Instituto Politécnico de Leiria
Escola Superior de Tecnologia e Gestão

IPv6@ESTG-Leiria
Mecanismos de transição

Luís Alberto Esteves Diogo
Óscar Filipe Correia Brilha

Leiria
Setembro de 2006

Instituto Politécnico de Leiria
Escola Superior de Tecnologia e Gestão

IPv6@ESTG-Leiria
Mecanismos de transição

Relatório final da cadeira de Projecto I, do curso de Licenciatura em Engenharia Informática e Comunicações, ano lectivo 2005/2006

Realizado entre Março e Julho de 2006

Autores:

Luís Alberto Esteves Diogo, nº 10429

Óscar Filipe Correia Brilha, nº 9061

Orientador: Prof. Nuno Veiga

Leiria

Setembro de 2006

Agradecimentos

Ao orientador Nuno Veiga pelas sugestões que nos apresentou e pela orientação dada.

Ao professor Mário Antunes pelas sugestões e ideias dadas que foram muito importantes para um aprofundamento de alguns temas abordados.

Ao Ricardo Santos e Ricardo Veríssimo pela camaradagem apresentada ao longo do tempo de elaboração do projecto.

Ao Vítor Santos e David Serafim pelo apoio dado e por terem elaborado um documento que nos serviu de ponto de partida para a elaboração do nosso projecto.

Ao nosso colega Nuno Cardoso pela preciosa ajuda nas configurações do sistema operativo Fedora Core 5.

Ás nossas famílias e namoradas pelo apoio dado.

Resumo

Este projecto tem como objectivo focar os diversos mecanismos de transição existentes explicando-os detalhadamente. É um documento que não explica o funcionamento do novo protocolo IPv6, este projecto centra-se essencialmente no capítulo da transição. Não é um documento para leigos em IPv6, no entanto optámos por desenvolver uma parte inicial que explica o essencial do protocolo em si para que se possa perceber o capítulo de transição nomeadamente os tipos de endereços. Foram também abordados alguns protocolos considerados importantes. Apesar da parte dos protocolos poderem não se enquadrar no capítulo de transição optámos por focar alguns deles uma vez que eles têm agora suporte para IPv4 e IPv6 podendo funcionar com os dois protocolos. Os principais mecanismos de transição para IPv6 assentam no uso de túneis, estes são focados neste documento de maneira bastante completa, primeiro numa abordagem teórica e posteriormente numa abordagem mais prática com aspectos relacionados com a implementação de casos concretos. O relatório está dividido em duas partes, uma primeira em que se foca o aspecto teórico da transição e uma segunda parte em que se elaboraram cenários de simulação.

Na transição para IPv6 não se podia deixar de falar em DNS, é também focado neste documento o estado do DNS para IPv6, o que se está a usar e o que se pensa fazer no futuro nomeadamente o uso dos novos registos A6.

São também abordados aspectos relacionados com os sistemas operativos mais usados, nomeadamente o *Windows XP* e Linux. Aspectos relacionados com as compatibilidades dos diversos mecanismos com estes sistemas operativos e modo de implementar algumas funcionalidades que à partida podem não estar disponíveis.

Índice

Lista de Siglas e Acrónimos

3GPP	3rd Generation Mobile System
AAAA	Authentication Authorization Accounting
ACK	Acknowledged
API	Aplication Programming Interface
APIPA	Automatic Private IP Addressing
ARP	Address Resolution Protocol
ARPANet	The Advanced Research Projects Agency Network
AS	Autonomous System
AVF	Active Virtual Forwarders
AVG	Active Virtual Gateway
BA	Binding Acknowledgement
BGP	Border Gateway Protocol
BIA	Bump-In-the-API
BIS	Bump-In-the-Stack
BOOTP	Bootstrap Protocol
BSD	Berkeley Software Distribution
BU	Binding Updates
CIDR	Classless Inter-Domain Routing
CLNS	ConnectionLess Network Service
CN	Correspondent node
DDoS	Distributed Denial-of-Service
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DNS-ALG	Domain Name Server – Aplication Layer Gateway
DNS-ALG	Domain Name Service – Application Level Gateway

DNSSEC	Domain Name System Security Protocol
DoS	Senial of Service
DSTM	Dual Stack Transition Mechanism
DUID	DHCP Unique Identifier
ED	Education Department
EGP	Exterior Gateway Protocol
EIC	Engenharia Informática e Comunicações
EIGRP	Enhanced Interior Gateway Routing Protocol
ESP	Encapsulating Security Payload
ESTG	Escola Superior de Tecnologia e Gestão
GLBP	Gateway Load Balancing Protocol
GNU	Gnu's Not Unix
GRE	Generic Routing encapsulation
HA	Home Agent
HÁ	Home Agent
HA	Home Agent
HSRP	Hot Standby Routing Protocol
IAID	Identity Association Identifier
IANA	Internet Assigned Numbers Authority
ICMP	Internet Control Message Protocol
ICMPv6	Internet Control Message Protocol para IPv6
ICV	Integrity Check Value
ID	Identifier
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IGMPv4	Internet Group Management Protocol version 4

IGP	Interior Gateway Protocol
IKE	Internet Key Exchange
IMS	IP Multimédia Subsystem
IOS	Internetwork Operating System
IP	Internet Protocol
IPL	Instituto Politécnico de Leiria
IPSec	IP Security
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISATAP	Intra-Site Automatic Tunnel Addressing Protocol
IS-IS	Intermediate System-to-Intermediate System
ISP	Internet Service Provider
L2TP	Layer 2 Tunneling Protocol
LD	Laboratory Development
LSA	Link State Advertisement
MAC	Mac address
MIPv4	Mobile IP version 4
MIPv6	Mobile IP version 6
MLD	Multicast Listener Discovery
MN	Mobile Node
MRU	Maximum Receive Unit
MTU	Maximum Transmission Unit
NAT	Network Address Translation
NAT-PT	Network Address Translation – Port Translation
ND	Neighbor Discovery
NSFNET	National Science Foundation Network

NTP	Network Time Protocol
OSI	Open System Interconnection
OSPF	Open Shortest Path First
OSPFv2	Open Shortest Path First version 2
OSPFv3	Open Shortest Path First version 3
OUI	Organizational Unit Identifier
PDA	Personal Digital Assistants
PDU	Protocol Data Unit
PPP	Point-to-Point Protocol
PTR	Pointer record
PXE	Pré-Boot Execution Environment
QoS	Quality of Service
RFC	Request For Comment
RIB	Routing Information Database
RIP	Routing Information Protocol
RPL	Routers Potencial List
RR	Return Routability
RR	Resource Record (DNS)
RSVP	Resource ReSerVation Protocol
SCTP	Stream Control Transmission Protocol
SIIT	Stateless IP/ICMP Translation algorithm
SPI	Security Parameters Index
TCP	Transmission Control Protocol
TEP	Tunel End Point
TLA	Top Level Aggregate
TLV	Type-Length-Value

ToS	Type of Service
TSIG	Transaction Signature
TTL	Time To Live
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunication System
USAGI	UniverSAl playGround for Ipv6
VoIP	Voice over IP
VPN	Virtual Private Network
VRRP	Virtual Router Redundancy Protocol
WAN	Wide Area Network
XP	Experience

Lista de Figuras

Lista de Tabelas

Introdução

O protocolo *Internet Protocol* (IP) é o protocolo responsável pelo transporte de dados na *Internet*. Actualmente, existem duas variantes deste protocolo: *IPv4* e *IPv6*.

Hoje em dia, o protocolo IPv4 ainda é o protocolo de rede mais usado no mundo.

Com o rápido crescimento da *Internet*, esta tecnologia (IPv4) tem-se esgotado e muitos esforços têm sido feitos para a sua evolução.

O protocolo IPv6 é uma nova versão do protocolo IP e encontra-se em plena fase de divulgação e cabe ao mundo académico divulgar e provar as inúmeras vantagens deste protocolo em relação ao seu antecessor, o protocolo IPv4.

O sucesso de uma nova tecnologia depende em grande parte da sua capacidade de adaptação à infra-estrutura já existente. Muitas tecnologias não obtiveram sucesso devido ao deficiente processo de transição.

A tecnologia IPv6 foi concebida de modo a que a sua transição seja efectuada suave e eficazmente. Este factor da transição foi tido como um factor muito importante na altura da sua concepção.

Durante os anos vindouros irão ocorrer duas etapas distintas, uma primeira em que existirão umas "ilhas" IPv6 num imenso "oceano" IPv4 e posteriormente o contrário, ou seja, quando todo o mundo tiver o protocolo ipv6 generalizado irão prevalecer umas ilhas IPv4 num imenso oceano IPv6.

Para assegurar a transição irão ser necessários mecanismos que permitam a coexistência dos dois protocolos, mecanismos esses que irão ser aprofundados neste projecto.

Para quem se questiona se o IPv6 será mesmo necessário, pensemos no rumo que as telecomunicações estão a tomar. Cada terminal, móvel ou fixo, terá um ou mais endereços IP. É impensável continuar com os endereços actuais. Porque?

O protocolo ipv6 irá prevalecer com toda a certeza, o que não se sabe é durante quanto tempo irá coexistir com o seu antecessor.

1 Porquê IPv6

O protocolo IPv6 está, de um modo geral, dividido em três características chave. A primeira é a capacidade que este protocolo tem no esquema de endereçamento IP que passa a ser composto por 128 bits. Pensado para a *Internet* de hoje em dia e para a *Internet* do futuro, esta nova versão protocolar permite assim uma enorme flexibilidade.

IPv6 foi concebido para ter em conta o crescimento exponencial da comunidade cibernautica disponibilizando um enorme número de endereços IP que poderão ser utilizados para qualquer equipamento que possua uma ligação à *Internet*;

A segunda baseia-se nas especificações do protocolo IPv6 que são mais claras e optimizadas, eliminando as características não utilizadas e obsoletas do protocolo anterior (IPv4) atingindo assim uma optimização do protocolo IP. A convergência das redes de telecomunicações futuras para a camada de rede comum, o protocolo IPv6, prevê o aparecimento de novos serviços sobre IP (*VoIP)*[1]*, streaming de vídeo em real-time*[2], etc). O protocolo IPv6 suporta intrinsecamente classes de serviço diferenciadas, em função das exigências e prioridades do serviço em causa.

A terceira é referente à mobilidade que está a tornar-se um factor muito importante na sociedade de hoje em dia. O protocolo IPv6 suporta a mobilidade dos utilizadores, onde estes poderão ser contactados em qualquer rede através do seu endereço IPv6 de origem.

1.1 Necessidade de um novo protocolo IP

O novo protocolo IPv6 oferece um completamente novo e bem estruturado protocolo IP que implementa todas as características de segurança (IPsec), Qualidade de Serviço (QoS – *Diffserv*[3] *e Intserv*[4]), e configuração (auto-configuração).

Este novo protocolo também foi desenhado tendo em conta as redes móveis que se prevêem, num futuro próximo, ser acedidas em qualquer lugar, a qualquer hora, sendo possível estar sempre on-line. IPv6 é considerado o *Backbone*[5] da futura sociedade da informação.

Factos e razões do protocolo IPv6:

- Espaço de endereçamento IPv4 praticamente esgotado;
- Número de dispositivos móveis ou mesmo dispositivos não móveis equipados com uma placa de rede para acesso à *Internet* irá crescer exponencialmente num futuro próximo (a continuação do uso do actual protocolo IPv4 iria criar "ilhas" de redes IP com mobilidade e segurança reduzidas);

[1] Voz sobre IP, telefonia IP.
[2] Transmissão de vídeo em tempo real.
[3] O método DiffServ opera sobre grandes volumes de dados em oposição a fluxos ou reservas individuais. Isto implica uma negociação para todos os pacotes de, por exemplo, um ISP, ou universidade.
[4] Em redes de computadores, o IntServ ou serviços integrados é um modelo que visa garantir a qualidade de serviço (QoS) em redes.
[5] Considerada a rede nativa.

- É o protocolo mandatário para as normas 3GPP, UMTS, IMS
- Tem um melhor e mais eficiente suporte no que respeita a segurança, qualidade de serviço e mobilidade;
- Reduz as despesas em operações de redes IP pois possui características que levam a um melhor planeamento e auto-configuração;
- Faz com que seja possível existir uma rede IP em qualquer lado fornecendo acesso à rede a qualquer hora e em qualquer lugar;
- Permite a computação em qualquer lugar e com isto uma enorme potencialidade em oportunidades de negócio e mudanças nos já existentes modelos de negócios;
- É considerado o *Backbone* da futura sociedade da informação;
- Já está a ser usado e a ser expandido, suportando todos os tipos de dispositivos [42].

1.2 Limitações IPv4

A versão corrente do protocolo IP (versão 4 ou IPv4) não mudou substancialmente desde a RFC 791, publicada em 1981. Esta versão provou ser robusta e facilmente implementável. Conseguiu sustentar uma rede informática (denominada ARPANet) pensada inicialmente somente para fins militares e académicos nos Estados Unidos tendo-se depois expandido globalmente para os dias de hoje ficando conhecida como a *Internet* [52].

No entanto o desenho inicial do protocolo IPv4 não antecipou os seguintes pontos:

- O crescimento exponencial da *Internet* levou a que os endereços IPv4 se tornassem escassos principalmente na China e na Índia. Devido a este facto, muitos utilizadores são forçados a usar um mecanismo denominado de NAT (*Network Address Translation*) para mapear múltiplos endereços IP privados para um único endereço IP público. O mecanismo NAT possibilita a reutilização do espaço de endereçamento IP privado, mas não suporta padrões base de segurança ao nível da camada de rede nem o correcto mapeamento de todas as camadas protocolares acima dessa camada. Este mecanismo também pode criar problemas quando duas organizações estão conectadas e utilizam um espaço de endereçamento IP privado. Para além disso, o crescimento dos dispositivos e aplicações ligados à *Internet* provam que o espaço de endereçamento IP público irá esgotar.
- A necessidade de segurança na *Internet*. As comunicações privadas feitas sobre a *Internet* requerem serviços de encriptação que protegem os dados de serem visualizados ou modificados durante o seu percurso. Para isso existe o mecanismo IPSec[6] que fornece esta segurança nos pacotes IPv4. No entanto este padrão é opcional e as soluções proprietárias prevalecem.
- O crescimento da *Internet* e a capacidade que os routers do *Backbone* desta têm para de manterem grandes tabelas de *Routing*. Devido ao modo como as redes IPv4 estão correntemente alocadas, as tabelas de *Routing* dos routers do *Backbone* da *Internet* contêm habitualmente mais de 85000 rotas.
- Necessidade de melhorar o suporte de entrega de pacotes de dados em tempo real – denominado QoS (*Quality of Service*). Os padrões de QoS existem para IPv4, mas a sustentação do tráfego em tempo real é feito pelo campo ToS (*Type of Service*) que identifica o *payload*[7] do tráfego recebido, que é feito tipicamente utilizando uma porta TCP ou UDP. Infelizmente o campo ToS tem uma funcionalidade limitada tendo sido mais tarde redefinido e interpretado localmente. Além disso, a identificação do tráfego utilizando a porta TCP ou UDP não é possível quando este está cifrado. [45]

[6] IPSecurity, protocolo de segurança para o protocolo IPv4 e agora também para IPv6. Opcional para IPv4, obrigatório para IPv6.
[7] Carga do pacote.

- Necessidade de configurar redes informáticas de um modo mais simples. A maior parte das configurações em IPv4 têm de ser efectuadas manualmente ou então utilizando uma gama estática de endereços IP distribuídos entre os terminais através do protocolo DHCP. Com o aumento de computadores e dispositivos que utilizam endereços IP, há a necessidade de simplificar e automatizar a configuração destes e outros esquemas de configuração que não são compatíveis numa administração de uma infra-estrutura DHCP.
- Falta de suporte para as aplicações emergentes. Estas novas aplicações são mais exigentes e requerem garantia de entrega de pacotes de dados em tempo real, garantia de disponibilidade de largura de banda e garantia de largura de banda segura.[45]

1.3 IPv6 vs IPv4

A *Internet* vai precisar de uma nova versão do protocolo IP. Dois factores vão orientar esta necessidade: encaminhamento e endereçamento. Actualmente, o encaminhamento na *Internet* baseado em endereços de 32-bit (IPv4) está saturado[8]. O IPv4 não dispõe de qualquer flexibilidade para a construção de uma hierarquia eficiente que possa ser agregada. Os esforços para gerir o encaminhamento continuam a aumentar.

O desafio do novo protocolo IPv6 é que a transição seja concluída antes que o endereçamento e encaminhamento via IPv4 se torne impossível. Existem duas exigências importantes em relação à transição, que são: flexibilidade de desenvolvimento e a possibilidade de máquinas com IPv4 comunicarem com máquinas IPv6.

Na tabela seguinte estão representadas as principais características e diferenças de ambos os protocolos IP.

IPv4	Vs	IPv6
Endereços IP de origem e de destino têm 32 bits (4 bytes) de tamanho.		Endereços IP de origem e de destino têm 128 bits (16 bytes) de tamanho.
Suporte IPSec é opcional.		Suporte IPSec é obrigatório.
Não identifica os pacotes pertencentes a um fluxo de QoS pois o tratamento dos pacotes IP efectuado pelos routers é feito sem o cabeçalho IP.		É possível a identificação dos fluxos de pacotes QoS por parte dos routers pois os pacotes IPv6 utilizam um campo destinado para o efeito denominado *Flow Label*.
A *Fragment*ação de pacotes IP é efectuada quer pela máquina de origem, quer pelos routers, atrasando a performance dos routers.		A *Fragment*ação é efectuada somente pela máquina de origem.

8 Não existem mais endereços disponíveis para atribuir.

Não tem restrições quanto ao tamanho dos pacotes na camada de rede e tem de ser capaz de unir novamente os pacotes de 576 bytes.	A camada de rede tem de suportar pacotes de 1280 bytes e de unir novamente os pacotes de 1500 bytes.
O cabeçalho inclui o *checksum*.	O cabeçalho não inclui o *checksum*.
O cabeçalho inclui o campo opções.	Toda a informação opcional está incluída no cabeçalho de estenção.
O mecanismo ARP usa *frames Broadcast* ARP *requests* para resolver endereços IPv4 para endereços da camada de ligação.	As *frames* ARP *requests* são substituídas por mensagens *Multicast Neighbor Solicitation*.
É usado o protocolo IGMP (*Internet Group Management Protocol*) para lidar com membros de uma rede local.	IGMP é substituído por mensagens MLD (*Multicast Listener Discovery*).
É utilizado o protocolo ICMP *Router Discovery* para determinar os endereços IPv4 para a melhor porta padrão de saída e é opcional.	O ICMP *Router Discovery* é substituído por mensagens ICMPv6 *Router Solicitation* e *Router Advertisement* e é obrigatório.
São usados endereços de *Broadcast* para enviar tráfego para todos os nós de uma subrede.	Não existem endereços IPv6 de *Broadcast*. Em vez de ser utilizado um endereço *Link-local* necessitando verificar todos os nós da rede, é utilizado um endereço *Multicast*.
Tem de ser configurado manualmente ou através do protocolo DHCP.	Não requer qualquer configuração manual nem precisa de recorrer ao protocolo DHCP para atribuição de endereços IP.
Utiliza uma base de dados de endereços IP de máquinas (A) no DNS (*Domain Name System*) para mapear nomes de máquinas para endereços IPv4.	Utiliza registos AAAA no DNS para mapear nomes de máquinas para endereços IPv6.
Utiliza um ponteiro (PTR) para os registos do domínio DNS IN-ADDR.ARPA para mapear endereços IPv4 para os respectivos nomes de máquinas.	Utiliza um ponteiro (PTR) para os registos do domínio DNS IP6.INT para mapear endereços IPv6 para os respectivos nomes de máquinas.

Tabela 1.1 – Comparação do Protocolo IPv6 com o protocolo IPv4.

[52]

1.4 Mobilidade

A mobilidade IPv6 permite que um nó IPv6 seja móvel e que ainda assim consiga manter conectividade, ou seja, um terminal móvel pode alterar a sua localização arbitrariamente numa rede IPv6 sem perder conectividade à rede.

Quando um nó IPv6 muda de localização, este poderá mudar a sua ligação. Quando isso acontece, o seu endereço IPv6 tem de ser alterado para que mantenha conectividade. Existem mecanismos que permitem solucionar este tipo de situação, tais como endereços *stateful*[9] e *stateless*[10] autoconfigurados para IPv6, existindo assim a possibilidade de serem alterados automaticamente. No entanto, estes mecanismos infelizmente terminam todas as ligações existentes do terminal móvel que estão a utilizar o endereço IP autoconfigurado na ligação antiga.

A chave da mobilidade IPv6 é, mesmo quando o nó móvel altera a sua posição e consequentemente o seu endereço IP, as ligações já existentes sejam mantidas.

A manutenção da ligação de terminais móveis não é feita modificando protocolos orientados à ligação (ex: TCP), mas sim conseguir lidar com a mudança de endereço IP da *Internet*. A camada de Transporte é completamente alheia de que o endereço IP foi alterado. A ligação é estabelecida com um endereço IP específico atribuído ao terminal móvel e mantém-se estabelecida independentemente do número de vezes que o terminal móvel mude a sua localização e o seu endereço IP.[52]

1.4.1 Funcionamento

Mobile Node (MN) – nó que pode mudar o seu ponto de conectividade de uma ligação para outra continuando a estar acessível através do seu *Home Address*[11].

Correspondent Node (CN) – nó com o qual um nó móvel está a comunicar. Este nó pode ser móvel ou estático. De destacar que este nó não requer necessariamente suporte MIPv6.

Home Agent (HA) – router da ligação de origem de um nó móvel que tem um endereço temporário registado (*care-of-address*). Enquanto o nó móvel está longe ligação de origem, o Home Agent intercepta os pacotes na ligação de origem destinados ao nó móvel em questão, encapsula-os e encaminha-os para o *care-of-address* registado.[3]

O endereço original (*Home Address*) é constituído por um prefixo válido no *link* da sua rede original (*Home network*). É através deste endereço que um nó correspondente (CN) irá comunicar com o nó móvel (MN), independentemente de onde este estiver. Quando o nó móvel muda de rede, ele mantém o *Home Address* e recebe outro endereço, o *care-of address*, constituído por um prefixo válido numa rede estrangeira. Este endereço é conseguido de forma *stateless* ou *stateful*. Desta forma, o nó móvel terá um *Home Address* e um ou mais *care-of address* quando se move entre redes distintas.

Para que seja possível saber onde se encontra o nó móvel, deve ser realizada uma associação entre *Home Address* e care-of address (*binding*). Esta associação do care-of address é feita pelo nó móvel, no *Home Agent* (HA). É realizada através de um *binding*

[9] Sem estado.

[10] Com estado.

[11] Endereço original

registration, onde o nó móvel envia mensagens designadas de *Binding Updates* (BU) para o HA. Este responde com uma mensagem *Binding Acknowledgement* (BA).

Os nós correspondentes (CN) no MIPv6 possuem "inteligência" para optimização de uma rota, ou seja, eles podem armazenar *bindings* entre *Home Address* e *care-of-address* de nós móveis (MN). Sendo assim, um nó móvel pode fornecer informações sobre a sua localização para CNs, através do *correspondent binding procedure*. Neste procedimento, é efectuado um mecanismo de autorização de estabelecimento de *binding*, chamado de *return routability procedure*. A figura a seguir mostra um cenário de mobilidade IPv6 com elementos básicos:

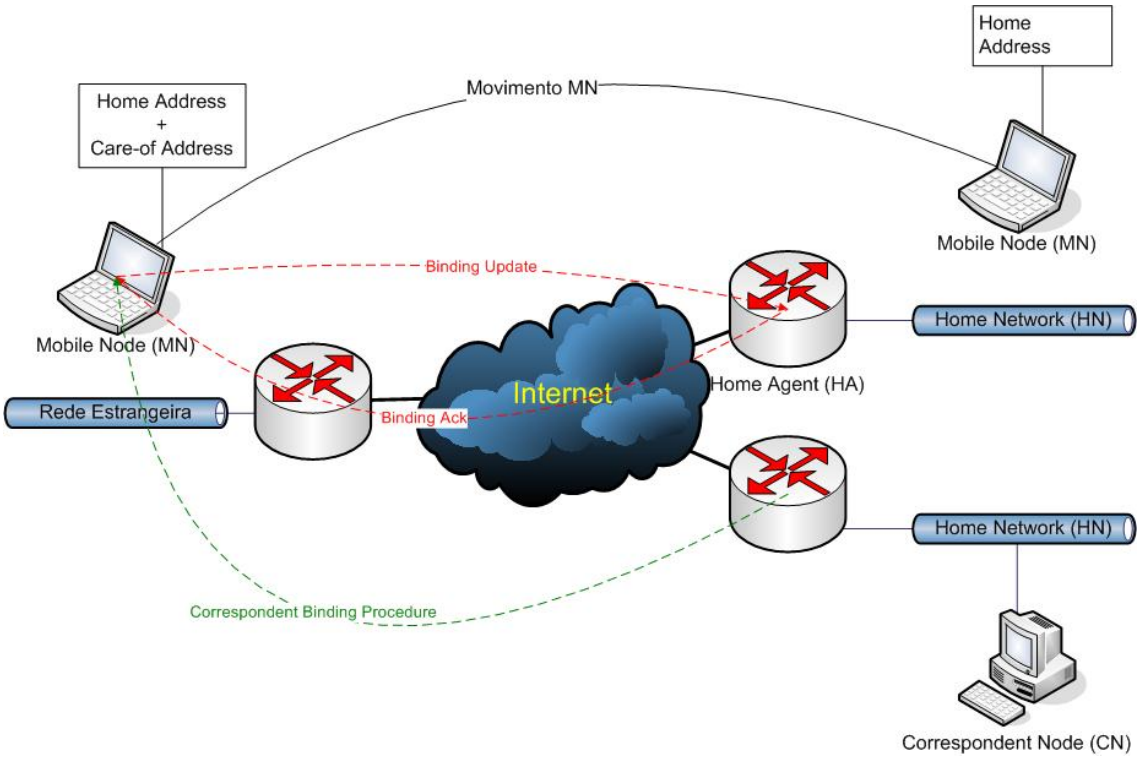


Figura 1.1 – Visão geral do MIPv6.

1.4.2 Segurança em MIPv6

Vários dos problemas existentes no MIPv4 não existem em MIPv6. Mobilidade em IPv6 não oferece problemas em "*ingress filtering*[12]", implementada em *firewalls* e routers. Isto acontece porque um MN usa sempre o seu *care-of-address*, obtido por autoconfiguração *stateful* ou *stateless*, na comunicação com os nós correspondentes. O mecanismo NAT deixa de ser necessário em redes IPv6. Portanto, problemas com NAT são exclusivos da versão IPv4. Confidencialidade e integridade de dados podem ser implementados usando o protocolo IPSec, mandatório no IPv6. Além disto, o protocolo IPv6 básico tenta resolver outros problemas de segurança existentes em IPv4, sendo mais adequado para a actual realidade encontrada na *Internet*.

Em MIPv6, o HA faz uso do protocolo *Neighbor Discovery* (ND), baseado na versão 6 do protocolo ICMP, para fazer o mapeamento entre endereços MAC e endereços IP. Assim, o protocolo IPv6 não necessita do protocolo ARP para esta funcionalidade.

[12] Técnica usada para provar que os pacotes provêm realmente da rede de onde aclamam vir.

Há uma forte tendência de que o protocolo ND elimine todas as ameaças existentes no protocolo ARP. É possível implementar, segundo a sua própria especificação, um esquema de autenticação para mensagens do protocolo, usando IPSec e chaves secretas manualmente configuradas. Em contrapartida, isto vai de encontro a uma das principais vantagens do IPv6 em relação ao IPv4: a existência de mecanismo de autoconfiguração, onde nenhuma intervenção do administrador é necessária para configurar os nós da rede.

Quanto à adopção de esquemas de autenticação, para tornar mais seguro o processo de *Binding Update*, o mecanismo IPSec nativo no IPv6 pode ser usado sem problemas. Para as mensagens de registo no HA, a autenticação pode ser feita usando o protocolo AH do IPSec, desde que seja possível ter um relacionamento prévio de segurança entre o MN e o HA. Isto significa que, por exemplo, uma chave secreta pode ser configurada de antemão por um administrador da rede. Entretanto, o MIPv6 inclui um processo de registo entre o MN e o CN. O CN é um nó que pode estar localizado em qualquer lugar da *Internet*, portanto, é impossível criar qualquer relacionamento de segurança entre estas partes sem algum mecanismo *Global* de autenticação automática. O uso de IPSec é proibitivo dentro deste cenário.

O draft[13] 18 do MIPv6 ([43]) introduz uma solução bastante inteligente para este problema, conhecida como "*Return Routability Procedure*". A ideia por trás do *Return Routability* (RR) é enviar paralelamente mensagens para ambos os endereços de um MN: *Home Address* e *care-of-address*. Desta forma, o CN é capaz de verificar que o MN pode ser alcançado por dois caminhos distintos. Pacotes enviados para o *care-of-address* seguem directamente para o MN no seu novo ponto de conexão. Já os pacotes enviados para o *Home Address*, são encaminhados para a sua rede *Home* e, daí, são transmitidos pelo HA para o MN, através de um túnel. Estas duas mensagens também transmitem "segredos" distintos, por ambos os caminhos. No nó móvel, uma função gera um valor usando estas informações secretas e transmite a saída na mensagem de BU para o nó correspondente. O CN, supondo que apenas o MN é capaz de obter os dois "pedaços" distintos da informação secreta através das mensagens anteriormente enviadas, recalcula o valor e verifica o resultado com a saída gerada no nó móvel. Se os valores forem iguais, o MN é autenticado.

De salientar que o RR não é totalmente seguro. Um potencial ataque, adequadamente localizado, pode capturar ambas as mensagens enviadas pelo CN com as informações secretas, e realizar o mesmo cálculo que o MN. Isto permitiria que mensagens de BU forjadas fossem criadas. Porém, se é possível capturar tais mensagens, também é possível implementar ataques similares contra o protocolo IPv6 sem mobilidade. Logo, o RR e MIPv6 não introduzem riscos adicionais ao protocolo IPv6 básico.[43]

[13] Esquema, rascunho.

2 Estrutura dos endereços IPv6

Os endereços IPv4 são representados num formato "*dotted-decimal*[14]" sendo divididos em grupos de 8 bits formando um conjunto de 32 bits. Cada byte é convertido para o seu equivalente decimal.

No caso dos endereços IPv6, estes são representados por 128 bits sendo divididos em grupos de 16 bits. Cada grupo é representado pelo seu equivalente hexadecimal (4 dígitos). Este formato é designado por "*colon-hexadecimal*".

Ex.:

Divisão em 16 bits:

0010000111011010 0000000011010011 0000000000000000 0010111100111011

0000001010101010 0000000011111111 1111111000101000 1001110001011010

Conversão de cada bloco de 16 bits para hexadecimal separados por ":":

21DA:00D3:0000:2F3B:02AA:00FF:FE28:9C5A

Este endereço IPv6 pode ser simplificado suprimindo os primeiros zeros de um bloco. No entanto, se um ou mais blocos consecutivos forem constituído somente por zeros, estes podem ser totalmente suprimidos colocando "::":

21DA:D3::2F3B:2AA:FF:FE28:9C5A

[52]

2.1 Tipos de endereços

Os endereços IPv6 identificam *interface*s, não nós. O nó é identificado por um endereço *Unicast* atribuído a qualquer uma das suas *interface*s.

2.1.1 *Unicast*

Um endereço *Unicast* é um identificador para uma única *interface*. Um pacote enviado para um endereço *Unicast* é entregue na *interface* identificada por esse endereço.

Existem vários tipos de endereços *Unicast* em IPv6, nomeadamente *Global Unicast* e *Link-local Unicast*. Existem também alguns endereços especiais que são subtipos dos endereços *Global Unicast* tais como endereços IPv4 encapsulados em IPv6. Outros tipos ou subtipos de endereços poderão ser adicionados futuramente.

Os nós IPv6 podem ter um pouco ou um considerável conhecimento da estrutura interna de um endereço IPv6, dependendo da regra que o nó segue (por exemplo computador vs

[14] Formato dos endereços IPv4 (ex: 192.168.20.2), dotted = ponto.

router). No mínimo, um nó pode considerar que os endereços *Unicast* (incluindo o dele próprio) não têm estrutura interna:

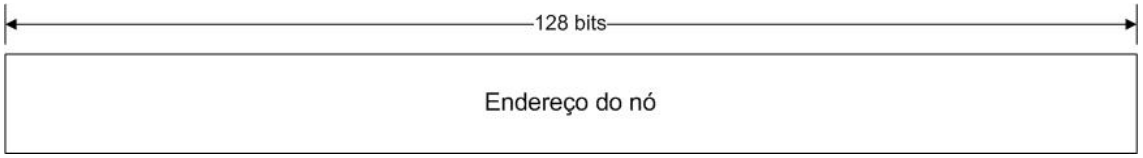


Figura 2.1 – Estrutura simples de um endereço *Unicast* IPv6.

Uma máquina ligeiramente mais sofisticada (mas ainda simples) pode adicionalmente estar ciente dos prefixos de subrede existentes para a ligação à qual está ligada, onde endereços distintos podem ter valores diferentes para a variável "n" representada na figura abaixo:

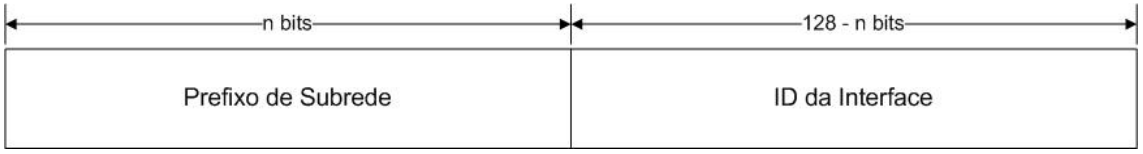


Figura 2.2 – Estrutura elaborada de um endereço *Unicast* IPv6.

[6]

2.1.1.1 Identificador de *interface*s

O identificador de *interface* em endereços IPv6 *Unicast* é utilizado para identificar *interface*s numa ligação. É obrigatório que estes sejam únicos dentro de um prefixo de subrede. O mesmo identificador de *interface* não pode ser atribuído a diferentes nós de uma ligação. Em alguns casos um identificador de *interface* será derivado directamente do endereço dessa mesma *interface*. O mesmo identificador de *interface* poderá ser usado em múltiplas *interface*s de um nó, desde que seja atribuído a redes diferentes.

De salientar que a unicidade de um identificador de *interface* é independente da unicidade dos endereços IPv6. Por exemplo, um endereço *Global Unicast* pode ser criado com significado local, tal como um endereço *Link-local* pode ser criado com significado universal.

Para todos os endereços *Unicast*, excepto os que começam pelo valor binário 000, é obrigatório que os identificadores de *interface* tenham 64 bits.

Os identificadores de *interface* com este formato são formados invertendo o bit 'x' da figura abaixo. Assim sendo, quando o bit 'X' é colocado a um (1) significa que o endereço IP tem significado universal; quando o bit 'X' é colocado a zero (0) significa que o endereço IP tem significado local.

