UNIVERSIDADE FEDERAL DE SANTA MARIA
COLÉGIO TÉCNICO INDUSTRIAL DE SANTA MARIA
CURSO SUPERIOR DE TECNOLOGIA EM REDES DE COMPUTADORES

IMPLANTAÇÃO DE UMA REDE UTILIZANDO OS PADRÕES DO PROTOCOLO IPv6

TRABALHO DE CONCLUSÃO DE CURSO

Raissa Monego Pedrozo

Santa Maria, RS, Brasil

2014

TCC/REDES DE COMPUTADORES/UFSM, RS MONEGO, Raissa Pedrozo Graduada 2014

IMPLANTAÇÃO DE UMA REDE UTILIZANDO OS PADRÕES DO PROTOCOLO IPv6

Raissa Monego Pedrozo

Trabalho de Conclusão de Curso (TCC) do Curso Superior de Tecnologia em Redes de Computadores, da Universidade Federal de Santa Maria (UFSM,RS), como requisito parcial para obtenção do grau de
Tecnólogo em Redes de Computadores

Orientadora: Prof. Dra. Simone Regina Ceolin
Coorientador: Prof. Ms. Renato Preigschadt de Azevedo

Santa Maria, RS, Brasil

2014

UNIVERSIDADE FEDERAL DE SANTA MARIA
COLÉGIO TÉCNICO INDUSTRIAL DE SANTA MARIA
CURSO SUPERIOR DE TECNOLOGIA EM REDES DE COMPUTADORES

A Comissão Examinadora, abaixo assinada, aprova o Trabalho de Conclusão de Curso

IMPLANTAÇÃO DE UMA REDE UTILIZANDO OS PADRÕES DO PROTOCOLO IPv6

elaborado por
Raissa Monego Pedrozo

COMISSÃO EXAMINADORA

Simone Regina Ceolin, Dra.
(Presidente/Orientadora)

Renato Preigschadt de Azevedo, Dr.
(Coorientador)

Bruno Augusti Mozzaquatro, Ms. (UFSM)

Alfredo Del Fabro Neto, Prof. (UFSM)

Santa Maria, 08 de janeiro de 2014.

RESUMO

TRABALHO DE CONCLUSÃO DE CURSO
CURSO SUPERIOR DE TECNOLOGIA EM REDES DE COMPUTADORES
UNIVERSIDADE FEDERAL DE SANTA MARIA

IMPLANTAÇÃO DE UMA REDE UTILIZANDO OS PADRÕES DO PROTOCOLO IPv6

AUTOR: RAISSA MONEGO PEDROZO

ORIENTADORA: SIMONE REGINA CEOLIN

COORIENTADOR: RENATO P. DE AZEVEDO

Data e Local da Defesa: Santa Maria, 08 de janeiro de 2014.

O rápido crescimento da rede mundial de computadores nos últimos anos gerou a necessidade de mais endereços IP. Então, torna-se inevitável a criação de um novo protocolo para resolver a escassez de endereços do protocolo IPv4. Desta forma, uma medida foi tomada, surgindo assim, o protocolo IPv6, o qual tornou-se a solução para este problema, bem como, contribuiu para a adição de novos recursos, como por exemplo, arquitetura hierárquica e novo formato de cabeçalho, dentre outras funcionalidades.

É inelutável que ocorra a migração entre os protocolos IPv4 e IPv6. Sendo assim, visando garantir as funcionalidades da Internet, esta mudança deve ser feita de forma gradual. O processo de transição entre os protocolos já está gradativamente em implantação, no entanto, ainda levará tempo para substituição total da tecnologia.

Sabendo que o andamento do processo de migração é um procedimento vagaroso, faz-se necessário o uso de mecanismos para que o protocolo IPv6 possa ser introduzido neste meio de profissionais da área de tecnologia, em ambientes corporativos e, consequentemente, atingir o escopo global de utilização.

Este trabalho propõe englobar funcionalidades do protocolo IPv6, bem como, suas características, estrutura do protocolo, endereçamento, roteamento e métodos de implantação, para aplicação de um projeto de migração de uma rede IPv4 para sua nova versão, IPv6.

Palavras-Chave: Tunelamento. IPv4. IPv6. Transição.

ABSTRACT

COMPLETION OF COURSE WORK
SUPERIOR COURSE OF TECHNOLOGY IN COMPUTER NETWORKS
FEDERAL UNIVERSITY OF SANTA MARIA

IMPLEMENTATION OF A NETWORK USING THE STANDARDS OF PROTOCOL IPv6

AUTHOR: RAISSA MONEGO PEDROZO

ADVISER: SIMONE REGINA CEOLIN

CO ADVISER: RENATO P. DE AZEVEDO

Defense Place and Date: Santa Maria, January 8th, 2014.

The rapid growth of the World Wide Web in recent years has created the need for more IP addresses. So, it becomes inevitable to create a new protocol to address the shortage of IPv4 protocol addresses. Thus, a measurement was taken, giving rise to the IPv6 protocol, which has become the solution to this problem, as well as contributed to the addition of new features, such as hierarchical architecture and new header format, among other features.

It is inevitable that migration occurs between the IPv4 and IPv6 protocols. Therefore, to ensure the functionality of the Internet, this change should be made gradually. The transition between the protocols is already being implemented gradually, however, it will take time to complete substitution of technology.

Knowing that the progress of the migration process is a slow procedure, it is necessary to use mechanisms for IPv6 can be introduced through this technology professionals in corporate environments and consequently achieve the global scope of use.

This paper proposes include IPv6 functionality as well as its characteristics, protocol structure, addressing, routing and deployment methods for the application of a migration project from an IPv4 network to the new version, IPv6.

Keywords: Tunneling. IPv4. IPv6. Transition.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

BIA	-	*Bump in the API*
BIND	-	*Berkeley Internet Name Domain*
BIS	-	*Bump in the Stack*
CANTIP	-	*Common Architecture for the Internet*
CIDR	-	*Classless Inter-domain Routing*
DHCP	-	*Dynamic Host Configuration Protocol*
DHCPv6	-	*Dynamic Host Configuration Protocol version 6*
DNS	-	*Domain Name System*
GRE	-	*Generic Routing Encapsulation*
HTTP	-	*Hypertext Transfer Protocol*
IANA	-	*Internet Assigned Numbers Authority*
IETF	-	*Internet Engineering Task Force*
IP	-	*Internet Protocol*
IPng	-	*Internet Protocol next generation*
IPv4	-	*Internet Protocol version 4*
IPv6	-	*Internet Protocol version 6*
LACNIC	-	*Latin American and Caribbean Internet Address Registry*
MTU	-	*Maximum Transmit Unit*
NAT	-	*Network Address Translation*
OSI	-	*Open Systems Interconnection*
RFC	-	*Requests for Comments*
RIR	-	*Regional Internet Registries*
SIP	-	*Simple Internet Protocol*
SIIT	-	*Stateless IP/ICMP Translation Algorithm*
TCP	-	*Transmission Control Protocol*
TI	-	Tecnologia da Informação
TRT	-	*Transport Relay Translator*
ToS	-	Tipo de Serviço
TUBA	-	TCP *and* UDP *with Bigger Addresses*
UDP	-	*User Datagram Protocol*
VoIPv6		*Voice over Internet Protocol v6*

SUMÁRIO

1 INTRODUÇÃO

No surgimento da *Internet* eram poucos os que acreditavam no sucesso desta, bem como na condição ou necessidade de ter um computador ou outro dispositivo que devesse conter um *Internet Protocol* (IP) para se comunicar. Passados os anos, notou-se um crescimento acelerado da *Internet*, expandindo assim, a necessidade de se dar mais atenção à serviços e tecnologias relacionadas à *Internet*. O IP é uma peça fundamental da *Internet*, sendo este protocolo capaz de endereçar dispositivos para que possa ser feita a comunicação entre quaisquer redes (HAGEN, 2006).

Segundo Comer (2007), pode-se observar na *Internet* uma mudança na característica comportamental. O cenário não é mais o mesmo, o uso de IPs tende a aumentar cada vez mais, e o número de endereços IPv4 esta acabando. Devido a demanda proveniente das necessidades do atual cenário tecnológico, o IPv4 necessita ser substituído em consequência do esgotamento de endereços e também de novas implementações que surgem no cenário de redes IP, como por exemplo, serviços voltados para aplicações móveis.

A criação do protocolo IPv6 visa suprir as deficiências do IPv4, sendo a principal delas, a falta de endereços IP. A evolução do protocolo traz consigo também algumas vantagens como mobilidade, segurança e padronização da tecnologia. Para suprir esta necessidade, na década de 90 foi criado o protocolo IPv6, gerando uma quantidade de endereços que pode ser considerada ilimitada para o cenário atual da rede mundial. O protocolo IPv4 será utilizado até que a transição para o protocolo IPv6 seja completamente concluída, durante a fase de transição estes protocolos devem coexistir (TANENBAUM; WETHERALL, 2011).

O presente trabalho pretende ampliar os conhecimentos e estudos da nova tecnologia IPv6, propondo a implantação de túneis IPv6 para que possa ser feita a comunicação entre uma rede IPv6 e a *Internet*. Serão apresentadas também, técnicas de transição entre os protocolos IPv4 e IPv6, como fundamentação teórica deste estudo.

A organização deste trabalho está segmentada em Capítulos. O Capítulo 2 apresenta a revisão bibliográfica, com conceitos relacionados. O Capítulo 3 relata os trabalhos relacionados a este estudo. O Capítulo 4 tem por escopo a proposta de desenvolvimento deste trabalho. O Capítulo 5 engloba os testes de validação de ambiente, juntamente com os resultados obtidos. Por fim, no Capítulo 6, encontram-se as considerações finais deste estudo, bem como, as propostas de trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

O escopo deste Capítulo centra-se nos principais alicerces para o desenvolvimento deste estudo, como por exemplo, descrever conceitos fundamentais para o compreendimento deste trabalho, bem como, elaborar uma pesquisa detalhada e realizar o levantamento de ferramentas que auxiliem na elaboração do mesmo.

Os conceitos referentes ao protocolo IPv6 e as técnicas de transição estão presentes no Capítulo 4. Além do exposto, é objetivo do capítulo apresentar e estudar técnicas de transição existentes, possibilitando a execução de um túnel para a comunicação entre os dois protocolos (IPv4 e IPv6) com a *Internet*.

A seguir são apresentados alguns conceitos importantes para elaboração deste trabalho. A Seção 2.1 expõe um modelo de sistema de comunicação. A Seção 2.2 descreve os princípios de protocolo e a pilha de protocolos TCP/IP. As Seções 2.3 e 2.4 descrevem os Protocolos IPv4 e IPv6, respectivamente. E por fim, nas Seções 2.5, 2.6 e 2.7 são apresentados os serviços DHCP, DNS e *Web Server*.

2.1 Sistema de Comunicação Básico

O princípio de um sistema de comunicação é o processo em que a informação é gerada, possibilitando o transporte de uma determinada informação através de um meio de comunicação, desde seu ponto de origem até o seu destinatário. A origem e o destinatário, no caso de redes de computadores, são equipamentos de tecnologia da informação.

A Figura 1 ilustra como se da o funcionamento de um sistema de comunicação. O transmissor é o responsável pelo envio de mensagens, através de um meio de comunicação até o receptor, cuja função é receber a mensagem.

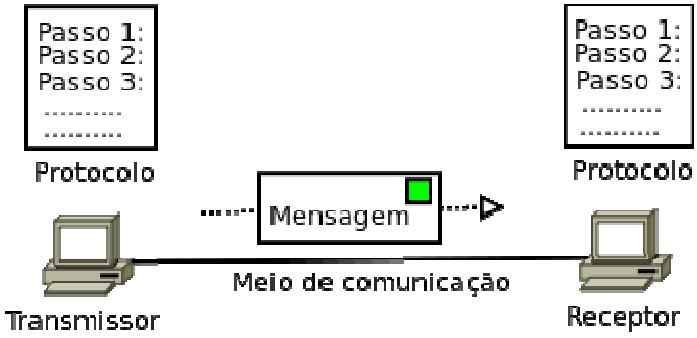

Figura 1: Sistema de Comunicação.
Fonte: wiki.sj.ifsc.edu.br/wiki/index.php/RES-2013-1.

2.2 Pilha de Protocolos TCP/IP

Para que computadores em rede possam trocar informações entre si é necessário que todos adotem as mesmas regras para o envio e o recebimento de informações. Esta coleção de regras é chamada de Protocolo, ou seja, para a troca de informações os computadores necessitam estar utilizando o mesmo protocolo de comunicação (KUROSE; ROSS, 2010).

Um protocolo pode ser definido como um conjunto de regras, controlando a troca de dados entre dois ou mais dispositivos (COMER, 2007). A seguir, serão especificados os principais protocolos utilizados no desenvolvimento deste trabalho.

A *Internet* possui aplicação prática de conectividade de redes de tecnologias distintas. Essa conectividade foi conseguida pelo uso do conjunto de protocolos TCP/IP. Estes protocolos foram criados com o intuito de realizar a intercomunicação de computadores. O TCP/IP executa essa conectividade em nível de rede, o que permite a comunicação entre aplicações em computadores de redes distintas sem a necessidade de conhecimento da topologia envolvida nesse processo (COMER, 2005).

Segundo Stallings (2005), o *Transmission Control Protocol* (TCP) tem como principal objetivo realizar a comunicação entre aplicações entre dois dispositivos diferentes. O protocolo TCP é um protocolo de nível de transporte muito utilizado, trabalha com mensagens de reconhecimento, especificação do formato da informação e mecanismos de segurança, possibilitando também o uso de várias aplicações voltadas à conversação.

No TCP/IP, o *Internet Protocol* (IP) especifica o endereçamento. Para transmitir informações através de uma rede TCP/IP, um computador deve conhecer o endereço IP do computador para o qual as informações serão enviadas. Sua função é transportar os datagramas de uma rede a outra na *Internet*. Ele é um protocolo de transmissão não orientada

à conexão, e por ser mais básico, não apresenta muitas características do TCP. Vale ressaltar que tais protocolos tem funções distintas, não sendo objetos de comparação, um com o outro.

A configuração destes protocolos tem como função controlar como a informação é passada de uma rede a outra, e como manusear o endereçamento contido nos pacotes. Outra característica é a flexibilidade de adaptação às tecnologias de redes existentes e futuras (como por exemplo a fibra ótica, uma tecnologia física existente), que é possível porque o TCP/IP foi concebido de forma independente das tecnologias de redes (COMER, 2005).

2.2 Protocolo IPv4

No início, o escopo da *Internet* era bastante reduzido, sendo utilizada basicamente em pesquisas de universidades, em algumas empresas e nas forças armadas dos Estados Unidos (TANENBAUM; WETHERALL, 2011). Nessa época seus inventores não tinham consciência de que este projeto desencadearia enormes proporções vivenciadas hoje. Sendo assim, diversos problemas foram originados concomitantemente com a expansão, como por exemplo, problemas com segurança de informações e a falta de endereçamentos IP (HAGEN, 2006).

Um *host* é uma máquina conectada na rede, representando um nodo nesta rede. Tal equipamento pode oferecer informações, serviços e aplicações para usuários. Na *Internet*, cada *host* tem um endereço *Internet Protocol* (IP), que codifica seu número de rede e seu número de *host*. Esta é uma combinação única, ou seja, duas máquinas conectadas a *Internet* não possuem o mesmo endereço IP.

O protocolo IPv4 é uma versão do protocolo IP, assim como o IPv6. A distribuição de endereços IPv4 foi feita de maneira desigual. Mesmo que a divisão de endereços IPv4 fosse feita de uma maneira igualitária para todas as regiões do mundo, hoje estaríamos sujeitos do mesmo jeito ao esgotamento de endereços IP (FLORENTINO, 2012).

Além do problema com o limite de endereços IP, o aumento da tabela de roteamento é outro fator importante para a substituição do protocolo IPv4, já que os roteadores devem manter sempre informações completas do roteamento da *Internet*. Se o número de entradas nas tabelas de roteamento crescerem indiscriminadamente, os núcleos dos roteadores serão forçados a pular rotas e porções da *Internet* ficarão inalcançáveis.

Outro fator é que, o protocolo não dispõe de nenhum mecanismo de segurança nativo para os dados que são transmitidos pela rede, possibilitando assim que um invasor intercepte uma conexão e tenha acesso aos dados.

Quando iniciou a comercialização da *Internet* as conexões eram feitas somente para computadores, não haviam celulares, *tablets*, entre outros. Em poucos anos os eletrodomésticos e eletrônicos em geral provavelmente também estarão conectados a *Internet*, ocasionando uma demanda maior de endereços IP para uso. Surge então a nova versão do protocolo IP, o IPv6, que busca acabar com a escassez de endereços.

É importante salientar que o IPv6 não é um complemento do IPv4, ou seja, não são compatíveis. O protocolo IPv6 é o substituto do IPv4, criado para resolver os problemas do IPv4, como a carência de endereços IP, podendo os dois funcionarem paralelamente com a ajuda de mecanismos de transição.

2.2.1 Endereçamento e Cabeçalho IPv4

A distribuição dos números IPs é controlada por diversas entidades, entre elas a *Internet Assigned Numbers Authority* (IANA). A responsabilidade sobre uma parte dos endereços é delegada pela IANA para cada um dos Registros Regionais da *Internet* (RIRs), que os gerenciam e distribuem dentro de suas respectivas regiões geográficas. No Brasil quem cumpre esta função é o *Latin American and Caribbean Internet Address Registry* (LACNIC).

A IANA fez um padrão de divisão de endereços, em classes para diminuir o desperdício de endereços. Este é apresentado no Quadro 1.

ENDEREÇOS IP PRIVADOS			
Classes	Número de End. por Rede	Intervalos de endereçamentos	Total de *Hosts*
Classe A	Até 256	0.0.0.0 até 127.0.0.0	Até 16.777.216
Classe B	Até 65.536	128.0.0.0 até 191.255.0.0	Até 65.536
Classe C	Até 16.777.216	192.0.0.0 até 223.255.255.0	Até 256

Quadro 1: Classes de endereçamento IPv4.
Fonte: MSDN - Microsoft.

Conforme o órgão padronizador da segmentação de endereços (IANA), estes devem atender aos seguintes requisitos:

- Os endereços IP da classe A são usados em locais onde é necessária somente uma rede, com grande quantidade de máquinas nela.

- Os endereços IP da classe B são usados nos casos onde a quantidade de redes é equivalente à quantidade de computadores.

- Os endereços IP da classe C são usados em locais que requerem grande quantidade de redes, mas com poucas máquinas em cada uma.

Com o endereçamento de classes cheias, muitas vezes para uma organização esse número de endereços seria muito pequeno ou, relativamente grande para uma só organização, o que indica falha na forma como é feita a divisão em classes (*classful*).

A Figura 2 apresenta um datagrama IPv4, que nada mais é do que um pacote da camada de rede , que consiste em um cabeçalho e a área de dados (KUROSE; ROSS, 2010). O campo do cabeçalho reservado para endereçamento possui 32 *bits*. Cada campo é especificado a seguir.

Número da versão: Quatro *bits* que especificam a versão do protocolo IP;

Comprimento do cabeçalho: Necessários para determinar onde começam os dados no datagrama, pois este pode conter um número variável de opções;

Tipo de Serviço (ToS): Estes *bits* estão incluídos no cabeçalho para poder diferenciar os diferentes tipos de datagramas IP que devem ser distinguidos uns dos outros. Como por exemplo, distinguir datagramas de tempo real (VoIP) de um outro que não é *real-time* (FTP);

Comprimento do Datagrama: Comprimento total do datagrama (cabeçalho e dados);

Identificador, *flags*, deslocamento de fragmentação: Estes fazem parte da fragmentação do protocolo IP. O campo identificador, é utilizado para identificar fragmentos do datagrama IP original. O campo *flags* serve para controlar ou identificar fragmentos. Por fim, o campo deslocamento de fragmentação mostra a posição relativa de um fragmento com relação ao datagrama como um todo;

Tempo de vida: Garante que datagramas não fiquem circulando na rede;

Protocolo: Indica o protocolo de camada de transporte especifico ao qual a porção de dados desse datagrama IP deverá ser passada. Só é usado quando um datagrama chega ao seu destino final;

Soma de verificação do cabeçalho: Auxilia um roteador na detecção de erros de *bits* em um datagrama IP recebido;

Endereços IP de fonte e destino: Quando é criado um datagrama, é inserido seu endereço IP no campo de endereço de fonte IP e também o endereço de destino final;

Opções: Permite que um cabeçalho seja ampliado;

Dados: Também conhecido como carga útil, contém o segmento da camada de transporte (TCP ou UDP) a ser entregue ao destino. Este campo pode carregar também outros tipos de dados, como por exemplo, mensagens ICMP.

