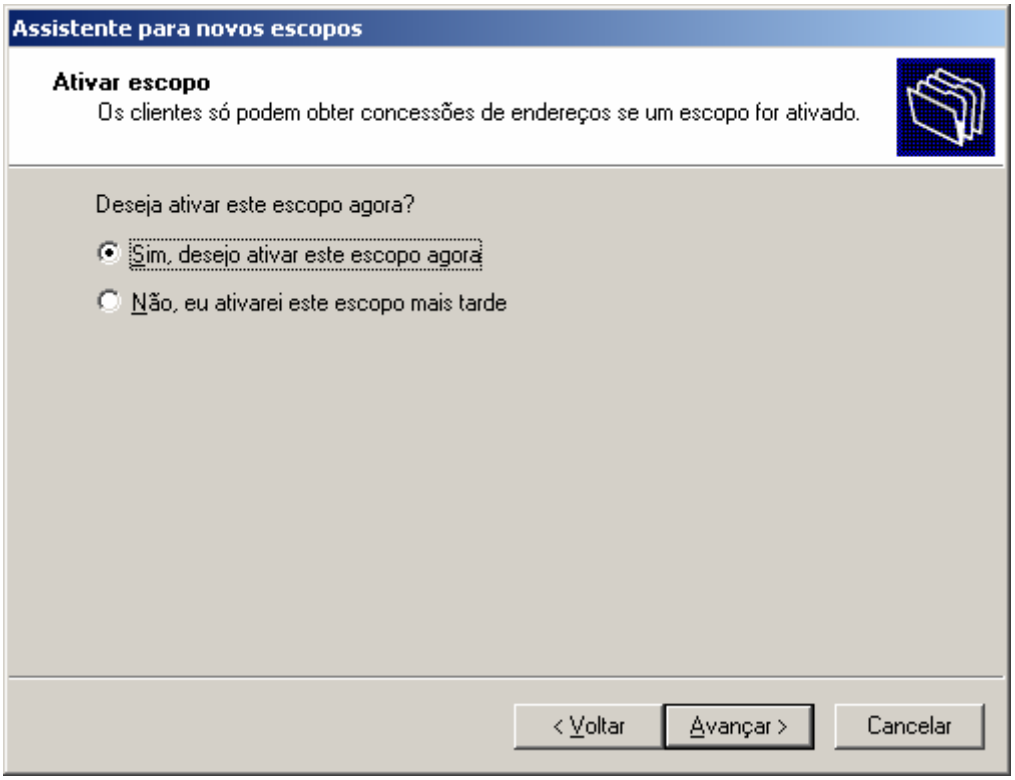

Figura 9.9 – DHCP no Windows 2003 Server: Ativando o escopo.

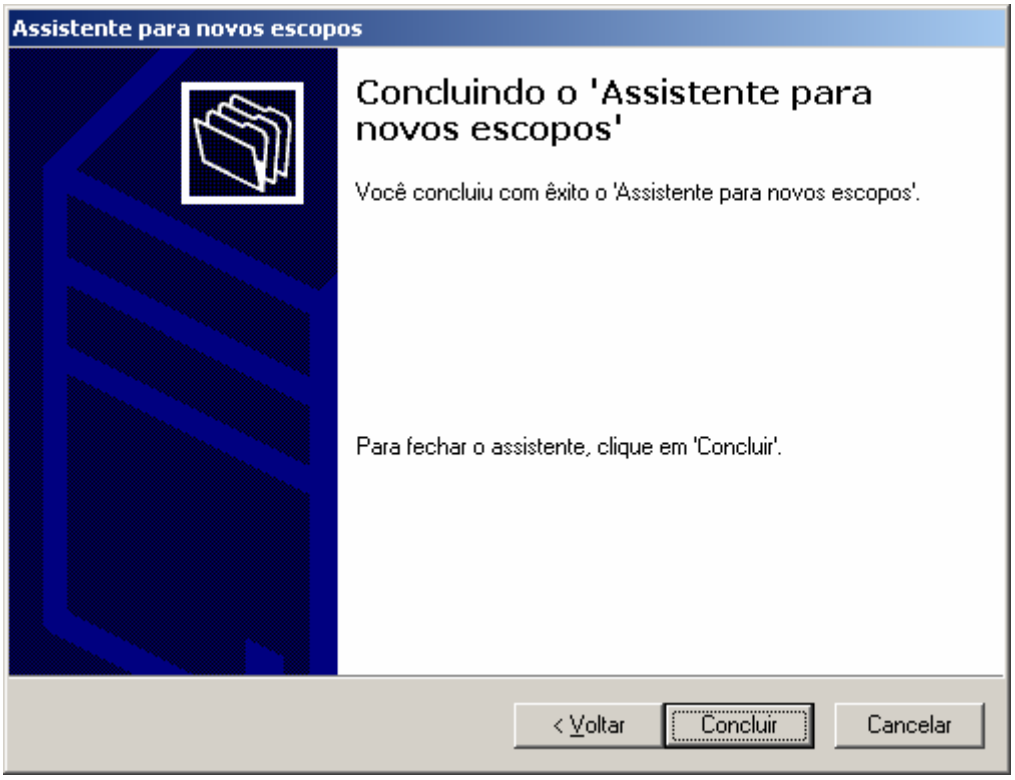

Figura 9.10 – DHCP no Windows 2003 Server: Concluindo a configuração do escopo.