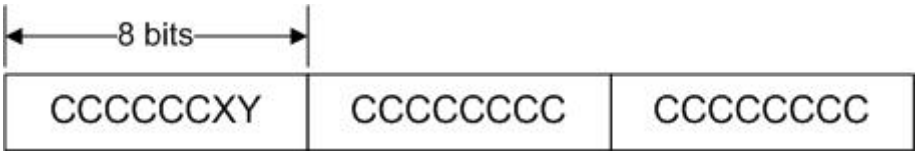


Figura 2.3 – Três primeiros octetos do identificador de *interface* IPv6.

Na figura acima estão representados os primeiros três octetos do identificador de uma *interface* de acordo com o standard da *Internet*. O bit "X" é o bit universal/local, o bit "Y" é o bit individual/local e os bits "C" correspondem aos bits *company_id* (identificadores de uma empresa).

A razão para que o bit 'X' seja invertido aquando da formação de um identificador de uma *interface*, é para facilitar os administradores de sistema de lidar com a configuração de identificadores não globais quando o hardware não os identifica. Esta situação faz sentido se pensarmos em ligações série e nas extremidades de um túnel. Os nós IPv6 não necessitam de validar estes identificadores de *interface*.

O uso do bit universal/local no identificador serve para permitir o desenvolvimento de tecnologia futura que possa tirar partido do identificador de *interface* com significado universal.[6]

2.1.1.2 Endereço não especificado

O endereço 0:0:0:0:0:0:0:0 (::) é conhecido como endereço não especificado. Este endereço nunca pode ser atribuído a um nó. Indica a ausência de um endereço. Um exemplo do seu uso é no campo *Source Address* de um pacote IPv6 enviado por um nó em inicialização antes de ter aprendido o seu endereço.[6]

2.1.1.3 Endereço *Loopback*

O endereço *Unicast* 0:0:0:0:0:0:0:1 (::1) é conhecido como o endereço *Loopback*. Este endereço pode ser usado por um nó para enviar pacotes IPv6 para ele próprio. Este endereço não deve ser atribuído a nenhuma *interface* física. È tratado como se fosse um endereço local de uma *interface* virtual (chamado *interface Loopback*) que tem um *link* imaginário que não vai a parte alguma.

O endereço *Loopback* não deve ser usado como se se tratasse do endereço de origem dos pacotes IPv6 que são enviados para fora do próprio nó. Os pacotes recebidos numa *interface* que tenham como endereço de origem o endereço *Loopback* devem ser descartados.[6]

2.1.1.4 Endereços *Global Unicast*

Formato dos endereços *Global Unicast* IPv6:

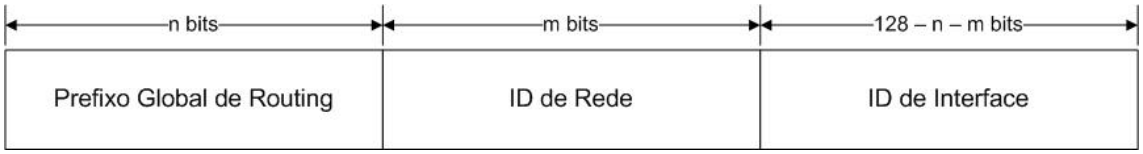


Figura 2.4 – Estrutura de um endereço IPv6 *Global Unicast.*

Legenda:

Prefixo *Global* de *Routing* – valor (normalmente estruturado hierarquicamente) atribuído a um sítio (cluster de subredes/*link*s).

ID de *interface* – definido em 2.1.1.1.

ID de subrede – identificador de uma ligação dentro de um sitio.

Todos os endereços *Global Unicast* que não começam com o valor binário 000 têm 64 bits para o campo *Interface* ID. Por outro lado, os endereços *Global Unicast* que começam com o valor binário 000, não têm este tipo de restrições relativamente ao tamanho e estrutura do campo *Interface* ID.

Um exemplo de endereços *Global Unicast* que começam com o valor binário 000 é um endereço IPv6 com um endereço IPv4 embutido.[6]

2.1.1.5 Endereços IPv6 com endereços IPv4 embutidos

Nestes endereços estão definidos dois tipos de endereços IPv6 que transportam um endereço IPv4 nos 32 bits menos significativos do endereço IPv6.

Estes são designados como:

- **Endereço IPv6 compatível com IPv4 –** definido para assistir a transição IPv6 e tem o seguinte formato:

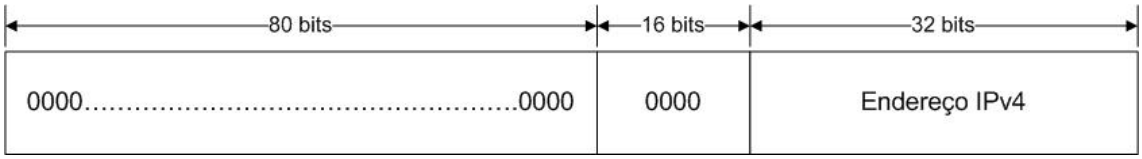


Figura 2.5 – Estrutura de um endereço IPv6 compatível com IPv4.

Nota: o endereço IPv4 tem de ser um endereço *Globally-Unique* IPv4 *Unicast*.

Os endereços IPv4 compatíveis com IPv6 estão **actualmente obsoletos** pois os mecanismos de transição IPv6 já não usam estes endereços.

- **Endereço IPv6 mapeado com IPv4 –** usado para representar os endereços dos nós IPv4 como endereços IPv6 e tem o seguinte formato:

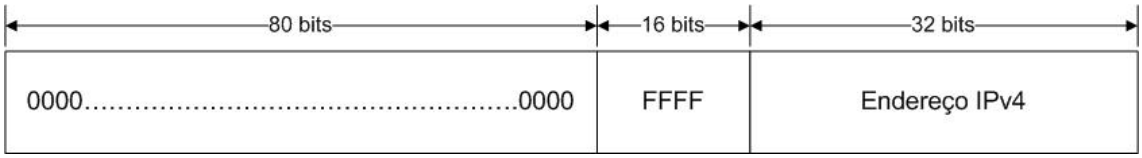


Figura 2.6 – Estrutura de um endereço IPv6 mapeado com IPv4.

[6]

2.1.1.6 Endereços *Link-local*

Utilizado para uma única ligação. Têm o seguinte formato:

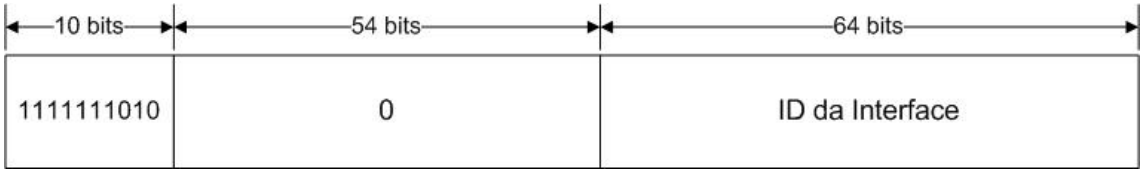


Figura 2.7 – Estrutura de um endereço IPv6 *Link-local.*

Estes endereços foram concebidos para endereçar uma ligação única ou para quando não estão envolvidos routers na ligação, e têm a finalidade de automatizar a sua configuração e/ou descobrir a vizinhança.

Os routers não devem encaminhar para outros *link*s pacotes que tenham como endereço de origem ou destino um endereço *Link-local*.[6]

2.1.1.7 Endereços *Site-local*

Os endereços *Site-local* foram desenhados inicialmente para serem usados para endereçamento dentro de um site, sem haver necessidade de um prefixo *Global*.

Estes endereços têm o seguinte formato:

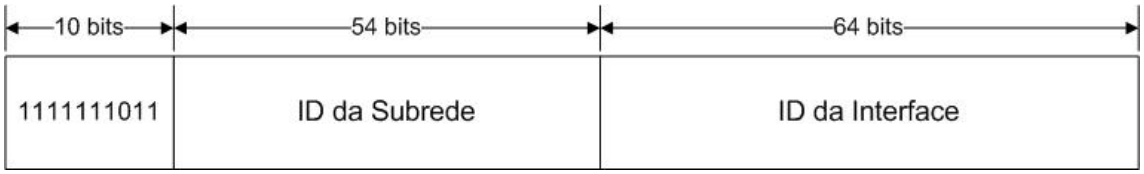


Figura 2.8 – Estrutura de um endereço IPv6 *Site-local.*

Nota: Os endereços *Site-local* estão actualmente obsoletos e não devem ser suportados em novas implementações.[6]

2.1.2 Endereços *Anycast*

Endereços *Anycast* são endereços que são atribuídos a mais que uma *interface* (normalmente pertencentes a nós distintos). Um pacote enviado para um endereço *Anycast* é encaminhado para a *interface* mais "próxima" que tenha esse endereço, de acordo com as métricas dos protocolos de encaminhamento.

Estes endereços são alocados do espaço de endereçamento *Unicast*, utilizando qualquer um dos formatos dos endereços *Unicast*. Assim, os endereços *Anycast* são esteticamente indistinguíveis de um endereço *Unicast*, ou seja, quando um endereço *Unicast* é atribuído a mais do que uma *interface*, é chamado de endereço *Anycast*. Os nós que têm este tipo de endereço atribuído devem ser explicitamente configurados para saberem que esse é um endereço *Anycast*.

Para qualquer endereço *Anycast* atribuído, existe um prefixo P mais longo que identifica a região topológica na qual todas as *interface*s pertencentes a esse endereço *Anycast* residem. Sem essa região identificada por P, o endereço *Anycast* tem de ser mantido como uma entrada separada do sistema de *Routing* (normalmente conhecida como "host route"). Fora da região identificada por P, o endereço *Anycast* pode ser agregado na entrada de encaminhamento para o prefixo P.

De salientar que, no pior caso, o prefixo P de um grupo *Unicast* pode ser nulo, ou seja, os membros do grupo podem não ter qualquer localidade topológica. Neste caso, o endereço *Anycast* tem de ser mantido como uma entrada de encaminhamento separada de toda a *Internet*, o que representa uma subida abrupta na quantidade de grupos "*Global*" *Unicast* que podem ser suportados. No entanto é provável que o suporte para grupos *Global Anycast* deixe de estar disponível ou então fica muito restrito.

Uma utilização provável dos endereços *Anycast* é identificar um grupo de routers pertencentes a uma organização que disponibilizam a *Internet*. Estes endereços podem ser usados como endereços intermediários num cabeçalho IPv6 *Routing*, para que um

pacote seja entregue via um fornecedor de serviços particular ou uma sequência de fornecedores de serviços.

Outras possíveis utilizações são identificar um grupo de routers ligados a uma rede ou um grupo de routers que disponibilizam a entrada a um domínio de *Routing*.[6]

2.1.2.1 Endereços *Anycast Required*

O endereço *Anycast Required* ou *Anycast Subnet-Router* é predefinido e tem o seguinte formato:

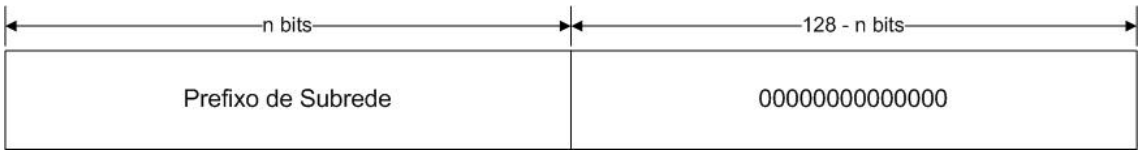


Figura 2.9 – Estrutura de um endereço IPv6 *Anycast Required.*

O "prefixo de subrede" num endereço *Anycast* é o prefixo que identifica um *link* específico. Este endereço *Anycast* é esteticamente o mesmo que um endereço *Unicast* para uma *interface* do *link* com o identificador da *interface* colocado a zero.

Os pacotes enviados para o endereço *Anycast Subnet-Router* serão devolvidos para um router da subrede. Todos os routers são obrigados a suportar o endereço *Anycast Subnet-Router* para as subredes para as quais eles têm *interface*s associadas.

O endereço *Anycast Subnet-Router* existe para que seja usado em aplicações onde um nó precise de comunicar com outro qualquer nó de um grupo de routers.[6]

2.1.3 Endereços *Multicast*

Um endereço IPv6 *Multicast* é um identificador para um grupo de *interface*s (normalmente em nós diferentes). Não existem endereços de *Broadcast* em IPv6. A sua função é substitui-los por endereços de *Multicast*.

Uma *interface* pode pertencer a um qualquer número de grupos *Multicast*. Estes endereços têm o seguinte formato:

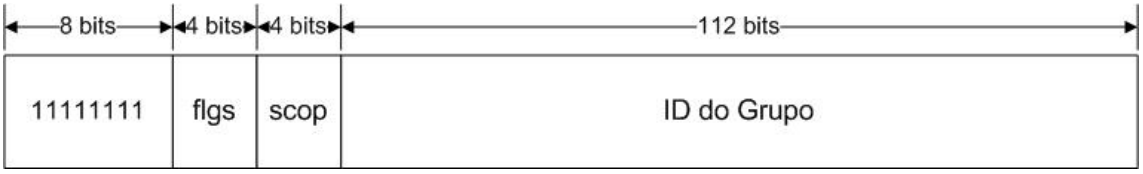


Figura 2.10 – Estrutura de um endereço IPv6 *Multicast.*

O valor binário 11111111 no início do endereço identifica o endereço como sendo um endereço *Multicast*.

A variável flag representa um conjunto de 4 *flags*:

O	R	P	T

A flag inicial está reservada e tem de ser inicializada a 0.

T = 0 – significa que um endereço *Multicast* está permanentemente atribuído ("*well known*"). Este endereço é atribuído pela IANA (*Internet Assigned Numbers Authority*).

T = 1 – significa que um endereço *Multicast* não está atribuído permanentemente ("*transient*" ou "*dynamically*" atribuídos).

Se **P = 0** – significa que o endereço *Multicast* não está atribuído baseado no prefixo de rede.

Se **P = 1** – T tem de estar definido com o valor 1.[29]

O valor da variável *scop*suporta 4 bits e é usada para limitar a abrangência do grupo *Multicast*. Pode assumir os seguintes valores:

0 – reservado

1 – *Interface-Local*

2 – *Link-local*

3 – reservado

4 – *Admin-Local*

5 – *Site-local*

6 – (não atribuído)

7 – (não atribuído)

8 – *Organization-Local*

9 – (não atribuído)

A – (não atribuído)

B – (não atribuído)

C – (não atribuído)

D – (não atribuído)

E – Global

F – reservado

Interface-Local utiliza apenas uma *interface* num nó e é útil apenas para transmissão de *Multicast* para a *Loopback*.

Link-local Multicast utiliza a mesma região topológica como o correspondente *Unicast*.

Admin-Local é o espaço mais pequeno que deve ser administrativamente configurado, ou seja, não é automaticamente derivado da conectividade física ou de qualquer outro modo.

Site-local tem como objectivo abranger um único sítio.

Organization-Local tem como objectivo abranger vários sítios pertencentes a uma única organização.

(não atribuído) está disponível para administradores para definirem regiões *Multicast* adicionais.

O identificador de grupo (*group ID*) identifica o grupo *Multicast*, de um modo permanente ou transitório. Um endereço *Multicast* configurado de modo permanente é independente do valor *scop*. Por exemplo, se for atribuído ao grupo "*NTP Servers Group*" um endereço *Multicast* permanente com um *group ID* 101 (hex), então:

FF01:0:0:0:0:0:0:101 – todos os servidores NTP estão na mesma *interface* (mesmo nó) que o emissor.

FF02:0:0:0:0:0:0:101 – todos os servidores NTP estão no mesmo *link* que o emissor.

FF05:0:0:0:0:0:0:101 – todos os servidores NTP estão no mesmo sítio que o emissor.

FF0E:0:0:0:0:0:0:101 – todos os servidores NTP estão na *Internet*.

Um endereço *Multicast* transitório (não permanente) faz sentido somente quando não se aplica nenhum valor à variável *scop*. Por exemplo, um grupo identificado por um endereço *Site-local Multicast* transitório FF15:0:0:0:0:0:0:101 de um sítio não se relaciona a um grupo que usa o mesmo endereço de um sitio diferente, nem a um grupo transitório que utilize o mesmo *group ID* com diferentes variáveis *scop*, nem a um grupo permanente com o mesmo *group ID.*

Os endereços *Multicast* têm de ser usados como endereços origem em pacotes IPv6 ou aparecer num qualquer cabeçalho de *Routing*.

Os routers não podem encaminhar nenhum pacote *Multicast* para lá do valor *scop* indicado por este no endereço *Multicast* de destino.

Os nós não podem originar um pacote para um endereço *Multicast* cujo campo *scop* contenha o valor reservado 0. Se for recebido um pacote destes, tem de ser descartado. Os nós não devem originar um pacote para um endereço *Multicast* cujo campo *scop* contenha o valor reservado F. Se um pacote destes for enviado ou recebido, deverá ser tratado com se de um pacote com endereço *Global* (*scop* E) *Multicast* de destino se tratasse.

[6]

2.1.3.1 Endereços *Multicast* predefinidos

Os endereços IP seguintes são predefinidos. Os IDs de grupo são definidos explicitamente para espaços de valores.

Não é permitido que a *flag* T seja igual a zero.

Endereços *Multicast* reservados:

FF00:0:0:0:0:0:0:0

FF01:0:0:0:0:0:0:0

FF02:0:0:0:0:0:0:0

FF03:0:0:0:0:0:0:0

FF04:0:0:0:0:0:0:0

FF05:0:0:0:0:0:0:0

FF06:0:0:0:0:0:0:0

FF07:0:0:0:0:0:0:0

FF08:0:0:0:0:0:0:0

FF09:0:0:0:0:0:0:0

FF0A:0:0:0:0:0:0:0

FF0B:0:0:0:0:0:0:0

FF0C:0:0:0:0:0:0:0

FF0D:0:0:0:0:0:0:0

FF0E:0:0:0:0:0:0:0

FF0F:0:0:0:0:0:0:0

Estes endereços *Multicast* estão reservados e não devem nunca ser atribuídos a nenhum grupo *Multicast*:

Endereços de nós:

FF01:0:0:0:0:0:0:1

FF02:0:0:0:0:0:0:1

Estes seguintes endereços *Multicast* identificam o grupo de nós IPv6 com espaço 1 (*Interface-Local*) ou 2 (*link-local*):

Endereços de routers:

FF01:0:0:0:0:0:0:2

FF02:0:0:0:0:0:0:2

FF05:0:0:0:0:0:0:2

Estes endereços identificam o grupo de routers IPv6 com espaço 1 (*Interface-Local*), 2 (*link-local*) ou 5 (*Site-local*).

Endereço *Solicited-Node*: FF02:0:0:0:0:1:FFXX:XXXX

Este endereço *Multicast* é processado em função dos endereços *Unicast* e *Anycast* de um nó. É formado pegando nos 24 bits menos significativos unindo-os ao prefixo FF02:0:0:0:0:1:FF00::/104, resultando num endereço *Multicast* que vai de FF02:0:0:0:0:1:FF00:0000 até FF02:0:0:0:0:1:FFFF:FFFF.

[6]

Exemplo:

O endereço *Multicast Solicited-Node* correspondente ao endereço IPv6 4037::01:800:200E:8C6C é FF02::1:FF0E:8C6C. Endereços IPv6 que diferem somente nos bits mais significativos serão mapeados para o mesmo endereço *Solicited-Node* reduzindo, no entanto, o número de endereços *Multicast* que um nó pode juntar.

Um nó tem de processar e juntar-se ao endereço *Multicast Solicited-Node* associado para todos os endereços *Unicast* e *Anycast* que tenham sido configurados para *interface*s de nós (manualmente ou automaticamente).[6]

2.2 Endereços *Required-Node*

Uma máquina é obrigada a reconhecer os seguintes endereços para se identificar ele próprio:

- Um endereço *Link-local* para cada *interface*;
- Quaisquer endereços *Anycast* ou *Unicast* adicionais que tenham sido configurados para as *interface*s dos nós (manualmente ou automaticamente);
- O endereço *Loopback*;
- Os endereços *Multicast All-nodes* definidos em 2.1.3.1.
- O endereço *Multicast Solicited-Node* para cada endereço *Unicast* ou *Anycast*.
- Endereços *Multicast* para todos os outros grupos aos quais o nó pertence.

[6]

Um router é obrigado a reconhecer todos os endereços que uma máquina é obrigada a reconhecer, mais os seguintes endereços ao identificar-se ele próprio:

- Os endereços *Anycast Subnet-Router* para todas as *interface*s que estão configuradas para funcionarem como um router;
- Todos os outros endereços *Anycast* para os quais o router foi configurado;
- Os endereços *Multicast* All-Routers definidos em 2.1.3.1.

[6]

2.3 Supressão de zeros

Alguns endereços IPv6 contêm longas sequências de zeros. Para simplificar a representação destes, essa sequência poderá ser substituída por "::", mais conhecida por "*double colon*".

Por exemplo, o endereço FE80:0000:0000:0000:2AA:FF:FE9A:4CA2 pode ser suprimido ficando com a seguinte forma: FE80::2AA:FF:FE9A:4CA2.

No entanto não é possível suprimir os zeros que estão a direita dos restantes caracteres. Por exemplo, dado o endereço FF02:30:0:0:0:0:0:5, não é possível simplifica-lo desta forma: FF02:3::5 mas sim FF02:30::5.

De salientar também que um endereço IPv6 não poderá conter duas vezes "::", ou seja, dado o endereço 2001:0DB8:0000:0000:123:4567:0000:CDEF, simplificando temos 2001:0DB8::123:4567:0:CDEF ou 2001:0DB8:0:0:123:4567::CDEF.

Para determinar quantos bits a zero podem ser representados por ::, conta-se o número de blocos do endereço comprimido, subtrai-se por 8 e multiplica-se por 16.

Ex.:

Endereço:

FF02::2

Conta:

(8-2) x 16 = 96 logo foram suprimidos 96 bits.

2.4 Prefixos

O prefixo é uma parte do endereço IP onde os bits assumem valores fixos ou podem ser bits de uma rota ou do identificador de uma rede. Estes 2 últimos prefixos são expressos como na notação IPv4 CIDR (*Classless Inter-Domain Routing*).

Um prefixo IPv6 é escrito no seguinte modo: endereço IPv6/tamanho do prefixo, onde endereço IPv6 é o endereço em questão e tamanho do prefixo é um valor decimal que especifica o número de bits contínuos a contar da esquerda do endereço (bits mais significativos).

Ex.:

Representações correctas:

2001:0DB8:0000:CD30:0000:0000:0000:0000/60

2001:0DB8::CD30:0:0:0:0/60

2001:0DB8:0:CD30::/60

Representações incorrectas:

2001:0DB8:0:CD3/60 – falta um zero no último bloco hexadecimal

2001:0DB8::CD30/60 – expandido fica 2001:0DB8:0000:0000:0000:0000:0000:CD30

2001:0DB8::CD3/60 – expandido fica 001:0DB8:0000:0000:0000:0000:0000:0CD3

Ao expandir as representações incorrectas, estas ficam todos diferentes do endereço IP original.

Tal como já foi referido, um prefixo de 64 bits é usado para redes individuais. Todas as subredes têm um prefixo de 64 bits. Qualquer outro prefixo inferior a 64 bits é uma rota ou uma gama de endereços que sumariza uma porção do espaço de endereçamento IPv6.

As implementações IPv4 utilizam uma notação "*dotted decimal*" do prefixo da rede a que estão associados conhecida como máscara de rede. Tal não existe em IPv6. Somente a notação com o tamanho do prefixo é suportada. Um prefixo IPv6 só é relevante para rotas ou gamas de endereços, não para endereços *Unicast* individuais.

Em IPv6 não existe a noção de identificador de subrede variável.

Numa subrede IPv6 de nível individual para os endereços IPv6 *Unicast* definidos correntemente, o número de bits usados para identificar a subrede é sempre 64 e o número de bits usados para identificar para identificar a máquina na subrede é sempre 64. No entanto, é permitido criar endereços IPv6 com tamanhos de prefixos à escolha [30] mas na prática o tamanho do seu prefixo é sempre 64 não sendo assim necessário

serem expressos. Por exemplo, não é necessário exprimir o endereço IPv6 *Unicast* FEC0::2AC4:2AA:FF:FE9A:82D4 em FEC0::2AC4:2AA:FF:FE9A:82D4/64.[52]

3 Estrutura dos Pacotes IPv6

Um pacote IPv6 é composto por um cabeçalho um, ou mais cabeçalhos de extensão e pelos dados a transportar (*upper-layer Protocol unit)*.

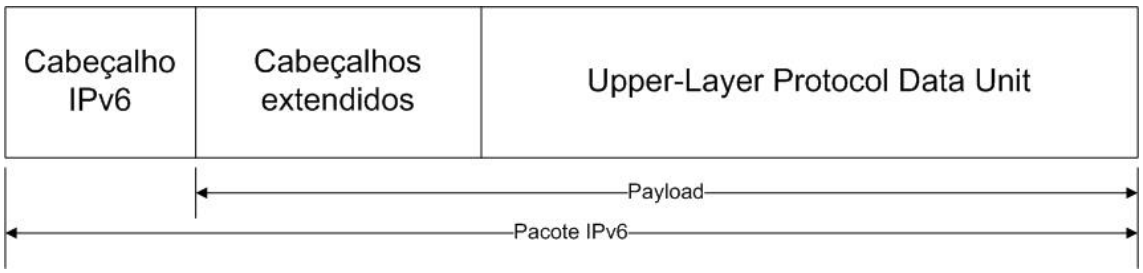


Figura 3.1 – Formato de um pacote IPv6.

3.1 Cabeçalho IPv6

O cabeçalho IPv6 está sempre presente e tem um valor fixo de 40 bytes. Consiste numa versão melhorada do cabeçalho IPv4 eliminando os campos que não são utilizados ou que são raramente utilizados. Adiciona um campo que oferece melhor suporte para tráfego em tempo real.

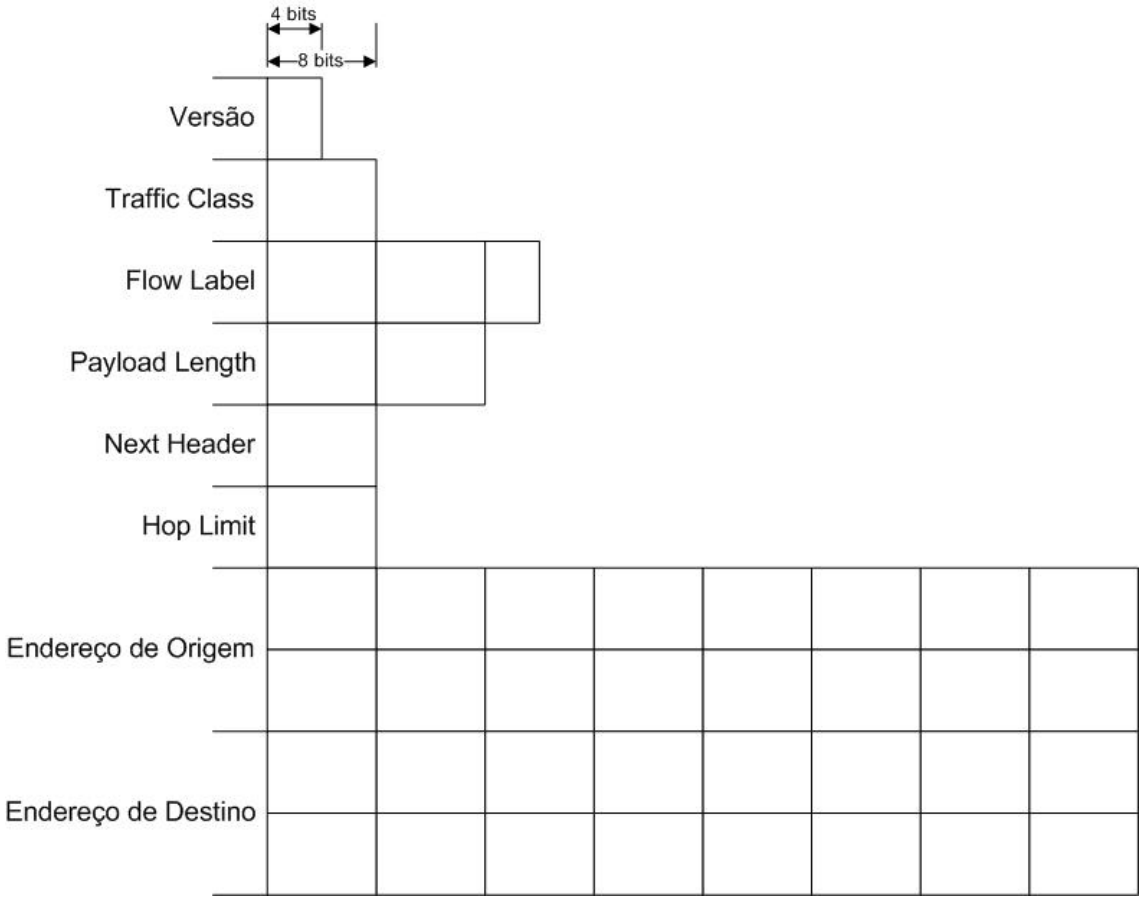


Figura 3.2 – Estrutura de um cabeçalho de um pacote IPv6.

3.1.1 Campos

- **Versão**

Indica a versão do IP e tem o valor 6. Tem 4 bits de tamanho. Este valor não é utilizado para que um pacote passe por qualquer camada do protocolo IPv4 ou IPv6. Esta identificação é feita recorrendo a um campo de identificação protocolar no cabeçalho da camada de ligação.

Por exemplo, um encapsulamento comum na camada de ligação na tecnologia *Ethernet* chamada *Ethernet II*, usa um campo de 16 bits para identificar o *payload* da *frame Ethernet*. Este campo, designado por *Ethernet II*, tem o valor 0x86DD quando se trata de um pacote IPv6. No entanto este campo é analisado antes do pacote passar pela camada protocolar apropriada.

- ***Traffic Class***

Indica a classe ou a prioridade de um pacote IPv6. Tem 8 bits de tamanho. Tem uma funcionalidade similar ao campo ToS (*Type of Service*) do protocolo IPv4. Os valores que este campo pode tomar ainda não estão definidos. É necessário que seja definido também um valor para fins experimentais.

- ***Flow Label***

Indica se um pacote pertence a uma sequência específica de pacotes entre a origem e o destino, requerendo tratamento especial por parte dos routers IPv6 intermediários. Este campo tem 20 bits de tamanho e é usado para quando não existe uma conexão padrão com QoS (*Quality of Service*) tais como as que requerem transmissão em tempo real (voz e vídeo). Por defeito, este valor assume o valor 0.

Podem existir múltiplos fluxos de dados entre a origem e o destino distintos apenas pelo valor do seu *Flow Label*. Tal como no campo *Traffic Class*, ainda não estão definidos valores exactos que este campo pode tomar.

- ***Payload Length***

Indica a quantidade de "carga" de um pacote IPv6. Tem 16 bits de tamanho.

Este campo inclui os cabeçalhos de estensão e a upper-layer PDU. Visto ter 16 bits de tamanho, um pacote IPv6 pode ter mais de 65535 bytes. Para valores superiores, é atribuído o valor 0 e a opção *Jumbo Payload* é usada nas opções *Hop-by-Hop* do cabeçalho de estensão.

- ***Next Header***

Indica quer o tipo do primeiro cabeçalho de estensão (se existir), quer o protocolo na camada acima PDU (tais como TCP, UDP, ICMPv6). Este campo tem um tamanho de 8 bits. No caso de ser indicado o protocolo da camada acima PDU, este campo usa os mesmos valores usados no campo do protocolo IPv4. Pode assumir os seguintes valores:

Valor	Descrição
0	Num cabeçalho IPv4: reservado e não usado Num cabeçalho IPv6: Seguidamente vem um cabeçalho *Hop-by-Hop* Option
1	*Internet Control Message Protocol* ICMPv4
2	*Internet Group Management Protocol* IGMPv4
4	IP dentro de IP (encapsulamento)
6	TCP
8	*Exterior Gateway Protocol* (EGP)
9	IGP – qualquer *Gateway* interior privado (usado pela Cisco para IGRP)
17	UDP
41	IPv6

43	Cabeçalho de *Routing*
44	Cabeçalho de *Fragment*ação
45	*Interdomain Routing Protocol* (IDRP)
46	*Resource ReSerVation Protocol* (RSVP)
50	Cabeçalho *Encrypted Security Payload*
51	Cabeçalho *Authentication*
58	ICMPv6
59	Não existe *Next Header* para IPv6
60	Cabeçalho *Destination Options*
88	EIGRP
89	OSPF
108	IP *Payload* Compression *Protocol*
115	*Layer 2 Tunneling Protocol* (L2TP)
132	*Stream Control Transmission Protocol* (SCTP)
134 – 254	Sem atribuição
255	Reservado

Tabela 3.1 – Descrição de cada valor que o campo *Next Header* pode assumir.

- ***Hop Limit***

Indica o número máximo de *links* sobre os quais um pacote IPv6 pode viajar antes de ser descartado. Tem um tamanho de 8 bits.

Este campo é similar ao campo TTL do protocolo IPv4 excepto o facto de não existir o histórico do tempo (em segundos) que o pacote foi pedido ao router.

Quando este campo assume o valor 0 no router, este manda uma mensagem ICMPv6 *Time Exceeded-Hop Limit Exceeded in transit* para a fonte e descarta o pacote.

- ***Source Address***

Indica o endereço IPv6 da máquina de origem. Tem 128 bits de tamanho.

- ***Destination Address***

  Indica o endereço IPv6 da máquina de destino. Tem 128 bits de tamanho. Na maioria dos casos, este campo é atribuído como sendo um endereço de destino final. No entanto, se estiver presente um cabeçalho *Routing extension*, este campo poderá ser definido para o endereço do próximo destino intermediário.[52]

3.2 Cabeçalhos de Estensão

Os cabeçalhos dos pacotes IPv4 incluem todas as opções. Deste modo todos os routers intermediários têm de verificar a sua existência e processá-los. Isto pode provocar degradação no encaminhamento de pacotes IPv4.

Com IPv6, as opções de entrega e encaminhamento de pacotes encontram-se nos cabeçalhos de estenção. O único cabeçalho de estensão que tem de ser processado em cada router intermediário é o cabeçalho de estensão *Hop-by-Hop Options*. Assim, a velocidade de processamento dos cabeçalhos IPv6 é melhorada, bem como a performance do encaminhamento de pacotes IPv6.

Os seguintes cabeçalhos de estensão IPv6 têm de ser suportados por todos os nós IPv6 e têm a seguinte ordem:

1. Cabeçalho *Hop-by-Hop Options*
2. Cabeçalho *Destination Options* (para destinos intermediários quando o cabeçalho de *Routing* está presente)
3. Cabeçalho de *Routing*
4. Cabeçalho *Fragment*
5. Cabeçalho de autenticação
6. Cabeçalho *Encapsulating Security Payload*
7. Cabeçalho *Destination Options* (para o destino final)

Num pacote IPv6 típico não existem cabeçalhos de estensão. Se for necessário algum tipo de tratamento especial ao pacote IPv6 por parte dos routers intermediários ou da máquina de destino, a máquina de origem adiciona ao pacote um ou mais cabeçalhos de estensão.