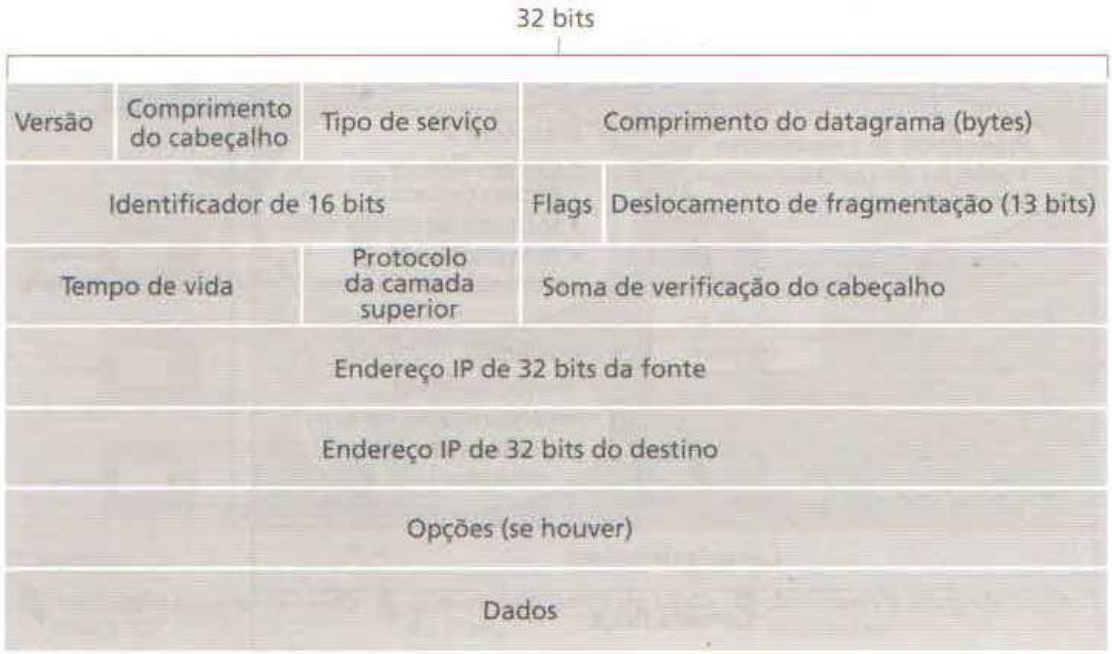

Figura 2: Formato do datagrama IPv4.
Fonte: Kurose; Ross (2010, p. 248).

O IPv4 possibilita um máximo de 4.294.967.296 endereços distintos, considerada uma quantidade suficiente para identificar todos os computadores na rede e suportar o surgimento de novas sub-redes. No entanto, com o acelerado crescimento da *Internet*, surgiu o problema da escassez dos endereços IPv4.

O fim de endereços IPv4 já é fato. Se provedores de *Internet* e também as empresas não iniciarem a migração para essa nova realidade, a adoção do IPv6, poderão gastar muito mais no futuro e ainda correrem o risco de falhas e não estarem prontos para competir com rivais que se anteciparem na transição (FLORENTINO, 2012).

Algumas medidas foram adotadas para postergar o término de endereços IPv4, dentre elas estão o *Classless Inter-domain Routing* (CIDR) e *Network Address Translation* (NAT). O CIDR tem como objetivo o fim de classes de endereçamento, permitindo alocar blocos com o tamanho necessário a cada rede, além desta funcionalidade, pode ser feita a agregação de maneira a permitir redução no tamanho da tabela de roteamento. Basicamente o CIDR tem por função alocar os endereços IP restantes em blocos de tamanho variável, sem levar em consideração as classes. A técnica NAT foi desenvolvida no intuito de um único endereço IP,

ou um pequeno número deles, permita que diversos *hosts*, de uma rede local (LAN), possam trafegar na *Internet* (MOREIRAS, 2012).

Segundo Tanenbaum e Wetherall (2011), a utilização destas técnicas são apropriadas como medidas paliativas para a escassez de endereçamento. Sendo que técnica NAT apresenta algumas desvantagens em relação ao proveito obtido com seu uso, como por exemplo, não permite conexão direta entre *hosts,* outro problema é a baixa escalabilidade, acarretando em alta taxa de processamento. E a eliminação das classes com a utilização de CIDR torna o encaminhamento mais complicado.

2.3 Protocolo IPv6

Diante da previsão de esgotamento de endereços IPv4 e da necessidade de novos endereços, em 1992, foi criado o grupo *Internet Protocol next generation* (IPng) pela *Internet Engineering Task Force* (IETF), com o intuito de realizar pesquisas para a geração de uma nova versão do protocolo IP, levando em consideração as seguinte características, as quais são principais para sua elaboração, segurança, escalabilidade, configuração e administração de rede, políticas de roteamento, transição e padronização da tecnologia (MOREIRAS, 2012).

Entre projetos pesquisados, foram analisadas e publicadas na RFC 1752, as três principais propostas de protocolos, *TCP and UDP with Bigger Addresses* (TUBA), *Common Architecture for the Internet* (CANTIP) e *Simple Internet Protocol* (SIP). As propostas estão apresentadas abaixo:

TUBA: aumentar o espaço para endereçamento IPv4 e torná-lo mais hierárquico, para evitar a necessidade de alteração dos protocolos da camada de transporte e aplicação. A migração deveria ser simples e em longo prazo, baseando-se na atualização de *hosts* e servidores DNS.

CANTIP: criado para ser um protocolo de convergência, permite a qualquer protocolo da camada de transporte ser executado sobre qualquer protocolo de camada de rede. Para isto é criado um ambiente comum entre os protocolos da *Internet*.

SIP: se fosse aprovado o projeto, este seria uma evolução do IPv4, sem haver mudanças drásticas e mantendo a interoperabilidade com a versão 4 do protocolo IP, forneceria uma plataforma para novas funcionalidades da *Internet*, aumentando o espaço para endereçamento de 32 *bits* para 64 *bits*.

Os projetos tiveram seus prós e contras, porém, o CANTIP foi descartado por não estar em pleno entendimento. O novo protocolo foi baseado em uma versão do SIP, utilizando 128 *bits* de endereçamento, em detrimento aos 32 *bits* do IPv4, e também usando elementos de transição do TUBA. A partir disto, foi criada a nova versão do protocolo IP, chamada de IPv6. A Figura 3 apresenta este histórico.

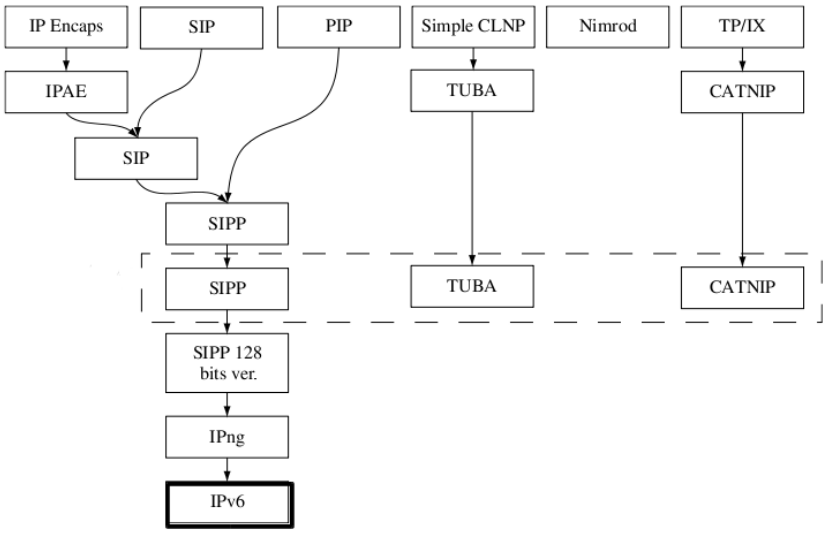

Figura 3: Criação do IPv6
Fonte: ipv6.br

Algumas características que foram sucesso para o IPv4, continuam contribuindo com o IPv6. Uma delas é o encaminhamento de pacotes sem conexão, chamados de datagramas. A RFC 2460 (1998), especifica as principais diferenças entre o protocolo IPv4 e seu sucessor, o IPv6:

Expansão na capacidade de endereçamento: o IPv6 teve uma mudança significativa em relação ao endereçamento. Enquanto o IPv4 utiliza apenas 32 *bits* de endereçamento, o IPv6 usa 128 *bits* de endereço, proporcionando uma gama de $3{,}4 \times 10^{38}$ novos endereços e também permitindo um endereço único para cada *host* no mundo;

Cabeçalho em formato simplificado: para reduzir o custo de processamento dos pacotes, alguns campos do cabeçalho IPv4 foram abolidos e outros são opcionais;

Novos cabeçalhos de extensão: as opções não estão mais no cabeçalho base, o que gera maior capacidade de introdução de novas opções no futuro e também permite um roteamento mais eficaz;

Identificar fluxo de dados: existem pacotes que requerem tratamentos especiais. Esta característica permite identificar estes pacotes.

Além da capacidade de atribuição de endereços, o IPv6 ainda permite novas funcionalidades como:

- Arquitetura hierárquica, encaminhamento eficaz de pacotes;
- Fornecimento de endereços válidos na *Internet*, a todos os dispositivos conectados;
- Arquitetura fim-a-fim;

A alta disponibilidade de endereços e prefixos de rede prove flexibilidade à arquitetura das redes e permite a hierarquia. Um prefixo de rede pode endereçar um país ou até mesmo um continente, dividindo estes em diversos níveis. Assim, os provedores poderão utilizar apenas um prefixo de rede para todos os seus usuários, anunciando somente uma rota a outros provedores, hierarquizando a estrutura de rede e reduzindo o tamanho de tabelas de roteamento.

Segundo Liu (2011), apenas 9% de Sistemas Autônomos de Internet possuem rotas para redes IPv4 e IPv6, o que torna difícil a utilização do novo protocolo da *Internet*. Nos EUA, onde o governo exigiu que agencias governamentais migrem suas redes para o novo protocolo, assim como *Sites* e Servidores terão de estar acessíveis a partir do novo protocolo, ou até mesmo, só poderão ser acessados utilizando o IPv6. É sabido que algumas empresas necessitam implementar o IPv6 rapidamente, como é o caso de operadoras móveis.

2.3.1 Datagrama IPv6

O datagrama do IPv6 é um pouco diferente do IPv4, principalmente por ter um cabeçalho base, seguido ou não por cabeçalhos de extensão, podendo estes, as vezes, serem maiores que o cabeçalho base. A Figura 4 ilustra a forma geral do datagrama IPv6.

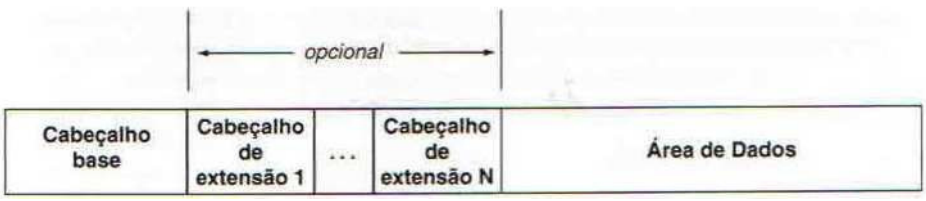

Figura 4: Forma geral do datagrama IPv6
Fonte: Comer (2007, p.322).

O cabeçalho base utiliza campos essenciais para a transmissão do datagrama. O cabeçalho de extensão apresenta campos adicionais, o qual não necessita ser processado por roteadores intermediários, fazendo com que o datagrama IPv6 seja mais simples. A maior parte do espaço do cabeçalho base IPv6 é destinado ao receptor e ao remetente. Além destes, o cabeçalho base inclui mais seis campos. O formato do datagrama IPv6 é apresentado na Figura 5.

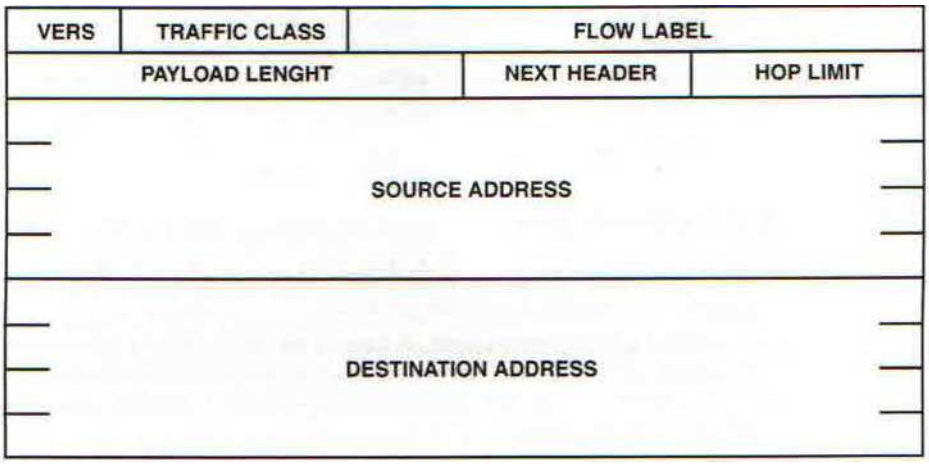

Figura 5: Cabeçalho Base IPv6
Fonte: Adaptado Comer (2007, p. 322).

Alguns campos do cabeçalho IPv6 são iguais ao do IPv4, e outros, foram retirados por serem irrelevantes. A descrição de cada campo é feita abaixo segundo Comer (2007):

Vers: Identifica a versão do protocolo, no caso IPv6.

Traffic Class: Identifica os pacotes por classes de serviços ou prioridade. Tem a mesma funcionalidade que o campo "Tipo de Serviço do IPv4".

Flow Label: Indica pacotes com mesmo fluxo de comunicação, separando o fluxo de cada uma das aplicações para realizar o tratamento especifico de cada pacote.

Payload Lenght: Substitui o campo "Tamanho total do IPv4", identifica a carga de dados transportados.

Next Header: Especifica o tipo de informação que segue o cabeçalho atual. Se houver cabeçalho de extensão, este campo dirá o tipo deste cabeçalho. No IPv4, era chamado de "Protocolo".

Hop Limit: Este campo indica o máximo de roteadores que o pacote pode passar. O datagrama será descartado se o *Hop Limit* chegar à zero antes que seu destino seja alcançado.

Source Address: Identifica o endereço de origem do pacote.

Destination Address: Utilizado para identificar o endereço de destino do pacote.

A Figura a seguir, apresenta a mudança de algumas funcionalidades do cabeçalho IPv6.

IPv4		IPv6
Tipo de Serviço	→	Classe de Tráfego
Tamanho Total	→	Tamanho dos Dados
Tempo de Vida (TTL)	→	Limite de Encaminhamento
Protocolo	→	Próximo Cabeçalho

Figura 6: Mudanças entres cabeçalhos IPv4/IPv6
Fonte: ipv6.br

As alterações no cabeçalho IPv4 para IPv6 permitiram que mesmo com quatro vezes maior o espaço de endereçamento do IPv6, o tamanho total do cabeçalho seja somente duas vezes maior que a sua versão anterior, ou seja, oito campos e 40 *Bytes* fixos.

2.3.1.1 Cabeçalhos de Extensão

Diferentemente do IPv4, onde o datagrama possui o campo “Opções”, no IPv6 este campo é tratado com Cabeçalho de Extensão, onde são manuseadas as opções adicionais.

Os cabeçalhos de extensão localizam-se entre o cabeçalho base e o cabeçalho da camada de transporte e não havendo nem tamanho fixo, nem quantidade estipulada. Se houverem vários cabeçalhos de extensão no mesmo pacote, estes formaram uma cadeia de cabeçalhos.

A utilização de cabeçalhos de extensão é feita para aumentar a velocidade de processamento nos roteadores, visto que o envio destes cabeçalhos deve seguir uma sequência definida, evitando que os nós intermediários tenham que processar toda a cadeia de cabeçalhos.

Segundo Moreiras (2012), os cabeçalhos importantes para todos os nós envolvidos no roteamento devem ser colocados em primeiro lugar e os importantes, apenas para o destinatário final, são colocados no fim da cadeia. A vantagem desta sequência é que o nó pode parar de processar os cabeçalhos assim que encontrar algum cabeçalho de extensão dedicado ao destino final, havendo certeza de não existirem mais cabeçalhos importantes a seguir. Estes cabeçalhos podem ser incluídos sem a necessidade de alteração no cabeçalho base. Conforme a RFC 2460, estão definidos alguns cabeçalhos de extensão, são eles:

Hop-by-Hop Options: Leva informações que devem ser examinadas por todos os nós intermediários do caminho até chegar ao seu destino. Identificado pelo número 00 no campo *Next Header* do cabeçalho base.

Destination Options: É processado apenas pelo nó de destino do pacote. Identificado pelo número 60.

Routing: Lista os nós intermediários que devem ser avistados até o pacote chegar ao seu destino final. Identificado pelo número 43.

Fragmentation: Deve ser utilizado quando um pacote IPv6 é maior que a *Maximum Transmit Unit* (MTU) do caminho. Identificado pelo número 44.

Authentication Header e Encapsulating Security Payload: fornece confidencialidade, autenticação e integridade do conteúdo do datagrama IPv6.

2.3.2 Endereçamento IPv6

Diferentemente do IPv4 que utiliza 32 *bits* para endereçamento totalizando aproximadamente 4,3 bilhões de endereços, o IPv6 utiliza 128 *bits*, para que cada *host* na *Internet* possa ter um endereço único, gerando 79 octilhões de vezes o número de endereços IPv4 em endereços IPv6. Isto equivale à, aproximadamente, 340 undecilhões de endereços.

O IPv4 representa seus endereços na forma decimal, sendo os 32 *bits* divididos em quatro grupos de 8 *bits* separados por ponto, como por exemplo 192.168.1.1. No IPv6, os endereços são representados de forma hexadecimal (0-F), divididos em oito grupos de 16 *bits*, separados por ponto e vírgula.

Um exemplo de endereço IPv6:

2001:C0A8:0000:A026:F0CA:0000:B3C4:0A0A

É importante lembrar que os caracteres podem ser tanto maiúsculos quanto minúsculos, além disso, é possível omitir zeros à esquerda, e também uma sequência de quatro zeros pode ser substituída somente por dois pontos, como mostra o exemplo abaixo:

2001:C0A8:0:A026:F0CA::B3C4:A0A

A representação dos prefixos de rede é apresentada na forma "endereço-IPv6/tamanho do prefixo", onde o tamanho do prefixo especifica a quantidade de *bits* a esquerda do endereço que abrangem o prefixo. Isso possibilita a inclusão de endereços de forma hierárquica, desta forma é possível diminuir o tamanho da tabela de roteamento e também tornar mais ágil o encaminhamento de pacotes.

Cada endereço IPv6 pertence a um de três tipos de endereços IPv6, que serão apresentados na Subseção a seguir.

2.3.2.1 Tipos de endereços IPv6

No IPv6, diferentemente do IPv4, não há endereço de *broadcast*, que é responsável por redirecionar um pacote a todos os nós da rede, esta função no IPv6 é atribuída a endereços *multicast*, que fazem parte dos três tipos de endereços definidos no IPv6. São eles:

Unicast: Identifica uma única interface, são utilizados para a comunicação entre dois nós, de modo que, cada pacote enviado a um endereço *unicast* é entregue a interface associada a este endereço. Como exemplo de aplicação, podem ser citados telefones VoIPv6.

Existem alguns tipos de endereços *unicast* IPv6, como por exemplo, *Global Unicast*, *Link-Local* e *Unique-Local* os quais serão especificados abaixo.

- *Global Unicast*: roteável globalmente e acessível na Internet IPv6. Formando por três partes: prefixo de roteamento global, identifica o tamanho do bloco atribuído a uma rede; identificação da sub-rede, identifica um enlace na rede; e identificação da interface, identifica uma interface dentro de um enlace.
- *Link Local*: utilizado apenas no enlace especifico onde a interface esta conectada.
- *Unique Local Address* (ULA): utilizados para comunicações locais, geralmente dentro de um mesmo enlace.

Anycast: Identifica um conjunto de interfaces, cada pacote enviado a um endereço *anycast* é encaminhado à interface do grupo mais próxima da origem do pacote. Um endereço *anycast* não pode ser configurado em um *host*, ele é associado somente a roteadores. É utilizado para fazer balanceamento de carga e também para descobrir serviços na rede, como DNS, HTTP, etc.