Cada cabeçalho de estensão tem de ter um máximo de 64 bits (8 bytes). Os cabeçalhos de estensão de tamanho fixo têm de ser múltiplos de 8 bytes. Por outro lado os cabeçalhos de estensão de tamanho variável contêm um campo *Header Extension Length* e têm de usar bits adicionais para que o seu tamanho seja múltiplo de 8 bytes.

O campo *Next Header* de um cabeçalho IPv6 e opcionalmente também com cabeçalhos de estensão formam um conjunto de ponteiros. Cada ponteiro indica o tipo de cabeçalho que aparece depois do cabeçalho até que o protocolo da camada acima seja identificado.

Este conjunto de ponteiros é representado na figura seguinte:

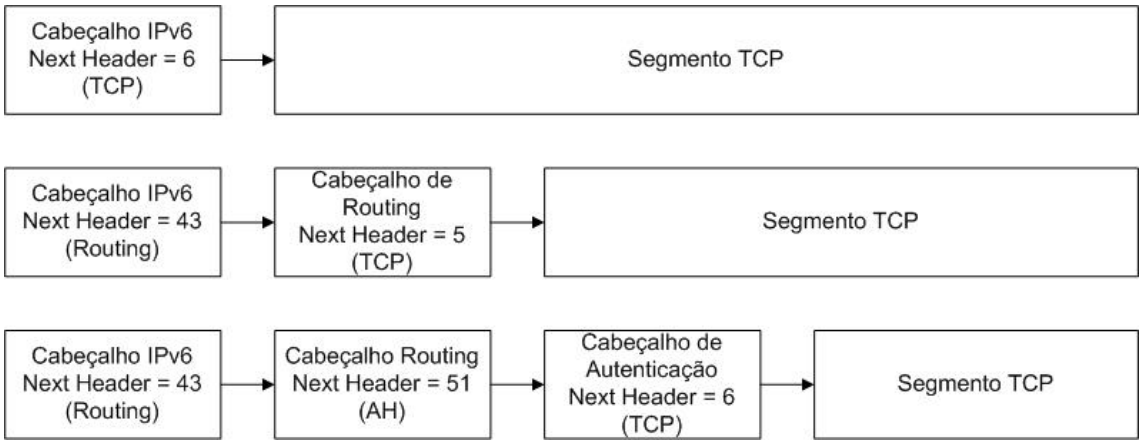


Figura 3.3 – Esquema dos possíveis cabeçalhos de estenção de um pacote IPv6.

3.2.1 Cabeçalho *Hop-by-Hop Options*

Cabeçalho utilizado especificamente para entrega de parâmetros a cada "salto" da viagem até ao destino. É identificado com o valor 0 no campo do cabeçalho IPv6 *Next Header*.

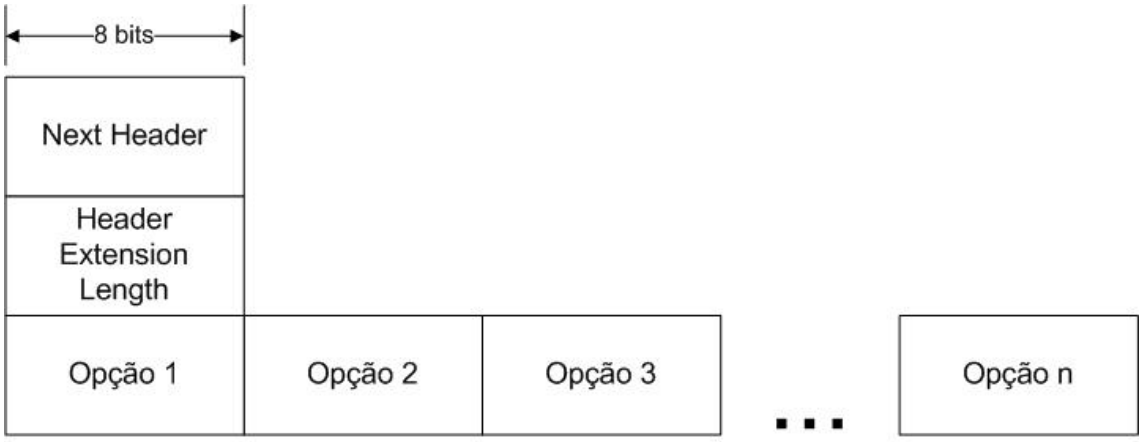


Figura 3.4 – Formato de um cabeçalho *Hop-by-Hop* de um pacote IPv6.

Este cabeçalho reúne os campos *Next Header, Header Extension Length* e *Options* que contém uma ou mais opções. O valor campo *Header Extension Length* é composto pelo número de blocos de 8 bytes contido no campo *Options Extension Header*, não incluindo os primeiros 8 bytes. No entanto, para um *Hop-by-Hop Options Header* de 8 bytes, o valor do campo *Options Extension Header* é 0.

As restantes opções são usadas para garantir limites de 8 bytes.

O campo *Header Extension Length* é um dos exemplos de como foram feitos os possíveis para optimizar o processamento de pacotes IPv6 nos routers intermediários. Uma das primeiras operações a ser feita aos pacotes IPv6 é determinar o tamanho do seu cabeçalho. Para garantir robustez na implementação de IPv6, os campos cujos valores validos comecem pelo valor 1 têm de ser verificados para que não permitam que

o valor 0 lhes seja atribuído. Esta verificação tem de ser feita antes de qualquer processamento adicional seja feito.

Correntemente, o valor 0 é um valor válido e não é preciso efectuar qualquer verificação de valores inválidos pois o número de bytes no *Hop-by-Hop Options Header* é calculado de acordo com a seguinte fórmula: (tamanho do *Header Extension* + 1) x 8, ou seja, é sempre adicionado o valor 1 para que nunca ocorra o caso deste campo assumir o valor 0.

Uma opção é um grupo de campos que descreve características específicas de entrega de pacotes. As opções são enviadas no cabeçalho *Hop-by-Hop Options* e no cabeçalho *Destination Options*. Cada opção está codificada no formato TLV (*Type-Length-Value*) que é normalmente utilizado nos protocolos TCP/IP

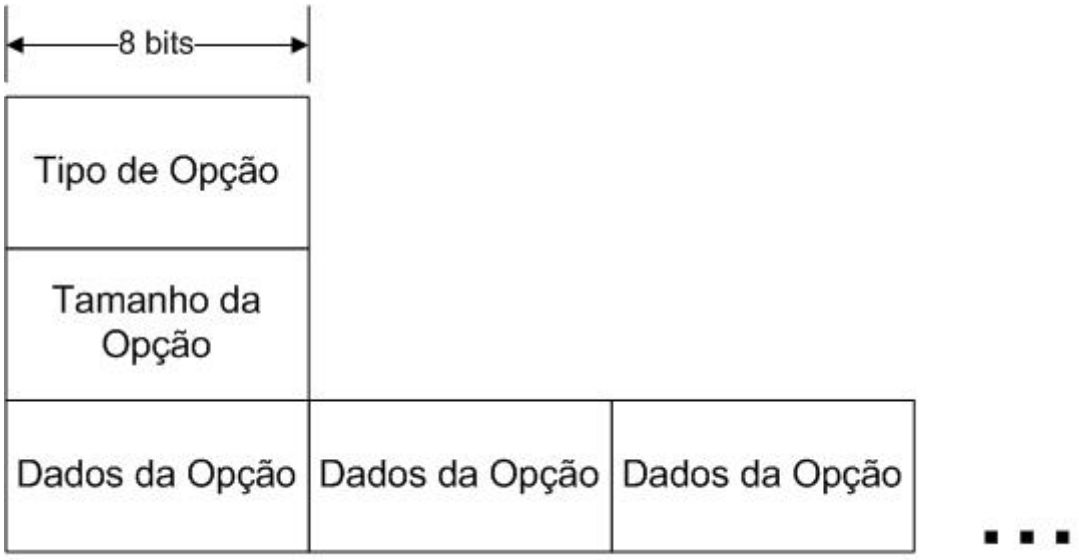


Figura 3.5 – Formato geral de uma opção IPv6.

O campo *Option Type* identifica a opção e determina a forma como ela vai ser tratada pelo nó.

O campo *Option Length* indica o número de bytes da opção, não incluindo os campos *Option Type* e *Option Length*.

O campo *Option Data* identifica os dados específicos associados à opção.

Uma opção pode ter necessidade de preparação para garantir que os campos específicos que se encontram dentro da opção se encontrem nos desejados limites. Por exemplo, é mais fácil processar um endereço IPv6 se este estiver limitado a 8 bytes.

A necessidade de preparação é expressa utilizando a notação xn + y, indicando que essa opção tem de começar nos limites de um byte igual ao inteiro múltiplo de x bytes mais y bytes desde o inicio do cabeçalho. Por outras palavras, a opção tem de começar na sequência 6, 10, 14…, relativamente ao início dos cabeçalhos *Hop-by-Hop Option* ou *Destination Options*. Para facilitar a preparação requerida, o encapsulamento ocorre antes de uma opção e é feita entre cada opção quando existem múltiplas opções definidas.

Campo *Option Type*

Dentro deste campo os dois bits mais significativos indicam a forma como a opção irá ser tratada quando o nó que vai processar a opção não reconheça o tipo de opção.

Valor (binário)	Acção tomada
00	Ignora a opção
01	Descarta o pacote silenciosamente
10	Descarta o pacote e envia uma mensagem ICMPv6 *ParameterProblem* para quem enviou o pacote se o campo *Destination Address* do cabeçalho IPv6 fôr um endereço *Unicast* ou *Multicast*.
11	Descarta o pacote e envia uma mensagem ICMPv6 *ParameterProblem* para quem enviou o pacote se o campo *Destination Address* do cabeçalho IPv6 não for um endereço *Multicast*.

Tabela 3.2 – Acções tomadas sobre um pacote consoante os dois bits mais significativos do campo *Option Type*.

O terceiro bit mais significativo do campo *Option Type* indica se a informação contida no campo pode ser modificada (=1) ou se não pode ser modificada (=0) no caminho até ao destino.

Opção *Pad1*

Esta opção é usada para inserir um único byte de preenchimento para que os cabeçalhos *Hop-by-Hop Options* e *Destination Options* fiquem limitados nos 8 bytes e para acomodar a preparação necessária das opções. A opção Pad1 não necessita de qualquer preparação.

Figura 3.6 – Valor do tipo de opção *Pad1* de um pacote IPv6.

Opção *PadN*

Utilizado para inserir dois ou mais bytes de preenchimento para que os cabeçalhos *Hop-by-Hop Options* e *Destination Options* fiquem limitados nos 8 bytes e para acomodar a preparação necessária das opções. A opção *PadN* não necessita de qualquer preparação.

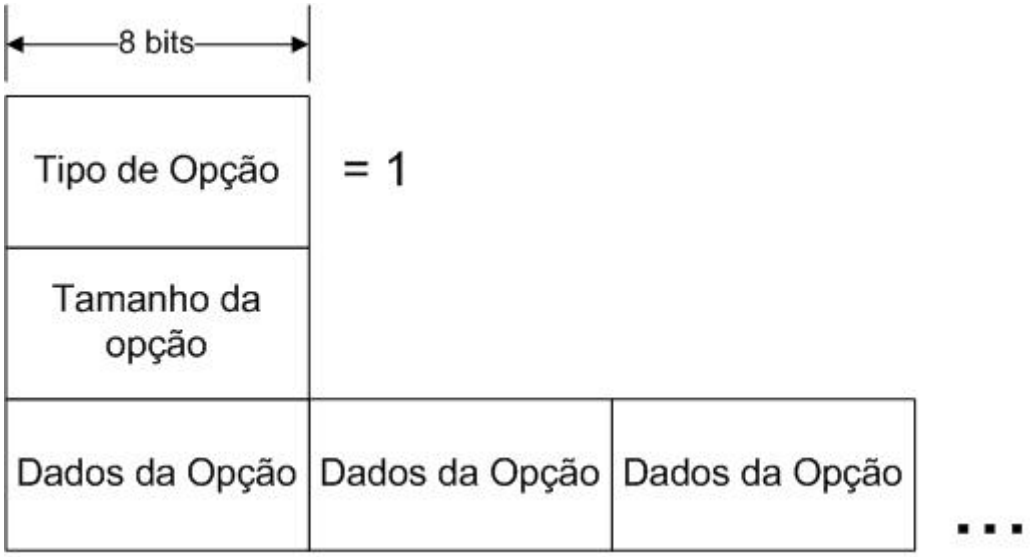


Figura 3.7 – Valor do tipo de opção *PadN* de um pacote IPv6.

Esta opção reúne um conjunto de parâmetros tais como o campo *Option Type* (definido a 1), o campo *Option Length* (definido com o número de bytes de preenchimento presentes) e 0 ou mais bytes de preenchimento. Com o campo *Option Type* definido a 1, a opção é ignorada se não for reconhecida e não pode ser alterada durante o percurso.

Opção *Jumbo Payload*

Esta opção é usada para identificar quando a carga do pacote é superior a 65535 bytes. A opção *Jumbo Payload* tem necessidade de um alinhamento do tipo 4n + 2.

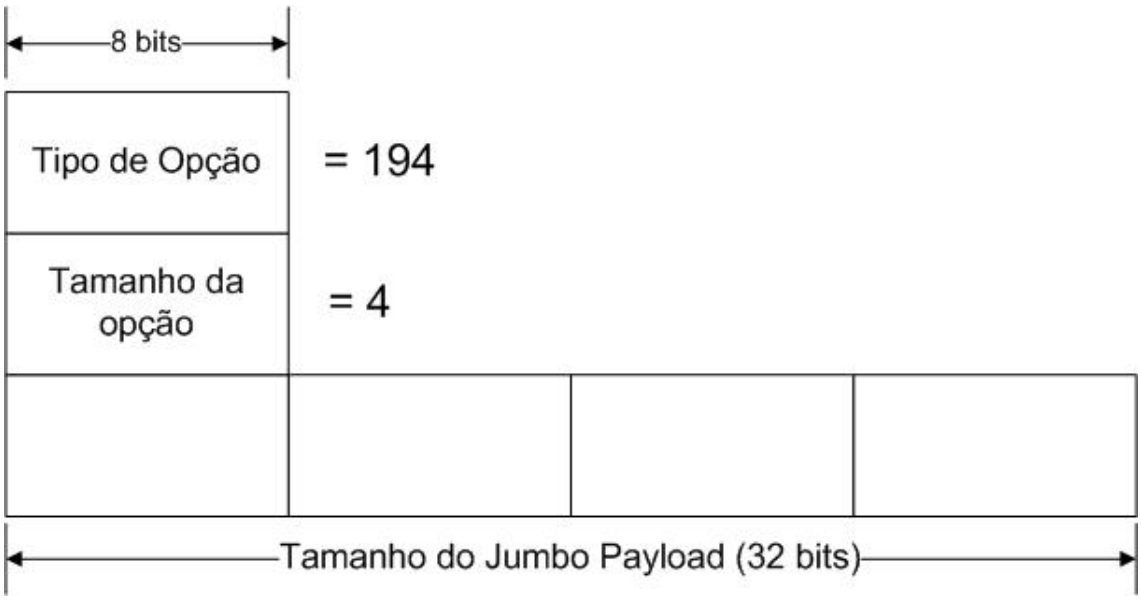


Figura 3.8 – Formato da Opção *Jumbo Payload* de um pacote IPv6.

Com esta opção, o tamanho do campo *Payload Length* no cabeçalho IPv6 deixa de indicar o tamanho da carga de um pacote IPv6. Em vez disso, o campo *Jumbo Payload Length* da opção *Jumbo Payload* indica o tamanho em bytes da carga de um pacote IPv6. Com um campo *Jumbo Payload Length* de 32 bits, o tamanho da carga pode atingir 4294967295 bytes. Um pacote IPv6 com uma carga superior a 65535 bytes é conhecido como *Jumbogram*. Com o campo *Option Type* definido a 194, o pacote é descartado e uma mensagem ICMPv6 *Parameter Problem* é enviada se a opção não for reconhecida e o endereço de destino não for um endereço *Multicast* e se a opção não puder ser alterada durante o seu trajecto.

Nota: As implementações *Microsoft* de IPv6 não suportam o uso de *Jumboprograms*.

Opção *Router Alert*

Esta opção é usada para indicar um router que os dados de um pacote requerem processamento adicional. A opção *Router Alert* requer um alinhamento de 2n + 0.

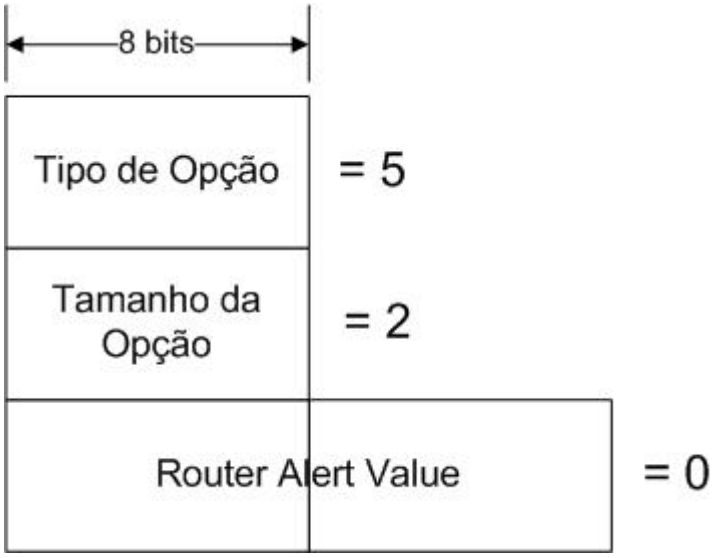


Figura 3.9 – Formato da opção *Router Alert* de um pacote IPv6.

Esta opção é também utilizada para *Multicast Listener Discovery* (MLD) e para *Resource ReSerVation Protocol* (RSVP). Com o campo *Option Type* definida com o valor 5, a opção é ignorada se não for reconhecida e não é permitida a sua alteração durante o seu percurso.

3.2.2 Cabeçalho *Destination Options*

Opções usadas para especificar parâmetros de entrega de pacotes quer para destinos intermediários quer para destinos finais. Este cabeçalho é identificado pelo valor 60 no cabeçalho anterior *Next Header*. Estas opções têm a mesma estrutura que as opções do cabeçalho *Hop-by-Hop*.

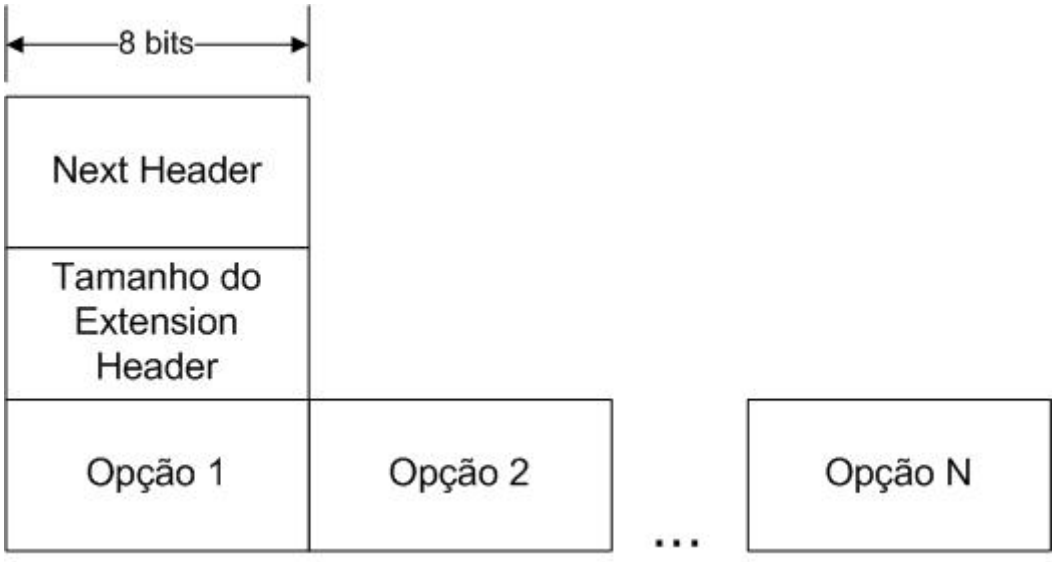


Figura 3.10 – Formato do cabeçalho *Destination Options* de um pacote IPv6.

As opções do *Destination Header* são usadas em 2 casos:

1 – Se um cabeçalho de *Routing* estiver presente especifica opções de entrega e processamento em cada destino intermediário. Neste caso, o cabeçalho *Destination Options* ocorre antes do cabeçalho de *Routing*.

2 – Se o cabeçalho de *Routing* não estiver presente ou se este cabeçalho ocorrer depois do cabeçalho de *Routing*, este cabeçalho especifica opções de entrega e processamento no destino final.

Opção *Binding Update*

Opção utilizada por um nó móvel para actualizar outro nó com o seu novo *care-of-address*.

Esta opção pode ser incluída num pacote já existente enviado para o seu destino ou num pacote que contenha somente o cabeçalho *Destination Options*. Neste último caso, ao campo *Next Header* do cabeçalho *Destination Options* é atribuído o valor 59 indicando que não existe mais nenhum cabeçalho seguinte (*Next Header*).

A opção *Binding Update* requer um alinhamento 4n + 2.

A estrutura desta opção é ilustrada no esquema abaixo:

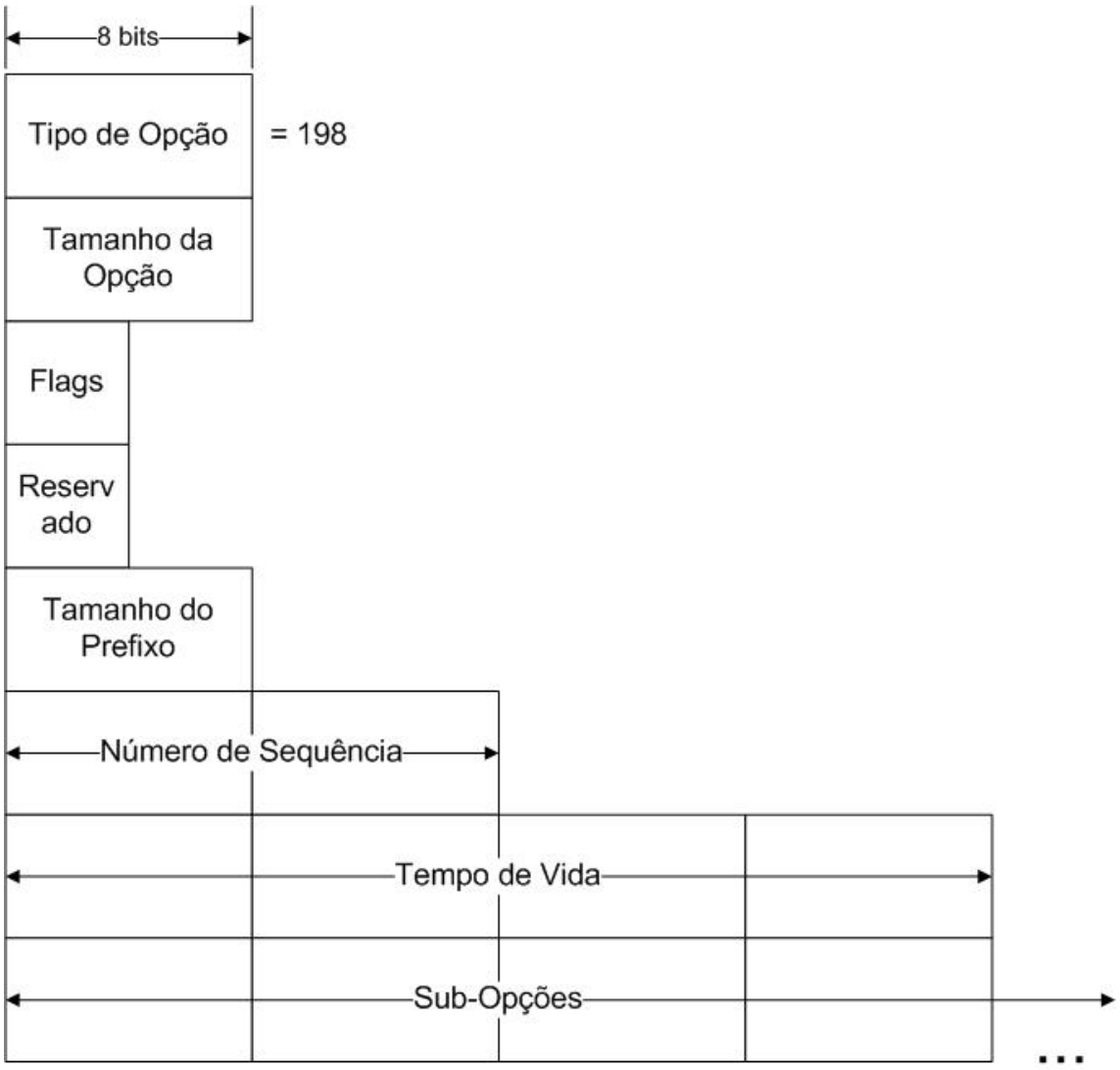


Figura 3.11 – Formato da opção *Binding Update* de um pacote IPv6.

- **Tipo de Opção**

Com o valor 198 atribuído a este campo, o pacote é descartado e uma mensagem ICMPv6 *Parameter Problem* é enviada se a opção não for reconhecida e se o endereço de destino não for um endereço *Multicast*. Esta opção não poderá ser modificada durante o seu percurso.

- **Tamanho da Opção**

Indica o tamanho da opção em bytes não incluindo os campos Tipo de Opção e Tamanho de Opção. Esta opção inclui as sub-opções (se presente).

- ***Flags***

Existem 4 *flags* de 1 bit que podem ser definidas neste campo, o que significa que este campo tem um tamanho de 4 bits. Estas *flags* têm o seguinte significado:

Bit mais significativo a 1:

***Acknowledge* (A) –** definido para indicar que o emissor está a pedir uma confirmação de ligação.

Segundo bit mais significativo a 1:

***Home Registration* (H) –** definido para indicar que o emissor está a pedir ao receptor para que este seja o home agent do nó móvel.

Segundo bit mais significativo a 1:

Router (R) – definido para indicar que o nó móvel é um router. Esta *flag* só pode ser definida se a *flag Home Registration* (H) também estiver definida.

Bit menos significativo a 1:

***Duplicate Address Detection* (D) –** definido para indicar que o emissor está a pedir ao receptor para verificar se existem endereços duplicados para o *Home Address* do nó móvel. Esta *flag* só pode ser activa se as *flags Acknowledge* (A) e *Home Registration* (H) estiverem activas.

- **Reservado**

Este campo contém 4 bits reservados que estão definidos com o valor zero.

- **Tamanho do Prefixo**

Este campo indica o tamanho do prefixo de subrede no campo *Home Address*. Os bits menos significativos do *Home Address* são os identificadores de *interface* do *Home Address* do nó móvel. O *Home Agent* usa este identificador de *interface* para determinar todos os tipos de endereços usados no *Home link* (*link-local*, *Site-local* e *Global*) e pode também verificar se existem endereços duplicados se tal for indicado.

Este campo é definido apenas se a *flag Home Registration* (H) estiver definida. Tem um tamanho de 8 bits.

- **Número de Sequência**

Campo que indica a sequência actualização da ligação e é usado pelo nó móvel para adaptar a actualização da ligação com a correspondente localização da ligação. O tamanho deste campo é de 16 bits

- **Tempo de Vida**

Campo que indica o número de segundos cuja ligação é considerada válida. O valor 0xFFFFFFFF indica tempo infinito. Por outro lado o valor 0 indica que a ligação é inválida e tem de ser eliminada. O tamanho deste campo é de 32 bits.

- **Sub-Opções**

Campo que pode incluir informação adicional e permite que a actualização da ligação seja estendida futuramente. Este campo utiliza o formato TLV, tal como o campo Opções. Correntemente, as Sub-Opções definidas para o *Binding Update* são:

Sub-Opção *Unique Identifier* que é composto por um campo *Type* de 8 bits (definido com o valor decimal 2), um campo *Length* também de 8 bits (definido também com o valor decimal 2) e um campo *Unique Identifier* de 16 bits. Esta última opção é usada para fazer a correspondência entre a actualização de uma ligação e um pedido de ligação.

Sub-Opção *Alternate Care-of-address* que é composto por um campo *Type* de 8 bits (definido com o valor decimal 4), um campo *Length* também de 8 bits (definido com o valor decimal 16) e um campo de 128 bits *Alternate Care-of-address*. Este campo é incluído quando o nó móvel quer indicar que o *Care-of-address* é diferente do endereço de origem do pacote que contém a actualização da ligação.

Opção *Binding Acknowledgement*

Opção usada para conhecer os parâmetros de uma actualização de ligação quando a *flag Acknowledge* (A) está definida. Tal como a opção *Binding Update*, a opção *Binding Acknowledgement* pode ser incluída num pacote existente que foi enviado para um nó móvel, ou num pacote que contém somente o cabeçalho *Destination Options*.

Esta opção requer um alinhamento do tipo 4n + 3.

A seguinte figura mostra a estrutura desta opção:

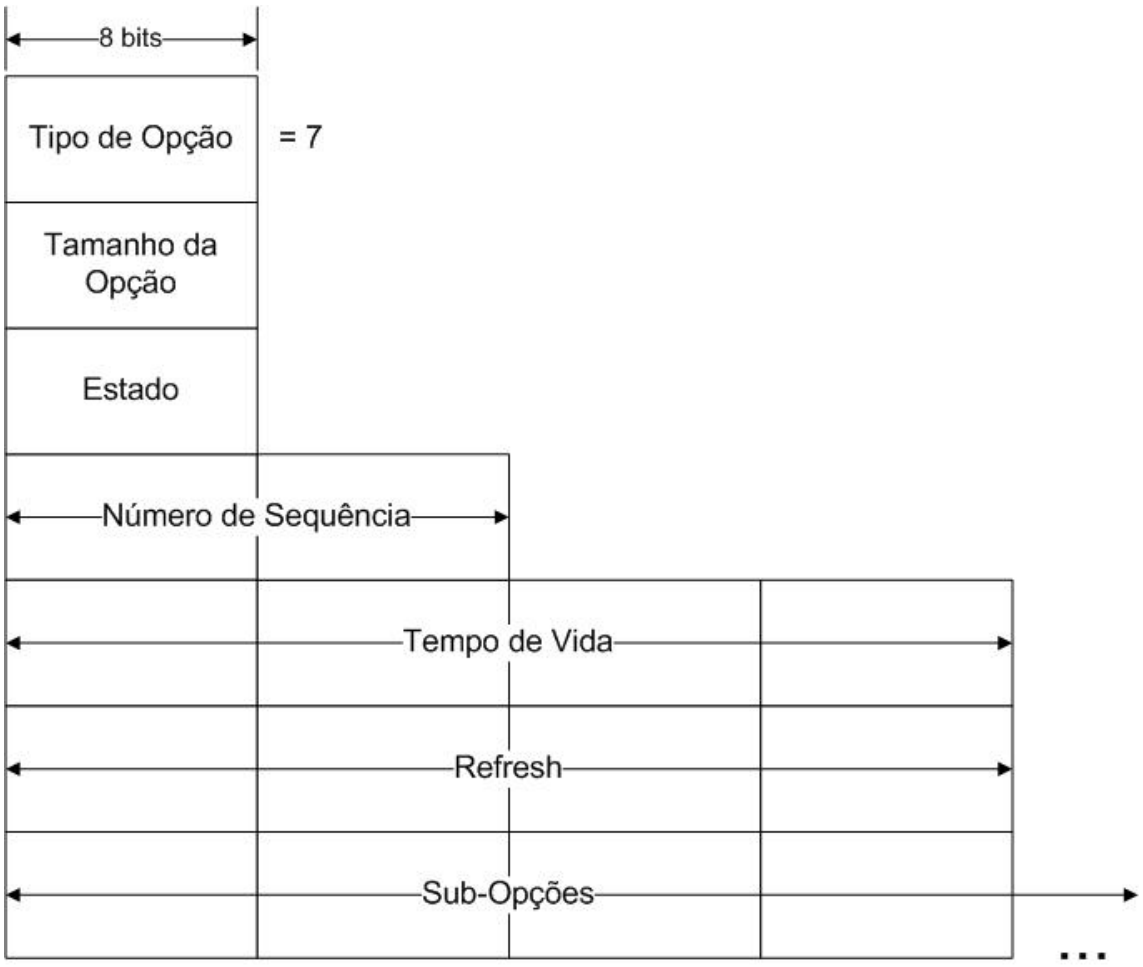


Figura 3.12 – Formato da opção *Binding Acknowledgement* de um pacote IPv6.

Os campos da opção *Binding Acknowledgement* são os seguintes:

- **Opção *Type***

Com esta opção definida com o valor 7, a opção é ignorada se não for reconhecida e não é permitida a sua alteração durante o seu percurso.

- **Opção *Length***

Opção que indica o tamanho da opção em bytes, não incluindo a opção *Type* nem a opção *Length*. A opção *Length* pode incluir sub-opções.

- **Estado**

Indica o estado da actualização de uma ligação. Se o seu valor decimal for inferior a 128, significa que o nó receptor aceitou com êxito a actualização da ligação. Para valores superiores a 127, significa que houve uma falha na actualização da ligação.

Este campo tem um tamanho de 8 bits.

A tabela abaixo indica os possíveis valores e respectivos significados que este campo pode assumir:

Valor do campo Estado (em decimal)	Descrição
0	Actualização da ligação aceite
128	Não especificado
130	Administrativamente proibido
131	Recursos insuficientes
132	Registo *Home* não aceite
133	Não é uma subrede home
136	Tamanho de identificador de *interface* incorrecto
137	Não é um *home agent* para este nó móvel
138	Detecção de endereço duplicado falhada

Tabela 3.3 – Descrição para cada valor do campo Estado.

- **Número de Sequência**

Campo que indica a actualização de ligação para este *Binding Acknowledgement* e tem o valor do campo Número de Sequência da opção *Binding Update* recebida. Este campo tem um tamanho de 16 bits.

- **Tempo de vida**

Campo que indica o número de segundos para os quais o nó de origem irá manter a sua ligação para o nó móvel. Se o nó de origem for o nó móvel *Home Agent*, o campo Tempo de Vida indica também o número de segundos que este será *Home Agent*. Este campo só é relevante se a actualização da ligação for recebida com sucesso. Este campo tem um tamanho de 32 bits.

- ***Refresh***

Campo que consiste num intervalo, em segundos, para o qual o nó móvel deve actualizar a sua ligação com o nó de origem. Este campo só é relevante se a actualização da ligação for recebida com sucesso. Tem um tamanho de 32 bits.

- **Sub-Opções**

As Sub-Opções podem incluir informação adicional e permitir que o *Binding Acknowledgement* seja estendido posteriormente. Correntemente não existem Sub-Opções definidas para a opção *Binding Acknowledgement*.

Opção *Binding Request*

Opção usada para fazer um pedido de ligação a partir de um nó móvel. Tal como na opção *Binding Update*, a opção *Binding Request* pode ser incluída num pacote já existente enviado para o nó móvel, ou num pacote que contenha somente o cabeçalho *Destination Options*.

Esta opção não necessita de qualquer tipo de alinhamento.

A figura seguinte mostra a estrutura desta opção:

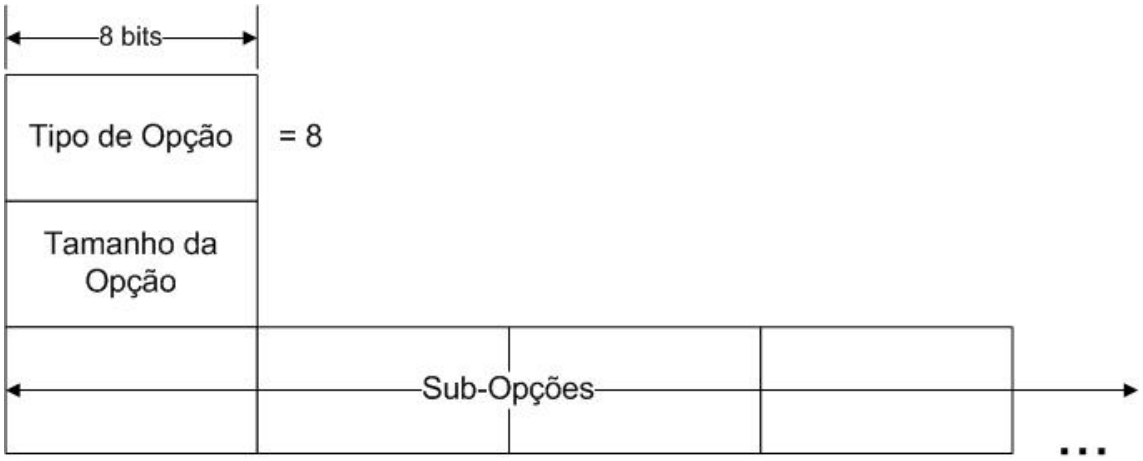


Figura 3.13 – Formato da opção *Binding Request* de um pacote IPv6.

- **Opção *Type***

Com esta opção definida com o valor 8, a opção é ignorada se não for reconhecida e não é permitida a sua alteração durante o seu percurso.

- **Opção *Length***

Opção que indica o tamanho da opção em bytes, não incluindo a opção *Type* nem a opção *Length*. A opção *Length*, se incluída, pode incluir sub-opções. A presença destas sub-opções é detectada através do valor desta opção, ou seja, se a opção *Length* for superior a 0, estão presentes sub-opções.

- **Sub-Opções**

As Sub-Opções podem incluir informação adicional e permitir que o *Binding Acknowledgement* seja estendido posteriormente. A única sub-opção definida para o pedido de ligação é a *Unique Identifier*, que é usada para fazer corresponder um pedido de ligação a uma actualização de ligação.

Opção *Home Address*

A opção *Home Address* é usada para indicar o endereço de origem do nó móvel. Esta opção está incluída na actualização de uma ligação e tem um alinhamento de 8n + 6.

A estrutura desta opção está representada na figura abaixo:

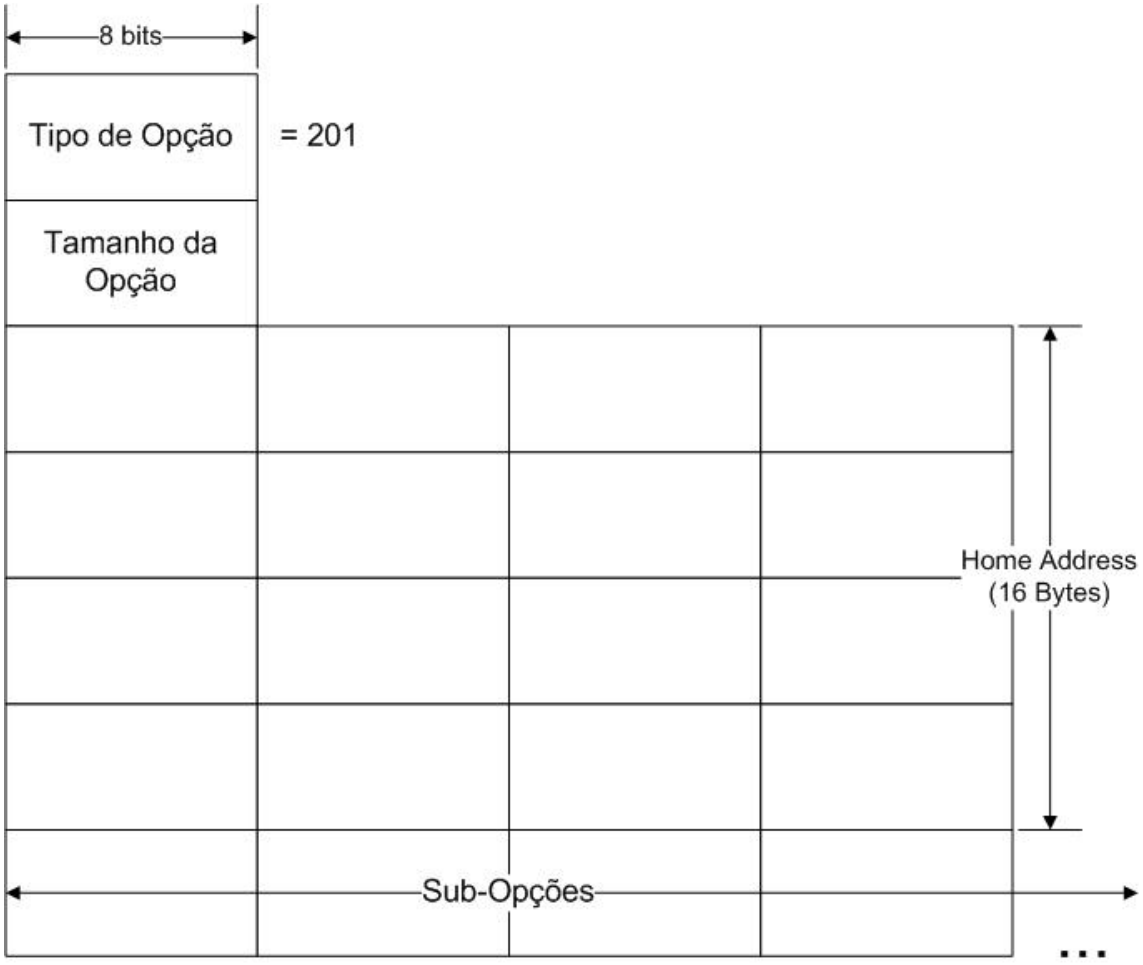


Figura 3.14 – Formato da opção *Home Address* de um pacote IPv6.

- **Opção *Type***

Com a opção *Type* definida com o valor 201, o pacote é descartado e é enviada uma mensagem ICMPv6 *Parameter Problem* se esta opção não for reconhecida e se o endereço de destino não for um endereço *Multicast*. Esta opção não pode ser modificada durante o seu percurso.

- **Opção *Length***

A opção *Length* indica o tamanho da opção em bytes, não incluindo a opção *Type* nem a opção *Length*. Nesta opção estão contidas as sub-opções caso existam. A presença

destas sub-opções é detectada através do valor da opção *Length*, ou seja, se opção *Length* for maior que 16 é sinal de que existem sub-opções.

- ***Home-Address***

O campo *Home Address* representa o endereço origem do nó móvel. O tamanho deste campo é de 128 bits.

- **Sub-*Options***

O campo sub-opções pode incluir informação adicional e permitir que o campo *Home Address* seja estendido no futuro. Correntemente não existem sub-opções definidas para o campo *Home Address*.

3.2.3 Cabeçalho de *Routing*

Os nós de origem IPv6 podem usar o cabeçalho de *Routing* para especificar uma rota de origem, que é composta por uma lista de destinos intermediários para que o pacote chegue até ao seu destino pelo trajecto correcto.

Este cabeçalho é identificado pelo valor 43 no cabeçalho anterior *Next Header*.

O cabeçalho de *Routing* tem a seguinte estrutura:

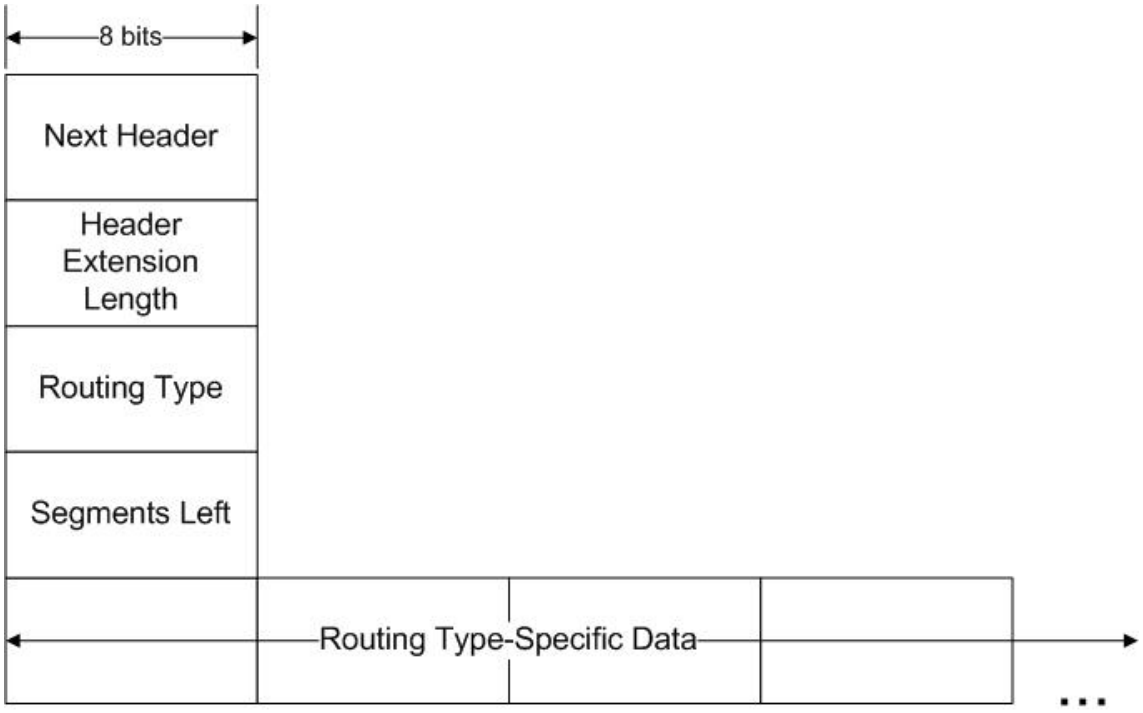


Figura 3.15 – Formato do Cabeçalho de *Routing* de um pacote IPv6.

- ***Next Header e Header Extension***

Definidos do mesmo modo que no cabeçalho de estensão *Hop-by-Hop Options*.

- ***Routing Type***

Identificador de uma variante particular de um cabeçalho de *Routing*.

- ***Segments Left***

Indica o número de destinos intermediários que ainda faltam ser visitados até ao destino final.

- ***Routing Type Specific Data***

Campo de tamanho variável, de formato definido pelo *Routing Type*.

[52]

Se, enquanto um pacote recebido está a ser processado, um nó encontra um cabeçalho de *Routing* com um *Routing Type* não reconhecido, o comportamento do nó depende do valor do campo *Segments Left*, tal como se segue:

***Segments Left* = 0**

O nó tem de ignorar o cabeçalho de *Routing* e processar o próximo cabeçalho do pacote, cujo tipo é identificado pelo campo *Next Header* do cabeçalho de *Routing*.

***Segments Left* ≠ 0**

O nó tem de descartar o pacote e enviar uma mensagem ICMP *Parameter Problem* para o endereço de origem, apontando para o não reconhecido *Routing Type*.[7]

O campo *Routing Type* assume o valor 0, é gerado um cabeçalho chamado *Type 0 Routing* que tem o seguinte formato:

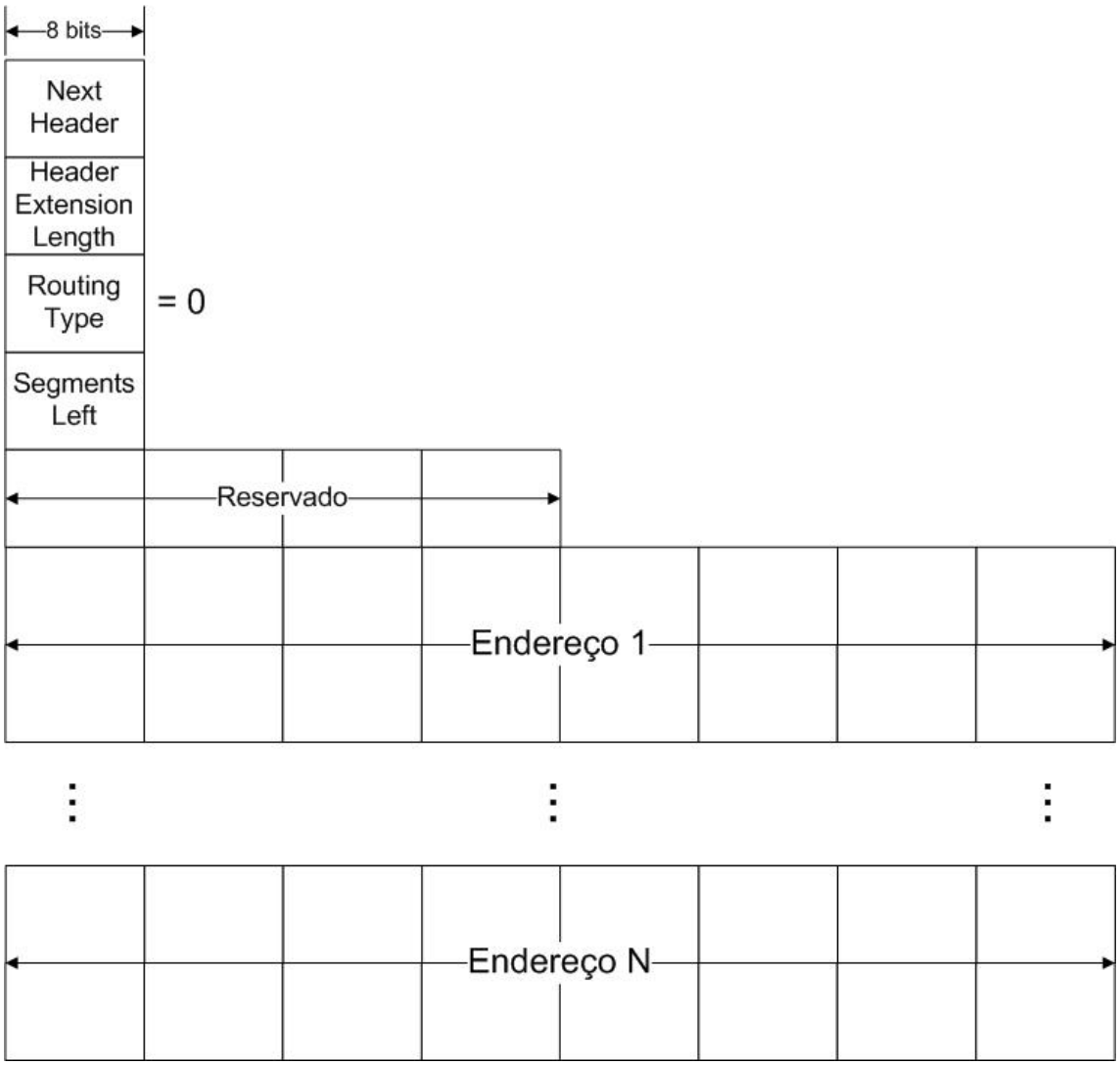


Figura 3.16 – Formato do cabeçalho *Type 0 Routing* de um pacote IPv6.

Quando o campo *Routing Type* está definido com o valor 0, o campo *Routing Type-Specific Data* é composto por um campo reservado de 32 bits e uma lista de endereços de destino intermediários, incluindo o endereço de destino final.

Quando o pacote é inicialmente enviado, o endereço de destino é o endereço do primeiro destino intermediário e o *Routing Type-Specific Data* é a lista de destinos intermediários adicionais e o endereço de destino final. O campo *Segments Left* é composto pelo número total de endereços incluídos no *Routing Type-Specific Data*.

Quando um pacote IPv6 chega a um destino intermediário o cabeçalho de *Routing* é processado no seguinte modo:

1. O endereço de destino corrente e o endereço na (N – *Segments Left* + 1)ª posição da lista de endereços é trocada, onde N corresponde ao número total de endereços do cabeçalho de *Routing*;
2. O campo *Segments Left* é decrementado;
3. O pacote é reencaminhado.

Quando o pacote chega ao destino final, o campo *Segments Left* foi colocado a 0 e a lista de endereços intermediários visitados pelo caminho até chegar ao destino final foi gravada no cabeçalho de *Routing*.

3.2.4 Cabeçalho *Fragment*

O cabeçalho *Fragment* é usado para a *Fragment*ação IPv6 e para reassemblar serviços. Este cabeçalho é identificado com o valor 44 no cabeçalho anterior *Next Header*. A figura seguinte mostra a estrutura de um cabeçalho *Fragment*:

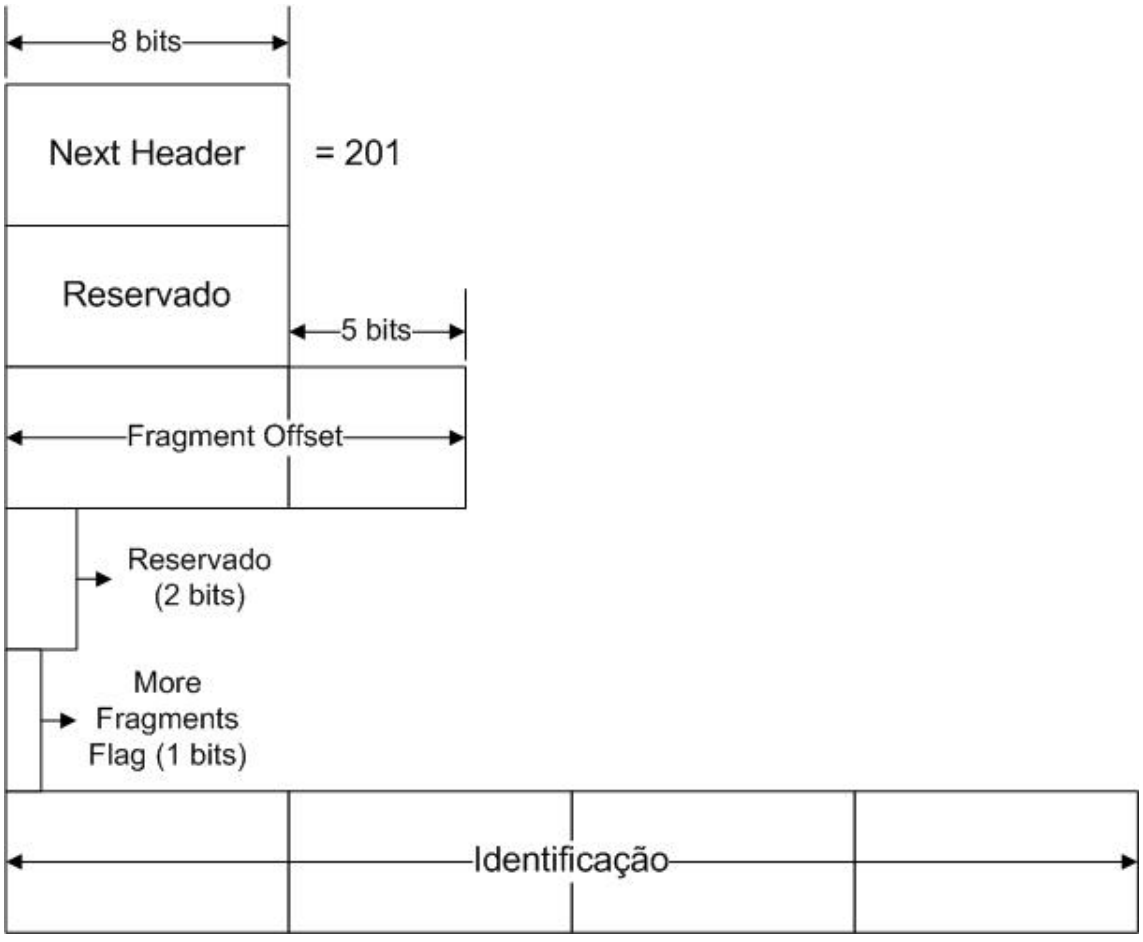


Figura 3.17 – Formato do cabeçalho *Fragment* de um pacote IPv6.

Este cabeçalho inclui um campo *Fragment Offset*, More *Fragments Flag* e um campo *Identification* que são usados do mesmo modo que os campos correspondentes de um cabeçalho IPv4.

Devido ao facto do campo *Fragment Offset* estar definido para blocos de *Fragmentos* de 8 bytes, o cabeçalho *Fragment* não pode ser usado para *jumbograms*. O número máximo que pode ser expresso para o campo *Fragment Offset* é 8191. No entanto este campo pode ser usado para indicar somente um *Fragmento* de dados que comece na posição 8192 x 8, ou seja, 65528.

Em IPv6, somente os nós de origem podem fragmentar *payload*s. Se o *payload* submetido pela camada protocolar acima for maior que a ligação ou caminho MTU, então o *payload* é fragmentado na origem e usa o cabeçalho *Fragment* para voltar a reunir a informação. Um router IPv6 não pode nunca fragmentar um pacote IPv6.

Como uma rede IPv6 não fragmenta uma os *payloads* de um modo transparente, os dados enviados de aplicações que desconhecem o caminho de destino, o MTU não será capaz de saber quando é que os dados que necessitam de ser fragmentados pela origem são descartados pelos routers IPv6. Este facto pode ser um problema para tráfego *Unicast* ou *Multicast* enviado como mensagens UDP ou outros tipos de mensagens que não usem TCP.

3.2.4.1 Diferença entre os campos de fragmentação IPv4 e IPv6

Existem algumas diferenças subtis entre os campos de fragmentação em IPv4 e IPv6.

Em IPv6, os bits usados para as *flags* de fragmentação são os três bits menos significativos dos 16 bits compostos pela combinação das *flags* de fragmentação e pelo campo *Fragment Offset*.

Em IPv4, o campo Identificação é composto por 16 bits em vez dos 32 bits em IPv6 e em IPv6 não existe a *flag Don't Fragment* pois os routers IPv6 nunca fragmentam tráfego, logo esta *flag* estaria sempre com o valor 1 para todos os pacotes IPv6. Assim sendo esta *flag* não é necessária em IPv6.

3.2.4.2 Processo de fragmentação IPv6

Quando um pacote IPv6 é fragmentado, é dividido inicialmente em partes fragmentadas e em partes não fragmentadas.

- A parte não fragmentada de um pacote IPv6 tem de ser processada por nós intermediários entre o nó de origem (onde deveria ser fragmentado) e o nó de destino. Este pacote é composto pelo cabeçalho IPv6, pelo cabeçalho *Hop-by-Hop Options*, pelo cabeçalho *Destination Options* (para destinos intermediários) e pelo cabeçalho de *Routing*.
- A parte fragmentada de um pacote original IPv6 tem de ser processado somente no nó de destino final. Este pacote é composto pelo cabeçalho *Authentication*, pelo cabeçalho *Encapsulating Security Payload*, pelo cabeçalho *Destination Options* (para o destino final) e pela camada protocolar acima PDU.

De seguida, os pacotes IPv6 fragmentados são formados. Cada pacote fragmentado é composto por uma parte não fragmentada, um cabeçalho *Fragment* e uma parte fragmentada. O esquema seguinte mostra o processo de fragmentação de um pacote IPv6:

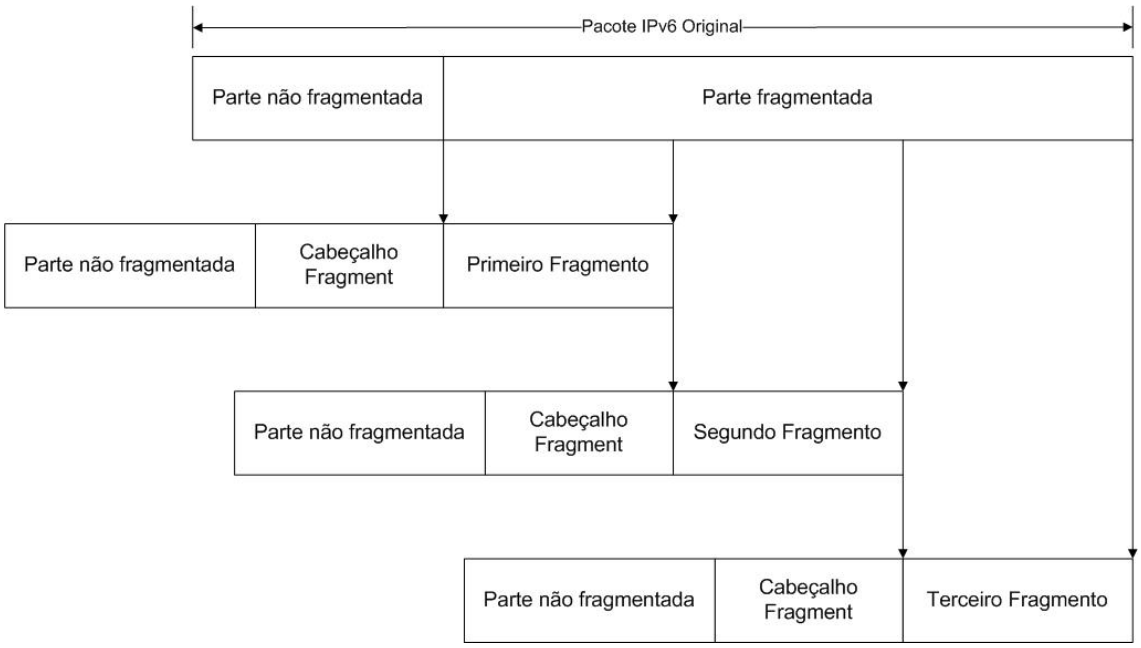


Figura 3.18 – Processo de fragmentação de um pacote IPv6.

Em cada Fragmento, o campo *Next Header* do cabeçalho *Fragment* indica o primeiro cabeçalho ou a camada protocolar acima da parte fragmentável original. O campo *Fragment Offset* do cabeçalho *Fragment* indica o offset (em 8 Bytes conhecidos como blocos de *Fragmentos*) do *Fragmento* relativo ao *payload* original. A *flag* More *Fragments* está definida em todos os fragmentos de pacotes excepto no último fragmento do pacote. Todos os fragmentos de pacotes criados a partir do mesmo pacote IPv6 têm de conter o mesmo valor no campo *Identification*.

A fragmentação de pacotes em IPv6 pode ocorrer quando a camada protocolar acima da máquina de origem envia um pacote IPv6 maior que o caminho MTU para uma máquina destino.

Os pacotes IPv6 enviados para destinos IPv4 (pacotes IPv4 com IPv6 embutido) podem receber um caminho MTU menor que 1280 Bytes. Neste caso, a máquina de origem envia pacotes IPv6 com um cabeçalho *Fragment* no qual o campo *Fragment Offset* está definido a 0 e a *flag More Fragments* não está definida e tem um *payload* menor que 1272 Bytes. O cabeçalho *Fragment* está incluído no pacote IPv6 para que a conversão IPv6 para IPv4 possa usar o campo *Identification* para que os pacotes IPv4 fragmentados possam chegar ao seu destino.

3.2.4.3 Chegada dos fragmentos de pacotes IPv6 ao destino

Os fragmentos dos pacotes IPv6 são encaminhados pelos routers IPv6 até ao endereço de destino final IPv6. Estes fragmentos podem tomar caminhos distintos até ao ser destino e chegar em ordem diferente (aleatório) em relação a ordem em que foram enviados. Para voltar a unir estes fragmentos, o protocolo IPv6 usa os campos *Source Address* e *Destination Address* do cabeçalho IPv6 e o campo *Identification* do cabeçalho *Fragment* para agrupar os fragmentos, tal como mostra a figura seguinte:

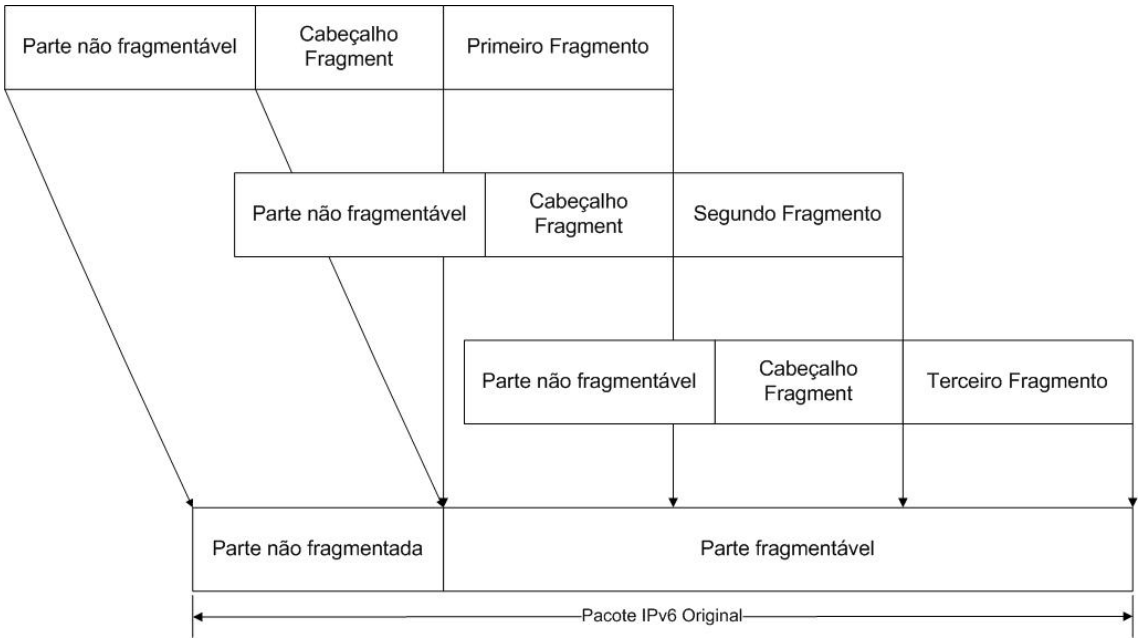


Figura 3.19 – Processo de agregação dos fragmentos IPv6.

Depois de todos os fragmentos chegarem ao seu destino, é calculado o tamanho original do *payload* e o campo *Payload Length* do cabeçalho IPv6 para voltar a unir os fragmentos dos pacotes é actualizado.

É recomendado uma reunião de fragmentos em 60 segundos antes de descartar o pacote parcialmente construído.

3.2.5 Cabeçalho *Authentication*

O cabeçalho *Authentication* disponibiliza autenticação de dados (verificação do nó que envia os dados), integridade dos dados (verificação se os dados forma alterados durante o seu percurso até ao destino final) e protecção *anti-replay* (garantia de que os pacotes capturados não podem ser retransmitidos e aceites como pacotes validos) para pacotes IPv6 incluindo campos do cabeçalho IPv6 que não mudam durante o seu percurso na rede.

Este cabeçalho é identificado pelo valor 51 no cabeçalho anterior *Next Header*.

A figura seguinte mostra a estrutura deste cabeçalho:

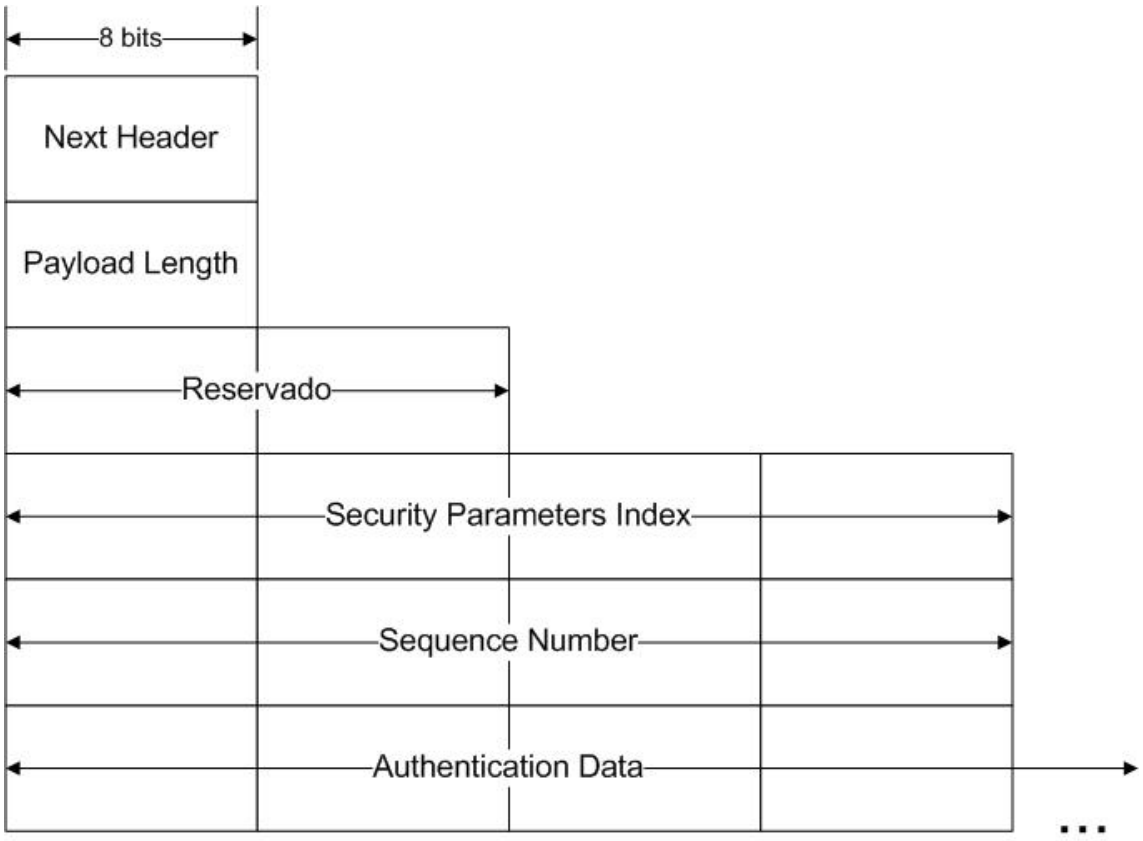


Figura 3.20 – Formato do cabeçalho *Authentication.*

O cabeçalho *Authentication* é composto por um campo *Next Header*, um campo *Payload Length* (número de blocos de 4 bytes no cabeçalho *Authentication*, sem contar com os primeiros dois), um campo *Reserved*, um campo Security Parameters *Índex* (SPI) que ajuda a identificar um *IP Security* (IPSec) específico, um campo *Sequence Number* que disponibiliza protecção *anti-replay* e finalmente um campo *Authentication Data* que contém um valor de verificação de integridade (ICV). ICV oferece autenticação e integridade dos dados.