Multicast: Identifica um conjunto de interfaces, porém um pacote quando enviado a um endereço *multicast* é enviado a todas as interfaces ligadas a este endereço. Estes endereços são utilizados para identificar grupos de interfaces, estas interfaces podem pertencer a mais de um grupo. Todos os pacotes enviados a um endereço *multicast* são entregues a todas as interfaces que compõem o grupo.

2.4 Técnicas de Transição

A coexistência dos dois protocolos ainda se dará por muito tempo, para que não cause grandes impactos no meio tecnológico. Por muito tempo, os dois protocolos terão que permanecer na rede simultaneamente.

Com o intuito de facilitar o processo de transição entre as duas versões do Protocolo Internet, algumas técnicas foram desenvolvidas para que toda a base das redes instaladas sobre IPv4 mantenha-se compatível com o protocolo IPv6 (SANTOS, 2008).

São objetivos da transição:

- Roteadores e máquinas devem ter seus serviços de rede trocados, sem que todos os outros no mundo tenham que mudar ao mesmo tempo;
- Nodos IPv6 devem poder se comunicar com outros nós IPv6, mesmo que a infraestrutura entre eles seja IPv4.

Os mecanismos de transição podem ser classificados em três e serão especificados a seguir, sendo eles: Pilha Dupla, Tradução e Túneis.

- **Pilha Dupla**

Esta técnica faz com que *hosts* possuam tanto suporte ao IPv4 quanto ao IPv6. Cada dispositivo deve ser configurado com um endereço IPv4 e um endereço IPv6. Este método

permite que os dispositivos estejam equipados com pilhas para os dois protocolos, assim os dispositivos terão capacidade de enviar e receber pacotes IPv4 e pacotes IPv6.

Um nó Pilha Dupla, irá portar-se como um nó IPv6 quando necessário e/ou como um nó IPv4 quando houver tal necessidade. Quando a comunicação for IPv6, este nó será IPv6. A mesma coisa ocorre com o protocolo IPv4, quando houver comunicação somente IPv4, este se comportará como um nó IPv4. Esta técnica não resolve o problema da escassez de endereços. O funcionamento da Pilha Dupla é apresentado na Figura 7.

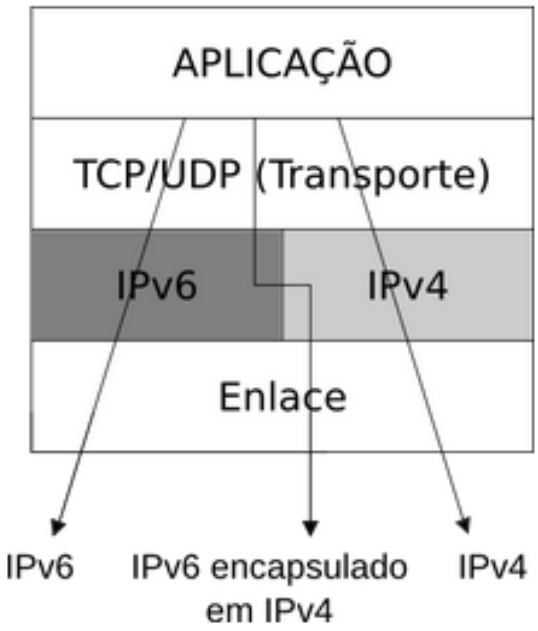

Figura 7: Funcionamento da Técnica de Pilha Dupla
Fonte: ipv6.br

Conforme demonstrado na parte superior da Figura 7, na camada de aplicação coexistem serviços de ambos protocolos, IPv4 e IPv6. Entretanto, como o cenário atual ainda é de ampla adoção do protocolo IPv4 e sabe-se que tal protocolo e seu sucessor não são compatíveis, uma medida para contornar tal situação (de acessar serviços IPv6 utilizando o procotolo IPv4 e vice-versa) é através do encapsulamento de pacotes. No caso, o demonstrado na Figura 9 é o IPv6 encapsulado em IPv4.

Também pode-se observar que na camada de transporte os protocolos utilizados (TCP e UDP) são os mesmos para ambas aplicações, tanto em IPv4 quando em IPv6. E na camada inferior, a de rede, encontram-se separadamente os dois protocolos. Por fim, constata-se, na Figura 9, que os enlaces utilizados para os dois protocolos são os mesmos.

A implementação de pilha dupla traz consigo a vantagem da desativação da pilha IPv4 quando for necessário, sem que a pilha IPv6 sofra qualquer alteração. Esta técnica pode estar aliada a outros mecanismos de transição (túnel e tradução). Porém, ocasiona uso adicional de memória e também de processando de CPU, devido a duas pilhas de protocolo funcionando concomitantemente, o serviço de DNS também se faz necessário para os dois protocolos, um para o protocolo IPv4 e outro para IPv6.

- **Tradução**

A técnica de tradução consiste em permitir comunicação entre nós com suporte apenas a IPv6 com nós que suportam apenas IPv4. Para tanto é feita a tradução de cabeçalhos IPv4 em cabeçalhos IPv6 e cabeçalhos IPv6 em IPv4, realizando conversões de endereços. As principais técnicas de tradução são SIIT, BIS, BIA e TRT.

Segundo Moreiras (2012) o uso desta técnica implica em não utilizar novas funcionalidades do IPv6, quando efetuada a comunicação com equipamentos que utilizam IPv4. A Figura a seguir apresenta como é feita a tradução.

Figura 8: Técnica de Tradução
Fonte: CGI.br

O mecanismo de tradução deve ser utilizado apenas quando não for possível a implementação de outra técnica de transição, ou como solução temporária. A principal característica da tradução é permitir que hospedeiros IPv6 possam comunicar diretamente com *host* IPv4 e vice-versa. Todavia apresenta limitações na topologia, pois respostas precisam ser recebidas através do mesmo roteador NAT na qual foram enviadas. Outra desvantagem é não suportar recursos do IPv6, como segurança.

- **Túneis**

A técnica de tunelamento consiste na transmissão de pacotes IPv6 sobre a *Internet* IPv4 já existente, sem que seja preciso realizar qualquer mudança em mecanismos de roteamento. Seu funcionamento consiste em encapsular o pacote IPv6 em um pacote IPv4. A Figura 9 ilustra o encapsulamento.

Figura 9: Encapsulamento IPv4/IPv6
Fonte: CGI.br

Os túneis podem ser classificados como roteador-a-roteador, *host*-a-roteador, roteador-a-*host*, e *host*-a-*host*.

A técnica de tunelamento é bastante útil, podendo fazer a conexão de *hosts* com IPv6 através de túneis, não sendo necessária a utilização de um provedor de *Internet* IPv6, pois é utilizada a infra estrutura IPv4 já existente. Desvantagens são principalmente o tempo e poder de processamento da CPU para encapsular e desencapsular pacotes.

As Subseções apresentadas abaixo demonstram três modelos de tunelamento estudados, para que posteriormente, seja possível a escolha do mais apropriado para a proposta deste trabalho.

2.4.1 *Tunnel Broker*

O *Tunnel Broker* baseia-se em servidores dedicados, tornando-se útil para a incorporação do IPv6. É uma técnica que permite que dispositivos, ou uma rede IPv4, obtenham conectividade IPv6, por meio de um túnel funcionando como um provedor.

Para ter acesso a um *Tunnel Broker*, é necessária uma requisição a um provedor de túnel, e realizar *download* de um *software* de configuração ou através de um comando <*sudo apt-get install gogo6*>. Após isso deve ser feita a instalação das dependências necessárias para sua utilização, em seguida finalizar a instalação. A Figura 10 mostra a topologia lógica de um

Tunnel Broker, onde o cliente solicita um túnel, é cadastrado no servidor do túnel e assim, o túnel é estabelecido.

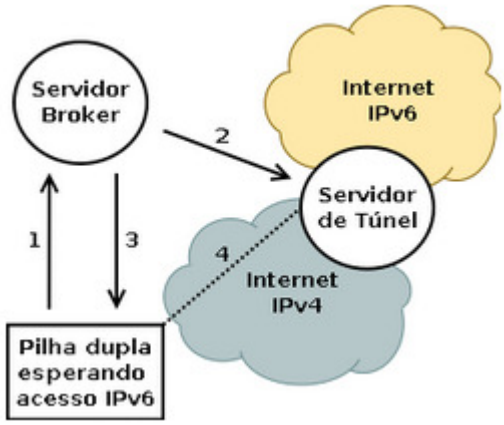

Figura 10: Topologia lógica *Tunnel Broker*
Fonte: ipv6.br

Conforme o diagrama, em 1 o cliente pilha dupla faz a solicitação do túnel via protocolo IPv4, logo em 2 o servidor *broker* efetua o cadastro do usuário no servidor de túnel na *Internet* (provedor). Em 3, o servidor broker retorna ao cliente parâmetros para criação do túnel. E por fim, em 4 obtem-se o estabelecimento do túnel.

A utilização de *Tunnel Broker* é recomendada para usuários que querem testar o IPv6, ou começar um processo de implantação em suas redes, onde seus provedores de acesso ainda não oferecem suporte ao protocolo. Sua implantação é bastante complexa, por não haverem *softwares* abertos para a funcionalidade de Servidor *Broker*.

2.3.2 Tunnel 6*over*4

O túnel 6*over*4 é apresentado de forma manual entre nós IPv6 conectados a uma rede IPv4. A técnica de tunelamento é conhecida por fazer o encapsulamento de um pacote IPv6 em um pacote IPv4. O protocolo 6*in*4 é o responsável por fazer o encapsulamento de um pacote juntamente com o protocolo 41, o qual consiste em encapsular um pacote IPv6 dentro de um pacote IPv4, adequando os endereços de destino e origem para IPv4 e inserindo no

cabeçalho o tipo 41. Processando o tipo 41, será removido o cabeçalho IPv4 e o pacote IPv6 é tratado. A Figura 11 exemplifica o funcionamento do 6*in*4.

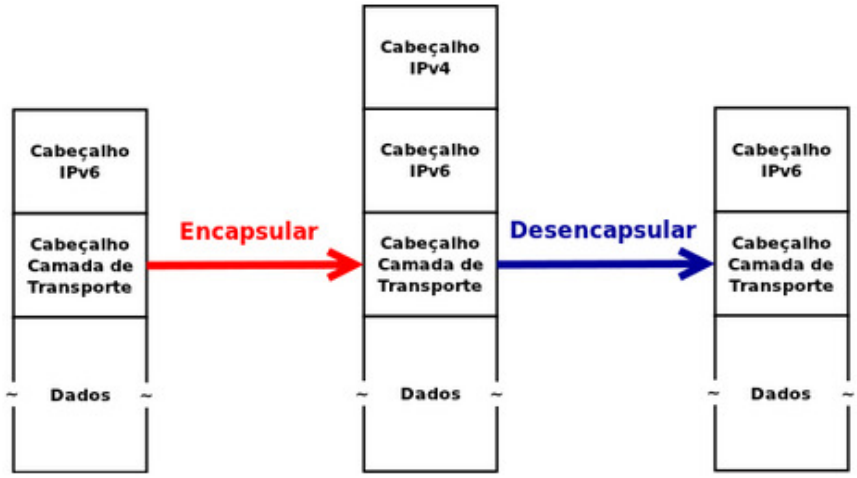

Figura 11: Técnica 6*in*4
Fonte: ipv6.br

A configuração do túnel 6*over*4 consiste em definir os IP (IPv4) de origem e de destino, que serão utilizados nas pontas do túnel. Quando um pacote é recebido pelo seu destino, ele é desencapsulado. O 6*over*4, faz a comunicação entre duas redes, no caso IPv6, distintas, as quais estão conectadas a *Internet* IPv4. A Figura 12 exemplifica o túnel 6*over*4.

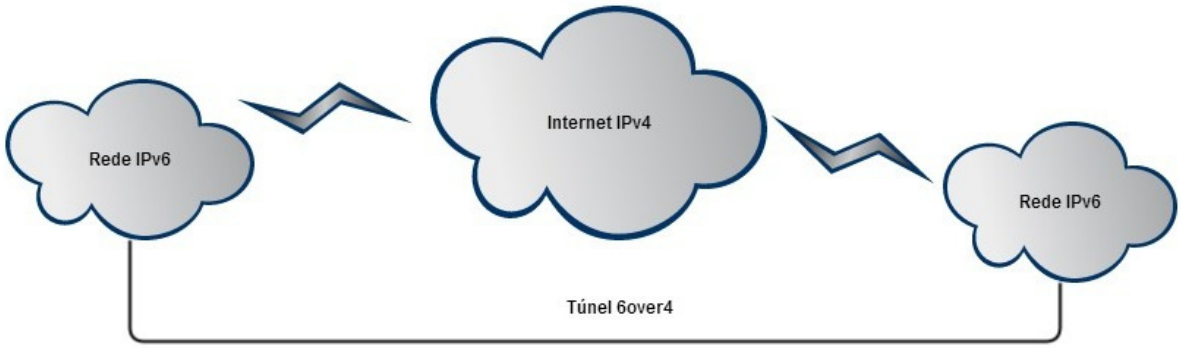

Figura 12: Túnel 6*over*4.
Fonte: Do Autor.

Técnicas de tunelamento fazem o encapsulamento de pacotes IPv6 em pacotes IPv4, ou vice-versa, permitindo que pacotes de um tipo, trafeguem por uma rede de outro tipo. Ao fazer este encapsulamento, assume-se no cabeçalho um protocolo do tipo 41, ou 6in4. Os túneis 6*over*4 utilizam este protocolo.

Exemplificando, estes túneis podem ser utilizados para contornar um equipamento ou enlace sem suporte a IPv6 numa rede, ou para criar túneis estáticos entre duas redes IPv6 através da *Internet* IPv4. Quanto ao seu modo de configuração, este é feito manualmente.

2.4.3 *Tunnel* GRE

O *Tunnel Generic Routing Encapsulation* (GRE) é um túnel estático, criado pela Cisco, com o intuito de encapsular vários tipos de protocolos, possibilitando a criação de um link ponto a ponto. Seu funcionamento e sua configuração é bastante semelhante ao túnel 6*over*4. É feito de forma manual, acarretando em transtornos na sua manutenção e gerenciamento. A Figura 13 mostra o encapsulamento do cabeçalho GRE.

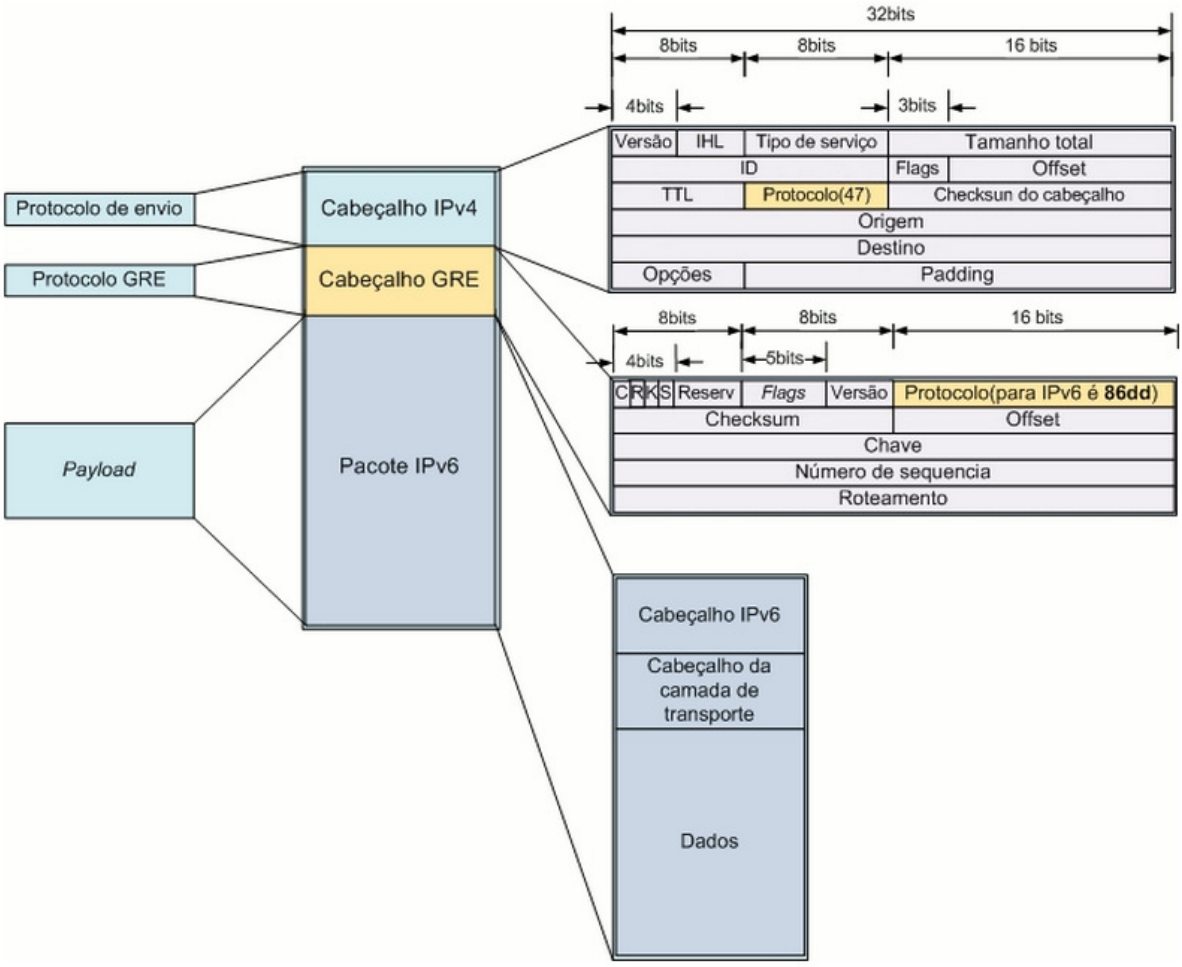

Figura 13: Encapsulamento do cabeçalho GRE
Fonte: ipv6.br

O encapsulamento do cabeçalho GRE, consiste em adicionar ao pacote original, o cabeçalho GRE e o IPv4, e enviar ao destino final. Chegando ao destino final o cabeçalho GRE e o cabeçalho IPv4 são removidos, deixando apenas o pacote original, encaminhando-o até o destino final. Na Figura 22 pôde-se observar o cabeçalho GRE encapsulado.

O funcionamento do *Tunnel* GRE consiste em adicionar um cabeçalho próprio (GRE+IPv4) aos pacotes à serem transmitidos e enviá-los ao destinatário, sendo assim, desencapsulados, retornando ao seu estado original.

Quanto ao seu método de configuração, este é manual (assim como no Tunnel 6*over*4), podendo gerar maiores trabalhos com manutenção e gerenciamento de túneis.

No Capítulo 4 são apresentadas as configurações necessárias para cada tipo de túnel.

2.5 Dynamic Host Configuration Protocol

Em uma rede de Arquitetura TCP/IP, todos computadores possuem um endereço IP distinto. O *Dynamic Host Configuration Protocol* (DHCP) por sua vez, é o protocolo que provê um meio para alocar estes endereços dinamicamente.

O DHCP é um serviço básico, que permite fácil utilização para as redes móveis e para quem necessita utilizar uma faixa de IPs limitada. Os endereços IP podem ser atribuídos de formas diferentes:

i) Configuração Manual: nesta configuração um endereço IP é atrelado manualmente à um *host* pelo administrador da rede. Para isso, é necessária a associação de um endereço existente no banco do servidor DHCP ao endereço MAC do adaptador de rede do *host*;

ii) Configuração Automática: o servidor DHCP é configurado de forma que quando um dispositivo acessar a rede, é atribuído um endereço IP automaticamente;

iii) Configuração Dinâmica: neste caso, um endereço IP é destinado a tal dispositivo e é necessária sua atualização periodicamente. Na configuração dinâmica é possível um mesmo endereço ser utilizado por equipamentos distintos em diferentes momentos, basta que, o equipamento deixe de utilizar este endereço para que quando solicitado um endereço novamente, este mesmo endereço possa ser fornecido para outro dispositivo. A Figura 14 apresenta o funcionamento do DHCP.