O cabeçalho *Authentication* não fornece serviços de confidencialidade de dados para as camadas acima do PDU pois não cifra os dados de modo que estes não possam ser vistos sem a respectiva chave cifrada. Para obter autenticação, integridade e confidencialidade de dados para todos os pacotes IPv6, pode ser usado ambos os cabeçalhos *Authentication* e *Encapsulating Security Payload*.

3.2.6 Cabeçalho *Encapsulating Security Payload*

O cabeçalho *Encapsulating Security Payload* (ESP) oferece confidencialidade, autenticação e integridade de dados, e serviços repetidos de protecção de dados para o *payload* encapsulado. O cabeçalho ESP não disponibiliza serviços de segurança para o cabeçalho ou cabeçalhos de estensão IPv6 que ocorram antes do cabeçalho ESP.

O cabeçalho ESP é identificado pelo valor 50 no cabeçalho anterior *Next Header*.

A figura seguinte mostra a estrutura deste cabeçalho:

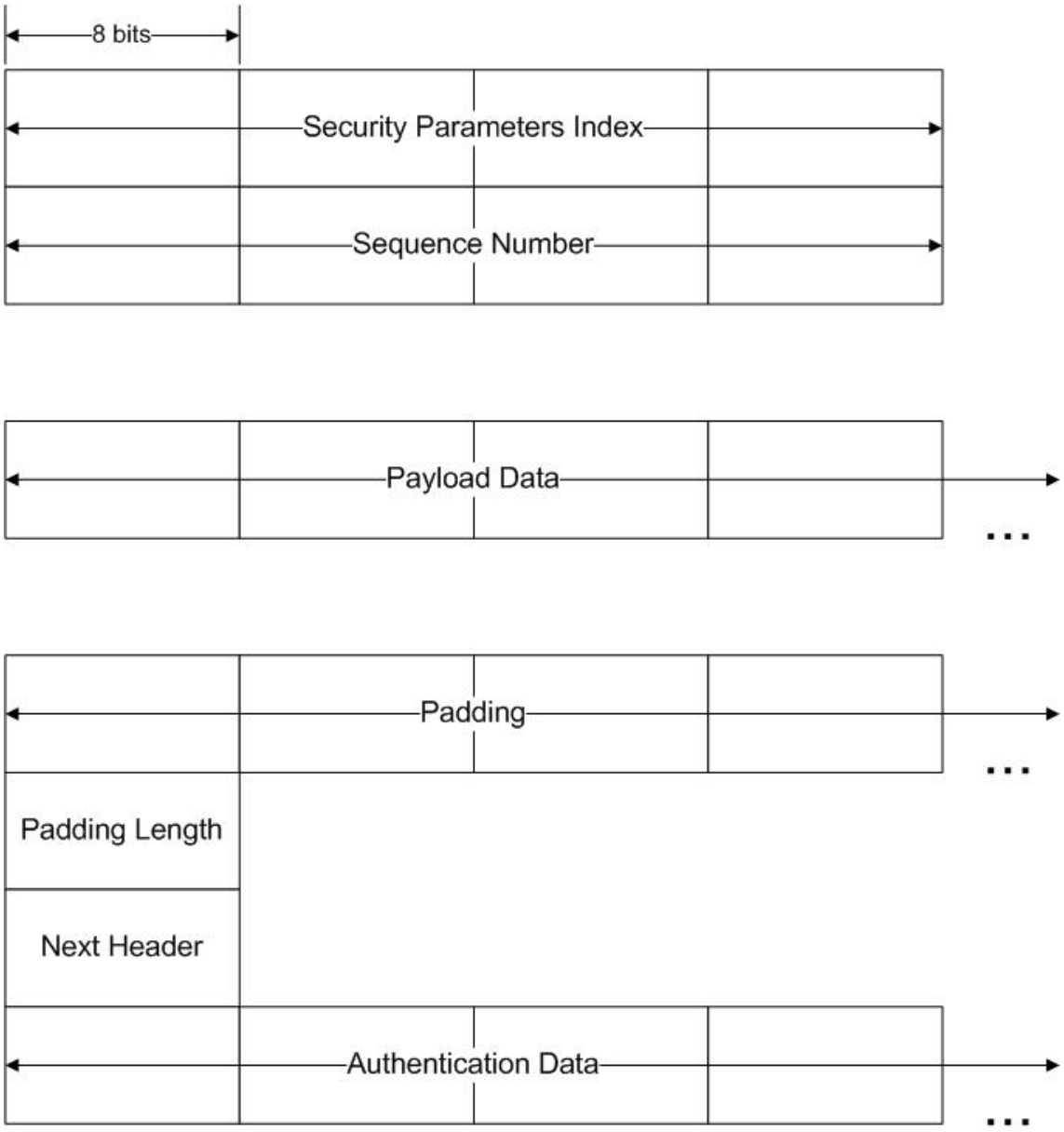


Figura 3.21 – Formato do cabeçalho *Encapsulating Security Payload.*

O cabeçalho ESP é composto por é composto por um campo *Security Parameters* Índex (SPI) que ajuda a identificar o IPSec SA, um campo *Sequence Number* que oferece protecção *anti-replay*, um campo *Padding* usado para garantir que o *payload* é múltiplo de 4 bytes e ajusta os blocos de dados para algoritmos de encriptação, um campo *Padding Length* que indica o tamanho do campo *Padding* em Bytes e finalmente um campo *Authentication Data* que contém o ICV.

4 IPv6 MTU

O protocolo IPv6 obriga a que a camada de ligação suporte um MTU mínimo de 1280 Bytes. As camadas de ligação que não suportem este MTU têm de fornecer mecanismos de fragmentação e de reunião que seja transparente para IPv6. Por outro lado, para as camadas de ligação que suportem um MTU configurável, é recomendado que estas sejam configuradas com um MTU de pelo menos 1500 Bytes (IPv6 MTU para o encapsulamento *ETHERNET II*). Um exemplo de um MTU configurável é o *Maximum Receive Unit* (MRU) de uma ligação PPP.

Tal como em IPv4, o protocolo IPv6 fornece um mecanismo *Path MTU Discovery* que utiliza uma mensagem ICMPv6 *Packet Too Big* para enviar pacotes de tamanho superior a 1280 Bytes.

As máquinas IPv6 de origem podem fragmentar *payloads* de camadas protocolares acima que sejam maiores que o MTU utilizando os mecanismos acima referidos. No entanto, a fragmentação de pacotes IPv6 não é aconselhável. Um nó IPv6 tem de ser capaz de reunir os fragmentos de um pacote fragmentado que tenha pelo menos 1500 Bytes de tamanho.[52]

5 Alguns Protocolos IPv6

Neste capítulo serão abordados protocolos considerados importantes na transição de IPv4 para IPv6. Muitos deles são protocolos já usados nos últimos anos em IPv4 mas agora foram actualizados para suportar o novo protocolo IPv6. Outros protocolos são relativamente recentes e já foram implementados com o suporte para IPv6.

5.1 *Border Gateway Protocol (*BGP)

O BGP é o protocolo de encaminhamento do *core* da *Internet*. Funciona mantendo uma tabela com os IP ou prefixos que permitem alcançar redes entres sistemas autónomos (AS).

É um "*path vector Protocol*". Não usa as métricas habituais de outros protocolos de encaminhamento mas sim decisões baseadas nas políticas das redes. Em Janeiro de 2006 este protocolo foi actualizado [21] substituindo a original [20].

O BGP foi criado para substituir o EGP (*Exterior Gateway Protocol*). O objectivo era descentralizar totalmente o sistema (*Internet*) removendo o *Backbone* da *Internet* NSFNET.

O multiprotocolo BGP para IPv6 tem como novas características o suporte para endereços IPv6 e a capacidade de *Multicast* entre domínios.[36]

5.2 IPv6 RIP

O protocolo RIP para IPv6 fornece as mesmas características que o RIP para IPv4.Como características novas tem o suporte para os endereços IPv6 e seus prefixos e o uso do endereço FF02::9 (endereço do grupo *Multicast* para os router RIP) como destino das mensagens de *update*.

No IOS Cisco cada processo IPv6 RIP mantém uma tabela local de encaminhamento denominada de *Routing Information Database* (RIB). O RIP RIB tem um conjunto das melhores rotas aprendidas a partir dos vizinhos. Se o RIP aprender a mesma rota de dois vizinhos diferentes mas com custos diferentes apenas será guardada no RIB local a que tem um custo menor. O RIB também guarda rotas obsoletas que o processo RIP anuncia aos seus vizinhos que também estão a usar RIP. O IPv6 RIP irá tentar inserir qualquer rota activa existente na RIB local na RIB principal denominada de "*master RIB*". Caso a mesma rota seja aprendida por outro protocolo de encaminhamento com uma distância administrativa melhor, a rota não irá ser adicionada ao RIB.

5.3 IPv6 EIGRP

O protocolo EIGRP para IPv6 funciona do mesmo modo que o EIGRP para IPv4. Tem como nova característica o facto do protocolo ser configurado nas *interface*s. Esta

característica elimina a necessidade do uso de endereços globais, acabando com o comando "NETWORK".

Novas características do protocolo em relação ao EIGRP IPv4:

- Requer a atribuição de um id ao router antes da inicialização do protocolo.
- O protocolo tem a característica *shutdown* comum às *interface*s dos routers e switch, para correr o EIGRP é necessário efectuar o comando "*no shutdown*".
- Característica de filtragem de rotas através do comando *distribute-list prefix-list*.
- Protocolo *Neighbor Discovery*.[68]

O EIGRP tem 4 componentes base:

- ***Neighbor Discovery*** – *Neighbor Discovery* é o processo que os routers utilizam para tomarem conhecimento de outros routers nas redes directamente ligadas. Os routers precisam de saber quando os routers vizinhos se tornam inalcançáveis (*unreachable*). O *Neighbor Discovery* é conseguido através do envio periódico de pequenos pacotes *hello*. Enquanto os pacotes de *hello* forem recebidos o IOS Cisco consegue determinar que routers vizinhos estão activos e em funcionamento. A partir do momento que está feito o levantamento dos routers vizinhos activos é efectuada a troca de informação de encaminhamento.

- ***Reliable Transport Protocol*** – este protocolo é responsável pela entrega ordenada dos pacotes EIGRP a todos os vizinhos. Suporta a transmissão de *Multicast* e *Unicast*. Alguns pacotes EIGRP necessitam de garantia de entrega, outros não, por questões de eficácia a garantia de entrega (*reliability*) só é fornecida quando necessário.

  Os pacotes de *hello* não necessitam de garantia de entrega, sendo assim o EIGRP envia um único *hello Multicast* com uma indicação de não ser necessário o ACK (*acknowledgment*). Outros tipos de pacotes como os pacotes de *update* já requerem os *updates*, isso mesmo é indicado (necessidade de *ack*) nos pacotes enviados.

  O *Reliable Transport Protocol* tem a capacidade de enviar pacotes *Multicast* de maneira muito rápida sempre que pacotes sem necessidade de *ack* estão pendentes, esta característica permite manter valores baixos de convergência na presença de várias velocidades de ligação (*speed links*).

- ***DUAL Finite State Machine***- responsável pela processo de decisão da escolha de rotas. Monitoriza todas as rotas anunciadas pelos vizinhos. Utiliza métricas que incluem distância e custo para uma selecção eficiente de uma rota livre de *loops*. Quando existem várias rotas para um vizinho o DUAL escolhe a que tem uma métrica menor (denominada *feasible distance*) e insere-a na tabela de encaminhamento. Outras rotas para o vizinho (com métricas maiores) que sejam recebidas são analisadas e o DUAL determina a "*reported distance*" para a rede em que se encontra. A *reported distance* é a métrica total anunciada por um *upstream router* de uma origem a um destino. O DUAL compara a *reported*

distance com a distância administrativa, caso seja inferior o DUAL considera a rota como uma rota sucessora e insere-a na tabela topológica da rede. Esta rota torna-se na rota sucessora caso a rota actual falhe, funciona como uma rota alternativa para se a principal falhar.

- ***Protocol-dependent modules***- quando não existem sucessores capazes de assegurar encaminhamento para uma rota que tenha falhado, mas os vizinhos continuam a anuncia-la, é necessário uma nova escolha de rota. Este é o processo em que o DUAL encontra um novo sucessor (nova rota que assegure o encaminhamento para determinado local). Sempre que ocorre uma alteração da topologia da rede o DUAL testa possíveis rotas para sucessor. A escolha de rota é um processo intenso, e deve ser evitado sempre que possível. Caso existam rotas que possam funcionar como sucessores estas são designados como tal de modo a evitar uma nova escolha de rota.

  Os *Protocol-dependent modules* são responsáveis pelas tarefas protocolares da camada três. Como exemplo temos o módulo EIGRP que é responsável pelo envio e recepção de pacotes EIGRP encapsulados em IPv4 ou IPv6.

5.4 OSPFv2 e OSPFv3

O protocolo OSPF é um protocolo de encaminhamento IP do tipo *Link-State* ao contrário dos outros existentes distance vector. Um protocolo do *Link-State* decide as rotas com base no estado dos *link*s das máquinas origem e destino. Podemo-nos referir a estado dos *link*s como *interface*s dos dispositivos de rede e a sua relação com outros dispositivos e *interface*s vizinhos. A informação associada a cada *interface* é a descrição da *interface*, o prefixo ipv6 correspondente, máscara de rede, o tipo de rede a que está ligada, os routers ligados, etc. Esta é a informação que é propagada em LSAs (*link* state-advertisement). Os dados das LSAs são armazenandos numa base de dados que quando sujeita ao algoritmo de Dijkstra[15] resulta na criação de uma tabela de encaminhamento OSPF. A diferença entre a base de dados e a tabela de encaminhamento é que a base de dados contem toda a informação enquanto que as tabelas apenas contém os caminhos mais curtos para destinos conhecidos através das *interface*s do router.

OSPFv2 vs OSPFv3

O OSPFv3 tem suporte para IPv6 ou seja, suporta o encaminhamento de prefixos IPv6 assim como o suporte para os endereços IPv6. Não é necessário criar um *Routing* process como em outros protocolos, activando o OSPF para IPv6 numa *interface* cria automaticamente o processo e a sua configuração associada. Tal como noutros protocolos para IPv6 o OSPF precisa de ser activado nas *interface*s onde se pretende usar.

[15] Edsger Wybe Dijkstra - foi um cientista da computação. Recebeu o Turing Award de 1972 pelas suas contribuições fundamentais na área das linguagens de programação.

Em IPv6 os utilizadores podem configurar vários prefixos IPv6 a usar numa *interface*. No OSPF todos os prefixos estão incluídos, ou são usados todos os prefixos ou não é usado nenhum, não é possível escolher apenas alguns prefixos para usar.

Ao usar o OSPF é possível não atribuir nenhum endereço IPv4 a uma *interface* do router, neste caso é necessário usar o comando router-id de maneira a atribuir um ID ao router antes de se iniciar o processo OSPF. Este passo é necessário pois quando existe um endereço IPv4 atribuído este é usado como router ID. Caso não exista nenhum endereço IPv4 atribuído é necessário configurar o router ID. Caso exista mais que um endereço IPv4 atribuído é usado o mesmo algoritmo utilizado na versão 2 do OSPF [22].

5.5 Integrated Intermediate System-to-Intermediate System (IS-IS)

IS-IS é um IGP (*Interior Gateway Protocol*) que anuncia informação *Link-State* pela rede de modo a criar uma imagem da topologia da rede. O IS-IS é um protocolo de encaminhamento hierárquico OSI (*Open Systems Interconnection*). O IS-IS é um protocolo de encaminhamento hierárquico que caracteriza os dispositivos de uma rede (um sistema intermédio) em dispositivos de nível 1 e dispositivos de nível 2.

Os dispositivos de nível 2 (relativos a um domínio) funcionam em áreas de dispositivos de nível 1(relativos a uma área) de modo a criar um *Backbone* de encaminhamento de intradominio. O IS-IS usa um algoritmo de encaminhamento que suporta os vários tipos de famílias de endereços existentes, IPv4, IPv6 e OSI.

5.5.1 IS-IS para IPv6

O IS-IS para IPv6 oferece os mesmos benefícios e funciona do mesmo modo que o protocolo IS-IS para IPv4. As diferenças são as mesmas que as encontradas na maioria dos protocolos já referidos anteriormente, ou seja, têm suporte para anunciar prefixos IPv6 (suporte prefixos OSI também). Este protocolo suporta também o modo de topologia única e modo de topologia múltipla.

Topologia única

Este modo permite que as *interface*s sejam configuradas com o protocolo IS-IS juntamente com outros tipos de protocolo (o IPv4 por exemplo). Todas as *interface*s têm de ser configuradas com famílias de endereços de rede idênticas (IPv4, IPv6). Para além disso todos os routers na área IS-IS (nível 1) ou domínio (nível 2) têm de suportar essa família de endereços em todas as *interface*s.

Topologia múltipla

Este modo permite que o IS-IS mantenha um conjunto de topologias independentes numa única área ou domínio. Este modo elimina a restrição de todas as *interface*s configuradas com IS-IS terem de suportar uma família de endereços idêntica, assim com a restrição de todos os routers na área ou domínio terem de suportar a mesma família de endereços de rede (IPv4, IPv6).

5.6 ICMPv6

Tal como no protocolo IPv4, a especificação do cabeçalho IPv6 e respectivos cabeçalhos de estensão não fornecem mecanismos de aviso de erros. Em vez disso, o protocolo IPv6 utiliza uma versão melhorada do protocolo *Internet Control Message Protocol* (ICMP) conhecida como ICMPv6. Este protocolo tem as funções comuns do ICMP em IPv4 tais como aviso de entrega de pacotes e encaminhamento de erros, e o simples serviço de *echo* para *troubleshooting*.

O protocolo ICMPv6 tem as seguintes vantagens face ao protocolo seu homólogo para IPv4:

- ***Neighbor Discovery***

  Neighbor Discovery (ND) é composto por uma série de cinco mensagens ICMP que cuidam da comunicação ponto-a-ponto de uma ligação. Este mecanismo substitui o *Address Resolution Protocol* (ARP), ICMPv4 *Router Discovery* e o ICMPv4 *Redirect Message* do protocolo IPv4.

- ***Multicast Listener Discovery***

  Multicast Listener Discovery (MLD) é composto por uma série de três mensagens ICMPv6 equivalentes à versão 2 do protocolo *Internet Group Management Protocol* (IGMP) para IPv4, para tratar dos membros *Multicast* de uma subrede.

5.6.1 Tipos de mensagens ICMPv6

- **Mensagens de erros**

  As mensagens de erros reportam erros na entrega ou encaminhamento de pacotes IPv6 por parte do nó destino ou mesmo por um router intermediário. O bit mais significativo do campo de 8 bits *Type* para todas as mensagens ICMPv6 é definido a zero. Assim, os valores válidos para este campo quando se trata de mensagem de erro estão definidos de 0 a 127. As mensagens de erro ICMPv6 podem ser: *Destination Unreachable*, *Packet Too Big*, *Time Exceeded* e *Parameter Problem*.

- **Mensagens de Informação**

  As mensagens de informação fornecem diagnósticos e funcionalidades adicionais das máquinas tais como MLD e ND. O bit mais significativo do campo de 8 bits *Type* para todas as mensagens ICMPv6 é definido a um. Assim, os valores válidos para este campo quando se trata de mensagem de informação estão definidos de 128 a 255.

  As mensagens de informação ICMPv6 podem ser: *Echo Request* e *Echo Reply*.

5.6.2 Cabeçalho ICMPv6

Um cabeçalho ICMPv6 é composto por o valor 58 no cabeçalho *Next Header* no cabeçalho imediatamente anterior. A sua estrutura é a seguinte:

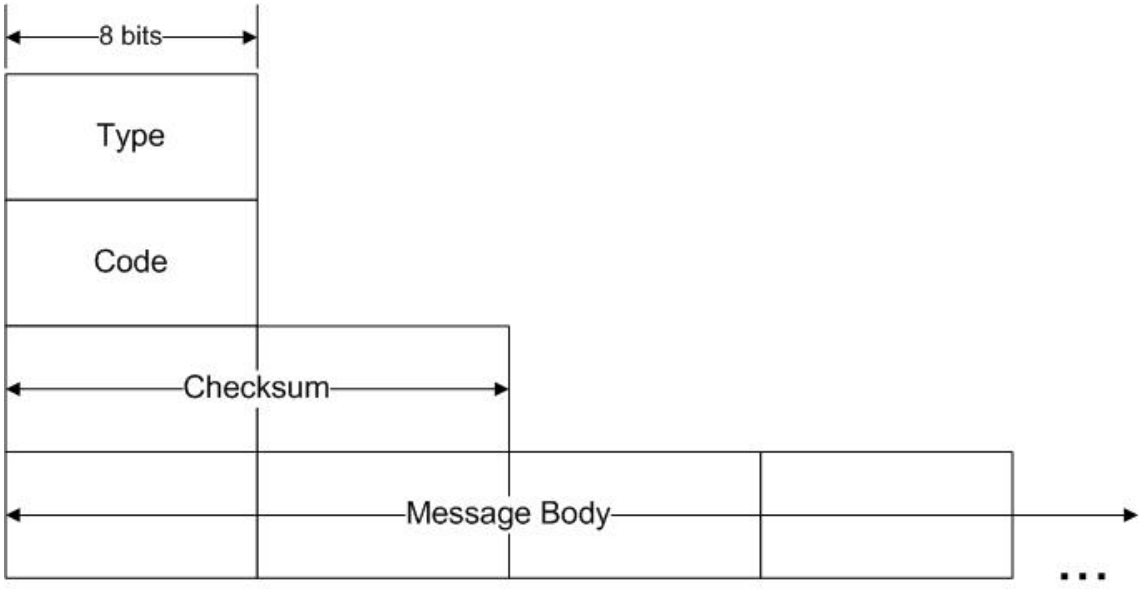


Figura 5.1 – Formato do cabeçalho de um pacote ICMPv6.

Campos:

- ***Type***

Campo que indica o tipo de mensagem ICMPv6. Tem um tamanho de 8 bits. Numa mensagem de erro ICMPv6, o bit mais significativo deste campo é colocado a 0. Numa mensagem de informação ICMPv6, o bit mais significativo deste campo é colocado a 1.

- ***Code***

Campo que diferencia múltiplas mensagens dentro de um dado tipo de mensagem. Tem um tamanho de 8 bits. Para a primeira ou única mensagem de um dado tipo, o valor do campo *Code* é 0.

- ***Checksum***

Campo que contém o *checksum* da mensagem ICMPv6. Tem um tamanho de 16 bits. O cabeçalho IPv6 é adicionado à frente da mensagem ICMPv6 quando o *checksum* é calculado.

- ***Message Body***

Campo que contém os dados específicos de ICMPv6.

5.6.3 Mensagens de erro

As mensagens de erro ICMPv6 reportam erros de encaminhamento e de entrega de pacotes e consiste nas seguintes mensagens:

- *Destination Unreachable* (ICMPv6 *Type* 1)
- *Packet Too Big* (ICMPv6 *Type* 2)
- *Time Exceeded* (ICMPv6 *Type* 3)
- *Parameter Problem* (ICMPv6 *Type* 4)

Para manter a largura de banda, as mensagens de erro ICMPv6 não são enviadas para todos os erros encontrados. Em vez disso, a taxa de envio destas mensagens é limitada com base no seguinte:

- **Um temporizador**

  A taxa de envio de uma mensagem de erro por fonte é de 7 milisegundos em 7 milisegundos.

- **Uma percentagem de largura de banda**

  A taxa de mensagens de erro enviadas por *interface* é uma percentagem P da largura de banda da ligação.

5.6.3.1 *Destination Unreachable*

Um router ou uma máquina de destino enviam uma mensagem ICMPv6 *Destination Unreachable* quando o pacote não pode ser encaminhado para o nó destino ou por um protocolo de uma camada superior. A seguinte figura mostra a estrutura de uma mensagem ICMPv6 *Destination Unreachable*:

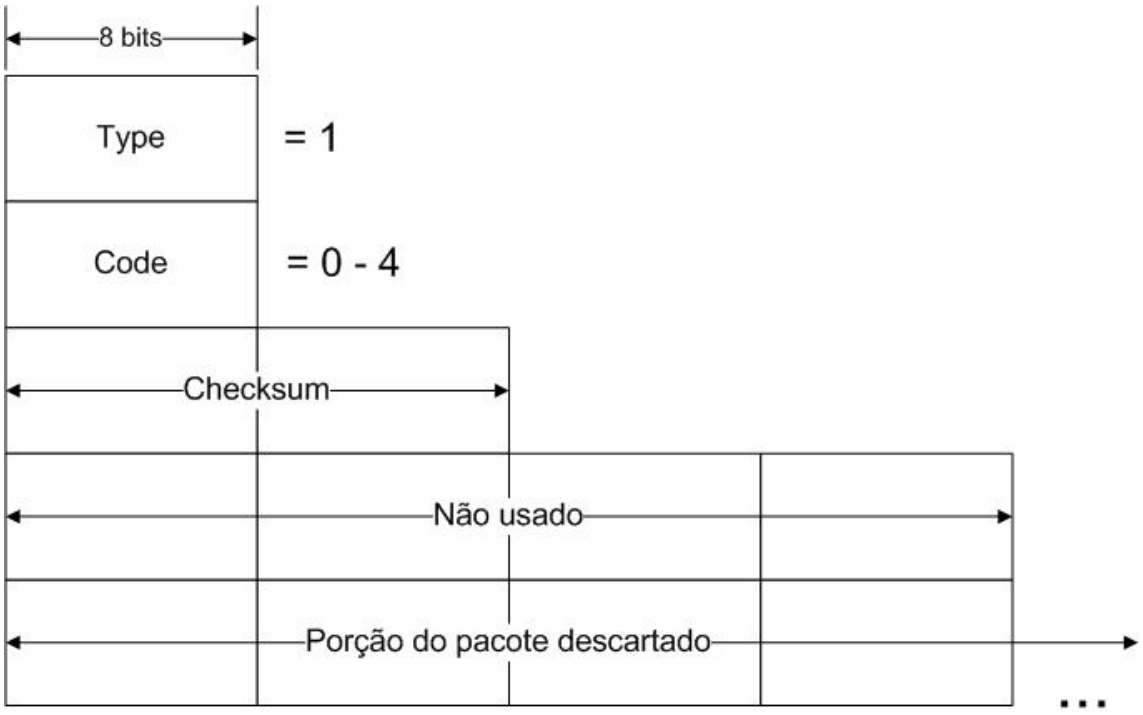


Figura 5.2 – Formato de uma mensagem ICMPv6 *Destination Unreachable.*

Numa mensagem *Destination Unreachable*, o campo *Type* tem o valor 1 e o campo *Code* tem um valor de 0 a 4. De seguida o *Checksum* é um campo de 16 bits. O campo seguinte trata-se da porção do pacote descartado que contém a mensagem ICMPv6 e não pode ter um tamanho superior a 1280 Bytes (MTU mínimo para IPv6). O número de Bytes do pacote descartado incluídos na mensagem variam se existirem cabeçalhos IPv6 estendidos.

A seguinte tabela mostra o significado de cada valor do campo *Code*:

Code	**Significado**
0	Sem rota para o destino
1	Comunicação com o destino está administritivamente proibida
2	Para além da abrangência do endereço de destino
3	Endereço inatingível
4	Porto inatingível
5	Falha na política de entrada/saída no endereço de origem
6	Rota para o destino rejeitada

Tabela 5.1 – Descrição de cada valor do campo *Code* de uma mensagem ICMPv6 que não chega ao seu destino.

[19]

5.6.3.2 *Packet Too Big*

Uma mensagem ICMPv6 *Packet Too Big* é enviada quando o pacote não pode ser encaminhado porque a ligação MTU da *interface* do router de que vai encaminhar o pacote é mais pequena que o tamanho do pacote IPv6.

Estrutura destes pacotes:

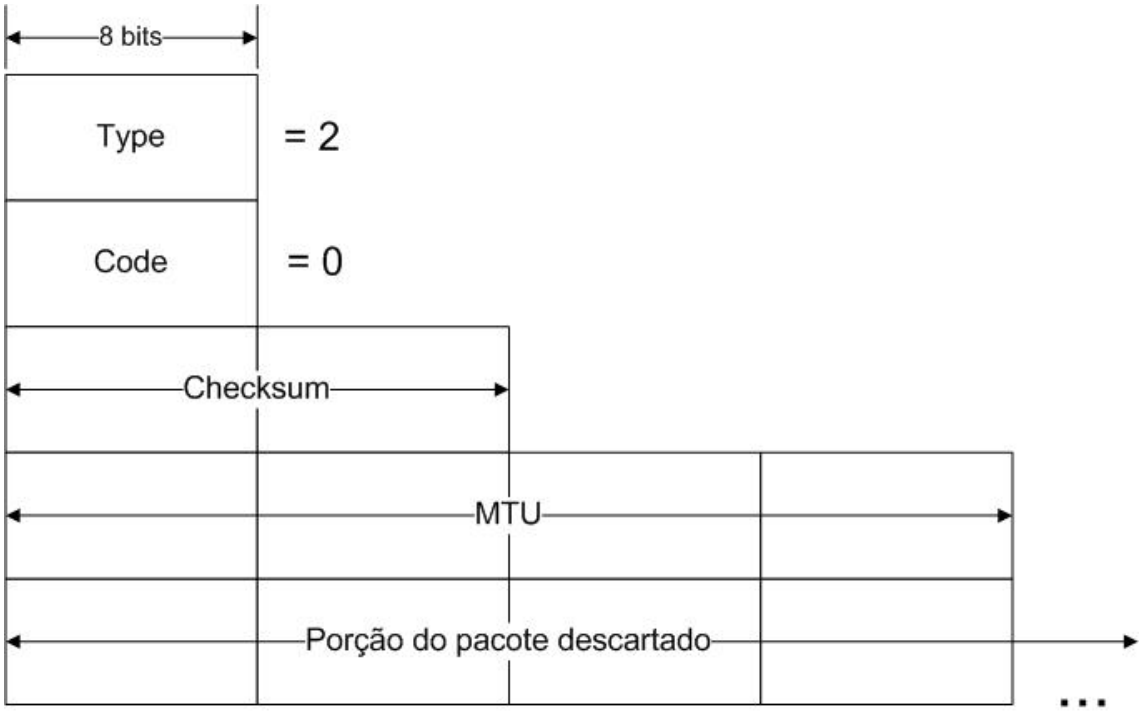


Figura 5.3 – Formato de uma mensagem ICMPv6 *Packet Too Big.*

Numa mensagem *Packet Too Big*, o campo *Type* é definido a 2 e o campo *Code* a 0. O campo MTU é um campo de 32 bits que contém o *link* MTU da *interface* cujo pacote foi encaminhado. O campo seguinte trata-se da porção do pacote descartado que contém a mensagem ICMPv6 e não pode ter um tamanho superior a 1280 Bytes (MTU mínimo para IPv6). O número de Bytes do pacote descartado incluídos na mensagem variam se existirem cabeçalhos IPv6 estendidos.

5.6.3.3 *Time Exceeded*

Um router normalmente envia uma mensagem ICMPv6 *Time Exceeded* quando o campo *Hop Limit* do cabeçalho IPv6 apresenta o valor 0. A figura seguinte mostra a estrutura destas mensagens:

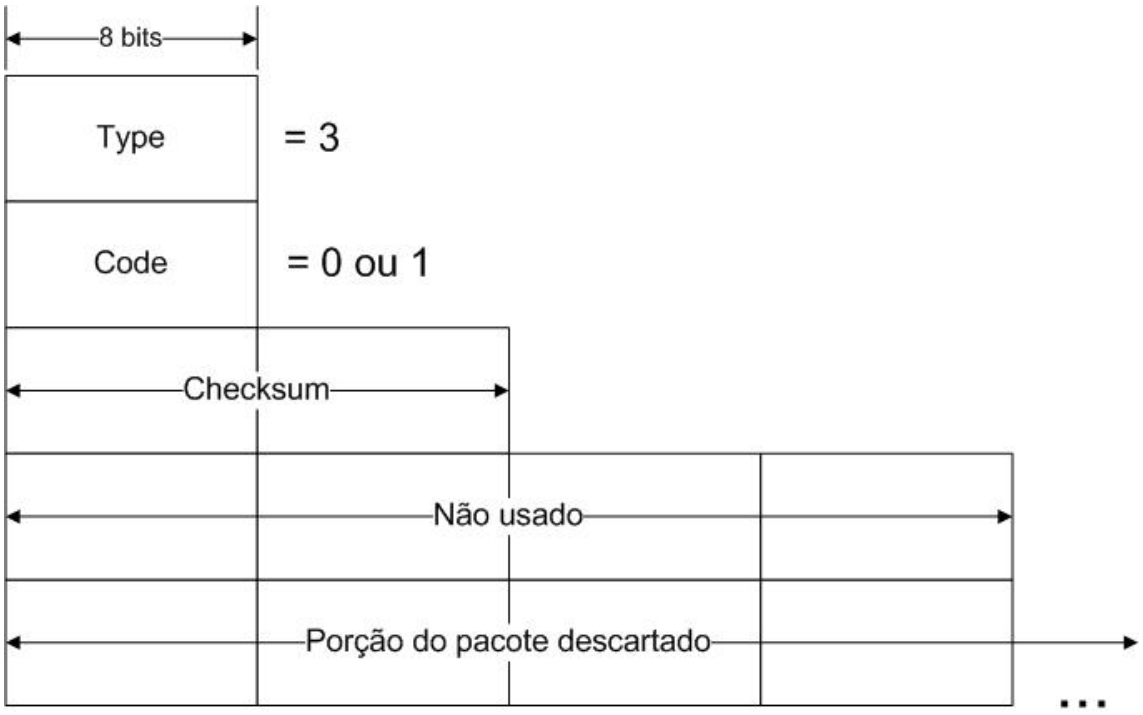


Figura 5.4 – Formato de uma mensagem ICMPv6 *Time Exceeded.*

O campo *Type* é definido a 3 e o campo *Code* pode assumir dois valores:

- **Valor 0 (*Hop Limit Exceeded in transit*)** – quando o valor do campo *Hop Limit* do cabeçalho IPv6 é decrementado para 0 ou, em raras ocasiões, o valor do campo *Hop Limit* de um pacote acabado de chegar ao seu destino tem o valor 0.

- **Valor 1 (*Fragment Reassembly Time Exceeded*)** – quando o tempo de reassemblagem dos *Fragmentos* do pacote na máquina destino expira.

Os campos *Checksum* e Não usado já foram explicados em 5.6.3.1.

Concluindo, esta mensagem indica que o valor do campo *Hop Limit* dos pacotes enviados não é grande o suficiente para o pacote chegar ao destino ou que ocorreu um *Routing* loop. O valor recomendado para o campo *Hop Limit* definido pelo nó emissor é duas vezes o diâmetro da rede, onde o diâmetro é o número máximo de ligações entre os nós mais distantes um do outro.