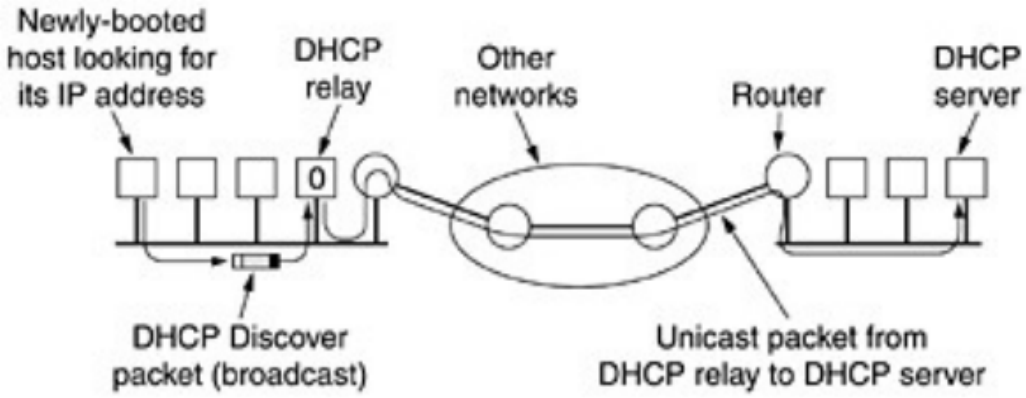

Figura 14: Funcionamento DHCP
Fonte: Tanenbaum (2011, p. 349).

O DHCP faz com que o endereçamento a máquinas seja um processo mais dinâmico do que estático. Geralmente, um novo usuário da rede solicita ao gerenciador um endereço IP válido. Esse usuário pode precisar desse endereço apenas esporadicamente ou até

temporariamente. Contudo, enquanto o endereço é atribuído a uma máquina, ninguém mais pode usá-lo (BUGALO et. al., 1999).

Para o desenvolvimento do presente estudo, será utilizado um servidor DHCPv6 configurado com distribuição de endereços dinâmicos, posteriormente, sua configuração e testes utilizando o mesmo serão especificadas.

2.6 Domain Name System

É sabido pela comunidade de profissionais de tecnologia da informação que o *Domain Name System* (DNS) administra endereços IP e nomes na *Internet*. Este procedimento é feito de forma transparente para o usuário final, deste modo, optou-se por utilizar um servidor deste tipo no presente estudo para resolução de nomes para a rede local implementada.

O protocolo DNS faz a criação de um esquema hierárquico de atribuição de nome baseado no domínio e de um sistema de bancos de dados distribuídos para implementar esse esquema de nomenclatura. Geralmente é usado para mapear nomes de *hosts* e destinos de mensagens de correio eletrônico em endereços IP (TANENBAUM; WETHERALL, 2011).

Nos próximos Capítulos, serão tratados o processo de desenvolvimento deste estudo. Sendo vistos trabalhos relacionados e, por conseguinte, a proposta deste estudo, contendo, os serviços implantados e as ferramentas necessárias para a execução do mesmo. Bem como, os procedimentos de instalação e configurações efetuadas para implantação da rede IPv6.

2.7 *Web Server*

Web server é um servidor que tem por objetivo fazer a hospedagem de aplicações, serviços e *websites*. Através deste tipo de servidor permite-se o compartilhamento de informações com usuários pela *Internet*. Neste caso, o serviço utilizado para disponibilização deste recurso foi através da instalação do pacote *apache2*.

3 TRABALHOS RELACIONADOS

Na literatura, existem diversos estudos sobre o protocolo IPv6, bem como a adoção do protocolo e técnicas de transição. Efetuou-se uma pesquisa analisando trabalhos já realizados, os quais foram realizados com o mesmo intuito deste trabalho, fazendo um levantamento do estado do protocolo IPv6 atualmente e também de técnicas utilizadas na transição entre o protocolo IPv4 e o IPv6.

Braga (2011) destaca as principais características do protocolo IPv6, como escalabilidade, gerenciamento e monitoramento e mobilidade, fazendo um estudo geral do protocolo IPv6 analisando suas melhorias em relação ao protocolo IPv4. Destacando também as técnicas de transição para estimular a migração de IPv4 para IPv6 e propondo que os profissionais de TI ampliem seus estudos sobre o protocolo IPv6 que provavelmente será o protocolo padrão da *Internet*.

Mohd, et. al. (2009) defendem o uso de IPv6,devido ao esgotamento de endereços IPv4 que deve ocorrer em pouco tempo. A única opção para esta escassez é fazer a migração de protocolos, podendo ser utilizados dois métodos, o de pilha dupla, o qual permite utilização de pilhas IPv4 para sites que ainda tem suporte IPv4 e pilha IPv6 para sites que já aderiram a esta nova tecnologia, e a técnica de tunelamento também é especificada, apresentado suas formas de implementação. As duas técnicas são de grande importância para a migração de protocolos.

Silveira (2012) propõem testes para a avaliação de técnicas de tradução e pilha dupla, as quais não são o objetivo deste trabalho. Buscando quais seriam as técnicas mais adequadas para o cenário de transição de protocolos atual, foram estudadas técnicas de tradução, ressaltando que a técnica de pilha dupla e NAT64/DNS64 são adequadas para navegação web e se adaptam aos dois protocolos.

Em um cenário atual de *Internet* onde é inevitável a transição entre os protocolos, propõe-se neste trabalho, fazer um túnel para a conexão de uma rede IPv6 com a *Internet*. Este será transparente ao usuário e em um futuro próximo, quando o protocolo IPv6 estiver incorporado à *Internet*, este túnel possa ser removido e a comunicação seja feita exclusivamente pelo protocolo IPv6.

4 TRABALHO PROPOSTO

Este Capítulo apresenta na Seção 4.1 a representação do ambiente de testes implementado, juntamente com a proposta de desenvolvimento deste estudo. A Seção 4.2 engloba a proposta de tunelamento e, por fim, as Seções 4.3, 4.4, 4.5 e 4.6 apresentam as Implantações do *Tunnel Broker*, do *Tunnel* 6*over*4, do *Tunnel* GRE e dos Serviços da Rede.

4.1 Ambiente de Testes

Esta Seção descreve a proposta de desenvolvimento deste estudo, onde serão feitas as implantações de diferentes técnicas de transição do protocolo IPv4 para o seu sucessor, o IPv6. Dentre as técnicas existentes hoje, optou-se por utilizar à de tunelamento, a qual foi a técnica de melhor acessibilidade, possibilitando uma pesquisa com ampla documentação. Onde, a partir desta pesquisa, foram feitas verificações e selecionadas as mais relevantes. O cenário apresentado abaixo na Figura 15 demonstra a proposta inicial do trabalho.

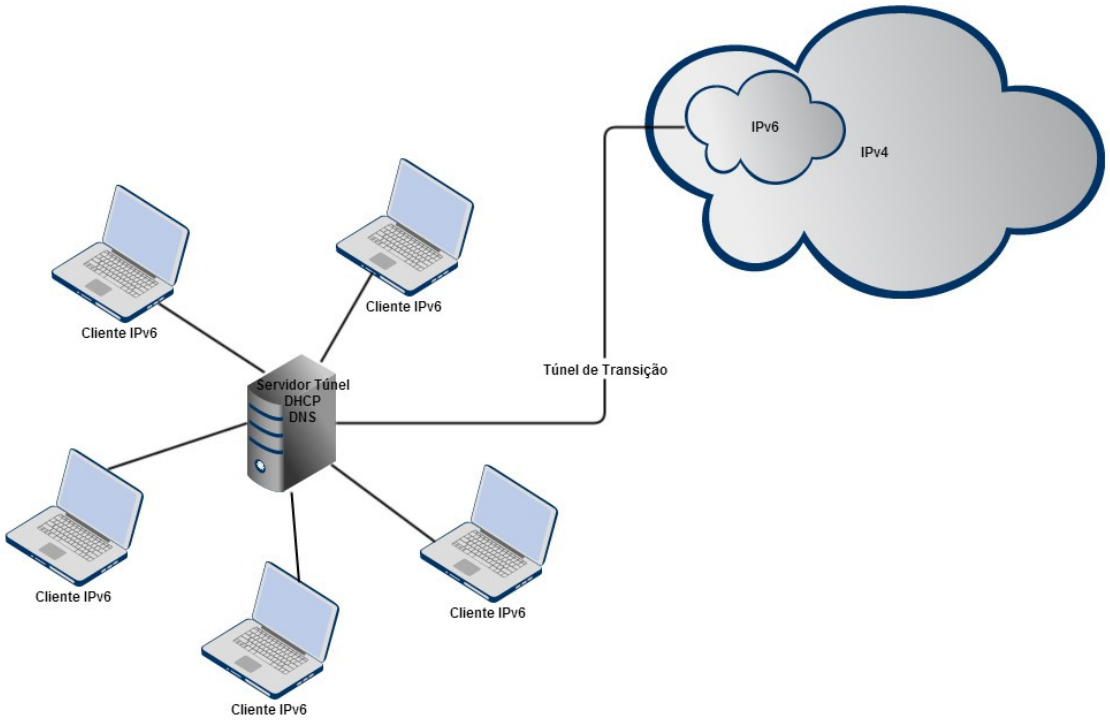

Figura 15: Ambiente de Testes Implementado
Fonte: Do Autor.

O cenário proposto, simula um ambiente de rede contendo *hosts* atuando como clientes em uma rede IPv6, onde nesta rede implementada existe um servidor DHCP, um servidor DNS e um servidor de túnel. Neste cenário os *hosts* são emulados em máquinas virtuais, contendo interfaces em IPv4 e IPv6 interligados a *Internet* IPv4 e, com o túnel ativo à *Internet* IPv6. Por fim, considerou-se a utilização do sistema operacional *Linux*, distribuição Ubuntu *Server* 12.04 LTS como o servidor e, às demais máquinas clientes nas plataformas *Linux* e *Windows*, com as distribuições *Desktop* 12.04 LTS e *Desktop* 13.10 para *Linux* e XP SP3 para o *Windows*.

Deste ponto em diante este Capítulo está estruturado da seguinte maneira: a Seção 4.1 apresenta a Proposta de Tunelamento. A Seções 4.2, 4.3 e 4.4 demonstram definições e configurações referentes à cada técnica de tunelamento implantada. Findando-se com a Seção 4.5, onde são mostrados os Serviços da Rede Implementada.

4.2 Proposta de Tunelamento

A proposta deste trabalho é a implementação de uma rede IPv6 fazendo a utilização de um túnel para que a partir dele, possa ser feita a comunicação via *Internet* IPv4. Este túnel deve estar ativo até que o IPv6 seja totalmente incorporado na *Internet*, logo após deve ser removido de forma transparente ao usuário final.

Para a realização do mesmo, entre todas as técnicas de tunelamento, foram estudadas três, as quais foram escolhidas por serem as mais atuais, sendo elas: *Tunnel Broker*, 6*over*4 e *Tunnel* GRE, para que posteriormente possa ser usada a técnica mais apropriada para obter conectividade IPv6 quando o provedor de *Internet* não a oferece.

4.3 *Tunnel Broker*

Esta Seção apresenta na Subseção 4.3.1 a Definição do Servidor *Broker* utilizada para implementação. Na Subseção 4.3.2 a Instalação do Serviço e, por fim, na Subseção 4.3.3 a Configuração do *Tunnel*.

4.3.1 Definição do Servidor *Broker*

Dentre às viabilizações de provedores de *Tunnel Broker* existentes atualmente, como por exemplo, a *Hurricane Electric* e a *SixXS*, foi selecionada a *Freenet* 6 para aplicação e utilização desta técnica de tunelamento. Tal serviço mostrou-se o mais adequado e eficaz, em tratando-se de funcionalidade de *Tunnel Broker*. Também pode-se observar, uma característica relevante bastante positiva, a simplicidade de implantação do mesmo.

O primeiro passo é escolher o local do servidor, dentre duas opções Montreal no Canadá ou Amsterdam na Holanda. Por questões de distância e o Canadá fazer parte do continente americano, foi o escolhido. O segundo passo é fazer um cadastro, no *site* <http://www.gogo6.com/freenet6/tunnelbroker>, para utilização do provedor de serviços de *Tunnel Broker Freenet* 6. Este cadastro é demonstrado na Figura 16, onde o número 1 é o endereço de e-mail do usuário que irá conectar ao provedor, os números 2 e 3 são os campos para inserção de Senha e Confirmação de Senha, nesta ordem. Nos números 4 e 5 são inseridos os caracteres solicitados do mecanismo antifraude. Findando-se, clicando em *Sign Up* representado pelo número 6 na Figura 16.

Figura 16: Cadastro no Provedor *Freenet* 6.
Fonte: www.gogo6.com/freenet6/account

4.3.2 Instalação do Serviço

O processo de instalação do *Tunnel Broker* via provedor *Freenet* 6 pode ser feito tanto em plataformas *Windows* quanto em *Linux*. Para este estudo, optou-se por utilizar a distribuição Ubuntu 12.04 LTS (*Precise Pangolin*) da plataforma *Linux*. Após, fazer o cadastro no *site*, deve-se logar no mesmo para realizar o *download* do *Source Code*, denominado '*gogoc-1_2-RELEASE.tar*', para que se possa dar continuidade à instalação. Ao concluir o *download* do arquivo, deve-se localizar o diretório onde o arquivo foi baixado e extraí-lo utilizando o comando a seguir (deve-se estar logado como superusuário).

```
# tar –xvf gogoc-1_2-RELEASE.tar
```

No arquivo de instalação, denominado '*INSTALL*' estão descritas as dependências exigidas para a instalação do serviço. Para instalá-las execute o seguinte comando, também como superusuário.

```
# apt-get update && sudo apt-get install gcc g++ openssl libssl-dev libcrypto* libpthread* libssl-dev libcurl4-openssl-dev
```

Deve-se também alterar o arquivo '*message.h*' localizado no diretório */gogoc-1_2-RELEASE/gogoc-messaging/gogocmessaging*, onde o caminho corresponde ao caminho do diretório onde o arquivo foi extraído. A alteração deve ser feita na linha 33, onde deve-se substituir por *#define MSG_MAX_USERDATA 64260*. Por fim, a última etapa do processo de instalação é obtida com a execução do seguinte comando, para montar o executável da aplicação.

```
# make && make installdir=/home/raissatunnel/gogoc-1_2-RELEASE install
```

Em distribuições *Linux* atuais, como por exemplo o Ubuntu 13.10, já é possível encontrar este serviço incluso no repositório da distribuição, simplificando o procedimento de instalação do serviço ao comando a seguir.

```
# apt-get install gogoc
```

4.3.3 Configuração do *Tunnel*

A configuração do *Tunnel Broker* via provedor *Freenet* 6 é fundamentada e centralizada em um único arquivo, denominado '*gogoc.conf*', localizado dentro do diretório de instalação do programa, no caso */home/raissatunnel/gogoc-1_2-RELEASE/bin/*. As configurações definidas no arquivo seguem na Figura 17.

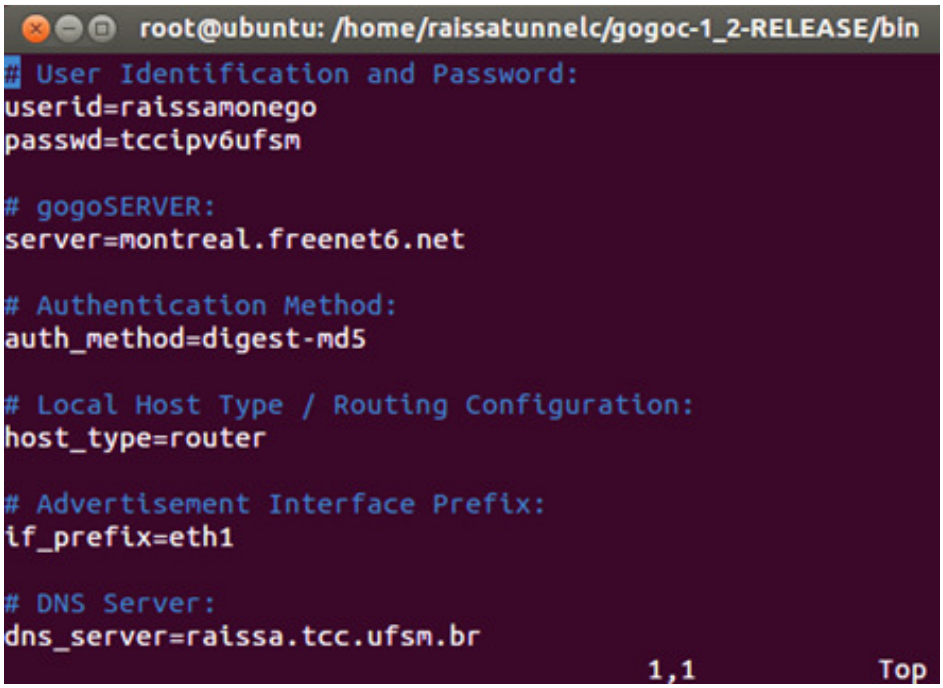

Figura 17: Configurações do *Tunnel Broker* via *Freenet6*.
Fonte: Do Autor.

Nas linhas 31 e 32, correspondentes no arquivo original de configuração, são inseridos usuário e senha <*raissamonego e tccipv6ufsm*> nesta ordem, para autenticação no provedor do *Túnel Broker Freenet* 6. Logo abaixo (na linha 53) é inserido o servidor, no caso <*montreal.freenet6.net*>. Depois, na linha 72, é definido o método de autenticação, no caso optou-se pela criptografia md5 <*digest-md5*>. Após (na linha 85) é configurado o modo para utilização do serviço, neste caso o modo optado é <*router*> para trabalhar como um servidor local em uma rede IPv6 ligando-se à *Internet* IPv6. A próxima opção, na linha 104, é a interface em que se encontra o servidor DHCP para LAN (*Local Area Network*) IPv6 <*eth1*>. Por fim, na linha 113, é definido o servidor DNS em IPv6 da rede <*raissa.tcc.ufsm.br*>. Para verificação do arquivo completo de configuração, consulte o Apêndice A.

4.4 *Tunnel* 6*over*4

O procedimento de implantação desta técnica de tunelamento será dividido em duas Subseções 4.4.1 Configuração de Interfaces e 4.4.2 Criação do *Tunnel* 6*over*4.

A topologia adotada para a execução do túnel 6*over*4 segue na Figura 18. Nesta podem-se observar dois equipamentos, *host*A e *host*B. Nota-se que a topologia representa duas redes IPv6 (*host*A e *host*B) comunicando-se através de uma rede IPv4. Os endereços IP

192.168.55.2/25 e 2013:db8::dcba/64 correspondem aos endereços IPv4 e IPv6 do *host*A. E os endereços IP 192.168.55.130/25 e 2013:db8::abcd/64 correspondem aos endereços IPv4 e IPv6 do *host*B.

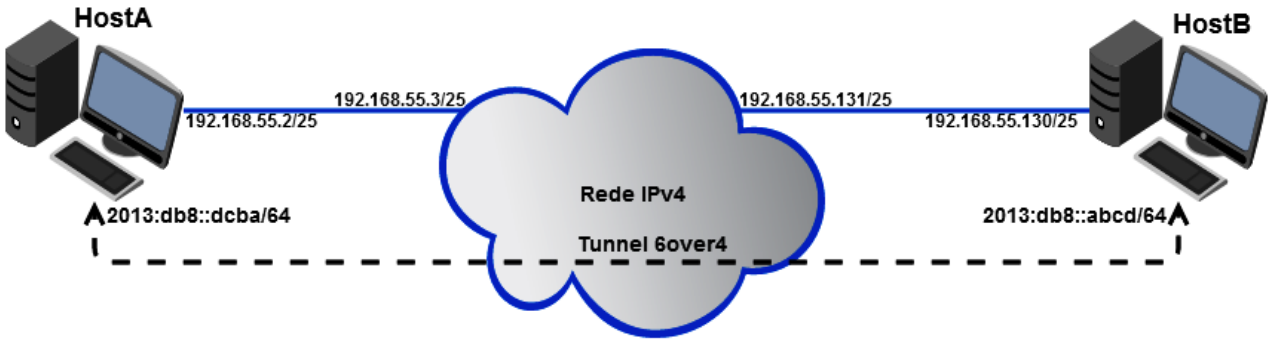

Figura 18: Topologia implantada do *Tunnel* 6*over*4.
Fonte: Do Autor.

4.4.1 Configuração de Interfaces

As interfaces foram configuradas manualmente, conforme mostram as Figuras 19 e 20. Nelas são mostrados os endereçamentos das interfaces dos dois *hosts* (A e B), utilizados para criação do *Tunnel* 6*over*4.

```
root@ubuntu: ~
root@ubuntu:~# ifconfig eth2
eth2      Link encap:Ethernet  HWaddr 00:0c:29:cb:27:e6
          inet addr:192.168.55.2  Bcast:192.168.55.127  Mask:255.255.255.128
          inet6 addr: 2013:db8::dcba/64 Scope:Global
          inet6 addr: fe80::20c:29ff:fecb:27e6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4638 errors:0 dropped:0 overruns:0 frame:0
          TX packets:550 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:403954 (403.9 KB)  TX bytes:58155 (58.1 KB)
          Interrupt:17 Base address:0x2424

root@ubuntu:~#
```

Figura 19: *Host*A do *Tunnel* 6*over*4.
Fonte: Do Autor.