5.6.3.4 *Parameter Problem*

Uma mensagem ICMPv6 *Parameter Problem* ocorre quando existe um erro no cabeçalho IPv6 ou num cabeçalho de estensão prevenindo processamento adicional. A figura seguinte mostra a estrutura destas mensagens:

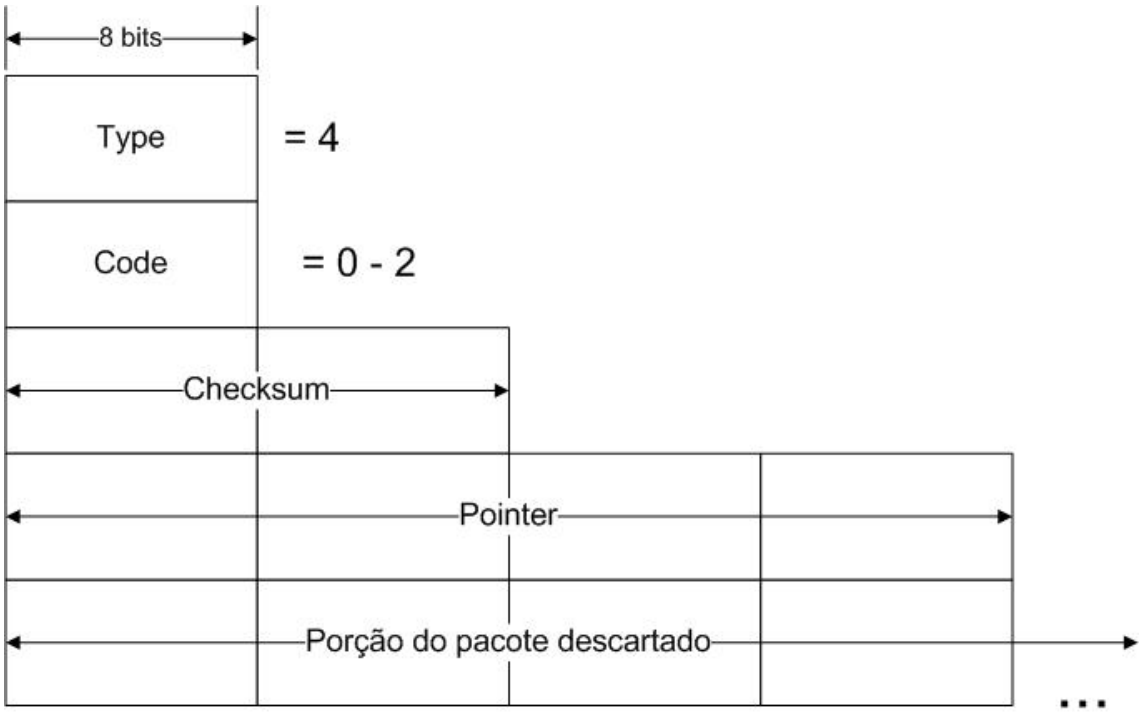


Figura 5.5 – Formato de uma mensagem ICMPv6 *Parameter Problem.*

Nestas mensagens, o campo *Type* está definido a 4 e o campo *Code* tem um valor entre 0 e 4. O campo *Pointer* indica o *byte offset* (iniciado a 0) do pacote IPv6 ao qual o erro foi encontrado. Este campo é definido com o offset correcto mesmo quando a localização do erro não está dentro da parte do pacote descartado.

A tabela seguinte mostra os valores e respectivos significados que o campo *Code* pode assumir:

Code	Significado
0	Campo do cabeçalho com erro
1	*Next Header Type* com erro
2	Opção IPv6 não reconhecida

Tabela 5.2 - Descrição de cada valor do campo *Code* de uma mensagem ICMPv6 com erros.

[52]

5.6.4 Mensagens de Informação

As mensagens de informação ICMPv6 fornecem capacidades de diagnóstico simples para ajuda em troubleshooting e são compostas por:

- *Echo Request*

- *Echo Reply*

Nota: Para ND e MLD existem mensagens ICMPv6 adicionais.

5.6.4.1 *Echo Request*

Uma mensagem ICMPv6 *Echo Request* é enviada para um destino para solicitar uma mensagem de *Echo Reply* imediata.

A seguinte figura mostra a estrutura destas mensagens:

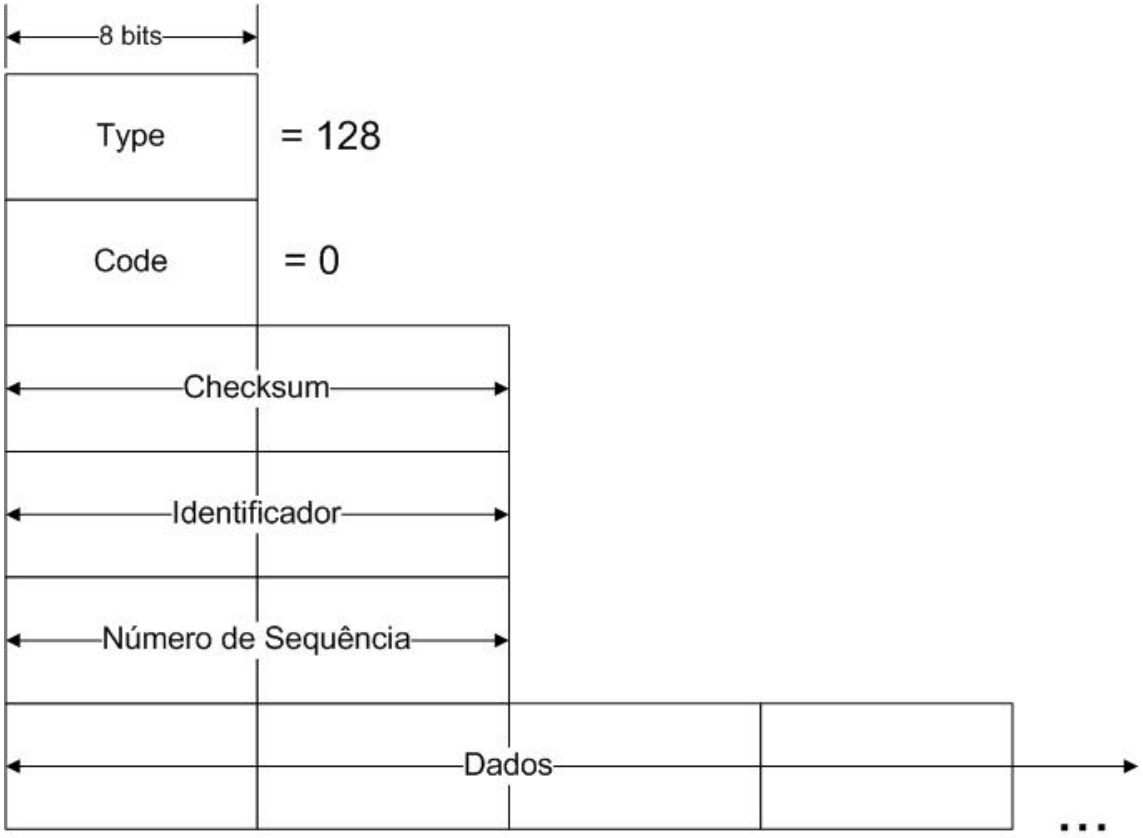


Figura 5.6 – Formato de uma mensagem ICMPv6 *Echo Request.*

Numa mensagem *Echo Request*, o campo *Type* tem o valor 128 e o campo *Code* o valor 0. Os valores dos campos Identificador e Número de Sequência são atribuídos pelas máquinas de origem para que estes possam corresponder às mensagens *Echo Reply* recebidas. Finalmente o campo Dados pode assumir o valor 0 ou mais bytes de dados que são também definidos pela máquina de origem.

As mensagens *Echo Request* podem ser enviadas para um endereço *Multicast*. Cada mensagem *Echo Request* enviada de um endereço *Unicast* atribuído a uma *interface* para um endereço *Multicast* deve ser correspondida com uma mensagem de *Echo Reply*.

5.6.4.2 Echo Reply

Uma mensagem ICMPv6 *Echo Reply* é enviada para responder a uma mensagem ICMPv6 *Echo Request*.

A seguinte figura mostra a estrutura destas mensagens:

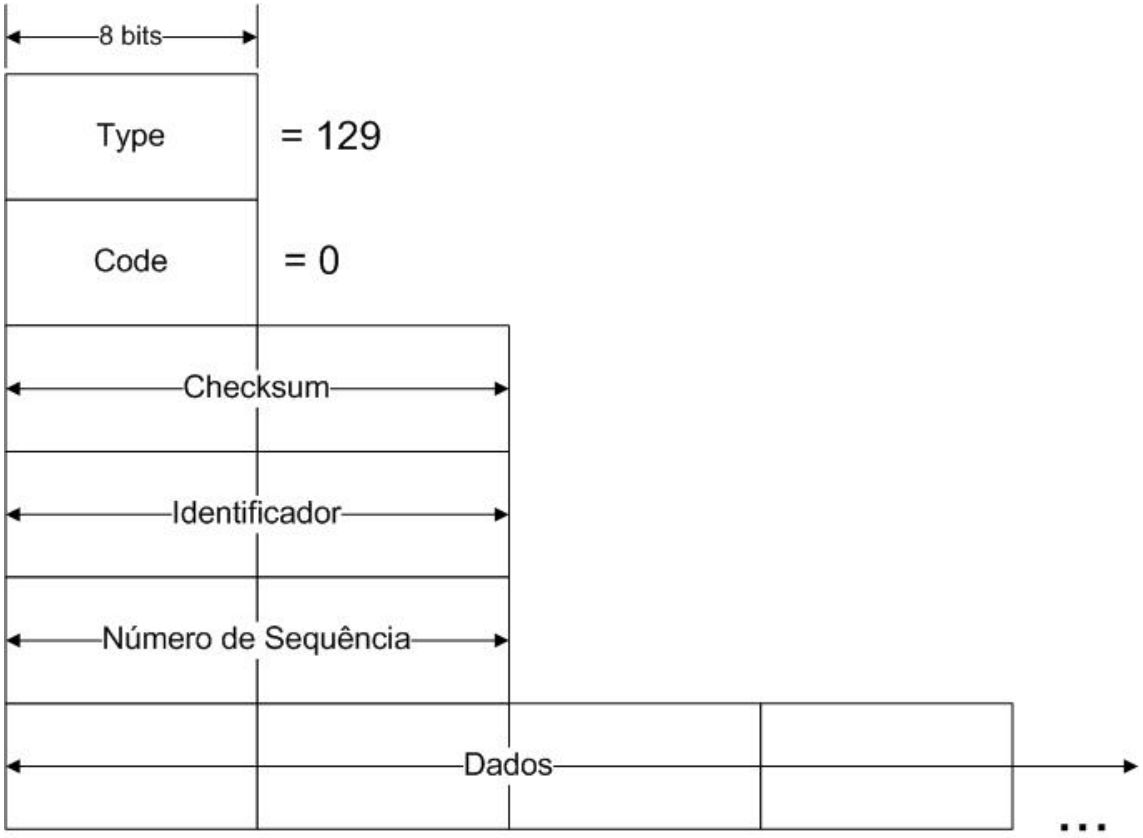


Figura 5.7 – Formato de uma mensagem ICMPv6 *Echo Reply*.

Numa mensagem *Echo Reply*, o campo *Type* tem o valor 129 e o campo *Code* o valor 0. Os campos Identificador, Número de Sequência e Dados são definidos com os mesmos valores que os da mensagem *Echo Request* correspondente.

5.7 Integração de DHCPv4 e DHCPv6

As implementações de clientes DHCPv4 usam um DHCP *Unique Identifier* (DUID) tal como está especificado na RFC sobre DHCPv6 [17]. O DUID é encapsulado numa opção que identifica o cliente DHCPv4.

Esta modificação foi feita para que, ao fazer a transição de IPv4 para IPv6, existam dispositivos de rede que necessitem de usar quer DHCPv6 quer DHCPv4. Os utilizadores destes dispositivos de rede têm assim uma estável, fácil e consistente ligação à rede informática seja qual for a versão do protocolo DHCP que estejam a usar no momento. Obviamente que as actualizações do DNS feitas pelo servidor DHCP de apoio ao cliente serão tratadas mais eficazmente.

Assim, para que tal implementação seja possível, é necessário modificar as RFC's 2131 (DHCP - [15]) e 2132 (*DHCP Options* and *BOOTP Vendor Extensions* - [16]).

5.7.1 Problemas

Podem ocorrer quatro problemas com esta integração de serviços:

- A identidade do cliente é passageira
- Os clientes podem acidentalmente ter múltiplos identificadores
- Os identificadores das RFC's 2131 (DHCP), 2132 (*DHCP Options* and *BOOTP Vendor Extensions*) e 3315 (DHCPv6) são incompatíveis.
- A RFC 2131 (DHCP) não precisa de um identificador de cliente.

5.7.1.1 Identidade do cliente é passageira

A RFC 2132 ([16]) recomenda que cada identificador de cliente seja gerado utilizando o endereço permanente da camada de ligação da *interface* de rede que o cliente está a tentar configurar. Um dos resultados desta recomendação é a de que, quando uma *interface* de rede de um computador de um cliente é substituída, a identidade do cliente é alterada. O cliente perde o seu endereço IP e quaisquer outros recursos associados ao seu identificador antigo.

5.7.1.2 Múltiplos Identificadores

Considerando um cliente DHCPv4 que tenha duas *interface*s de rede: uma ligada à rede por cabo e outra ligada à rede sem fios. O cliente DHCPv4 será bem sucedido ao configurar uma ou ambas as *interface*s de rede. De acordo com a especificação corrente, cada *interface* de rede receberá um endereço IP diferente. O servidor DHCPv4 trata cada *interface* de rede como se fosse um cliente DHCPv4 completamente independente.

No entanto, quando o cliente apresenta informação que precisa ser actualizada, (tal como acontece no DNS) o nome que é apresentado será o mesmo em ambas as *interface*s, mas o identificador apresentado será diferente. Isto faz com que uma das *interface*s ficará com o nome e irá mantê-lo até que este seja válido, mesmo que perca a ligação à rede, enquanto que a outra *interface* de rede nunca obterá o nome.

Em alguns casos, isto é o resultado desejado: quando somente uma *interface* de rede está ligada, o endereço IP é publicado de vez em quando; quando a *interface* de rede ligada à rede não é aquela que está publicada. Quando existem duas *interface*s de rede, às vezes a *interface* correcta é que é publicada, outras vezes não.

Esta situação é provável de ocorrer com os computadores portáteis modernos que normalmente têm uma placa *Ethernet* wireless e uma placa *Ethernet* por cabo. Quando um utilizador tem a possibilidade de se ligar por cabo, este poderá querer mais velocidade e privacidade fornecida pela ligação por cabo. Mas o mesmo utilizador pode desligar a ligação por cabo e continuar ligado a rede pela rede wireless. Quando ocorre uma transição deste tipo, dentro do esquema corrente, se o endereço da *interface* por cabo é aquela que é publicada, este cliente será visto pelos computadores que se tentem

ligar como se tivesse conectividade intermitente, apesar da conectividade à rede seja contínua através da porta wireless.

Outro caso comum de um identificador duplicado ocorre quando um monitor de boot (por exemplo Pré-Boot Execution Environment (PXE)) especifica um identificador para um cliente DHCP, e de seguida o sistema operativo carregado pelo boot loader especifica um identificador diferente.

5.7.1.3 Identificadores Incompatíveis

A opção Identificador de Cliente é usada pelos clientes e servidores DHCPv4 para identificar clientes. Nalguns casos, o valor da opção identificador de cliente é usada para mediar o acesso a recursos (por exemplo, o nome de domínio do cliente publicado através do servidor DHCP). As RFC's 2132 ([16]) e 3315 ([17]) especificam métodos diferentes para derivar os identificadores de clientes. Estes métodos garantem que os identificadores de DHCPv4 e de DHCPv6 serão diferentes. Isto significa que a mediação de acesso a recursos usando estes identificadores não irá funcionar correctamente em casos onde um nó poderá ser configurado usando DHCPv4 e noutros em DHCPv6.

5.7.1.4 DHCPv4 não necessita de um identificador de cliente

A RFC 2131 (DHCP - [15]) permite que um servidor DHCPv4 identifique clientes quer use a opção identificador de cliente (enviada pelo cliente) quer use somente o endereço de rede (caso o identificador de cliente não seja enviado). Assim o formato do identificador de cliente recomendado pela RFC 2131 ([15]) tem problemas tal como foi explicado em 5.7.1.2 e 5.7.1.3.

5.7.2 Soluções

Para resolver os problemas apresentados na secção 5.7.1, os identificadores de cliente DHCPv4 têm de ter as seguintes características:

- Têm de ser persistentes, de modo a que um identificador de cliente não possa ser alterado pelo simples facto de a placa de rede do computador seja adicionada ou removida.

- Tem de ser possível para o cliente ser representado ele próprio como tendo mais que uma identidade na rede, por exemplo, de modo a que um cliente com duas *interface*s de rede possa comunicar ao servidor DHCPv4 que essas duas *interface*s de rede têm de receber dois endereços IP distintos, mesmo que estejam ligadas na mesma ligação.

- Em situações em que o cliente DHCPv4 tenha associado mais que uma identidade na rede simultâneamente, nunca pode ser possível o servidor

DHCPv4 determinar que os dois identificadores de rede pertencem ao mesmo computador.

- Nalgumas situações que possa ser desejável para um cliente DHCP ter a mesma identidade em duas *interface*s de rede, de modo a que se ambas estiverem ligadas à mesma rede, ambas recebam o mesmo endereço IP. Nestes casos, tem de ser possível para o cliente usar exactamente o mesmo identificador para cada *interface*.

- Os servidores DHCPv4 que não estejam de acordo com esta especificação, mas que estão de acordo com a especificação mais antiga de identificadores de clientes, têm de lidar correctamente com os identificadores de clientes enviados pelos clientes que estão de acordo com esta nova especificação.

- Os servidores DHCPv4 que estão de acordo com esta nova especificação têm de lidar correctamente com os clientes DHCPv4 que não utilizem esta nova especificação, excepto quando, ao configurar estes clientes, ocorram comportamentos idênticos aos descritos em 5.7.1.

- O uso por parte dos clientes DHCPv4 do campo *chaddr* dos pacotes DHCPv4 como identificador tem de ser descontinuado.

- Os identificadores de cliente DHCPv4 utilizados por máquinas com pilha dupla têm de usar a mesma string que identifica a máquina quer seja em DHCPv4 como em DHCPv6. Por exemplo, um servidor DHCPv4 que use a identidade de um cliente para actualizar o DNS para suporte a um cliente DHCPv4, tem de registar a mesma identidade do cliente no DNS que será registado no servidor DHCPv6 para suporte do cliente DHCPv6 que está a correr nessa máquina e vice-versa.

Para que todos os pontos (excluindo o último) sejam cumpridos, é necessário construir um identificador de cliente DHCPv4 composto por duas partes: uma parte que tem de ser única para a máquina na qual o cliente está a correr; outra parte que tem de ser única para a identidade da rede apresentada. O DHCP *Unique Identifier* (DUID) e a *Identity Association Identifier* (IAID) satisfazem estas necessidades.

Para que o último ponto seja cumprido, é necessário usar o DUID para identificar o cliente DHCPv4.

Assim, juntando todas as necessidades, o DUID e a IAID descritos na RFC 3315 ([17]) são a única solução possível.

Ao cumprir estas regras, um cliente concordante com esta nova especificação de DHCPv4 irá interagir correctamente com ambos os servidores DHCPv4 (concordantes e não-concordantes com a nova especificação). Quer o servidor quer o cliente sejam não-concordantes, a integração do protocolo DHCPv4 com DHCPv6 não é possível embora não haja perda de funcionalidade.

5.7.3 Alterações à RFC 2131 – DHCPv4

Deixa de ser possível que o identificador de cliente possa conter o endereço de hardware idêntico ao conteúdo do campo *chaddr* ou possa conter outro tipo de identificador como por exemplo o nome DNS.

A RFC 2131 ([15]) afirma que um cliente pode escolher explicitamente fornecer o identificador através da opção identificador de cliente. Se o cliente fornecer o identificador de cliente, este tem de usar o mesmo identificador em todas as mensagens seguintes e o servidor tem de usar esse identificador para identificar o cliente. Se o cliente não fornecer a opção identificador de cliente, o servidor tem de usar o conteúdo do campo *chaddr* para identificar o cliente. Este conceito tem de ser alterado para: um cliente tem de fornecer explicitamente um identificador de cliente através da opção identificador de cliente. O cliente tem de usar o mesmo identificador para todas as mensagens.

Esta RFC também afirma que o uso do campo *chaddr* como único identificador único pode causar resultados inesperados pois esse identificador pode estar associado a uma *interface* de rede que poderá ser movida para um novo cliente. Alguns sites podem escolher usar o número de série do construtor (MAC *address*) como identificador de cliente para evitar alterações inesperadas no endereço de rede do cliente devido à mudança de *interface*s de rede entre computadores. Os sites podem também escolher se usam um nome DNS como identificador de cliente, fazendo com que o endereço fique associado ao nome DNS em vez de ficar associado a um hardware específico. Tudo isto agora deve ser alterado para: um cliente DHCP não pode contar com o campo *chaddr* para o identificar.

Deixa de ser opcional que um cliente possa incluir um identificador único, ou seja, o cliente terá de ter um identificador único.

Estas alterações não libertam os servidores DHCPv4 da obrigação de usar o campo *chaddr* como identificador se o cliente não enviar a opção identificador de cliente. Preferencialmente eles obrigam os clientes a enviar a opção identificador de cliente e a não contar com o campo *chaddr* para identificação. Os servidores DHCPv4 têm de usar o campo *chaddr* como identificador quando o identificador de cliente não é enviado, de modo a que possa suportar clientes antigos que não obedeçam a estes novos critérios.

5.7.4 Alterações à RFC 2132 – *DHCP Options* and *BOOTP Vendor Extensions*

Esta RFC indica que o identificador de cliente consiste num identificador que garante exclusividade. Nesta nova versão do protocolo DHCP o identificador de cliente consiste num tipo de campo cujo valor é normalmente 255, seguido de um campo de 4 bytes IAID, seguido por sua vez o DUID para o cliente.

Deixa tambémde existir um tamanho mínimo de 2 bytes na opção que especifica uma tabela com os tamanhos MTU que são usados na execução do Path MTU Discovery.[18]

6 Pilha protocolar

Com a chegada do protocolo IPv6 os novos dispositivos passaram a utilizar uma Pilha protocolar TCP/IP dupla, em que a camada de rede suporta os dois tipos de endereços (IPv4 e IPv6).

Pilha TCP/IP

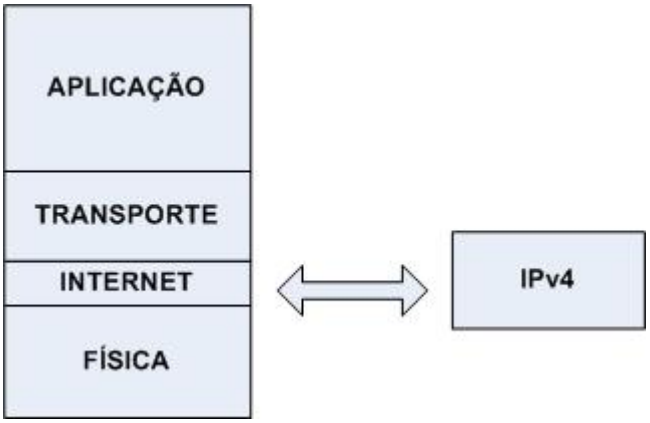


Figura 6.1 – Esquema da pilha TCP/IP IPv4.

Pilha TCP/IP dupla

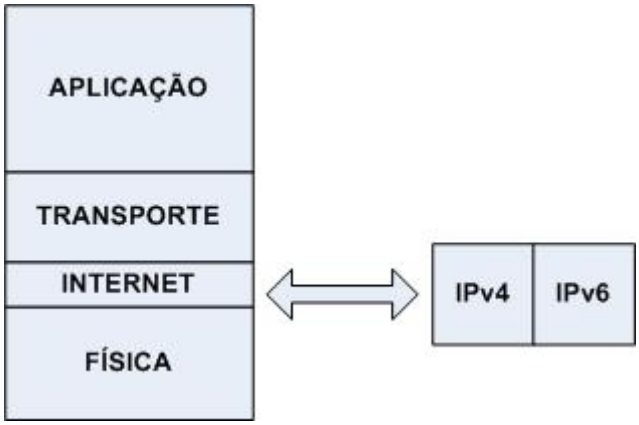


Figura 6.2 – Esquema da pilha TCP/IP IPv4 e IPv6 (pilha dupla).

Funcionamento da pilha TCP/IP em tunneling

O esquema seguinte pretende mostrar o funcionamento da pilha quando existem mecanismos de túneis. Como se pode ver pelas imagens apresentadas a pilha dupla consiste basicamente no suporte dos dois Protocolos (IPv6 e IPv4) na camada de rede.

Figura 6.3 – Esquema da pilha TCP/IP num túnel.

6.1 *Dual Stack Transition Mechanism* (DSTM)

O DSTM foi desenvolvido para permitir os operadores de desenvolver uma infra-estrutura de encaminhamento. Por outro lado, as máquinas podem ter suporte IPv4 e IPv6 activo para lhes permitir comunicar com qualquer outra máquina na *Internet* sem a necessidade de tradução de endereços IP.

A operação DSTM é bastante simples de efectuar e pode ser vista como o oposto do mecanismo ISATAP.

DSTM apresenta um *router* com pilha dupla que age como um *Tunnel End Point* (TEP) ligado na periferia entre uma rede IPv4 e uma rede IPv6. A abrangência do TEP é tipicamente uma rede pequena ou média. Este mecanismo assume que a rede entre as máquinas e o TEP é IPv6-*only*.

Quando os nós tentam comunicar com outros nós IPv6, a sua comunicação trabalha usando os padrões e encaminhamento IPv6. Quando uma máquina DSTM tenta comunicar com uma máquina IPv4, pede um endereço IPv4 de um servidor (ex. DHCPv6). A máquina fornece esse endereço ás aplicações que começam a comunicação normal utilizando a pilha protocolar IPv4. Os pacotes de saída são encapsulados em cabeçalhos IPv6 e enviados por um túnel até ao TEP, que por sua vez são desencapsulados e encaminhados para a rede IPv4. Os pacotes de chegada (da máquina IPv4), por sua vez, são enviados pelo mesmo túnel até à máquina IPv6 que definiu o endereço IPv4 no campo *destination* do cabeçalho IPv4.

Este mecanismo assume que o TEP oculta o endereço IPv4 correspondente a cada endereço IPv6 quando um pacote é recebido. Deste modo, o TEP é capaz de encontrar o endereço IPv6 correcto depois de receber um pacote IPv4.

No entanto, numa rede pública, alguém malicioso pode facilmente alterar a cache do TEP enviando um pacote falso contendo um endereço IPv4 (no endereço de origem do cabeçalho encapsulado) que está atribuído a outro dispositivo, fazendo com que o TEP altere a associação entre o endereço IPv4 e o endereço correspondente IPv6.

7 Suporte IPv6 nas Aplicações e Sistemas Operativos

As pilhas protocolares de IPv4 e IPv6 vão ter de existir ambas num nó durante um longo período de tempo.

No caso das aplicações, estas devem ser capazes de lidar com ambas as versões do protocolo IP durante um longo período de tempo também. Um sistema operativo com pilha dupla não tem intenção de ter aplicações IPv4 e aplicações IPv6.

No entanto as aplicações IPv6 podem ser independentes da pilha protocolar de um nó.

As aplicações capazes de lidar com IPv6 e IPv4 têm de trabalhar correctamente em nós IPv4.

Durante o período de transição das aplicações, os administradores de sistema podem ter várias versões da mesma aplicação. Assim, os utilizadores têm dificuldade na selecção da versão correcta que suporte a versão IP requerida.

Para evitar este tipo de problemas, deve-se ter aplicações híbridas que suportem ambos os protocolos (IPv4 e IPv6).

Existem vários cenários de transição nas aplicações:

- Aplicações IPv4 num nó com pilha dupla
- Aplicações IPv6 num nó com pilha dupla
- Aplicações IPv4/IPv6 num nó com pilha dupla
- Aplicações IPv4/IPv6 num nó IPv4

7.1 Aplicações IPv4 num nó com pilha dupla

Neste cenário, o protocolo IPv6 é adicionado ao nó mas as aplicações IPv6 ainda não estão disponíveis ou instaladas. No entanto o nó implementa pilha dupla. As aplicações IPv4 só conseguem lidar com ligações IPv4 e aceitar/estabelecer ligações de/para nós que implementem uma pilha IPv4.

Para permitir que uma aplicação comunique com outros nós que usam IPv6, a primeira prioridade é alterar a aplicação para suporte IPv6. Existem outras soluções que utilizam o mecanismo Bump-In-the-Stack (BIS) ou o mecanismo Bump-In-the-API (BIA) mas o seu uso não é recomendado pois têm alguns problemas associados.

Ambos os mecanismos BIS e BIA têm as habituais devantagens associadas com a tradução de endereços: endereços imbutidos noutros endereços causam problemas e o log de endereços pode ser erróneo.

No entanto também têm algumas vantagens relativamente ao NAT: a tarefa de traduzir os endereços das máquinas fica a cargo de cada máquina, melhorando a sua escalabilidade.

7.1.1 Bump in the Stack (BIS)

Bump in the Stack é basicamente uma variante do mecanismo SIIT. No entanto a sua motivação é ligeiramente diferente. Suponhamos que temos um *software* que queremos que funcione sobre IPv6, mas que não existe nenhuma versão desse *software* que funcione sobre IPv6. BIS é um mecanismo que faz com que este tipo de *software* possa ter conectividade IPv6.

Exemplo:

Se o *software* descrito anteriormente tenta consultar por exemplo o site www.qualquercoisa.com que tem como endereço 2001:db8::abcd. Ao tentar fazer essa consulta, o componente BIS responsável por mapear os endereços, utiliza um endereço IPv4 de uma pool de endereços configurada por este (por exemplo 192.168.1.1) para representar a máquina em questão e retorna este IPv4 para o *software*. Deste modo o *software* usa este endereço normalmente. Deste modo, o mecanismo BIS intercepta os pacotes vindos da *stack* IPv4 que têm como destino o endereço 192.168.1.1 e usa o mecanismo SIIT para reescrevê-los como pacotes IPv6 que têm como destino o endereço 2001:db8::abcd. Os pacotes que vêm na direcção oposta são traduzidos do mesmo modo.

7.1.2 Bump in the API (BIA)

Bump in the API é uma variante do BIS. Opera de um modo similar ao BIS: um mapeador de endereços intercepta pacotes e retorna um endereço IPv4 falso para máquinas IPv6. A aplicação utiliza o endereço IPv4 normalmente.

No entanto, a diferença do BIA em relação ao BIS é relativa às funções da livraria do BIA *connect*, *bind* e *getpeername* terem conhecimento destes endereços falsos traduzindo-os de/para endereços IPv6.

BIA tem a vantagem relativamente ao BIS de a tradução de endereços ser simplificada pois o cabeçalho IP não é traduzido.

7.2 Aplicações IPv6 num nó com pilha dupla

Tal como já foi referido no ponto anterior, as aplicações devem ser convertidas para suportar IPv6. A maneira mais fácil para essa conversão é substituir as referências para as API's IPv4 antigas com as novas API's com mapeamento um-para-um. Assim a aplicação é compatível somente com IPv6 e não pode funcionar em nós somente compatíveis com IPv4. Assim é necessário ter duas versões da aplicação: uma compatível só com IPv6 e outra compatível somente com IPv4. Este caso não é desejável pois assim é necessário gerir ambas as aplicações em vez de uma e é necessário também escolher qual a versão a usar em cada máquina.

A maioria das aplicações de pilha dupla permite que aplicações IPv6 comuniquem com nós IPv4 e IPv6. Os pacotes IPv4 que têm como destino aplicações IPv6 chegam a este pois o seu endereço é mapeado da seguinte forma: o endereço IPv6 ::FFFF:x.y.z.w representa o endereço IPv4 x.y.z.w. A seguinte figura esquematiza o funcionamento destas aplicações.

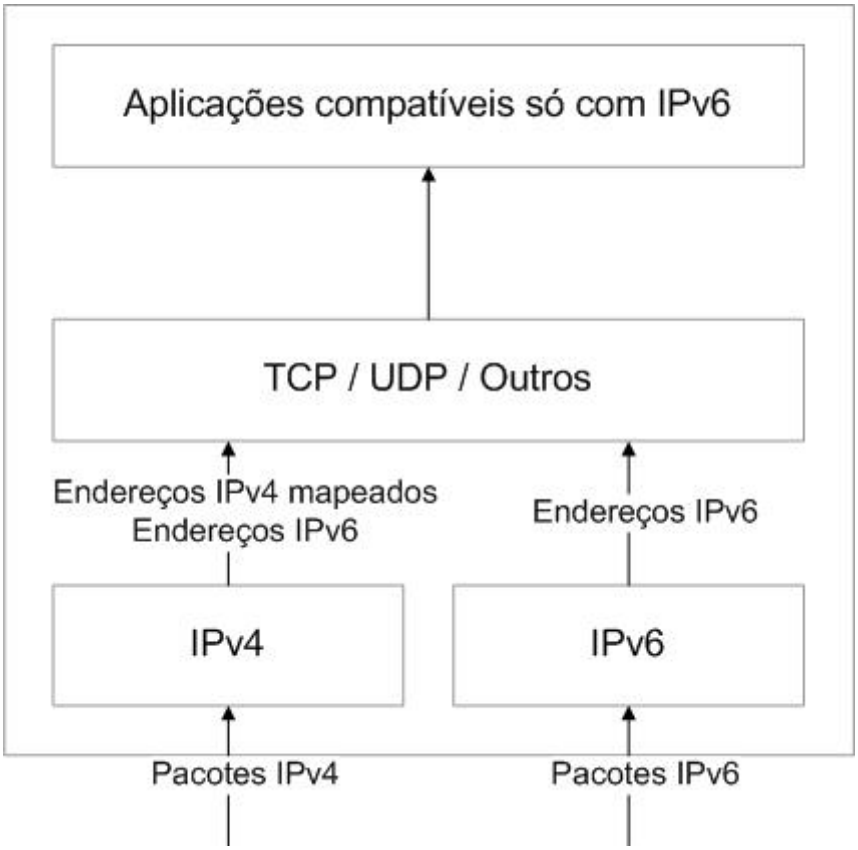


Figura 7.1 – Esquema do funcionamento das aplicações IPv4 e IPv6.

É necessário considerar dois casos distintos quando se desenvolve uma aplicação que suporte ambos os protocolos IP:

- Verificar se a aplicação pode (ou deve) suportar ambos os protocolos através de endereços IPv4 mapeados ou se a aplicação deve suportar ambos os protocolos separadamente.
- Verificar se os sistemas aos quais as aplicações são usadas suportam IPv6.

7.3 Aplicações IPv4 e IPv6 num nó com pilha dupla

Esta transição é a mais aconselhável. Durante o período de transição para IPv6, as aplicações que suportam ambos os protocolos IPv4 e IPv6 devem ser capazes de comunicar com outras aplicações, independentemente da versão da pilha protocolar ou da aplicação no nó.