O IP *192.168*.55.2/25 é um endereço estático utilizado para comunicação com o *host*B utilizando o protocolo IPv4. Conclui-se a comunicação com a adição de uma rota para a rede do *host*B, bem como sua rota de retorno. O endereço IPv6 *2013:db8::dcba/64* é o estático utilizado para comunicação via IPv6 pelo túnel 6*over*4.

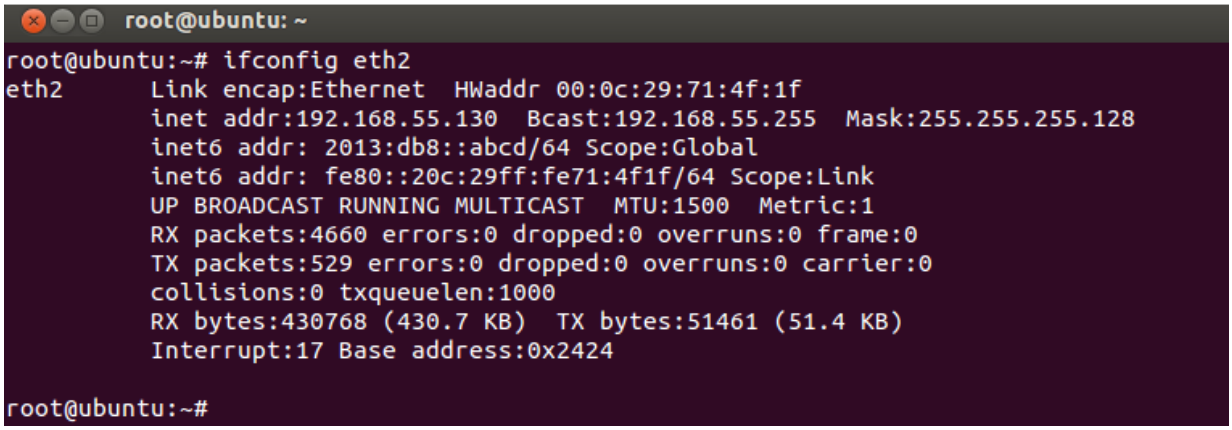

Figura 20: *Host*B do *Tunnel* 6*over*4.
Fonte: Do Autor.

O endereço *192.168.55.130/25* é o endereço estático utilizado para comunicação com o *host*A via IPv4, conforme dito anteriormente, para que haja esta comunicação deve ser adicionada uma rota de retorno para a rede do *host*A. O endereço *2013:db8::abcd/64* é o utilizado para comunicação IPv6 pelo túnel 6*over*4.

4.4.2 Criação do *Tunnel* 6*over*4

Uma interface própria de comunicação é gerada em cada *host* do túnel no processo de implantação. Tais interfaces são apresentadas nas Figuras 21 e 22, para o *host*A e para o *host*B, respectivamente. O nome das interfaces (*6over4*) é definido no momento da geração do túnel.

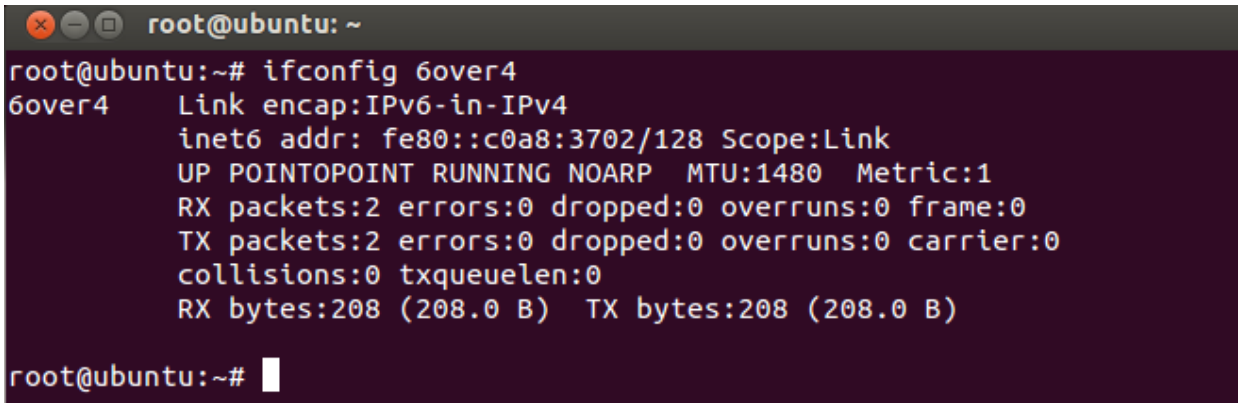

Figura 21: Interface 6*over*4 no *host*A
Fonte: Do Autor.

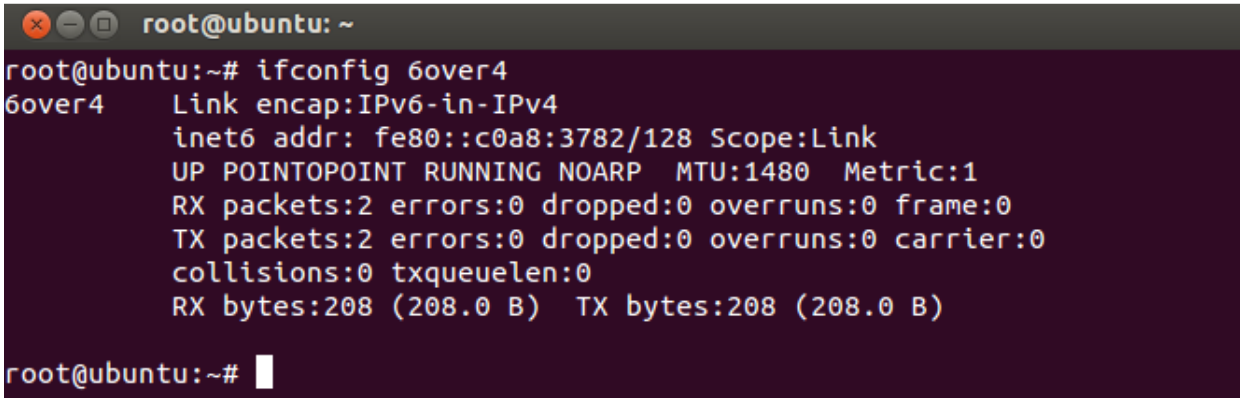

Figura 22: Interface 6*over*4 no *host*B
Fonte: Do Autor.

A criação do túnel 6*over*4 deve ser efetuada manualmente, tanto no *host*A quanto no *host*B, onde os seguintes comandos (como superusuário) devem ser executados (o primeiro no *host*A e o segundo no *host*B).

```
# ip tunnel add 6over4 mode sit ttl 64 remote 192.168.55.2 local 192.168.55.130 && ip link set dev 6over4 up && ip -6 route add 2013:db8::dcba dev 6over4
```

```
# ip tunnel add 6over4 mode sit ttl 64 remote 192.168.55.130 local 192.168.55.2 && ip link set dev 6over4 up && ip -6 route add 2013:db8::abcd dev 6over4
```

6*over*4 é o rótulo dado à interface do túnel, o ip *192.168.55.2* é o endereço remoto do *host*B. Já o endereço 192.168.55.130 é o correspondente local para que haja a comunicação. O comando *ip link set 6over4 up* serve para ativar a interface. Por fim, último comando serve para criação da rota utilizando o túnel 6*over*4. Para uma configuração de inicialização automatizada com o sistema foram adicionados tais comandos correspondentes no arquivo */etc/rc.local* onde o túnel é iniciado automaticamente quando os *hosts* são ligados. No Apêndice B encontra-se o arquivo do *host*B para implantação do tunelamento 6*over*4.

4.5 *Tunnel* GRE

O processo de implantação deste túnel será dividido em duas Subseções 4.5.1 Configuração de Interfaces e 4.5.2 Criação do *Tunnel* GRE.

A topologia adotada para a criação do túnel GRE segue na Figura 23. Verificam-se nesta ilustração, dois equipamentos, *host*A e *host*B. Pode-se observar que a topologia representa duas redes IPv6 (*host*A e *host*B) comunicando-se através de uma rede IPv4. Os endereços IP 192.168.55.1/25 e 2013::abc:a/64 correspondem aos endereços IPv4 e IPv6 do *host*A. E os endereços IP 192.168.55.129/25 e 2013::abc:b/64 correspondem aos endereços IPv4 e IPv6 do *host*B.

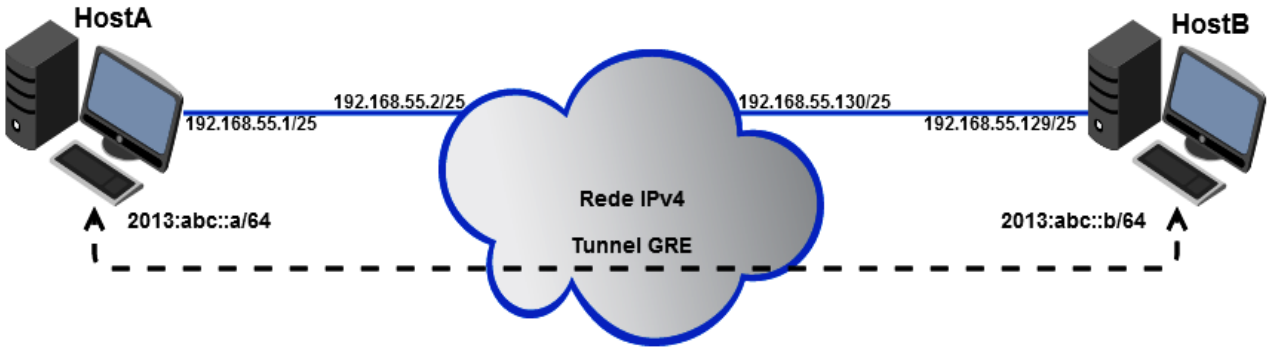

Figura 23: Topologia adotada no *Tunnel* GRE.
Fonte: Do Autor.

4.5.1 Configuração de Interfaces

A configuração de interfaces foi feita manualmente, conforme mostram as Figuras 24 e 25, as quais demonstram à disposição das interfaces dos dois *hosts* utilizados para comunicação via *tunnel* GRE.

```
root@ubuntu: ~
root@ubuntu:~# ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 00:0c:29:cb:27:dc
          inet addr:192.168.55.1  Bcast:192.168.55.127  Mask:255.255.255.128
          inet6 addr: fe80::20c:29ff:fecb:27dc/64 Scope:Link
          inet6 addr: 2013:abc::a/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3418 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1491 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:819990 (819.9 KB)  TX bytes:128548 (128.5 KB)
          Interrupt:16 Base address:0x20a4

root@ubuntu:~#
```

Figura 24: *Host*A do *Tunnel* GRE.
Fonte: Do Autor.

O endereço *192.168.55.1/25* é utilizado para comunicação com o outro *host* utilizando o protocolo IPv4. Para isto deve-se adicionar uma rota para a rede do *host*B, e vice-versa. O endereço IPv6 *2013:abc::a/64* é o utilizado para comunicação via IPv6 pelo túnel.

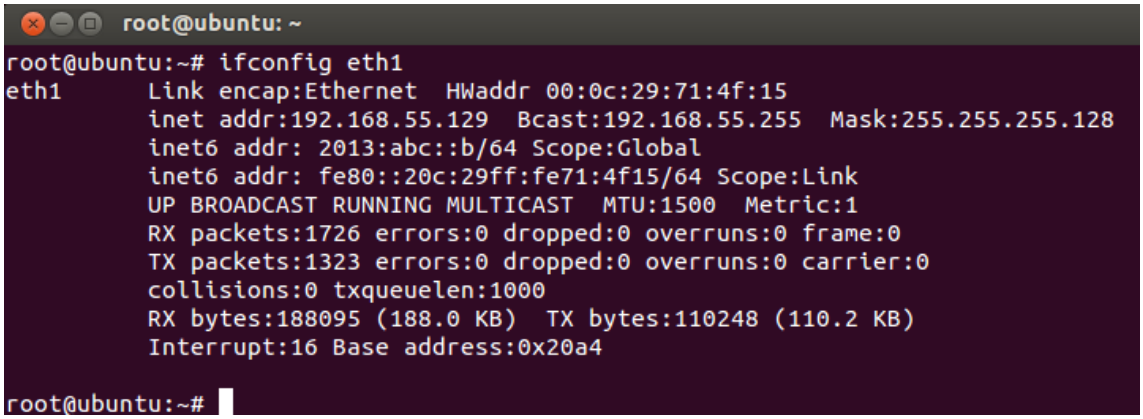

Figura 25: *Host*B do *Tunnel* GRE.
Fonte: Do Autor.

O endereço *192.168.55.129/25* é utilizado para comunicação com o *host*A utilizando o protocolo IPv4. Para que haja comunicação deve ser adicionada uma rota para a rede do *host*A, conforme citado anteriormente. O endereço IPv6 *2013:abc::b* é o utilizado pelo túnel GRE.

4.5.2 Criação do *Tunnel* GRE

A implantação do túnel cria uma interface de comunicação própria em ambos os *hosts* do *tunnel* GRE. Tais interfaces estão ilustradas nas Figuras 26 e 27 para o *host*A e *host*B, respectivamente. O rótulo das interfaces "*gre*" é definido na criação do túnel.

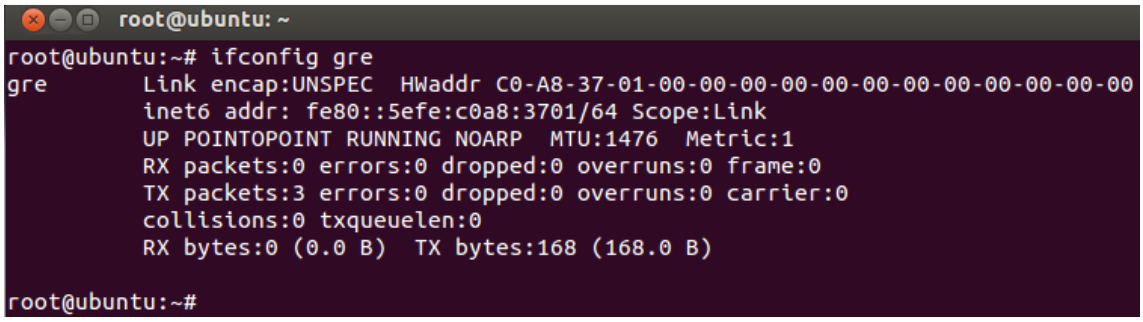

Figura 26: Interface *gre* no *host*A.
Fonte: Do Autor.

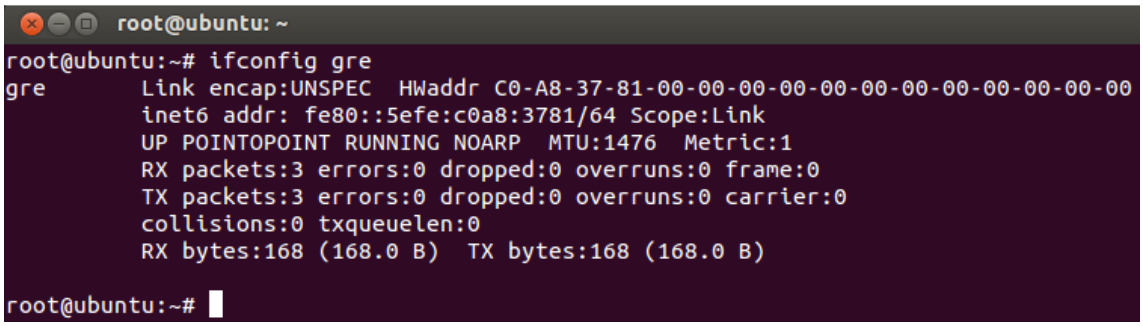

Figura 27: Interface *gre* no *host*B.
Fonte: Do Autor.

Para se efetuar o processo de criação do túnel GRE deve ser feito, manualmente nos dois *hosts*, o seguinte comando, como superusuário:

```
# ip tunnel add gre mode gre ttl 64 remote 192.168.55.1 local 192.168.55.129 && ip link set dev gre up && ip -6 route add 2013:abc::a dev gre
```

Onde *gre* após o *add* é o rótulo dado à interface do túnel, o IP *192.168.55.1* é o endereço remoto, isto é, do *host*B. Já o endereço *192.168.55.129* é o correspondente local para comunicação. O comando *ip link set gre up* serve para ativar a interface. E por fim, o comando seguinte é para criação da rota utilizando o túnel GRE.

Conforme explicado anteriormente, o comando do *host*B fica na forma à seguir, executado como superusuário. Para uma configuração de inicialização automatizada com o sistema foram adicionados tais comandos correspondentes no arquivo */etc/rc.local* onde o túnel é iniciado automaticamente quando os *hosts* são ligados. No Apêndice C encontra-se o arquivo do *host*B.

```
# ip tunnel add gre mode gre ttl 64 remote 192.168.55.129 local 192.168.55.1 && ip link set dev gre up && ip -6 route add 2013:abc::b dev gre
```

4.6 Serviços da Rede Implementada

Alguns serviços são de grande importância para o funcionamento de redes de computadores. No presente trabalho, serão utilizados três serviços básicos para adicionar funcionalidades na rede IPv6 implementada. Esta Subseção contemplara a descrição dos serviços utilizados neste estudo, sendo na Subseção 4.6.1 o DHCPv6, na seguinte 4.6.2 o DNS e, por fim na Subseção 4.6.3 o *Web server*.

Visto que estes três serviços (DHCP, DNS e *Web server*) são utilizados para somar funcionalidades à rede local implementada, suas respectivas configurações e instalações encontram-se, por completo, nos Apêndices no final deste trabalho.

4.6.1 DHCPv6

Para a configuração do servidor DHCP, foi utilizado o serviço *isc-dhcp-server*, o qual possui embutido o DHCPv6 (versão para se trabalhar com o protocolo IPv6). O serviço DHCPv6 permite passar parâmetros de configuração para fazer o endereçamento de equipamentos com o protocolo IPv6, oferecendo a capacidade de alocação automática de endereços de rede reutilizáveis com bastante flexibilidade.

O DHCPv6 atua na camada de transporte utilizando pacotes UDP para transmitir informações entre o cliente e o servidor. Para consultar a instalação e configuração do Servidor DHCPv6, verifique o Apêndice D.

4.6.2 DNS

Neste estudo, a configuração do Servidor DNS foi disponibilizada através do serviço *Berkeley Internet Name Domain* (BIND). Para este trabalho foi utilizada a versão 9 (instalou-se pacote *bind9*). Nesta versão já está presente o suporte à resolução de nomes de *hosts* endereçados com o protocolo IPv6 e a realização de consultas que solicitam endereços IPv6. Além disso este serviço é capaz de responder a estas consultas, bem como, consultar servidores de nomes em IPv6.

O procedimento de instalação do BIND9, bem como, sua configuração, encontram-se no Apêndice E.

4.6.3 *Web Server*

O servidor *web* (*Web server*) é um equipamento que tem por objetivo fazer a hospedagem de aplicações, serviços e *websites*. Através deste tipo de servidor permite-se o compartilhamento de informações com usuários pela *Internet*. Neste caso, o serviço utilizado para disponibilização deste recurso foi através da instalação do pacote *apache2*. A versão do Sevidor *Web* instalada foi o Apache 2.2.14.

Primeiramente, foi editado o arquivo */etc/network/interfaces*, estabelecendo-se um endereço IPv6 estático na interface desejada, o endereço escolhido foi *2013:0a26::c0a8:0a0a*, utilizando a máscara /64, a qual é um padrão do IPv6. Por fim, o *gateway* adotado foi o *2013::1*. A Figura 28 mostra esta configuração.

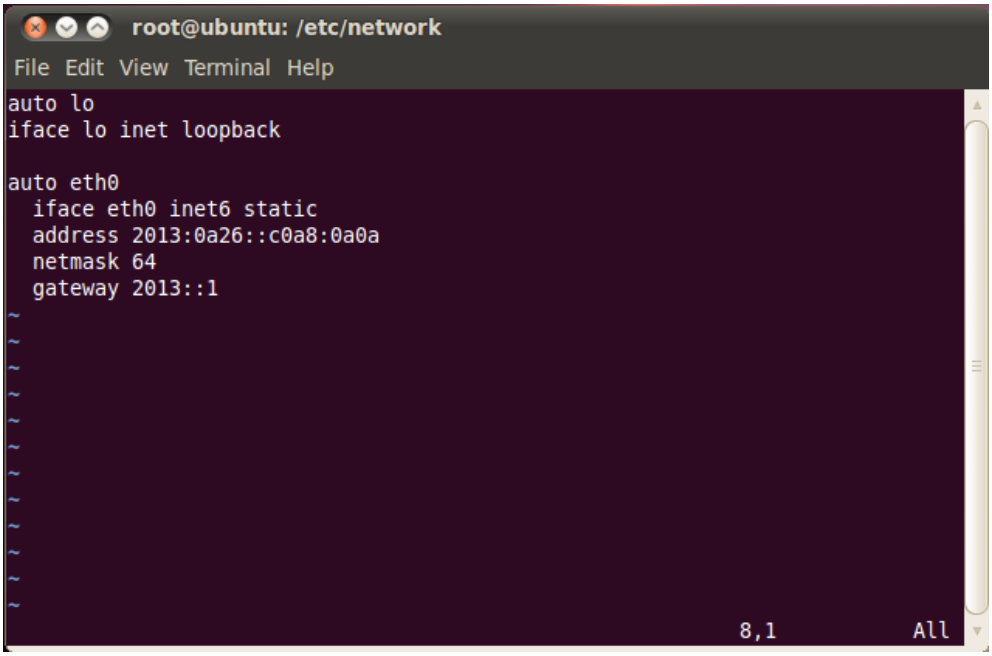

Figura 28: Configuração do arquivo */etc/network/interfaces.*
Fonte: Do Autor.

O próximo passo é permitir que este *host* comunique-se com os demais utilizando o protocolo IPv6. Para permitir o roteamento IPv6, é necessário habilitar o encaminhamento de pacotes do protocolo. Para isto, deve-se descomentar a linha correspondente à tal ação, no caso, *net.ipv6.conf.all.forwarding=1*, localizada no arquivo */etc/sysctl.conf*, como apresentado na Figura 29. Concluído este procedimento verifica-se a nova configuração com o comando *sudo sysctl -w net.ipv6.conf.all.forwarding=1*. Os demais procedimentos de configuração e

instalação do serviço podem ser encontrados no Apêndice F.

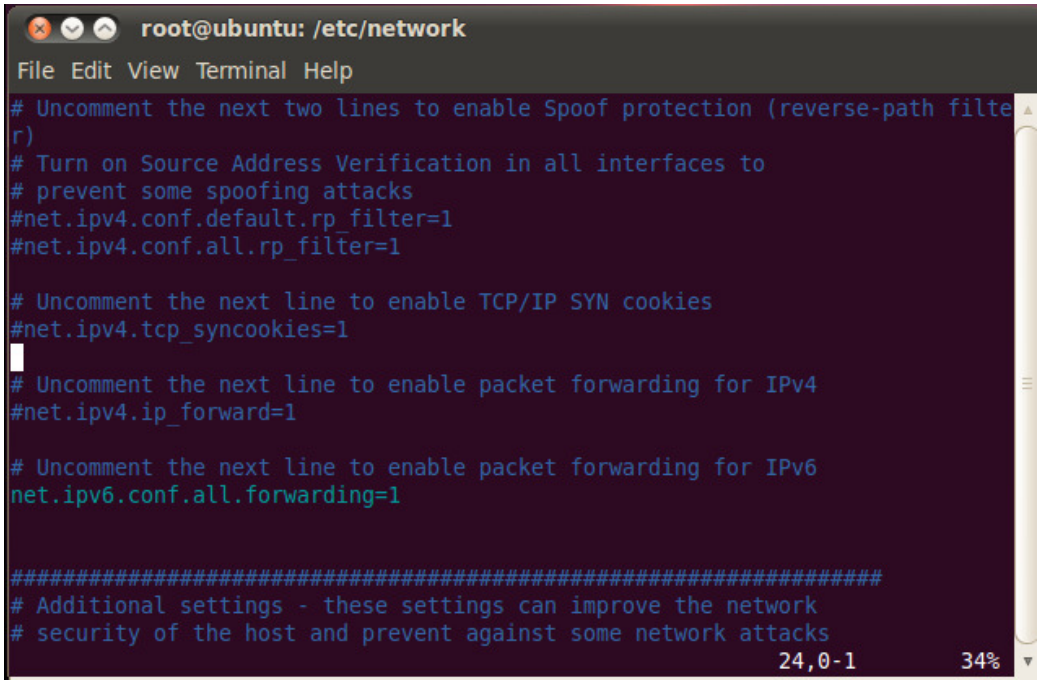

Figura 29: Habilitando roteamento IPv6.
Fonte: Do Autor.

Vale ressaltar que os procedimentos supra citados, bem como demais configurações encontradas nos Apêndices referentes aos serviços utilizados (DHCPv6, DNS e *Web server*), foram efetuados na Distribuição Ubuntu 12.04 LTS. Na Figura 30, é apresentada a página *web* executando um servidor *web* utilizando o protocolo IPv6.

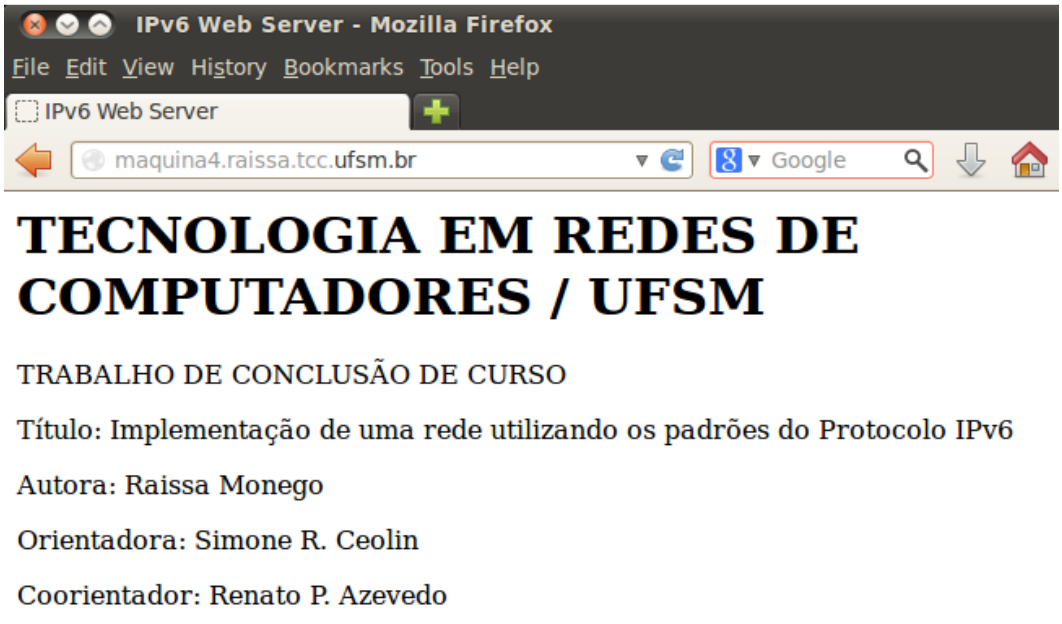

Figura 30: Página *web* implementada na rede local utilizando o Protocolo IPv6.
Fonte: Do Autor.

5 TESTES E RESULTADOS

Nesta Seção são apresentados os resultados obtidos com a implantação das técnicas de tunelamento selecionadas, bem como os testes de validação de funcionalidade para aplicação em um ambiente. O cenário de testes implementado segue na Figura 31.

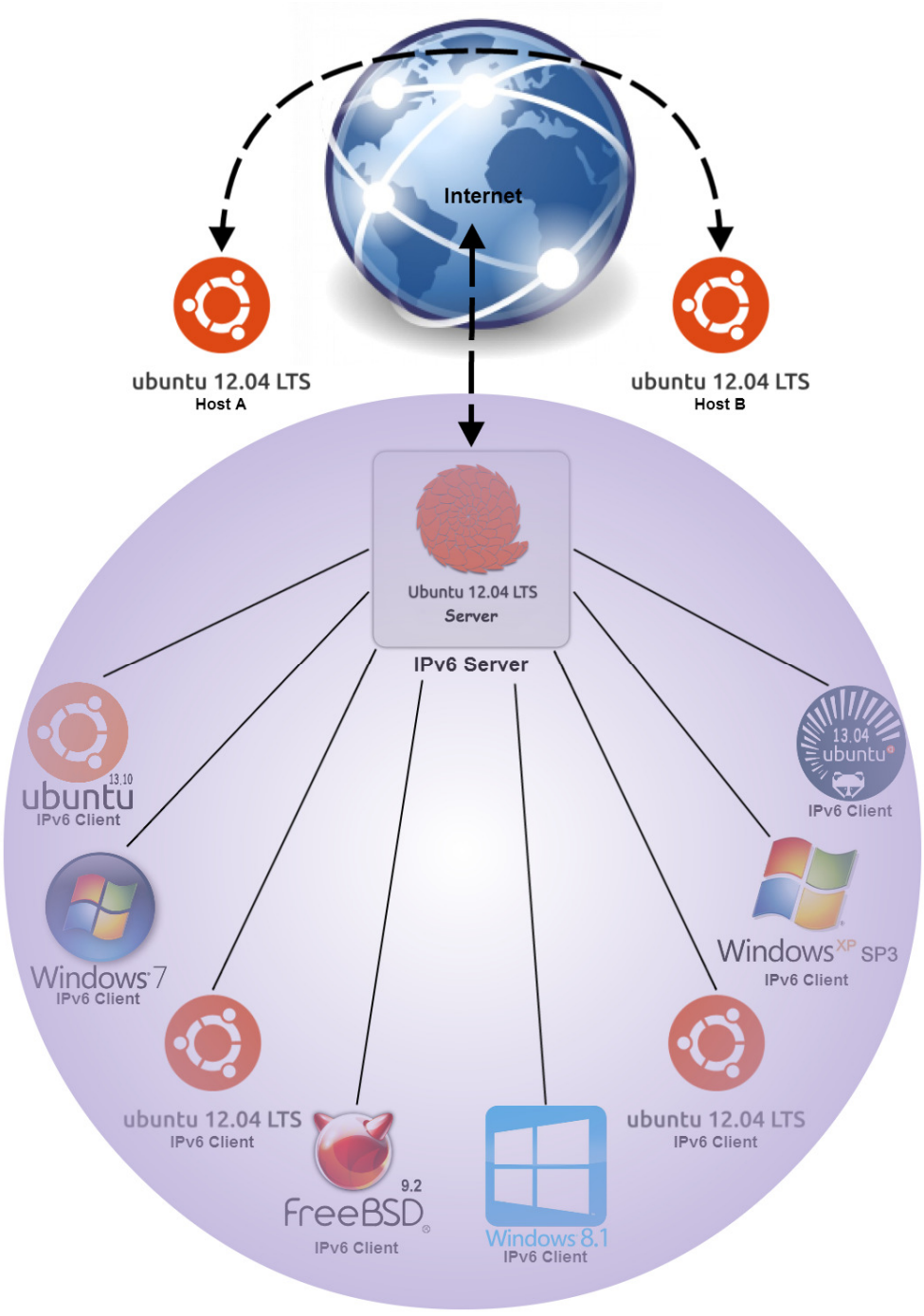

Figura 31: Ambiente de Testes Implementado.
Fonte: Do Autor.

Como pôde-se analisar na ilustração do Cenário criado, foi utilizado um Servidor IPv6 principal, implantado na distribuição Ubuntu 12.04 LTS *Server*. Tal *host* é o cerne deste estudo, pois o mesmo tem por funcionalidade a distribuição de endereços IPv6 para clientes de uma rede local (por meio do serviço DHCPv6), a disponibilização de um serviço para resolução de nomes em IPv6 (servidor DNS implantado com a utilização do serviço BIND 9) e, o mais importante a comunicação com a *Internet* IPv6 via *Tunnel Freenet* 6 (Gogo6), servindo o mesmo como um Servidor de Tunelamento para os demais *hosts* da rede interna.

A rede local implantada conta com oito *hosts*, sendo destes cinco *hosts LINUX* e outras três distribuições *WINDOWS*. Das distribuições *Linux* são: dois *hosts* Ubuntu 12.04 LTS *Desktop*, um Ubuntu 13.04 *Desktop*, um Ubuntu 13.10 *Desktop* e um FreeBSD 9.2 *Desktop*. Das distribuições *Windows* são mais três máquinas, sendo uma com *Windows* XP SP3, outra com *Windows Seven* SP1 32 *bits* e, por fim, um *Windows* 8.1 64 *bits*.

O restante representado no ambiente gerado, é a comunicação em IPv6 entre dois computadores Ubuntu 12.04 LTS *Desktop*, sendo eles: *host*A e *host*B. Foram implantadas para tal comunicação duas técnicas de tunelamento, já abordadas no Capítulo 4. Tais técnicas foram através da utilização do *Tunnel* 6*over*4 e do *Tunnel* GRE, ambas para comunicação via IPv6 na *Internet* IPv4.

A seguir serão apresentados, na Seção 5.1 os testes e resultados para validar o Servidor de *Tunnel Freenet* 6 (*Tunnel Broker*). Na Seção 5.2 são apresentados os testes e resultados para validação do *Tunnel* 6*over*4 e, por fim, na Seção 5.3 os testes e resultados para validar a comunicação quando utilizado o *Tunnel* GRE.

5.1 Validação do *Tunnel Broker*

O procedimento para validação do *Tunnel Broker* foi colocá-lo em funcionamento na rede e obter-se acesso à comunicação com a *Internet* via protocolo IPv6. Para isto, deve-se estabelecer uma conexão com um provedor de acesso *Tunnel Broker*. Conforme descrito no Capítulo 4, o túnel selecionado para tal conexão, foi o *Tunnel Freenet* 6, também conhecido como Gogo6. Consulte o Capítulo 4 para verificar a configuração de funcionamento do *Tunnel Freenet* 6 utilizada.

A execução deste túnel dar-se-á através do comando descrito a seguir, onde o caminho */home/raissatunnel/gogoc-1_2-RELEASE/* corresponde ao local onde o serviço foi previamente instalado.

```
# cd /home/raissatunnel/gogoc-1_2-RELEASE/bin/ && ./gogoc
```

Antes de se estabelecer a conexão com o provedor de acesso do *Tunnel Broker Freenet* 6, foi feia tentativa de acesso ao sítio do Google IPv6 <ipv6.google.com>. No entanto, conforme previsto a conexão ao endereço do protocolo IPv6 não foi bem sucedida. Segue na Figura 32 a tentativa de acesso antes da execução do túnel.

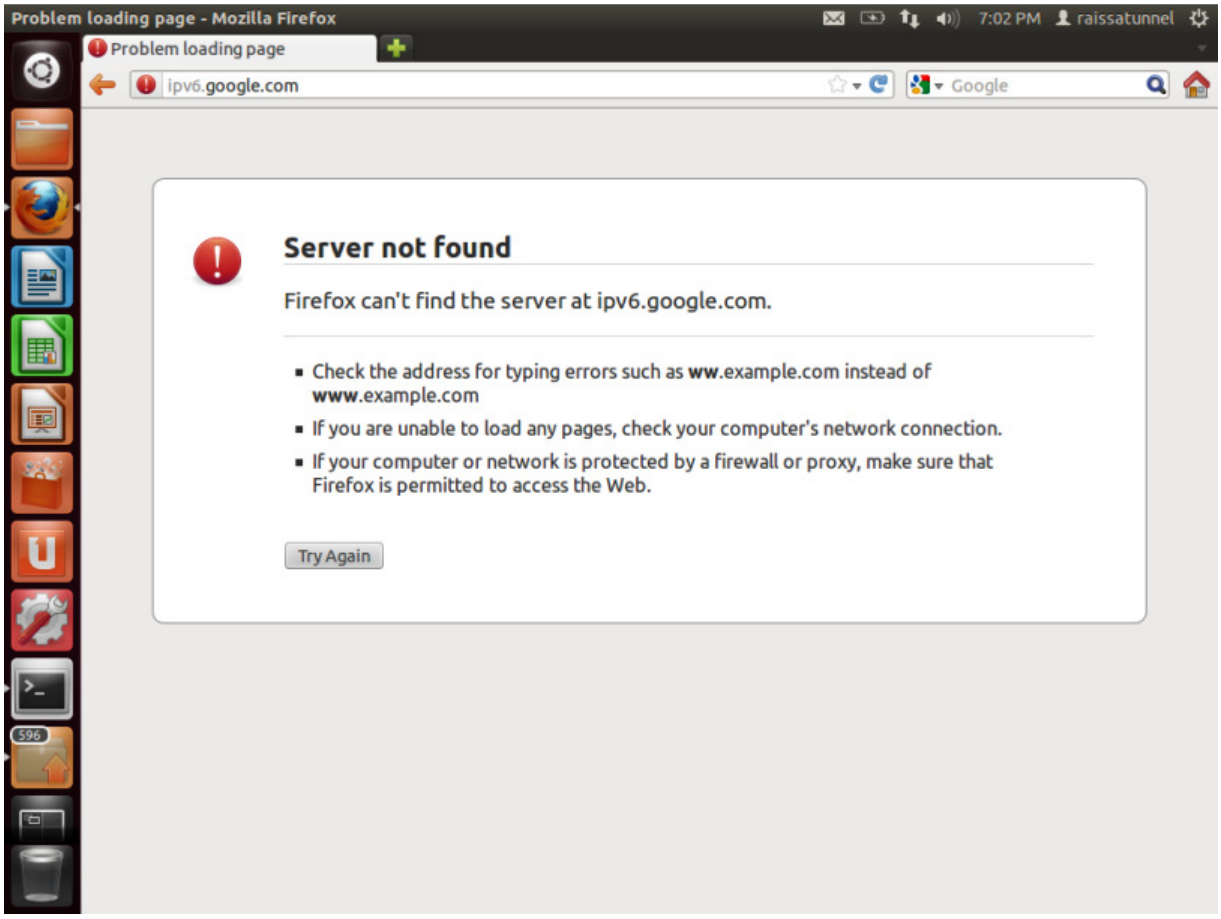

Figura 32: Sítio IPv6 do Google antes da execução do *Tunnel Freenet* 6.
Fonte: Do Autor.

Após, executado o comando descrito anteriormente, para se estabelecer a conectividade com o provedor de *Tunnel Broker Freenet* 6, foi feita nova tentativa de acesso ao mesmo endereço IPv6 do Google <ipv6.google.com>, onde obteve-se sucesso. A Figura 33 demonstra a conexão estabelecida ao sítio.

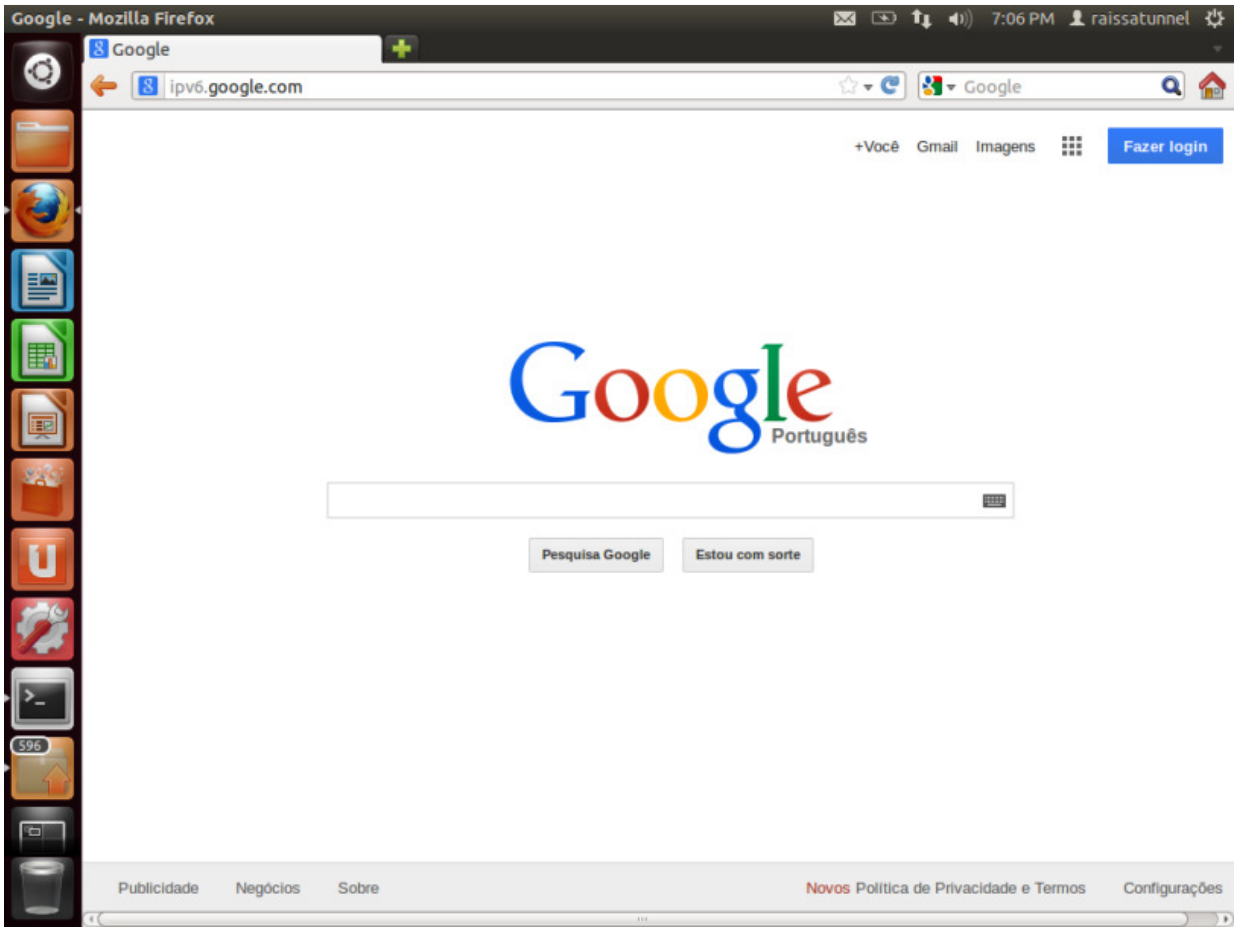

Figura 33: Sítio IPv6 do Google após executar o *Tunnel Freenet* 6.
Fonte: Do Autor.