As aplicações preferem IPv6 por defeito se o nó remoto e respectiva aplicação suportarem IPv6. No entanto, se a comunicação IPv6 falhar, as aplicações independentes da versão IP tentam automaticamente uma comunicação em IPv4.

7.4 Aplicações IPv4 / IPv6 num nó IPv4

O caso mais importante neste ponto é o suporte da aplicação em sistemas onde o suporte IPv6 pode ser ligado ou desligado dinamicamente pelos utilizadores. Num sistema destes a aplicação deve ser capaz de lidar com a situação do suporte IPv6 estar desligado. Outro cenário é quando uma aplicação é instalada em sistemas antigos que não suportam IPv6. Neste caso a aplicação deve ser desenvolvida julgando caso a caso para verificar se faz sentido ter suporte para este tipo de sistemas.

7.5 Considerações sobre a migração das aplicações para IPv6

As modificações mínimas para que aplicações IPv4 funcionem com IPv6 são baseadas no tamanho e formato diferentes de endereços IPv4 e IPv6.

A seguinte lista sumariza as dependências das versões IP mais comuns nas aplicações:

- Formato de apresentação de um endereço IP: String ASCII que representa o endereço IP, string "*dotted-decimal*" para IPv4 e string hexadecimal para IPv6.
- Camada de transporte da API: funções para estabelecer e trocar comunicação.
- Resolução de nomes e de endereços: funções para fazer a conversão de nomes e de endereços IP.
- Dependências IP especificas: selecção do endereço IP, enquadramento da aplicação e armazenamento de endereços IP.
- Aplicações *Multicast*: um endereço tem de encontrar os endereços IPv6 equivalentes para endereços *Multicast* IPv4.[5]

8 Tipos de nós

- **Nó IPv4 (*IPv4 only node*) –** nó que tem apenas IPv4 implementado. Só pode comunicar com outros nós IPv4. A maioria dos routers e anfitriões actuais são *IPv4 only node*.

- **Nó IPv6 (*IPv6 only node*) –** nó que tem apenas IPv6 implementado. Só pode comunicar com outros nós IPv6. Não são actualmente a maioria existente, no entanto cada vez vão aparecendo mais nós deste tipo. Com o desenvolvimento da tecnologia relacionada com a mobilidade (telemóveis PDA etc.) espera-se que estes nós comecem a afirmar-se face aos nós IPv4.

- **Nó IPv6/IPv4 –** nó que tem suporte para os dois protocolo (IPv4 e IPv6). Pode comunicar com nos IPv4 ou IPv6

Para que a coexistência possa ocorrer os nós sejam IPv4 ou IPv6 irão comunicar através de uma infra-estrutura IPv4. Na fase actual podemos considerar que a infra-estrutura é uma mistura de IPv4 e IPv6. Esta combinação irá manter-se por muitos anos ainda até que a esmagadora maioria de nós e anfitriões usem o Protocolo IPv6 como protocolo principal.

9 Túneis

Os túneis podem ser feitos no formato *end-to-end* (em que o ponto de entrada do túnel é uma máquina que envia pacotes e o ponto de saída do túnel o destino final), ou entre uma máquina e um router.

Independentemente do ponto de entrada e de saída dos túneis, ambos os pontos têm de ter um endereço IPv4 e um endereço IPv6. Assim, em qualquer caso, o ponto de entrada e de saída dos túneis têm de ter suporte IPv4 e IPv6.

Os mecanismos de *tunneling* funcionam da seguinte forma:

- Encapsulam os pacotes IPv6 em pacotes IPv4 e vice-versa, o que significa que também podem ser usados para ligações IPv4 sobre redes nativas IPv6.
- O valor do campo *Protocol* do cabeçalho IPv4 é 41.
- A extremidade de um túnel pede a retransmissão dos pacotes fragmentados, encaminha os pacotes para uma rede IPv6 e o campo *Hop Limit* (correspondente ao campo TTL em IPv4) é reduzido para 1, ou seja, o túnel é "transparente" para o IPv6.
- Os nós que executam o encapsulamento e o desencapsulamento têm de ser nós com pilha dupla.

9.1 Tipos de túneis

Existem dois mecanismos de túneis:

- Túneis manualmente configurados
- Túneis automáticos

Os túneis automáticos dividem-se nos seguintes tipos:

- 6to4
- ISATAP
- GRE
- 6over4

Tipo	Apropriados para:	Notas
Manual	Ligações ponto a ponto simples, podem ser usados em intranet como em ligações com locais exteriores à rede	Apenas pode transportar pacotes IPv6

6to4	Ligação ponto a multiponto (*point-to-multipoint*), são usados para efectuar a conectividade entre locais IPv6 isolados.	Os IP têm o prefixo 2002:::/16
6over4	Ligação ponto a multiponto (*point-to-multipoint*), são usados para efectuar a conectividade entre locais IPv6 isolados.	Este tipo de túnel está obsoleto sendo substituído pelos túneis 6to4
ISATAP	Ligação ponto a multiponto (*point-to-multipoint*), são usados para efectuar ligação entre locais IPv6 dentro de uma intranet	Podem ser usados endereços IPv6 *Unicast*
GRE	Ligações ponto a ponto simples, podem ser usados em ligações intranet e em ligações com locais exteriores à rede.	Podem transportar vários tipos de pacotes e *Connectionless Network Service* (CLNS[16])
Teredo	Usado para permitir conectividade ao *Backbone* Ipv6 a máquinas que se encontrem por trás de mecanismos NAT	Os routers Cisco não suportam este mecanismo

Tabela 9.1 – Aspectos dos vários tipos de túneis existentes.

9.2 Túneis configurados manualmente

Os túneis manualmente configurados são simples de usar e eficazes.

Nestes túneis os endereços IPv6 e IPv4 são manualmente configurados (atribuídos) nas *interface*s dos routers destino e origem.

Este tipo de configuração é usada quando se pretende uma conectividade permanente ou dedicada e requer configuração específica em cada extremidade do túnel. Como é necessária configuração nos routers em cada ponta do túnel requer um bom conhecimento acerca da topologia da rede. Este tipo de configuração ponto a ponto fornece uma segurança bastante robusta a tráfego injectado no troço.

Túneis manuais são usados em ligações *Router-to-Router* e *Host-to-Router*.

No entanto, é claro que este mecanismo pode ter problemas de escalabilidade quando se considera um vasto número de locais que precisem de comunicar uns com os outros. Supondo que, por exemplo, 1000 locais pretendem comunicar uns com os outros. Cada local tem de ter configurado um túnel com 999 pontos de saída para cada prefixo IPv6. Se um ponto de saída do túnel falhar, é necessário altera-lo nos outros 999 locais. Desta forma, os túneis automáticos são mais robustos a falhas na rede.

[16] tuneis GRE sobre redes CLNS activam os túneis cTunnels (Cisco CLNS Tunnels) para operarem com equipamento de rede de outros fabricantes.

9.3 Túneis automáticos

Nos túneis automáticos os endereços IPv6 das extremidades do túnel são configurados (atribuídos) com base nos endereços IPv4. Os endereços IPv6 derivam dos endereços IPv4 e são atribuídos aos routers automaticamente sem necessidade de configurações manuais. A maioria dos túneis automáticos foi planeada para coexistir durante muito tempo com o IPv4 e IPv6 de modo a garantir uma migração simples e eficaz para o protocolo IPv6.

9.3.1 Túneis 6to4

O objectivo dos túneis 6to4 é o de permitir a comunicação de vários domínios IPv6 (ilhas IPv6) sobre um *Backbone* IPv4 sem ser necessário configurar o túnel nos routers de fronteira situados nesses domínios IPv6. Desta forma, este tipo de túneis foram desenvolvidos para permitir que os routers de fronteira descubram automaticamente o endereço IPv4 da outra extremidade do túnel.

Os túneis 6to4 estabelecem o endereço das extremidades do túnel com base no endereço IPv4. O principal objectivo e propósito dos túneis 6to4 é o de permitir a comunicação entre máquinas IPv6 sem a configuração explícita de túneis, usando para isso *relay router*s.

Estes endereços são usados em locais onde exista pelo menos um endereço IPv4 público. Para endereços privados são usados endereços ISATAP.

São usados alguns routers denominados *relay router*s (este routers são disponibilizados publicamente na *Internet*) que têm a função de assegurar a um utilizador comum o acesso à rede IPv6.

Estes túneis são usados em ligações entre dois routers em extremidades opostas do túnel, ou entre um router de uma das extremidades e um terminal.[37]

Foi proposto IPv4 *Anycast* para ser possível encaminhar os pacotes para o *relay router* mais próximo de entre os múltiplos *relay router*s existentes na *Internet*. No entanto os operadores também precisam de endereços *Unicast* para identificar cada *relay router*.

No entanto não há garantia de que o *relay router* esteja perto da origem ou do destino, ou seja, a origem pode estar perto do destino enquanto que o *relay router* poderá estar longe de ambos resultando em significativos atrasos. Para além disso qualquer nó pode enviar pacotes encapsulados para o *relay router* com um endereço IPv6 de origem falso (endereço da vítima). O *relay router* por sua vez encaminhará esses pacotes para o seu destino pretendido.

Um endereço IPv6 típico destes túneis é uma combinação do prefixo único 2002::/16 (especificado pela IANA para este propósito) e um endereço único IPv4 de 32 bits.

A especificação 6to4 diz que os 32 bits após 2002::/16 são o endereço IPv4 do *Gateway* da rede em questão. Este foi o método encontrado para que os pacotes consigam encontrar o seu caminho na rede.

Em baixo, está representado o formato de um pacote 6to4:

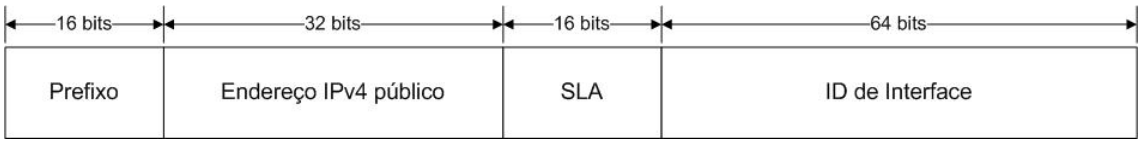


Figura 9.1 – Formato de um pacote 6to4.

Exemplo:

Suponhamos que temos o IP 192.168.2.199 (este IP é usado apenas para exemplo uma vez que não é um endereço usado na *Internet*), o seu prefixo IPv6 seria 2002:c0a8:2c7::/48.

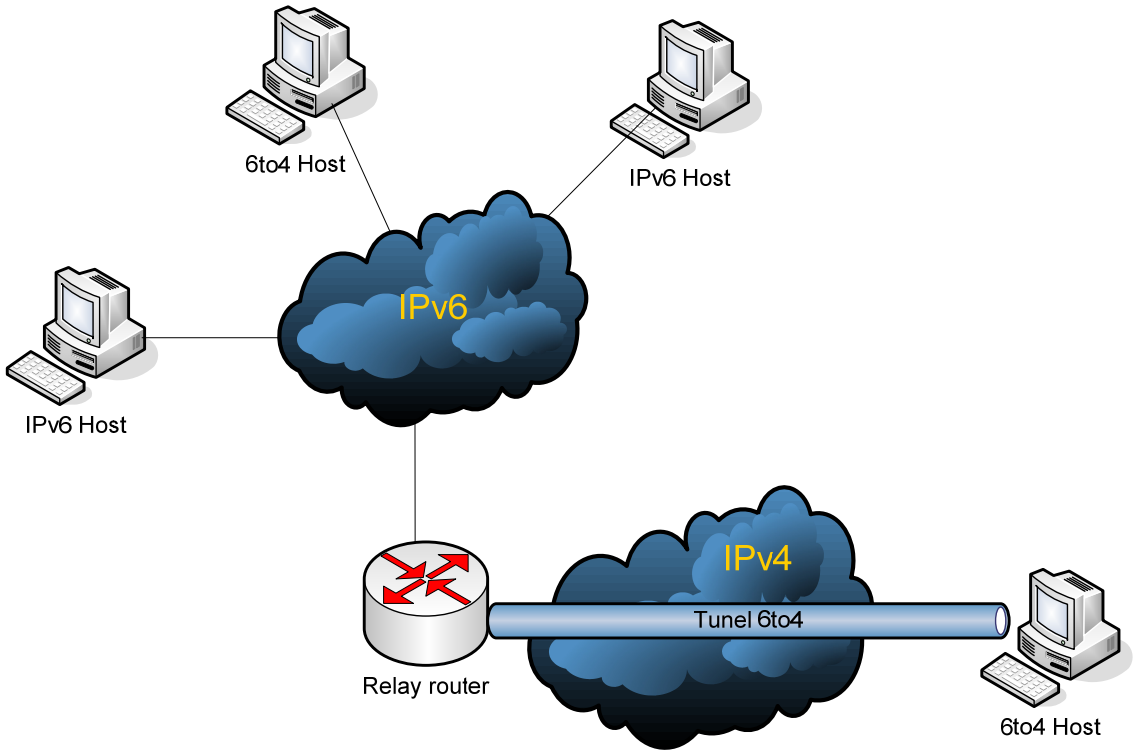


Figura 9.2 – Exemplo da implementação de um túnel 6to4.

Nota: os endereços IPv4 compatíveis com Ipv6 têm um formato diferente dos endereços 6to4. Os endereços IPv4 compatíveis não são usados nos túneis 6to4.

***Relay router*:** router que faz a transição entre endereços 6to4 e endereços IPv6 puros.

Lista dos *relay router*s existentes actualmente:

Global				
Nome	**Localização**	**Largura de banda**	**Contacto**	**Notas**
`2002:c058:6301::`	*Global*	n/a	n/a	Ver RFC-3068. Este é um endereço

				Anycast para o *relay router* mais proximo
América do Norte				
Nome	**Localização**	**Largura de banda**	**Contacto**	**Notas**
6to4.ipv6.*Microsoft*.com	*Redmond, WA? / -*	?	*Microsoft*	
Ipv6-lab-gw.cisco.com	*San Jose / Sprint?*	100 mbps	Cisco	Cisco's IPv6 Page
Asia				
Nome	**Localização**	**Largura de banda**	**Contacto**	**Notas**
6to4.ipv6.aarnet.net.au	*Sydney, Austrália*	100 mbps	AARNET NOC	Apenas para a austrália
kddilab.6to4.jp	*Tokyo, Japan / ?*	100 mbps	kddilab	Open
6to4.ipv6.ascc.net	*Taipei, Taiwan / ?*	100 mbps	Academia Sinica Computing Center	Experimental
Africa				
6to4.ipng.unix.za.net	*Cape Town, South Africa / ?*	48 mbps	Univ.of Cape Town	
Europa				
Nome	**Localização**	**Largura de banda**	**Contacto**	**Notas**
6to4.ipv6.bt.com	*Adastral Park, UK / ?*	10 mbps	Stuart Prevost	
skbys-00-00.6to4.xs26.net	*Banska Bystrica, Slovakia*	34 mbps	Access to Six	
6to4.ipv6.uni-leipzig.de	*Leipzig, Germany*	100 mbps	Uwe	Experimental

6to4.ipv6.fh-regensburg.de	*Regensburg, Germany*	34 mbps	Hubert Feyrer	Experimental

Tabela 9.2 – Características e localização dos *Relay router*s.

Nota: Os endereços privados do tipo 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 ou Automatic Private IP Addressing (APIPA) (169.254.0.0/16) usados no *Windows 98* e *Windows* 2002 não são "*Globally routable*", ou seja, não funcionam com os túneis 6to4.

Na rede da "ESTG-Leiria" os endereços atribuídos na sala de projecto são do tipo 192.168.0.0/16, ao se efectuar o comando "ipv6 if" é possível observar que não é atribuído nenhum ip (resultante do mecanismo 6to4) à *interface* 3 (*interface* dos túneis 6to4). Os túneis 6to4 apenas funcionam com endereços públicos.

Como é descoberto o IP do *relay router*

Para que não seja necessária a configuração manual do endereço do *relay router* foi estabelecido um endereço *Unicast* 192.88.99.1 (2002:c058:6301::) quando convertido para 6to4) para enviar pacotes para os *relay router*s. A rede 192.88.99.0/24 foi alocada com a finalidade de apontar para rotas que apontem para *relay router*s (routers estes que usam o IP *Anycast*) de modo a que se possa escolher qual o melhor *relay router* a adoptar.

9.3.1.1 Segurança

Os routers que implementam este tipo de túnel têm vulnerabilidades:

Os routers 6to4 têm de aceitar os pacotes de todos os *relay router*s 6to4. Não é possível saber se o *relay router* existe ou se é de confiança. Por outro lado os *relay router*s têm de aceitar pacotes dos routers 6to4 e de máquinas IPv6 nativas sem qualquer verificação.

Estes routers são também alvo de potenciais ameaças tais como:

- Ataques de DoS ou DDoS a componentes 6to4 (router ou *relay router*) que podem resultar na não disponibilidade deste.
- Podem ser usados para refletirem ataques de DDoS.
- "Service Theft": utilização não autorizada de serviços dos *relay router*s.
- Ataques locais de *Broadcast* IPv4.
- Ataques de *Neighbor Discovery*.

9.3.1.2 Prevenção de ataques

Poderão ser implementadas várias técnicas simples de prevenção a ataques quer a routers 6to4 quer a *relay router*s 6to4.

Para os routers 6to4 pode-se aplicar as seguintes técnicas de prevenção:

- Verificação da correspondência dos pacotes IPv4 com os pacotes IPv6 encapsulados (2002::/16).
- Não permitir que endereços IPv4 "estranhos" privados, *Multicast*, etc sejam encapsulados.
- Rejeitar endereços IPv6 errados tais como endereços *link* local ou *Multicast*.
- Prevenir o encaminhamento de pacotes para outros sítios 6to4 através dos *relay router*s 6to4.
- Rejeitar pacotes vindos de outro local 6to4 através de um *relay router*.

Para os *relay router*s 6to4 devem ser implementadas as seguintes medidas:

- Rejeição de pacotes IPv4 de routers 6to4 que não tenham o endereço IPv4 de origem correcto ou o seu equivalente endereço IPv6 encapsulado correcto.
- Rejeição de pacotes enviados para a rede IPv6 que não tenha endereços IPv6 universais.
- Rejeição de pacotes de routers 6to4 para endereços 6to4.
- Implementação de filtros e Access Lists para a rede IPv6.

9.3.1.3 Encapsulamento / desencapsulamento

Os pacotes são encapsulados em pacotes IPv4 com o campo *Protocol* = 41.

Ao chegar ao router destino a parte IPv4 é ignorada e apenas a parte IPv6 é analisada, ou seja, o encaminahmento a partir deste ponto é em IPv6, a parte IPv4 não é utilizada para encaminhamento a partir deste ponto.

Nos casos em que um dos extremos do túnel possui tanto um endereço 6to4 como um endereço IPv6 nativo, deve ser usado o endereço 6to4. Casos os dois extremos possuam endereços Ipv6 nativos e endereços 6to4 deve ser feita uma opção em que: ou se usam os endereços nativos ou se usam os endereços 6to4. A configuração por default deve adoptar o uso de endereços nativos.

9.3.1.4 Vantagens

- Túneis suportados pelo IOS da Cisco.
- Configurações simples do lado do cliente.
- Túnel automático. Não é necessário configurar um *relay router*. Estes já estão criados algures no *Backbone* IPv6.
- Túneis facilmente adaptáveis.
- Solução que facilita o endereçamento IP dinâmico de uma empresa.
- Túnel existente somente durante uma sessão.
- Um túnel 6to4 requer somente ser configurado uma vez do lado do ISP, ficando o *relay router* 6to4 disponivel simultâneamente para várias empresas.
- Qualquer falha num router secundário pode ser reparada sem envolver o ISP.

9.3.1.5 Desvantagens

- Comunica somente com outros nós que estão configurados com endereços 6to4. No entanto se for utilizado um *relay router* 6to4 é possível comunicar com nós configurados com endereços IPv6 nativos.
- Conta com a infraestrutura de encaminhamento da rede IPv4. No entanto, o encaminhamento pode não ser tão eficiente quanto a conectividade IPv6 nativa ou túneis configurados.

9.3.2 Túneis 6over4

Os túneis 6over4 fornecem um método trivial para gerar um endereço IPv6 *Link-local* a partir de um endereço IPv4 e um mecanismo para efectuar *Neighbor Discovery* sobre Ipv4.

O endereço *Link-local* é gerado atribuindo aos 32 bits menos significativos do endereço IPv6 o endereço IPv4 do host.

Exemplo:

Se tivermos o seguinte endereço IPv6 fe80:0000:0000:0000:0000:0000:, e o endereço 192.0.2.142, o endereço *Link-local* gerado será o: fe80:0000:0000:0000:0000:0000:c000:028e que corresponde a fe80::c000:028e

NOTA: Este tipo de túnel está obsoleto.

9.3.2.1 Atribuição de um endereço *Multicast*

Para que se possa usar o ICMPv6 *Neighbor Discovery* é necessário o uso de *Multicast*.

Um pacote *Multicast* ipv6 é encapsulado num pacote *Multicast* ipv4 com o destino 239.192.x.y onde o x e o y são o penúltimo e último byte do endereço ipv6 *Multicast* respectivamente.

9.3.2.2 *Neighbor Discovery*

Tendo um endereço *Link-local* e um endereço *Multicast* é possível a um host descobrir a "vizinhança" através do ICMPv6.

Os túneis 6over4 baseiam-se fortemente na disponibilidade *Multicast* do protocolo IPv4, disponibilidade essa que não é tão grande como se desejaria (o *Multicast* é relativamente recente assim como o IPv6), são túneis com limitações principalmente devido a não serem suportados pela maioria dos sistemas operativos actuais.

9.3.3 *Intra-Site Automatic Addressing Protocol* (ISATAP)

É um mecanismo em tudo semelhante ao 6over4 com a excepção de não usar o *Multicast* do protocolo IPv4. Esta particularidade torna-o ligeiramente mais complexo que o mecanismo 6over4 ou 6to4. Apesar de ser um mecanismo de túnel muito

semelhante aos outros, foi concebido para transporte de tráfego IPv6 numa intranet permitindo a comunicação entre máquinas IPv6 através de infra-estruturas IPv4.

Como neste tipo de túnel não é usado *Multicast* é necessário uma RPL (*routers potencial list*). Cada router desta lista é sondado pelo ICMPv6 para se determinar quais estão em funcionamento assim como para efectuar a auto configuração dos endereços *Unicast* (obtendo a lista de prefixos que podem ser usados).[72]

Os endereços ISATAP têm o seguinte formato:

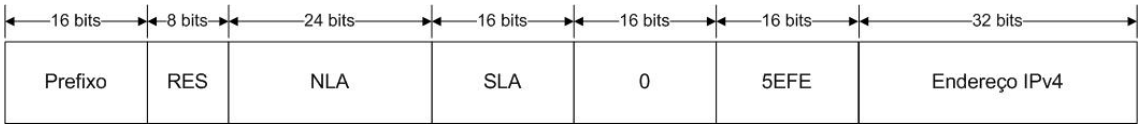


Figura 9.3 – Formato dos endereços ISATAP.

Prefixo é um qualquer prefixo de 64 bits válido para endereços *Unicast* do IPv6.Inclui os endereços locais de ligação (fe80::/64), os prefixos locais de site e prefixos globais.

RES (Reserved) é um campo reservado para posterior uso.

NLA (Next Level Aggregation) é o identificador de uma área (sítio).

SLA (Site Level Aggregation) contém a subrede de uma organização (disponíveis 2^{16} subredes).

0 e 5efe (0000:5EFE) é o identificador de *interface*s de 32 bits Globalmente exclusivo, constituído a partir da combinação do OUI (*Organizational Unit Identifier*) atribuído à IANA (*Internet* Assigned Numbers Authority) (00-00-5e) e um tipo que indica um endereço incorporado do IPv4 (FE).

Endereço IPv4 corresponde ao endereço *Unicast* IPv4 atribuído. Este campo é tratado como um endereço IPv4.

Os endereços ISATAP são apropriados para a ligação de hosts IPv6 intra-site, ou seja hosts IPv6 que se encontrem numa intranet IPv4. Para ligações com o exterior desta rede são usados túneis e endereços 6to4. Considera a rede IPv4 existente como uma *link* layer (ligação de dados) para o IPv6 e olha para os outros nós como potenciais hosts/routers IPv6.[9]

O ISATAP activa o túnel automático quer se usem endereços IPv4 privados ou globais.

9.3.3.1 Encapsulamento e desencapsulamento

Quando um nó ISATAP recebe um datagrama IPv4 com o campo protocolo=41 que não pertence a uma *interface* configurada para túnel, verifica se o endereço IPv4 e *interface* destino correspondem a alguma *interface* IPv6 configurada com um endereço ISATAP (os últimos dois octetos são o endereço IPv4). Caso exista, o mecanismo de desencapsulamento verifica se o endereço IPv4 origem está correcto e de acordo com o endereço encapsulado (endereço ISATAP), ou seja verifica se o endereço IPv4 origem está contido num endereço ISATAP, ou, se o endereço IPv4 faz parte de uma RPL (*Routers Potencial List*).

Para os pacotes cujo endereço IPv4 está incorrecto para a *interface* ISATAP é desencadeado um mecanismo para determinar se o pacote pertence a outra *interface* de túnel ou não.

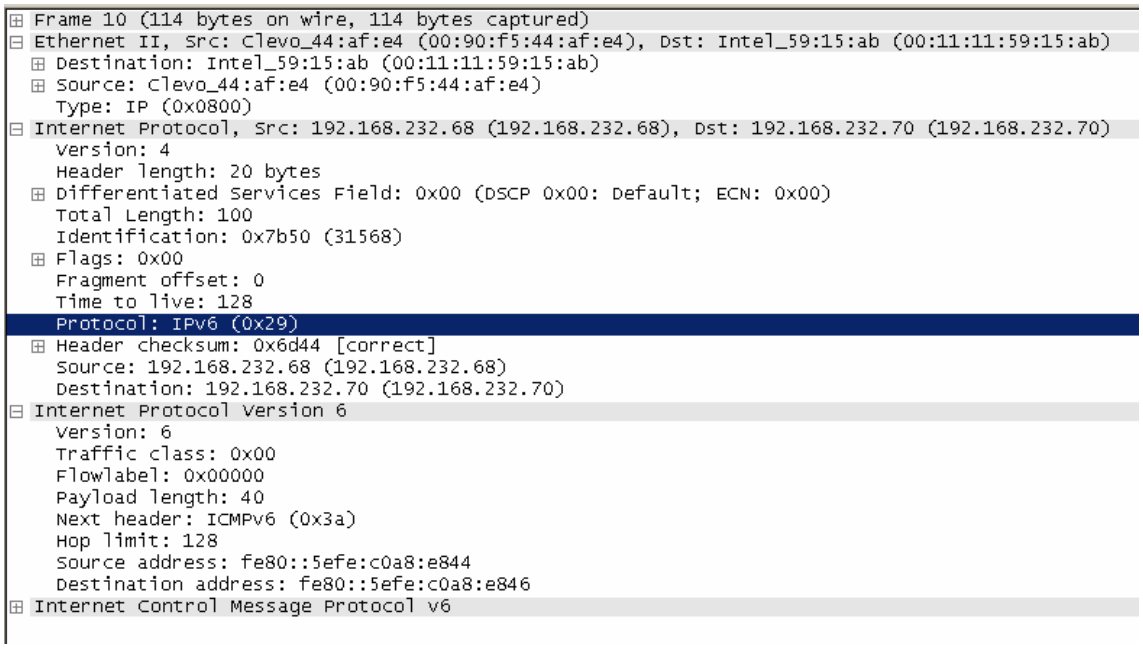


Figura 9.4 – Pacote ISATAP capturado na máquina destino.

Na linha seleccionada está o campo "*Protocol*" com o valor 41 em hexadecimal (0x29), este valor indica que o endereço origem é proveniente de um túnel. Ao saber que o endereço provém de um túnel verifica se existe alguma *interface* ISATAP com o endereço IPv4 embutido para determinar qual a máquina destino.

9.3.3.2 Vantagens

- Túnel automático.
- Encaminhável na infraestrutura IPv6.
- Possibilidade de colocar vários endereços ISATAP num único prefixo IPv6.
- Pode ser usado em túneis 6to4.
- Garante conectividade IPv6 entre máquinas de uma intranet IPv4.

[73]

9.3.3.3 Desvantagens

- Risco de segurança. A ligação IPv4 virtual tem de ser delimitada na periferia da rede para que as máquinas IPv4 externas à rede não possam fazer parte da ligação ISATAP. Isto é feito negando a passagem do proto-41[17] pela *firewall*.[38]

[17] Pacotes com o campo *Protocol* = 41

9.3.4 Generic Routing Encapsulation (GRE)

O tráfego IPv6 pode ser transportado através de túneis GRE. Estes túneis são usados para fornecer ligação ponto a ponto (*point-to-point*). Tal como acontece nos túneis manuais, os túneis GRE fornecem ligação entre dois pontos usando um túnel por cada *link* (ligação). Os pacotes designados para serem enviados pelo túnel (já encapsulados com um cabeçalho de um protocolo, por exemplo o IP) são encapsulados com um novo cabeçalho (cabeçalho GRE) e colocados no túnel com o endereço de destino do final do túnel. O cabeçalho GRE tem a seguinte estrutura:

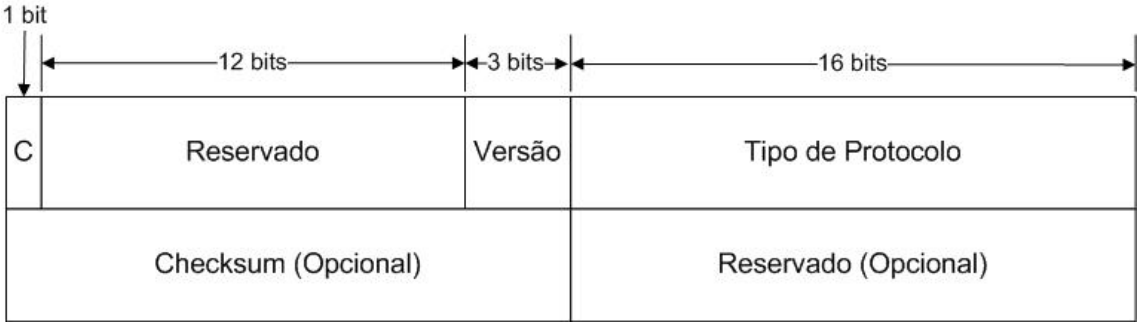


Figura 9.5 – Estrutura do cabeçalho do protocolo de túneis GRE.

- **Campo C:** bit que indica se o *checksum* está presente ou não, visto este ser opcional. Se estiver a 1 os campos opcionais contêm informação válida

- **Campo Reservado:** se primeiros 4 bits estiverem a 1, o receptor deve descartar o pacote a menos que este implemente a RFC 1701 ([10] GRE – Outubro de 1994). Os bits seguintes deste campo estão reservados para posterior utilização.

- **Campo Versão:** tem de conter o valor 0.

- **Campo Tipo de Protocolo:** contém o tipo de protocolo do pacote. Estes tipos de protocolos estão definidos na RFC 1700 ([11]). Se algum tipo de protocolo não estiver listado nesta RFC, então o pacote terá de ser descartado.

- **Campo *Checksum* (Opcional):** campo que contém o *checksum* de todas as palavras de 16 bits do cabeçalho GRE e do *payload* do pacote.

- **Campo Reservado (Opcional):** campo reservado para uso futuro e tem de ser transmitido com o valor 0.

Os túneis não são associados a um protocolo de transporte, no entanto neste tipo de túnel o IPv6 ou IPv4 é usado como protocolo de transporte.

O principal motivo para o uso destes túneis é a necessidade de ligação segura entre dois routers fronteira ou entre um router fronteira e um terminal. Também neste caso os routers e terminais têm de ter suporte para pilha dupla.

9.3.4.1 Vantagens:

- Fornecem redes locais multi-protocolares sob um *Backbone* com protocolo único.
- Contornam redes que contenham protocolos com um número de saltos limitado (*Hop Limit*).
- Ligam sub-redes descontínuas.
- Permitem VPN's através de WAN's.

[47]

9.3.4.2 Desvantagens:

- Se o túnel GRE estiver em baixo por qualquer motivo, os dispositivos que estão nos extremos do túnel não são notificados da falha. A sua rota correspondente é mantida na rota de encaminhamento dos dispositivos mesmo que a rota não esteja activa. O resultado é o tráfego ser encaminhado por uma rota inválida, sendo, consequentemente, os pacotes perdidos. A única solução é remover manualmente a rota estática inválida.[48]

9.3.5 Túneis Teredo

O principal propósito dos túneis Teredo é o de proporcionar conectividade IPv6 a máquinas que se encontrem por trás de um mecanismo NAT. Com a escassez de endereços IPv4 muitas instituições e pequenas redes informáticas usam o mecanismo NAT para que apenas seja necessário um ou dois endereços públicos por instituição.

Os túneis 6to4 não funcionam em máquinas que estejam por trás de NAT, pois o NAT apenas traduz TCP e UDP ficando o campo protocolo do cabeçalho IP (essencial para indicar a existência de um endereço proveniente de um túnel) descartado. Os túneis 6to4 apenas funcionam se a máquina que faz as funções de 6to4 router for a mesma que assegura o NAT.

Os túneis Teredo permitem que uma máquina que se encontre numa rede IPv4 por trás de um mecanismo NAT se possa ligar ao *Backbone* do IPv6 sem qualquer envolvimento da infra-estrutura existente. [12]

Os pacotes deste tipo de túnel automático têm o seguinte formato:

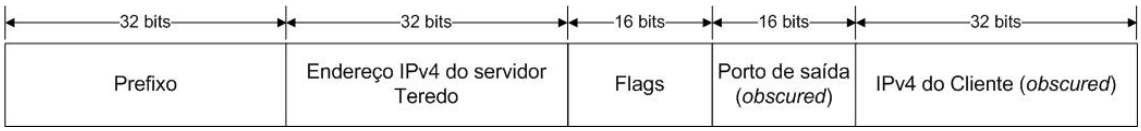


Figura 9.6 – Formato dos endereços Teredo.

Prefixo: prefixo usado pelo Teredo que é o mesmo para todos os endereços Teredo (ex: actualmente a Microsoft usa o prefixo 3FFE:831F::/32).

Endereço IP do servidor teredo: IP do servidor Teredo.

Flags: flags que têm como objectivo identificar o tipo de Nat que está a ser usado (cone nat, restricted nat). Ex: o valor 8000 nesta flag indica que o tipo de Nat é um cone Nat, o valor 0 indica que é um restricted nat.

Porto de saida: este campo é preenchido da seguinte maneira: efectua-se com o valor do porto um "XOR" com 0xFFFF. Ex: o porto 5000 daria 0xEC77 (5000 = 0x1388, 0x1388 XOR 0xFFFF = 0xEC77). O campo porto do cliente teria o valor 0xEC77.