Conforme descrito anteriormente, sabe-se que o *Tunnel Broker Freenet* 6 implantado tem por funcionalidade estabelecer conectividade com o protocolo IPv6 à *hosts* de uma rede local implementada. Sendo assim, apesar de este ser um Cliente do provedor *Freenet* 6 para acesso à *Internet* IPv6, ao mesmo tempo, é um Servidor de *Tunnel Broker* aos seus *hosts* da rede local para acesso à sítios IPv6, trabalhando de forma similar à um roteador com suporte à tecnologia IPv6. As Figuras 34 e 35 demonstram um acesso utilizando o protocolo IPv6 via servidor de túnel e via cliente deste servidor na rede local, nesta ordem.

Figura 34: Sítio IPv6 do Terra acessado através do Servidor *Tunnel Broker*.
Fonte: Do Autor.

Figura 35: Sítio IPv6 acessado por *host* na rede local implementada.
Fonte: Do Autor.

5.2 Validação do *Tunnel* 6*over*4

Para colocar em funcionamento esta técnica de tunelamento denominada 6*over*4, devem-se efetuar as configurações descritas na Subseção 4.3.2 do Capítulo anterior. Concluídos os procedimentos de configurações deve-se verificar à conectividade entre os dois *hosts* do túnel, no caso representados por *host*A e *host*B. Segue na Figura 36 o teste de conectividade realizado do *host*A [2013:db8::abcd] para o *host*B [2013:db8::dcba], através do comando *ping6*.

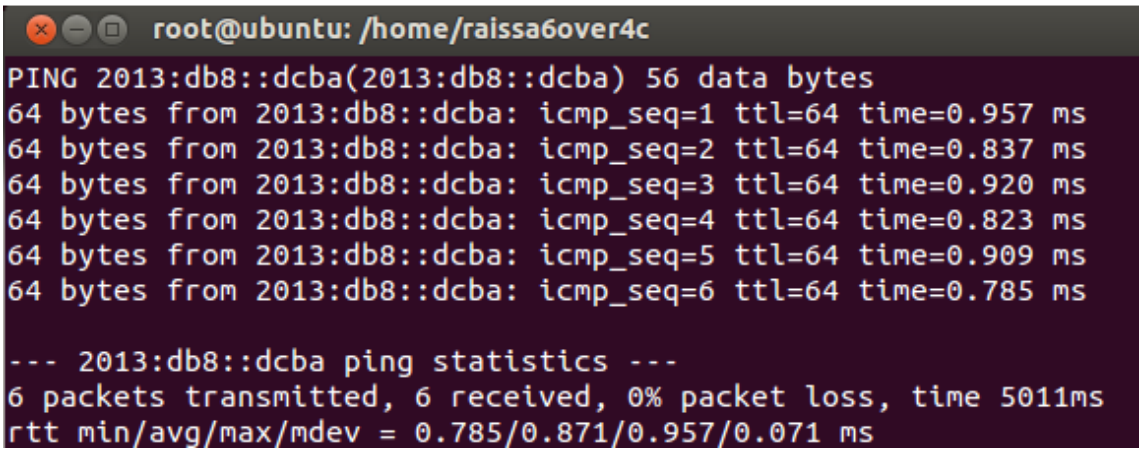

Figura 36: Teste de conectividade através do *Tunnel* 6*over*4.
Fonte: Do Autor.

A validação de tal comunicação via *Tunnel* 6*over*4 pode ser comprovada com a análise dos pacotes recebidos no *host* destinatário, no caso, o *host*B. Tal verificação foi concedida a partir da auditoria destes pacotes com o *Wireshark*, este é um *software* para análise de protocolos em tráfego de rede. A seguir na Figura 37 pode-se verificar o tipo de protocolo encapsulado (Protocolo 41) na comunicação estabelecida, o qual define que a técnica de tunelamento é a 6*over*4.

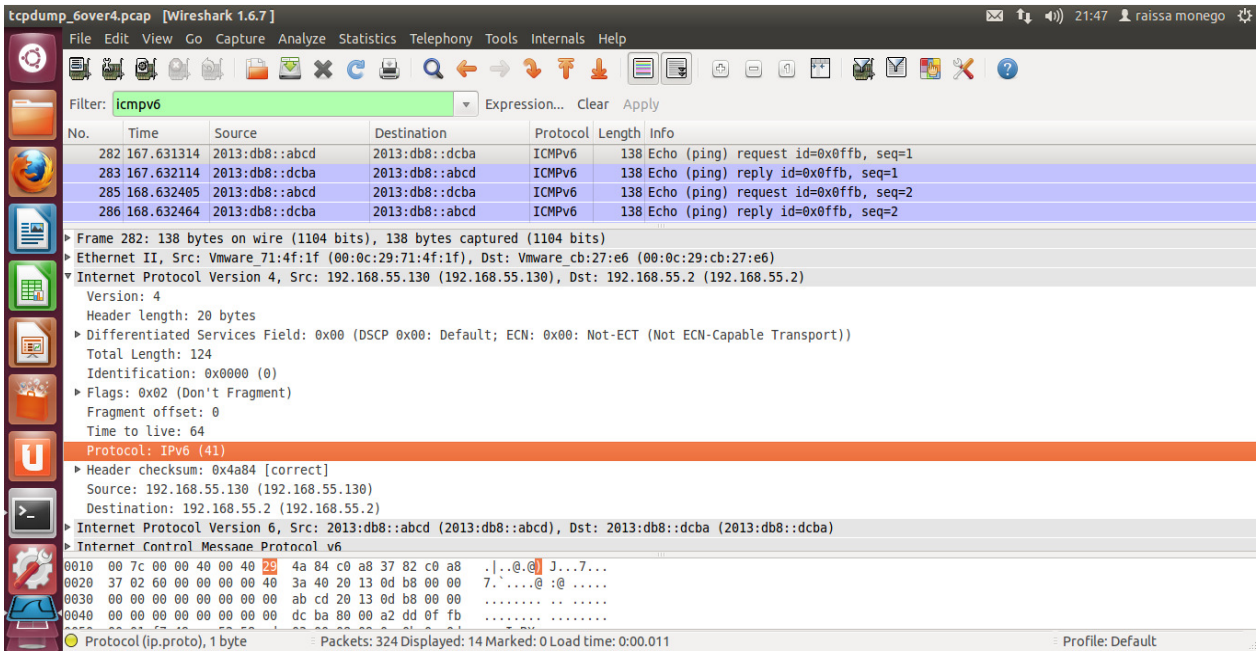

Figura 37: Verificação de comunicação via *Tunnel* 6*over*4.
Fonte: Do Autor.

Onde pode-se observar, na camada *Internet Protocol Version* 4, os endereços de origem e destino, no caso [192.168.55.130] e [192.168.55.2], correspondendo respectivamente à *host*A e à *host*B, nesta ordem. Também pode ser verificado, na camada *Internet Protocol Version* 6, que o endereço IPv6 de destino do *host*A é [2013:db8:: abcd], e o correspondente de origem do *host*B é [2013:db8::dcba]. Por fim, o mais importante, é analisar o protocolo utilizado na comunicação, onde é indicado o Protocolo IPv6 (número 41), que corresponde ao *Tunnel* 6*over*4.

5.3 Validação do *Tunnel* GRE

O funcionamento desta técnica de tunelamento denominada GRE (*Generic Routing Encapsulation*), depende das configurações descritas na Seção 4.3.3 do Capítulo anterior. Após efetuadas as configurações, deve-se testar o funcionamento do túnel. Para isto será feito uma verificação de conectividade entre os dois *hosts* do túnel, neste caso *host*A e *host*B. O mecanismo utilizado para verificar a conectividade entre *host*A [2013:abc::a] e *host*B [2013:abc:: b] foi o *ping6*, tal teste de conectividade é ilustrado na Figura 38.

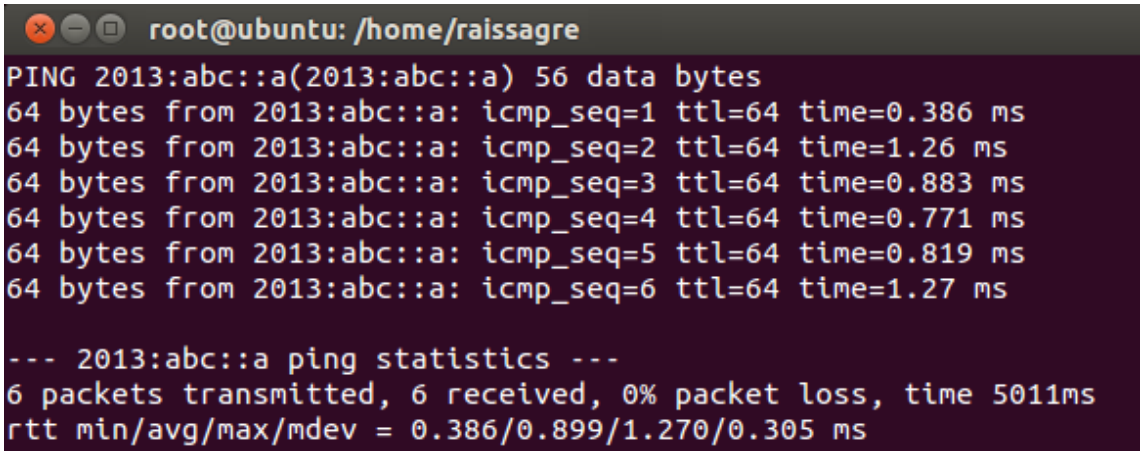

Figura 38: Teste de conectividade através do *Tunnel* GRE.
Fonte: Do Autor.

O processo de validação para se comprovar que esta comunicação foi feita através do *Tunnel* GRE pode ser observada com uma análise dos pacotes recebidos no *host*A, o destinatário. Esta auditoria foi feita a partir da análise dos protocolos tráfegos na comunicação testada. Segue na Figura 39 a tela de captura obtida no *Wireshark*, em que pôde-se verificar o tipo de protocolo encapsulado, o Protocolo GRE.

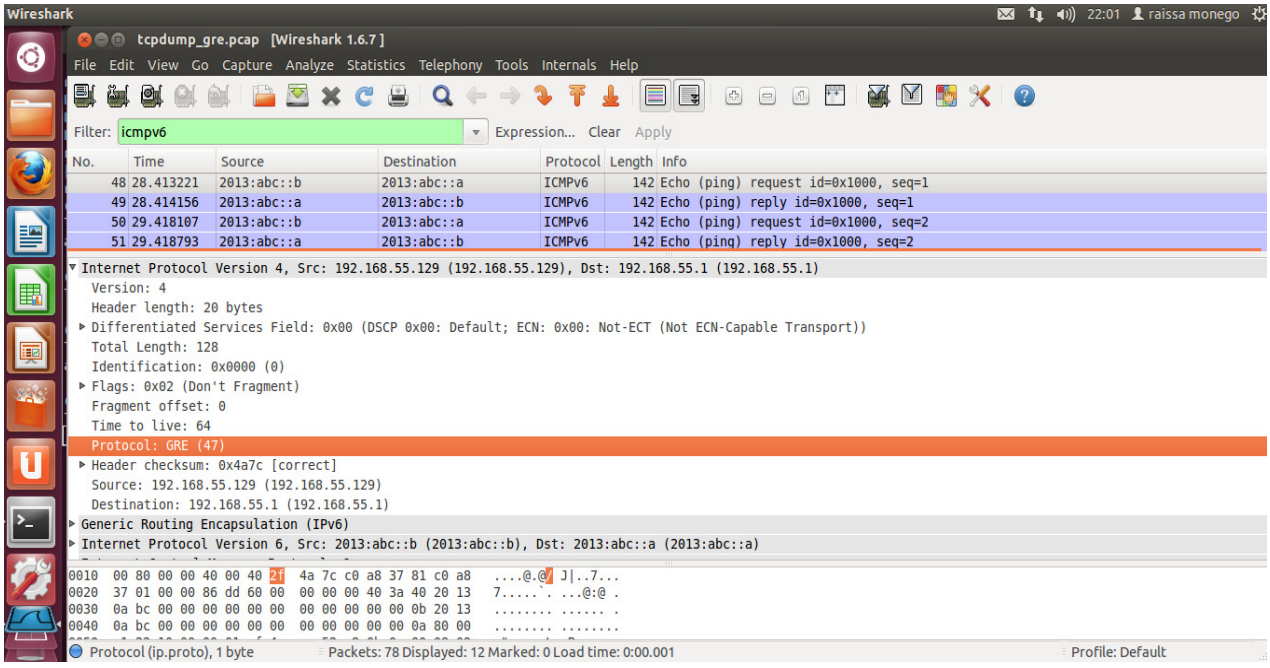

Figura 39: Verificação de comunicação via *Tunnel* GRE.
Fonte: Do Autor.

Pode-se observar na Figura 39, uma semelhança ao *Tunnel* 6*over*4, que na camada *Internet Protocol Version* 4, também contém os endereços IPv4 de origem (*host*B) e destino (*host*A), no caso [192.168.55.129] e [192.168.55.1]. Observam-se também, na camada *Internet Protocol Version* 6, os endereços IPv6 de origem e de destino, correspondentes ao *host*A [2013:abc::a] e *host*B [2013:abc::b], nesta ordem. Por fim, outro fator importante, é verificar o protocolo que foi utilizado na comunicação, onde deve ser o Protocolo GRE (número 47), o qual indica que o *Tunnel* é do tipo GRE.

6 CONSIDERAÇÕES FINAIS

As técnicas 6*over*4 e GRE mostraram-se métodos eficazes para a comunicação entre redes IPv6 através da *Internet* IPv4, no entanto, ambas técnicas de tunelamento limitam-se à troca de informações entre dois *hosts* específicos na rede, de forma similar à uma comunicação ponto-a-ponto.

Este estudo propôs a criação de um servidor túnel para que uma rede local IPv6 possa acessar endereços IPv6 já implantados na *Internet* de nova geração (Protocolo IPv6), como por exemplo os sítios do Google e do Terra. Sendo assim, as técnicas de tunelamento 6over4 e GRE foram implantadas e testadas em um segmento de rede próprio. Por este motivo, optou-se por utilizar a técnica de tunelamento para contemplar os aspectos desta proposta, tal técnica é denominada *Tunnel Broker*.

A técnica de tunelamento *Tunnel Broker* apresentou-se como a solução para o problema proposto neste estudo, para acessar endereços do protocolo IPv6 na *Internet*. Para isto, foi implemtentada uma rede local IPv6, com um servidor de túnel *Broker*. Onde este conectava-se à um provedor de acesso de *Tunnel* IPv6, o *Freenet* 6, também conhecido por Gogo6. Desta forma, todos os *hosts* da rede local conectavam-se à *Internet* IPv6 através deste servidor de *Tunnel Broker Freenet* 6.

Tal servidor da rede local, além de ser um Servidor de *Tunnel* (o qual é o cerne deste estudo), também possui os serviços de DHCPv6, BIND e Apache. O servidor DHCPv6 tem por finalidade à distribuição de endereços IPv6 aos *hosts* da rede interna. O servidor DNS (BIND) tem o objetivo de resolução do nome de *hosts* da rede local, como por exemplo o sítio <maquina4.raissa.tcc.ufsm.br> hospedado em um *host* da rede interna. E o servidor *web* (Apache) é utilizado para possibilitar a hospedagem de sítios IPv6 locais, sendo uma ideia para trabalho futuro fazer com que estes sites sejam disponibilizados na *web* IPv6 quando esta infraestrutura estiver concluída.

Quanto aos trabalhos futuros, poderá ser implementada a técnica de Pilha Dupla, a fim de, esta rede IPv6 implementada conseguir acesso a sítios atuais (IPv4), tornando os dois protocolos compatíveis por meio da técnica de Pilha Dupla, esta consiste em um nodo comportar-se como IPv4 e/ou IPv6, conforme necessário. Outro trabalho interessante é implantar esta rede IPv6 com a utilização de técnicas de transição em um ambiente corporativo, apresentando todas as funcionalidades do protocolo e tornar mais simples a transição entre protocolos.

A migração para a nova versão do protocolo é um processo que irá ocorrer naturalmente, vindo a acontecer de forma gradual. Em pouco tempo este procedimento de transição expandir-se-á na *Internet*. Sendo assim, este trabalho objetivou uma contribuição para auxiliar profissionais da área de Tecnologia da Informação na implantação deste protocolo, o IPv6.

7 REFERÊNCIAS BIBLIOGRÁFICAS

ARAÚJO, B. O.; ALMEIDA, I. S.; SILVA, L. A. **IPv6 – Funcionalidades e Métodos de Transição.** Trabalho de Conclusão de Especialização em Redes de Computadores e Telecomunicações. UNIFACS, 2011.

BRADNER, S.; MANKIN, A. **[RFC - 1752] - The Recommendation for the IP Next Generation Protocol.** Network Working Group. December, 1995. Disponível em: <http://www.ietf.org/rfc/rfc1752.txt>. Acessado em: 25/09/2013.

BRAGA, F. G. **A continuidade da Internet passa pelo IPv6.** Trabalho de Conclusão de Curso. São Paulo – SP, 2011.

BUGALLO, A. M. D.; Barros, M. A.;Torres, W. R. **Introdução ao DHCP**. Disponível em: <http://www.rnp.br/newsgen/9911/dhcp.html#ng-dhcp>. Acesso em: 10/10/2013.

CARISSIMI, A. da S.; ROCHOL, J.; GRANVILLE, L. Z. **Redes de Computadores**. Porto Alegre, Bookman, 2009.

COMER, D. E. **Redes de Computadores e a Internet:** Abrange Transmissão de Dados, Ligações Inter-Redes, Web e Aplicações. 4 ed. Bookman, 2007.

COMER, D. E. **Interligação em rede com TCP/IP - Princípios, Protocolos e Arquitetura**. Vol. 1. 5 ed. Rio de Janeiro: Campus, 2005.

DEERING, S; HINDEN, R. **[RFC-2460] - Internet Protocol, Version 6 (IPv6).** Network Working Group. Dezembro, 1998. Disponível em: <http://www.ietf.org/rfc/rfc2460.txt>. Acessado em: 08/09/2013.

DOMINGOS, F. D. **TÉCNICA DE TRANSIÇÃO ENTRE REDES IPV4/IPV6.** Revista de Ciencias Exatas e Tecnologia, Vol. 1, 2011.

DURAND, A; FASANO, P; GUARDINI, I. **[RFC - 3053] IPv6 Tunnel Broker.** Network Working Group. Janeiro, 2001. Disponível em: <http://www.ietf.org/rfc/rfc3053.txt>. Acessado em: 17/10/2013.

FARINACCI, D; LI, T; HANKS, S; MEYER, D; TRAINA P. **[RFC - 2784] Generic Routing Encapsulation (GRE).** Network Working Group. Março, 2000. Disponível em: <http://www.ietf.org/rfc/rfc2784.txt>. Acessado em: 17/10/2013.

FLORENTINO, A. A. **IPv6 na Prática.** 1 ed. São Paulo, Coleção Academy, 2012.

HAGEN, S. IPv6 Essentials. 2 ed. O'Reilly Book, United States of America, 2006.

IPv6 – Aplicabilidade. O Esgotamento dos Endereços IPv4. Disponível em: <https://sites.google.com/site/ipv6aplicabilidade/o-esgotamento-dos-enderecos-ip-1>. Acessado em: 15/10/2013.

KUROSE, J. F.; ROSS, K. W. **Redes de Computadores e a Internet:** Uma abordagem Top-Down. 5 ed., São Paulo. Addison Wesley, 2010.

LIU, C. **DNS and BIND on IPv6.** O'Reilly Book, United States of America, 2011.

MOHD, K. S.; HASSAN, R.; PATEL, A. **A comparative Review of IPv4 and IPv6 for Research Test Bed**. Department of Computer Science, Universiti Kebangsaan Malaysia. Malaysia, 2009.

MOREIRAS, A. M., et al**. Curso IPv6 básico**, 2010. Disponível em: <http://www.ipv6.br/download>. Acesso em: 13/10/2013.

MOREIRAS, A. M., et al. **Técnicas de Transição do IPv4 para o IPv6**, 2012. Disponível em: <http://www.ipv6.br/download>. Acesso em: 13/10/2013.

NORDMARK, E; GILLIGAN, R. **[RFC - 4213] Basic Transition Mechanisms for IPv6 Hosts and Routers.** Network Working Group. Outubro, 2005. Disponível em: <http://www.ietf.org/rfc/rfc4213.txt>. Acesso em: 17/10/2013.

SANTOS, R. R. **Curso de IPv6 básico.** 1 ed. São Paulo, 2009.

SILVEIRA, A. M. **Rede IPv6 com Integração IPv4.** Trabalho de Conclusão de Curso. São José - SC, 2012.

STALLINGS, W. **Redes e Sistemas de Comunicação de Dados:** Teoria e aplicações corporativas. 5 ed., Rio de Janeiro. Elsevier, 2005.

TANENBAUM, A. S.; WETHERALL, D. J. **Redes de Computadores.** 5 ed., Rio de Janeiro: Elsevier, 2011.

APÊNDICE A– Arquivo de Configuração do *Tunnel Broker* via *Freenet* 6

```
#-----------------------------------------------------------------------------------------------------
# $Id: gogoc.conf.in,v 1.1 2009/11/20 16:53:12 jasminko Exp $
#-----------------------------------------------------------------------------------------------------

##########################      READ ME!      ################################
#
# Welcome to the gogoCLIENT configuration file.
# In order to use the client, you need to modify the 'userid', 'passwd' and
# 'server' parameters below depending on which of these situations applies:
#
# 1. If you created a Freenet6 account, enter your userid and password below.
#    Change the server name to "broker.freenet6.net" and auth_method to 'any'.
# 2. If you would like to use Freenet6 without creating an account,
#    do not make any modifications and close this file.
# 3. If this software was provided by your ISP, enter the userid, password and
#    server name provided by your ISP below.
#


########################## BASIC CONFIGURATION ##########################

#
# User Identification and Password:
#   Specify your user name and password as provided by your ISP or Freenet6.
#   If you plan to connect anonymously, leave these values empty.
#   NOTE: Change auth_method option if you are using a username/password.
#
#   userid=<your_userid>
#   passwd=<your_password>
#
userid=raissamonego
passwd=tccipv6ufsm

#
# gogoSERVER:
#   Specify a gogoSERVER name or IP address (provided by your ISP or
#   Freenet6). An optional port number can be added; the default port number
#   is 3653.
```

```
#
#   Examples:
#     server=hostname # FQDN
#     server=A.B.C.D  # IPv4 address
#     server=[X:X::X:X] # IPv6 address
#     server=hostname:port_number
#     server=A.B.C.D:port_number
#     server=[X:X::X:X]:port_number
#
#   Freenet6 account holders should enter authenticated.freenet6.net,
#   otherwise use anonymous.freenet6.net.
#   Your ISP may provide you with a different server name.
#
#server=anonymous.freenet6.net
#server=authenticated.freenet6.net
server=montreal.freenet6.net

#
# Authentication Method:
#
#  auth_method=<{anonymous}|{any|passdss-3des-1|digest-md5|plain}>
#
#   anonymous:      Sends no username or password
#
#   any:            The most secure method will be used.
#   passdss-3des-1: The password is sent encrypted.
#   digest-md5:     The password is sent encrypted.
#   plain:          Both username and password are sent as plain text.
#
# Recommended values:
#   - any:          If you are authenticating a username / password.
#   - anonymous:    If you are connecting anonymously.
#
#auth_method=anonymous
#auth_method=any
auth_method=digest-md5

########################## ROUTING CONFIGURATION ##########################
# Use these parameters when you wish the client to act as a router and provide
# IPv6 connectivity to IPv6-capable devices on your network.
```

```
#
# Local Host Type:
#   Change this value to 'router' to enable IPv6 advertisements.
#
#   host_type=<host|router>
#
host_type=router

#
# Prefix Length:
#   Length of the requested prefix. Valid values range between 0 and 64 when
#   using V6*V4 tunnel modes, and between 0 and 32 when using V4V6 tunnel mode.
#
#   prefixlen=<integer>
#
prefixlen=64

#
# Advertisement Interface Prefix:
#   Name of the interface that will be configured to send router advertisements.
#   This is an interface index on Windows (ex: 4) and a name on Linux
#   and BSD (ex: eth1 or fxp1).
#
#   if_prefix=<interface name>
#
if_prefix=eth1

#
# DNS Server:
#   A DNS server list to which the reverse prefix will be delegated. Servers
#   are separated by the colon(:) delimiter.
#
#   Example: dns_server=ns1.domain:ns2.domain:ns3.domain
#
dns_server=raissa.tcc.ufsm.br


########################## ADVANCED CONFIGURATION ##########################
```

```
#
# gogoCLIENT Installation Directory:
#   Directory where the gogoCLIENT will be installed. This value has been
#   set during installation.
#
gogoc_dir=/home/raissatunnelc/gogoc-1_2-RELEASE/

#
# Auto-Retry Connect, Retry Delay and Max Retry Delay:
#  When auto_retry_connect=yes, the gogoCLIENT will attempt to reconnect
#  after a disconnection occurred. The time to wait is 'retry_delay' and that
#  delay is doubled at every 3 failed consecutive reconnection attempt.
#  However, the wait delay will never exceed retry_delay_max.
#
#
#   auto_retry_connect=<yes|no>
#   retry_delay=<integer: 0..3600>
#   retry_delay_max=<integer: 0..3600>
#
#   Recommended values: "yes", 30, 300
#
auto_retry_connect=yes
retry_delay=30
retry_delay_max=300

#
# Keepalive Feature and Message Interval:
#   Indicates if and how often the client will send data to keep the tunnel
#   active.
#
#   keepalive=<yes|no>
#   keepalive_interval=<integer>
#
#   Recommended values: "yes" and 30
#
keepalive=yes
keepalive_interval=30

#
# Tunnel Encapsulation Mode:
```

```
#   v6v4:    IPv6-in-IPv4 tunnel.
#   v6udpv4: IPv6-in-UDP-in-IPv4 tunnel (for clients behind a NAT).
#   v6anyv4: Lets the broker choose the best mode for IPv6 tunnel.
#   v4v6:    IPv4-in-IPv6 tunnel.
#
#   Recommended value: v6anyv4
#
tunnel_mode=v6anyv4

#
# Tunnel Interface Name:
#   The interface name assigned to the tunnel. This value is O/S dependent.
#
#   if_tunnel_v6v4 is the tunnel interface name for v6v4 encapsulation mode
#   if_tunnel_v6udpv4 is the tunnel interface name for v6udpv4 encapsulate mode
#   if_tunnel_v4v6 is the tunnel interface name for v4v6 encapsulation mode
#
#   Default values are set during installation.
#
if_tunnel_v6v4=sit1
if_tunnel_v6udpv4=tun
if_tunnel_v4v6=sit0

#
# Local IP Address of the Client:
#   Allows you to set a specific address as the local tunnel endpoint.
#
#   client_v4=<auto|A.B.C.D (valid ipv4 address)>
#   client_v6=<auto|X:X::X:X (valid ipv6 address)>
#     auto: The gogoCLIENT will find the local IP address endpoint.
#
#   Recommended value: auto
#
client_v4=auto
client_v6=auto

#
# Script Name:
#   File name of the script to run to install the tunnel interface. The
#   scripts are located in the template directory under the client
```

```
#   installation directory.
#
#   template=<checktunnel|freebsd|netbsd|openbsd|linux|windows|darwin|cisco|solaris>
#
#   Default value is set during installation.
#
template=linux

#
# Proxy client:
#   Indicates that this client will request a tunnel for another endpoint,
#   such as a Cisco router.
#
#   proxy_client=<yes|no>
#
#   NOTE: NAT traversal is not possible in proxy mode.
#
proxy_client=no


############################## BROKER REDIRECTION ##############################

#
# Broker List File Name:
#   The 'broker_list' directive specifies the filename where the broker
#   list received during broker redirection will be saved.
#
#   broker_list=<file_name>
#
broker_list=tsp-broker-list.txt

#
# Last Server Used File Name:
#   The 'last_server' directive specifies the filename where the address of
#   the last broker to which a connection was successfully established will
#   be saved.
#
#   last_server=<file_name>
#
last_server=tsp-last-server.txt
```

```
#
# Always Use Last Known Working Server:
#   The value of the 'always_use_same_server' directive determines whether the
#   client should always try to connect to the broker found in the
#   'last_server' directive filename.
#
#   always_use_same_server=<yes|no>
#
always_use_same_server=no


################################### LOGGING ################################

#
# Log Verbosity Configuration:
#   The format is 'log_<destination>=level', where possible values for
#   'destination' are:
#
#   - console  (logging to the console [AKA stdout])
#   - stderr   (logging to standard error)
#   - file     (logging to a file)
#   - syslog   (logging to syslog [Unix only])
#
#   and 'level' is a digit between 0 and 3. A 'level' value of 0 disables
#   logging to the destination, while values 1 to 3 request increasing levels
#   of log verbosity and detail. If 'level' is not specified, a value of 1 is
#   assumed.
#
#   Example:
#     log_file=3   (Maximal logging to a file)
#     log_stderr=0 (Logging to standard error disabled)
#     log_console= (Minimal logging to the console)
#
#   - Default configuration on Windows platforms:
#
#     log_console=0
#     log_stderr=0
#     log_file=1
#
```

```
#   - Default configuration on Unix platforms:
#
#     log_console=0
#     log_stderr=1
#     log_file=0
#     log_syslog=0
#
#log_console=
#log_stderr=
#log_file=
#log_syslog=

#
# Log File Name:
#   When logging to file is requested using the 'log_file' directive, the name
#   and path of the file to use may be specified using this directive.
#
#   log_filename=<file_name>
#
log_filename=gogoc.log

#
# Log File Rotation:
#   When logging to file is requested using the 'log_file' directive, log file
#   rotation may be enabled. When enabled, the contents of the log file will
#   be moved to a backup file just before it reaches the maximum log file size
#   specified via this directive.
#
#   The name of the backup file is the name of the original log file with
#   '.<timestamp>' inserted before the file extension. If the file does not
#   have an extension, '.<timestamp>' is appended to the name of the original
#   log file. The timestamp specifies when the rotation occurred.
#
#   After the contents of the log file have been moved to the backup file, the
#   original file is cleared, and logging resumes at the beginning of the file.
#
#   log_rotation=<yes|no>
#
log_rotation=yes
```

```
#
# Log File Rotation Size:
#   The 'log_rotation_size' directive specifies the maximum size a log file may
#   reach before rotation occurs, if enabled. The value is expressed in
#   kilobytes.
#
#   log_rotation_size=<16|32|128|1024>
#
log_rotation_size=32

#
# Deletion of rotated log files:
#   The 'log_rotation_delete' directive specifies that no log backup will be
#   kept. When rotation occurs, the file is immediately wiped out and a new
#   log file is started.
#
#   log_rotation_delete=<yes|no>
#
log_rotation_delete=no

#
# Syslog Logging Facility [Unix Only]:
#   When logging to syslog is requested using the 'log_syslog' directive, the
#   facility to use may be specified using this directive.
#
#   syslog_facility=<USER|LOCAL[0-7]>
#
syslog_facility=USER


# end of gogoc.conf
#------------------------------------------------------------------------
```

APÊNDICE B – Arquivo para Configuração do *Tunnel* 6*over*4

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

#interfaces config
sudo ifconfig eth2 192.168.55.130/25 up
sudo ifconfig eth2 inet6 add 2013:db8::abcd/64 up

#route network config
/sbin/route add -net 192.168.55.0/24 dev eth2

#6over4
/sbin/ip tunnel add 6over4 mode sit ttl 64 remote 192.168.55.2 local 192.168.55.130
/sbin/ip link set dev 6over4 up
/sbin/ip -6 route add 2013:db8::dcba dev 6over4

exit 0
```

APÊNDICE C – Arquivo para Configuração do *Tunnel* GRE

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

#interfaces config
sudo ifconfig eth1 192.168.55.129/25 up
sudo ifconfig eth1 inet6 add 2013:abc::b/64 up

#route network config
/sbin/route add -net 192.168.55.0/24 dev eth1

#gre
/sbin/ip tunnel add gre mode gre ttl 64 remote 192.168.55.1 local 192.168.55.129
/sbin/ip link set dev gre up
/sbin/ip -6 route add 2013:abc::a dev gre

exit 0
```

APÊNDICE D – Instalação e Configuração DHCPv6

O primeiro passo para instalação do serviço inicia-se com o pacote *isc-dhcp-server*, instala-se o mesmo através do comando abaixo:

```
# apt-get install isc-dhcp-server
```

Concluída a instalação, deve-se efetuar a criação do arquivo de configuração do servidor DHCPv6, denominado *dhcpd6.conf*. Este deve estar localizado no diretório */etc/dhcp/*. Tal arquivo serve para adição dos parâmetros para execução do servidor.

```
# Configuration file DHCPv6 server
# The ddns-updates-style parameter
ddns-update-style none;
update-static-leases off;
# option definitions common to all supported networks...
#option domain-name "example.org";
#option domain-name-servers ns1.example.org, ns2.example.org;

#max-lease-time 7200;

#default-lease-time 2592000;
#preferred-lifetime 604800;
#option dhcp-renewal-time 3600;
#option dhcp-rebinding-time 7200;

#allow leasequery;

option dhcp6.name-servers 2013:0a26::c0a8:1;
option dhcp6.domain-search "raissa.tcc.ufsm.br";

#option dhcp6.info-refresh-time 21600;
# If this DHCP server is the official DHCP server for the local
# network, the authoritative directive should be uncommented.
authoritative;

# Use this to send dhcp log messages to a different log file
#log-facility local7;
# Subnets DHCP server
#subnet 10.5.5.0 netmask 255.255.255.224 {
```

```
#  range 10.5.5.26 10.5.5.30;
#  option domain-name-servers ns1.internal.example.org;
#  option domain-name "internal.example.org";
#  option routers 10.5.5.1, router.example.com;
#  option broadcast-address 10.5.5.31;
#  default-lease-time 600;
#  max-lease-time 7200;
#}

# Hosts Pool for Dynamic Addresses
shared-network IPv6 {
  subnet6 2013:0a26:: /64 {
       range6 2013:0a26::c0a8:2 2013:0a26::c0a8:ffff;
  }
}
# Hosts Fixed Addresses
# host IPv6clientFixed {
#  host-identifier option dhcp6.client-id 00:01:00:02:08:00:27:d2:2e:3e;
#  fixed-address6 2013:0a26::c0a8:100;
#}

# Hosts Dynamic Addresses with assigned DNS Server
# host IPv6client {
#  host-identifier option dhcp6.client-id 00:01:00:02:08:00:27:d2:2e:3e;
#  host-identifier option dhcp6.client-id 00:01:00:02:MAC;
#  option dhcp6.name-servers 2013:0a26::c0a8:1;
#}
```

Terminada esta etapa, o próximo procedimento é a definição da interface para distribuição de endereços IPs. Para este estudo a interface definida foi a interface *eth1*, tal procedimento é configurado no arquivo */etc/default/isc-dhcp-server*.

```
#Defaults for dhcp initscript
#sourced by /etc/init.d/dhcp
#installed at /etc/default/isc-dhcp-server by the maintainer scripts
#

#This is a POSIX shell fragment
#
```

```
#On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#Separate multiple interfaces with spaces, e.g. “eth0 eth1”.
INTERFACES="eth1"
```

Por fim, para concluir o procedimento de instalação e configuração do servidor DHCPv6, deve-se efetuar a inicialização do mesmo. Tal etapa é concluída com a execução do comando à seguir, vale ressaltar que tal comando deve ser executado como superusuário no terminal do servidor.

```
# /etc/init.d/isc-dhcp-server6 start
```

APÊNDICE E – Instalação e Configuração do DNS

Sua instalação é simples, basta utilizar o comando abaixo para iniciar.

```
# apt-get install bind9
```

Após devem ser feitas alterações no arquivo */etc/bind/named.conf.local*, onde são configuradas as zonas de domínio do DNS Reverso e do DNS, para isto é necessária a criação um arquivo de configuração para cada um. Estes, foram criados com os nomes de *db.rdns* (DNS Reverso) e *db.dns* (DNS), os quais recebem os endereços e configurações necessárias para seu funcionamento e são encontrados no diretório */etc/bind/*.

Dentro do arquivo *named.conf.local* são inseridas as seguintes informações:

```
zone "8.a.0.c.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.6.2.a.0.3.1.0.2.ip6.arpa" {
type master;
notify no;
file "/etc/bind/db.rdns";
allow-query { any; };
};
zone "raissa.tcc.ufsm.br" {
type master;
file "/etc/bind/db.dns";
allow-query { any; };
};
```

O arquivo *db.rdns*, que é a configuração do DNS Reverso, foi configurado da seguinte forma:

```
; Arquivo reverso do dominio raissa.tcc.ufsm.br
$TTL   800
$ORIGIN 8.a.0.c.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.6.2.a.0.3.1.0.2.ip6.arpa.
@       IN SOA ns.raissa.tcc.ufsm.br. maq4.raissa.tcc.ufsm.br. (
2013111201 ; Serial
604800 ; Refresh
4000 ; 86400 Retry
```

```
2419200 ; Expire
800 ) ; 604800 Negative Cache TTL
@       IN NS ns.
;@      IN NS raissa.
a.0.a.0 IN PTR maquina4.raissa.tcc.ufsm.br.
```

Por fim, o arquivo *db.dns*, foi configurado desta forma:

```
; Arquivo dns do dominio raissa.tcc.ufsm.br
;
$TTL  800
@       IN SOA ns.raissa.tcc.ufsm.br. maq4.raissa.tcc.ufsm.br. (
              2013111201 ; serial
              604800 ; refresh
              4000 ; 86400 retry
              2419200 ; expire
              800 ) ;
@       IN NS ns.
@       IN NS raissa.tcc.ufsm.br.
@       IN AAAA 2013:0a26::c0a8:0a0a
@       IN AAAA ::1
maquina4        IN AAAA 2013:0a26::c0a8:0a0a
```

Para testar o serviço, deve ser feita a reinicialização do *bind9*, com o comando apresentado a seguir:

```
# service bind9 restart
```

APÊNDICE F – Instalação e Configuração do *Web server*

O procedimento de instalação do servidor *web* é bastante simples, pois todas suas dependências estão inclusas no pacote *apache2*, para isto, deve-se executar, como superusuário, o comando abaixo no terminal.

```
# apt-get install apache2
```

Após concluída a instalação do servidor, foi feita alteração no arquivo */var/www/index.html*, onde está o repositório principal de hospedagem padrão do Servidor *Web* Apache. Visto que objetivo inicial era de testar a implantação do serviço na rede local, utilizando o protocolo IPv6, o arquivo da página *web* para testes ficou simplificado, segue abaixo o código na linguagem HTML.

```
<html>
<body>
<h1>TECNOLOGIA EM REDES DE COMPUTADORES / UFSM</h1>
<title>IPv6 Web Server</title>
<p>TRABALHO DE CONCLUS&Atilde;O DE CURSO</p>
<P>T&iacute;tulo: Implementa&ccedil;&atilde;o de uma rede utilizando os padr&otilde;es do Protocolo IPv6</P>
<P>Autora: Raissa Monego</P>
<p>Orientadora: Simone R. Ceolin</p>
<p>Coorientador: Renato P. Azevedo</p>
</body>
</html>
```

Finalizam-se instalação e configurações do servidor *web*, com a reinicialização do serviço para que as configurações sejam concluídas, visto que o mesmo inicializa-se automaticamente ao ser instalado. Para tal procedimento, deve-se efetuar o comando abaixo, executando-o como superusuário.

```
# service apache2 restart
```