IPv4 do Cliente: este campo é preenchido da seguinte maneira: efectua-se com o IP do cliente um "XOR" com 0xFFFFFFFF. **Ex:** o IP 131.107.0.1 daria 7C94:FFFE (131.107.0.1 = 0x836B0001, 0x836B0001 XOR 0xFFFFFFFF = 0x7C94FFFE). O campo Cliente IPv4 teria o valor 7C94:FFFE. [39]

Exemplo:

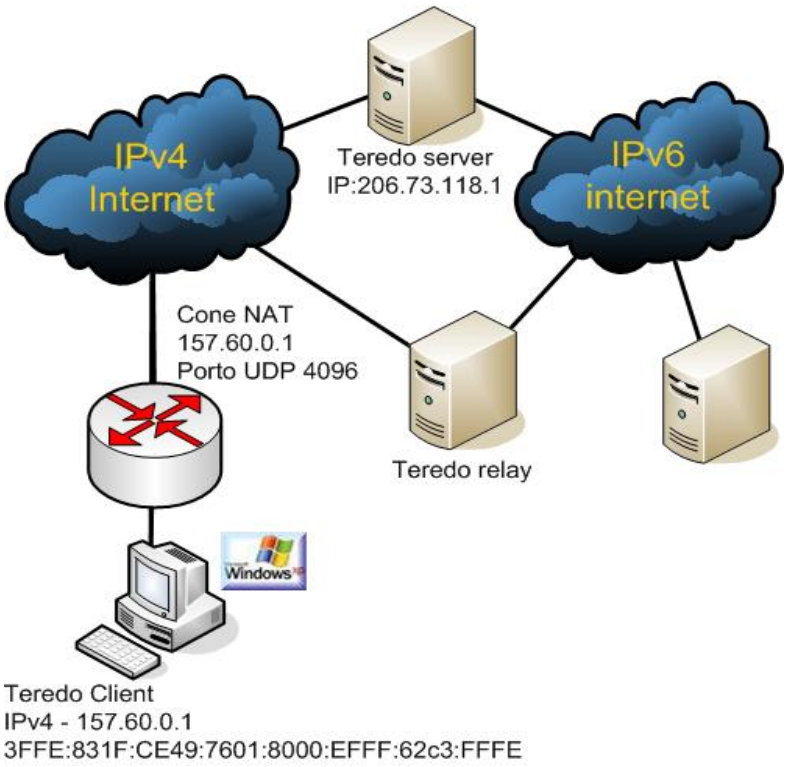


Figura 9.7 – Exemplo de um túnel Teredo.

3FFE:831F é o prefixo Teredo que a Microsoft está a usar actualmente.

CE49:7601 é a versão hexadecimal do ip 206.73.118.1.

8000 é a flag que indica que se encontra por trás de um "cone NAT".

EFFF é o porto com o formato "*obscured*"(4096= 1000 xor 0xFFFF = EFFF).

62C3:FFFE é o IP do cliente com o formato "*obscured*" (157.60.0.1 = 9D3c:0001, 9D3c:0001 xor 0xFFFFFFFF = 62C3:FFFE).

Existem vários tipos de nós Teredo:

- Cliente teredo
- Servidor Teredo
- *Teredo relays*
- *Teredo host-specific relay*

Cliente teredo: é uma máquina que tem ligação IPv4 à *Internet* através do NAT. Para distinguir os cliente Teredo dos outros é usado o prefixo (2001:0000::32).

Servidor Teredo: é usado para ajudar o cliente Teredo a inicializar o túnel assim como na sua configuração. Um servidor Teredo não encaminha tráfego com excepção dos pings IPv6.Um servidor Teredo pode suportar vários clientes uma vez que a largura de banda dispendida com cada um é mínima.

Existem actualmente alguns servidores conhecidos:

- **Teredo.ipv6.*Microsoft*.com**: servidor por defeito para os clientes *Windows*).
- **Teredo.remlab.net:** servidor experimental Miredo (solução open source de túneis IPv6 teredo).
- **Teredo.ipv6.wanadoo.fr:** servidor Wanadoo (ISP francês).

Teredo Relays: são as máquinas que estão na outra extremidade do túnel. Estas máquinas encaminham o tráfego IPv6, necessitam portanto de muita largura de banda e não podem suportar muitos Clientes Teredo. Cada Teredo relay tem um limite de máquinas IPv6 que pode servir. Encaminha o tráfego entre clientes Teredo.

Um Teredo Relay precisa de muita largura de banda, para alem disso precisa de exportar a rota através do prefixo Teredo IPv6 (2001::/32) para outras máquinas IPv6. Assim um Teredo Relay recebe tráfego de máquinas IPv6 através de clientes Teredo e reencaminha-o depois através de **UDP/IPv4**. Analogamente também recebe tráfego **UDP/IPv4** e reencaminha-o para a rede IPv6 nativa.

Um administrador de rede pode implementar um Teredo relay privado numa rede (um Campus de uma Universidade por exemplo), no entanto implementar um Teredo Relay *Global* requer a capacidade de exportar rotas BGP (*Border Gateway Protocol*). Apenas grandes ISP têm a capacidade de exportar rotas BGP.

Nota: BGP é o protocolo de *Routing* do "*core*" da *Internet*.[36]

Teredo Host-Specific Relay: Teredo Relay cuja abrangência do serviço é limitada ás máquinas a si ligadas. Não são necessárias especiais atenções à largura de banda ou ao encaminhamento. Um computador com um *Host-Specific Relay* irá usar Teredo para comunicar com clientes Teredo mas irá restringir-se ao seu principal fornecedor de IPv6 para alcançar a restante *Internet* IPv6.

Quando usar túneis Teredo?

Os tuneis Teredo devem ser usados como último recurso, sempre que possível deve-se usar ligação IPv6 directa (quando disponível) ou túneis 6to4 (quando é possível configurá-los na mesma máquina onde o NAT está configurado). Apesar dos túneis Teredo funcionarem bem sem o NAT, é sempre melhor opção usar os túneis 6to4 (mais robustos nestas situações) quando não existe um mecanismo NAT.

Caso se verifique alguma destas condições:

- Tenha uma ligação IPv6 nativa
- Tenha uma ligação IPv4 directa com um endereço público
- A máquina que se pretende ligar ao IPv6 se encontre atrás de um simmetric NAT (não compatível com protocolo de túneis teredo).

É desaconselhada a utilização de túneis Teredo.

Nota: o software IOS Cisco não suporta até à data o mecanismo de túneis Teredo

9.3.5.1 Vantagens

- Trafego IPv6 passa pelo mecanismo NAT de forma transparente.
- Não é necessário prefixos IPv6 globais.
- Pequena sobrecarga no servidor Teredo, o que faz com que exista somente este ponto de falha e de segurança.

9.3.5.2 Desvantagens

- Sobrecarga extra no encaminhamento IPv6.
- Necessidade de muitos routers relay Teredo para garantir a eficiência do encaminhamento IPv6.
- Sobrecarga no controlo e gestão da rede.
- Não é compatível com todo o tipo de NAT.
- Só é possível configurar um endereço IPv6 em cada extremidade do túnel, não sendo por isso possível ligar várias máquinas a um túnel Teredo.

[50], [39]

9.4 Routers a usar nas extremidades do túnel

Dos routers existentes na rede ou nos possíveis routers a adquirir é preciso ter em conta os seguinte factores:

- O router deve ser actualizado com software com funcionalidades IPv6
- O router tem de ter pilha dupla
- Para os mecanismos de túnel, o router deve possuir uma ligação IPv4 fiável com outro router remoto.

10 Compatibilidade com sistemas operativos

Suporte ISATAP para os seguintes sistemas operativos mais conhecidos e utilizados:

ISATAP			
Plataforma	**Cliente**	**Servidor**	**Observações**
Linux (Versão do Kernel **igual** a 2.4)	✓	✓	Incluído em USAGI kernel patches
Cisco IOS	✗	✓	Incluído oficialmente nas ED – Releases 12.2(15)ZJ,12.2(15)T2, 12.2(14)SY, 12.2(14)SX1, 12.2(14)S2 and in the LD-Release 12.3(1)
Windows XP	✓	✗	Mais estável a partir do SP2

Tabela 10.1 – Alguns sistemas operativos que suportam túneis ISATAP.

Suporte 6to4 para os seguintes sistemas operativos mais conhecidos e utilizados:

6to4			
Plataforma	**Cliente**	**Servidor**	**Observações**
Linux (Versão do Kernel **igual** a 2.4)	✓	✓	
Cisco IOS	✓	✓	
Windows XP	✓	✓	Mais estável a partir do SP2

Tabela 10.2 – Alguns sistemas operativos que suportam túneis 6to4.

Suporte Teredo/Miredo para os seguintes sistemas operativos mais conhecidos e utilizados:

Teredo/Miredo			
Plataforma	**Cliente**	**Servidor**	**Observações**

Linux (Versão do Kernel **igual** a 2.4)	✓	✓	Em Linux tem o nome Miredo.
Cisco IOS	✗	✓	
Windows XP	✓	✗✓	Mais estável a partir do SP2. Existe uma versão beta de um Teredo server para usar em Windows 2003.

Tabela 10.3 – Alguns sistemas operativos que suportam túneis Teredo/Miredo.

10.1 *Windows XP*

O protocolo IPv6 do *Windows XP* pode utilizar os seguintes métodos de configuração:

- Configuração automática utilizando endereços *stateless*.
- Configuração manual.

10.1.1 Configuração automática

O protocolo IPv6 do *Windows XP* suporta a configuração automática para endereços *stateless*. Os nós IPv6 criam automaticamente endereços locais únicos para todas as *interface*s de rede local. Os nós IPv6 utilizam as mensagens de anúncio de router recebidas para configurar automaticamente:

- Um router predefinido.
- A predefinição do campo *Hop Limit* no cabeçalho de IPv6.
- A determinação se o nó deve utilizar um protocolo de configuração de endereços *stateless*, como o DHCPv6 (*Dynamic Host Configuration Protocol* para IPv6), para endereços e outros requisitos de configuração. O protocolo IPv6 do *Windows XP* não suporta o DHCPv6 ou qualquer outro protocolo de configuração de endereços sem estado.
- Os temporizadores utilizados nos processos de *Neighbor Discovery*.
- A unidade máxima de transmissão (MTU, *Maximum Transmission Unit*) da ligação local.
- A lista de prefixos de rede definidos para a ligação. Cada prefixo de rede contém o prefixo de rede IPv6 e a respectiva duração válida e preferencial. Caso seja indicado, um prefixo de rede é combinado com a *interface* para criar uma configuração de endereço IPv6 *stateless* para a *interface* de recepção. Um prefixo de rede também cria um intervalo de endereços para nós na ligação local.
- Endereços compatíveis com IPv4 numa *interface* de *tunneling* automática.
- Endereços 6to4 numa *interface* 6to4 *tunneling* para todos os endereços públicos IPv4 atribuídos ao computador.

- Endereços ISATAP (*Intrasite Automatic Tunnel Addressing Protocol*) numa *interface* automática para todos os endereços IPv4 atribuídos ao computador.
- Rotas para prefixos exteriores à ligação (se o o prefixo de endereço exterior à ligação for anunciado por um router com o *Windows XP*).

10.1.2 ISATAP

Por definição o protocolo IPv6 do *Windows XP* configura automaticamente o endereço ISATAP fe80::5fef:w:x:y:z na Automatic *Tunneling Pseudo-Interface* para cada endereço de IPv4 atribuído a um nó[1].

Exemplo:

Endereço IPv4: 213:58:34:5

Endereço ISATAP correspondente: fe80::5efe:213:58:34:5 %2

Nota: o %2 indica o *interface* atribuído, neste caso o *interface* 2.

Sendo assim suponhamos o seguinte exemplo:

Resultado do comando ipv6 if 2:

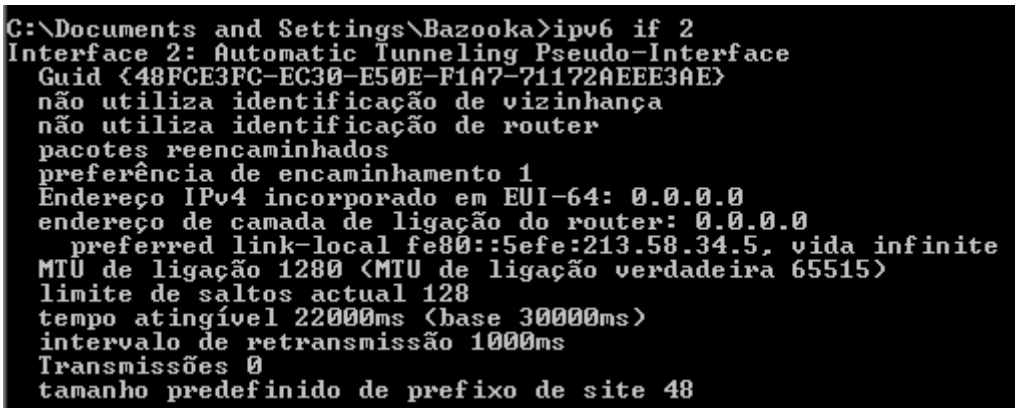


Figura 10.1 – Descrição da *interface* ISATAP do *Windows XP*.

Através de endereços ISATAP é possível dois nós comunicarem entre si através da intranet de uma infra-estrutura IPv4. Não é no entanto possível comunicar com máquinas exteriores a esta rede sem que sejam efectuadas alterações no router de saída da rede.

Para que se possa comunicar com uma máquina exterior à rede é necessário que o router fronteira da rede anuncie qual o prefixo de endereço *Global* a usar pelo host na criação do seu endereço ISATAP. Os routers fronteira são na maioria dos casos routers 6to4 ligados à *Internet*. A máquina constrói o seu endereço ISATAP com base no prefixo fornecido pelo router.

Exemplo:

Router local com ligação ao 6bone anuncia o prefixo de endereço *Global* 3000::/64

Endereço IPv4 do host: 10.40.1.29

Prefixo anunciado pelo router fronteira: 3000::/64

Endereço ISATAP: 3000::200:5EFE:10.40.1.29

Existe no entanto uma limitação grande nos endereços ISATAP quando utilizamos máquinas *Windows* como router. Não existe nenhum mecanismo para que o router possa anunciar qual o prefixo a usar.

È necessário configurar o endereço manualmente.

Para configurar o endereço manualmente na Automatic *Tunneling Pseudo-Interface* (id da *interface* 2) usa-se o comando "ipv6 adu". Para configurar o endereço do exemplo anterior ficaria: ipv6 adu 2/2002:836B:1:5:200:5EFE:10.40.1.29.

O prefixo de 64 bits da Automatic *Tunneling Pseudo-Interface* (id da *interface* 2) tem de ser manualmente adicionado à tabela de encaminhamento de ipv6 através do comando ipv6 rtu 2002:836B:1:5::/64 2.

Um host tem de ter uma rota predefinida, apontando para um endereço ISATAP que corresponda à *interface* de intranet do router fronteira da rede[1].

Exemplo:

Endereço IPv4 da *interface* intranet do router: 172.16.0.1

O host A terá de ter uma rota predefinida (::/0) que utilize o endereço ISATAP FE80::200:5EFE:172.16.0.1.

O comando para configurar esta rota é: ipv6 rtu ::/0 2/FE80::200:5EFE:172.16.0.1.

10.1.3 6to4

O suporte para os routers e máquinas 6to4 é fornecido no serviço 6to4 incluído no protocolo IPv6 para *Windows XP*. O serviço 6to4:

- Configura automaticamente endereços 6to4 na *interface* designada por 6to4 *Tunneling Pseudo-Interface* (*interface* com o ID 3) para todos os endereços IPv4 públicos atribuídos a *interface*s do computador.
- Cria automaticamente uma rota 2002::/16 que encaminha todo o tráfego 6to4 com a 6to4 *Tunneling Pseudo-Interface* (*interface* com o ID 3). Todo o tráfego reencaminhado por esta máquina para destinos 6to4 é encapsulado com um cabeçalho IPv4.
- Efectua automaticamente uma consulta ao DNS pelo nome 6to4 .Ipv6.*Microsoft*.com para obter o endereço IPv4 do router de encaminhamento 6to4 da *Microsoft* na *Internet*. Pode-se utilizar o comando netsh *interface* ipv6 6to4 set relay para especificar o nome de DNS para consulta.

Router local sem ligação ao 6bone

É utilizado um endereço *Global* baseado no 6to4 para estabelecer ligação a um router com ligação a outros locais e anfitriões do 6to4 e ao 6bone.

Resultado do comando ipv6 if 3:

```
C:\Documents and Settings\Brilha>ipv6 if 3
Interface 3: 6to4 Tunneling Pseudo-Interface
  Guid {A995346E-9F3E-2EDB-47D1-9CC7BA01CD73}
  não utiliza identificação de vizinhança
  não utiliza identificação de router
  preferência de encaminhamento 1
    preferred global 2002:c80f:f02::c80f:f02, vida infinite
  MTU de ligação 1280 (MTU de ligação verdadeira 65515)
  limite de saltos actual 128
  tempo atingível 18000ms (base 30000ms)
  intervalo de retransmissão 1000ms
  Transmissões 0
  tamanho predefinido de prefixo de site 48
```

Figura 10.2 – Descrição da *interface* 6to4 do *Windows XP*.

Através da configuração automática do serviço 6to4, qualquer máquina com o protocolo IPv6 para *Windows XP* configurado com um endereço IPv4 público, é automaticamente configurado como uma máquina 6to4.[72]

10.1.4 Teredo

O túnel Teredo de uma máquina *Windows XP* fica automaticamente inactivo se o computador já tiver sido membro de algum domínio. Um computador que faça parte de um domínio é mais provável que esteja ligado a uma rede que tenha evoluído para IPv6 nativo ou ISATAP. No entanto, um computador que seja membro de um domínio pode também beneficiar de conectividade IPv6 através de um túnel Teredo.

Este tipo de túnel, no *Windows XP* fica inactivo quando detecta que está por "trás" de um mecanismo de tradução NAT simétrico.

Resultado do comando ipv6 if 8:

```
C:\Documents and Settings\Brilha>ipv6 if 8
Interface 8: Pseudo-Interface de túnel Teredo
  Guid {E64F78B0-3C96-4940-BCA8-B5FEEA110B6F}
  zonas: link 8 site 5
  cabo desligado
  utiliza identificação de vizinhança
  utiliza identificação de router
  preferência de encaminhamento 2
  endereço de camada de ligação: 0.0.0.0:0
    preferred link-local fe80::5445:5245:444f, vida infinite
    multicast interface-local ff01::1, 1 refs, não referenciável
    multicast link-local ff02::1, 1 refs, não referenciável
  MTU de ligação 1280 (MTU de ligação verdadeira 1280)
  limite de saltos actual 128
  tempo atingível 22500ms (base 30000ms)
  intervalo de retransmissão 1000ms
  Transmissões 0
  tamanho predefinido de prefixo de site 48
```

Figura 10.3 – Descrição da *interface* Teredo do *Windows XP*.

[46]

10.1.5 Limitações

- Não envia mensagens DNS sobre IPv6 (O *Windows* Server 2003 não tem esta limitação).
- Não suporta a partilha de ficheiros nem de impressoras (O *Windows* Server 2003 não tem esta limitação).
- Não tem suporte para o Winlnet, IPHelper e API's DCOM

[74]

10.2 Fedora Core 5

O suporte IPv6 em distribuições Linux está directamente relacionado com a versão do Kernel destas mesmas. Assim, a versão do Kernel utilizada no Fedora *Core* 5 foi a 2.6.15.

O código das politicas base do Kernel (versão 2.6.6) foi modificado para suportar IPv6 e actualizado para fornecer politicas de segurança genérica para que as versões do Kernel mais antigas continuem a funcionar, facilitando a actualização do Kernel.

Ambas as implementações de *interface*s IPv4 e IPv6 de *Multicast* joining visíveis para o sistema foram actualiazadas confome os últimos padrões. A partir desta versão (2.6.6) do Kernel, o suporte IPv6 está já bastante fiável.

Para que as distribuições Linux tivessem uma implementação IPv6 de acordo com as inúmeras RFC's relativas ao seu funcionamento e arquitectura e de acordo com as últimas especificações IPv6, surgiu um projecto que trabalha somente no desenvolvimento de suporte IPv6 para sistemas Linux designado como USAGI.

10.2.1 ISATAP

O suporte ISATAP em distribuições Linux foi suspenso devido a uma reclamação de patente não existindo por isso suporte ISATAP para as versões mais recentes do Kernel. No entanto a IETF IPR revela que não é necessária qualquer licença por parte dos implementadores desta tecnologia.

Devido a toda esta situação, o projecto USAGI deliberou que o desenvolvimento do suporte ISATAP em sistemas Linux terá uma prioridade baixa.[38]

10.2.2 6to4

O FC5 tem suporte total para este tipo de túnel. A sua implementação é fácil e está explicada no documento em anexo.

10.2.3 Miredo

Miredo é um tipo de túnel Teredo open-source para linux. É possível configurá-lo para agir como um cliente Teredo, um relay Teredo ou um servidor Teredo, tendo também como objectivo oferecer conectividade IPv6 por trás de dispositivos NAT.

Este tipo de tunel foi inicialmente desenvolvido e mantido por Remi Denis-Courmont como um projecto voluntário. Inclui todos os componentes da arquitectura Teredo.

Actualmente é implementado sob os termos de GNU General Public License em várias distribuições Linux (incluindo no FC5).

O túnel Miredo é compatível com os drivers Teredo incluídos no *Windows XP* (Service Pack 2), *Windows* 2003 e supostamente no *Windows Vista*.

11 Estrutura DNS

Um *Domain Name System* (DNS) é necessário para uma coexistência fiável dos protocolo IPv4 e IPv6. O DNS é das ferramentas mais importantes da *Internet* actual e ganha uma importância ainda maior com a progressiva adopção do protocolo IPv6. O utilizador comum acede a uma página através de um nome e não através de um IP, mesmo que pensemos em pessoas especializadas em informática que saibam usar os endereços IP para aceder a recursos isso torna-se um pouco mais difícil com os novos endereços IPv6. A actualização da infra-estrutura DNS passa pela implementação nos servidores DNS de suporte para resolução de endereços IPv6 para nomes e vice-versa.

No entanto o "upgrade" para o suporte de endereços IPv6 não é simples uma vez as aplicações assumem que as consultas apenas devolvem endereços de 32 bits IPv4.[13]

11.1 Extensões da estrutura DNS para suporte do protocolo IPv6

- Criado um novo tipo de registo para os endereços IPv6 (registo AAAA)
- Criado um novo domínio para consultas baseadas no nome do endereço (domínio IP6.INT)
- Os mecanismos de consultas existentes foram actualizados para que efectuem consultas em endereços IPv4 e IPv6.

Estas mudanças foram efectuadas de modo a ser compatível com o software existente. **O suporte actual para registos IPv4 é mantido**.

A infra-estrutura DNS deve conter os seguintes registos:

- Registo dos nós IPv4 nativos (*IPv4 only node*s) e registo dos nós IPv4/IPv6.
- Registo de nós AAAA para nós IPv6 nativos (*IPv6 only node*s) e IPv4/IPv6.
- Registos PTR no domínio IN-ADDR.ARPA para nós IPv4 nativo e IPv4/IPv6
- Registos PTR no domínio IP6.ARPA para IPV6 nativo e IPv4/IPv6 (Opcional)
- Novo tipo de registo *DNAME*[18] análogo ao *CNAME* para registos A6 (Opcional).

11.2 Bind 9

[18] Registo que permite o funcionamento dos registos A6 uma vez que permite a consulta de endereços concatenados (chain records)

O software mais comum de servidor de DNS é o BIND3. Originalmente desenvolvido pela universidade de Berkeley é actualmente mantido pelo ISC4. Este software divide os servidores de nomes em três tipos:

- Servidor primário que contém todos os dados do domínio.
- Servidor secundário que copia a base de dados do servidor primário.
- Servidor de cache que constrói uma base de dados DNS através dos pedidos efectuados.

Apenas os servidores primários e secundários podem ser considerados iterativos. No Bind, o lado servidor denomina-se named. Este software é utilizado inclusive pela maioria dos *root name servers*. O Bind tem suporte IPv6 desde a versão 9, embora seja possível a utilização de

11.2.1 Melhorias e mudanças:

- Suporte para IPv6
- DNSSEC e TSIG (Funções de segurança);
- Suporte a hardware com multiprocessadores;
- Dynamic update para uso em redes DHCP;
- Novos tipos de registros (A6);
- Views (Permite várias visualizações do espaço de nomes);

[51]

11.3 *Resource records* (Registos) AAAA

Os quatro As (AAAA) é um nome relativo aos endereços de rede indicando que os endereços IPv6 são quatro vezes maiores que os endereços IPv4. Os registos estão estruturados praticamente do mesmo modo que os registos IPv4 de 32 bits, mas com um tamanho maior (o valor do campo *Type* é 28 em decimal).

Cada registo AAAA contém um endereço IPv6.

Exemplo:

$ORIGIN exemplo.com.
host 3600 IN AAAA 3ffe:8050:201:1860:42::1

[49]

11.3.1 Consulta AAAA

A consulta de um determinado registo AAAA para um dado nome de domínio devolve todos os registos AAAA associados a esse domínio. Não é efectuado qualquer processamento adicional.

11.4 Domínio IP6.INT

O objectivo da criação deste domínio foi o de mapear endereços IPv6 com nomes de domínio.

Um endereço IPv6 é representado por um nome com uma sequência de "nibbles"[19] separados por um ponto com o sufixo "IP6.INT". A sequência de "nibbles" é codificada em ordem inversa, ou seja, os "nibbles" de menor ordem são codificados primeiro, seguindo-se do segundo de menor ordem e assim sucessivamente.

Cada "nibble" é representado por um número hexadecimal.

Exemplo:

Usando o Formato Nibble o nome para o endereço: 2345:00C1:CA11:0001:1234:5678:9ABC:DEF0

A pesquisa será feita com o nome: 0.f.e.d.c.b.a.9.8.7.6.5.4.3.2.1.1.0.0.0.1.1.a.c.1.c.0.0.5.4.3.2.ip6.int.

Nota: Este domínio está obsoleto sendo subsituido pelo IP6.ARPA

11.5 Domínio IP6.ARPA

No domínio IP6.ARPA os registos usam um formato "*bit-string*".

Este domínio também funciona com o formato *nibble*.

Exemplo:

O endereço IP 4321::1:2:3:4:567:89AB será

\[x43210000001000200030004056789AB/128].IP6.ARPA

Os registos em IP6.ARPA começam obrigatoriamente pela barra ("\") e a sequência de dígitos tem de estar entre parênteses rectos e iniciado pelo caractere ("x"). Ao contrário dos registos em IP6.INT a ordem dos dígitos é igual à do endereço IP.

O exemplo seguinte também representa a resolução inversa mostrada anteriormente:

\[x43210000/32]. \[x0001/16]. \[00020003000 4056789AB/80].IP6.ARPA

11.6 Regras para selecção de endereços

Para a resolução nome-endereço, o nó onde foi efectuada a consulta precisa de determinar o conjunto de endereços que irão servir de endereços origem e destino. De maneira a reencaminhar os pacotes para o sítio certo.

[19] termo que designa a agregação de quatro bits, um nibble corresponde a um dígito hexadécimal.

Num cenário IPV4 nativo isto não é um problema, no entanto num cenário onde coexistam IPv4 e IPv6 as coisas tornam-se um pouco mais complexas uma vez que cada nó tem mais de um tipo de endereço. Geralmente um nó tem um endereço IPv4 e vários endereços IPv6. A estrutura DNS terá de devolver um resultado com os endereços IPv4 e também com os resultados IPv6 que existam nos seus registos.

11.7 *Resource records* A6

Na estrutura DNS foram recentemente criados para além do *Resource records* (RR) "AAAA" os RR "A6". Actualmente e durante alguns anos ainda usar-se-ão os "AAAA", no entanto prevê-se que os RR transitem gradualmente para os "A6" (o valor do campo "*Type*" é 38 em decimal).

Para facilitar a transição os registos A6 poderão guardar registos "AAAA".

O formato A6 tem umas diferenças significativas em relação aos "AAAA". Pretende tornar toda a estrutura DNS ainda mais hierarquizada e torná-la mais flexível.

Os RR A6 são mais flexíveis, é possível criar vários tipos de registos: registos completos ou registos concatenados.

Formato A6:

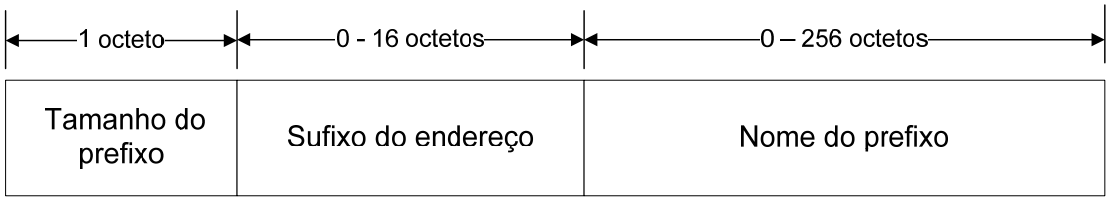


Figura 11.1 – Formato dos registos A6.

- Tamanho do prefixo ocupa oito bits com um inteiro com valores compreendidos entre 0 e 128
- Sufixo de endereço IPv6. Este campo deve conter octetos suficientes para conter um número de bits igual a 128 menos o tamanho do prefixo
- Nome do prefixo codificado como um nome de domínio.

Resolução Nome para endereço:

Os registos A6 podem conter num único registo o endereço completo (registos completos) ou apenas parte dele (registos concatenados), neste caso no registo está guardada informação relativa à localização da restante parte do endereço.

Exemplo:

Registo A6 completo:

```
$ORIGIN exemplo.com.
 host 3600 IN AAAA 3ffe:8050:201:1860:42::1
```

Registo A6 concatenado:

Por exemplo, um host localizado na empresa "company". Esta empresa possui dois ISPs fornecendo endereçamento IPv6. Cada um desses dois ISPs especificam o prefixo que fornecem. Assim, na empresa do nosso exemplo:

```
$ORIGIN exemplo.com.
host 3600 IN A6 64 0:0:0:0:42::1 company.exemplo1.net
host 3600 IN A6 64 0:0:0:0:42::1 company.exemplo2.net
```

O ISP1 vai utilizar:

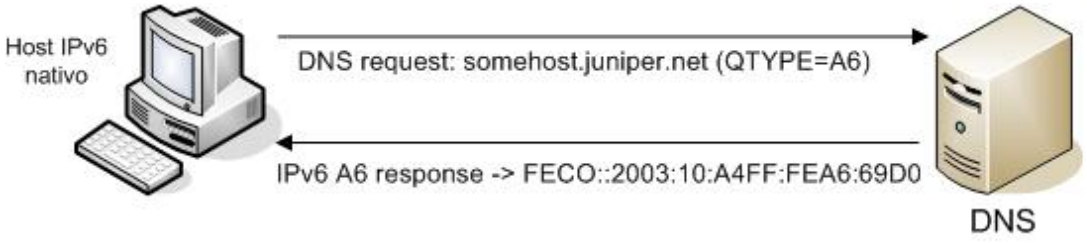
```
$ORIGIN exemplo1.net.
company 3600 IN A6 0 3ffe:8050:201:1860::
```

O ISP2 vai utilizar:

```
$ORIGIN exemplo2.net
company 3600 IN A6 0 1234:5678:90ab:fffa::
```

Quando se procura host.exemplo.com, o resolver vai encontrar dois registros A6 parciais. Com o nome adicional vai procurar o restante.

Outro Exemplo:




```
somehost.juniper.net.   A6 64        10:A4FF:FEA6:69D0 Juniper.net
Juniper.net             A6 48         2003:: provider.tla.org.
Provider.tla.org.       A6 0         FEC0::
Somehost.juniper.net    A   192.168.1.101
```

Figura 11.2 – Exemplo de uma consulta DNS usando registos A6.

Uma aplicação que efectue uma consulta de um endereço IPv6 irá fazer com que o resolver do DNS aceda a vários registos A6. Poderão ser devolvidos vários endereços IPv6 mesmo que o nome consultado seja dono de apenas um registo A6.

Para se obter um endereço pertencente a um nome um cliente DNS terá de obter um ou vários registos A6 (denominados de A6 *chain records*).

Exemplo:

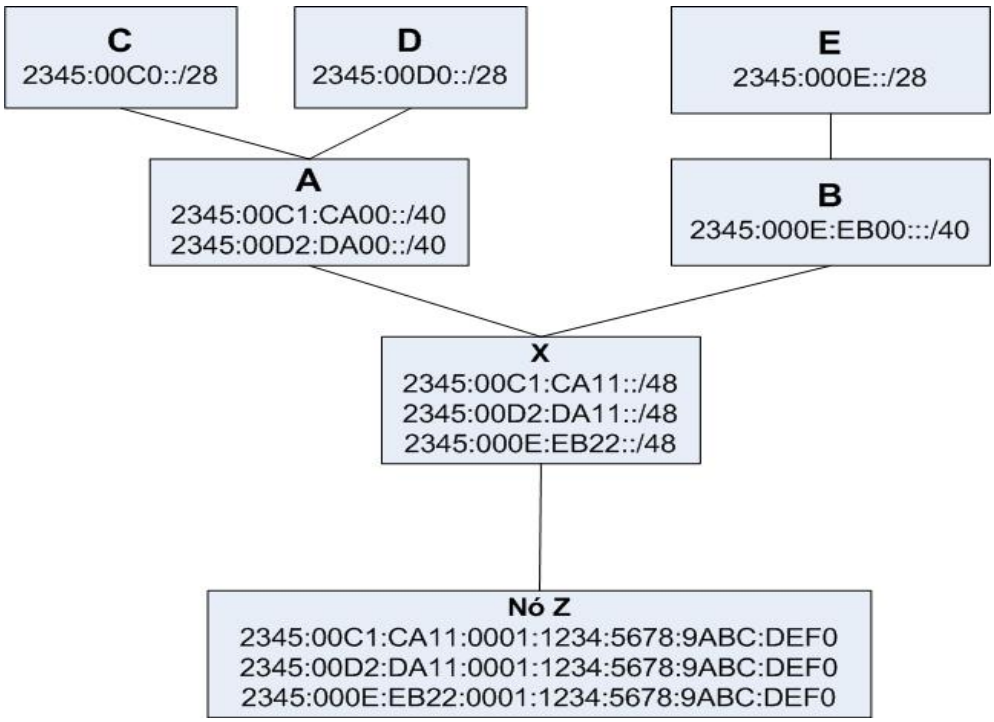


Figura 11.3 – Exemplo de como se distribuem os vários registos A6 (registo concatenado).

No exemplo anterior temos um local X que tem dois "fornecedores DNS" A e B. O fornecedor A tem dois fornecedores C e D. O fornecedor B tem apenas um fornecedor E. Para simplificar o exemplo supõe-se que os fornecedores "C", "D" e "E" estão todos ao mesmo nível de agregado (*toplevel aggregate* TLA) com um prefixo 2345.

Sendo o prefixo 2345 comum aos três os fornecedores C, D e E terão os prefixos 2345:00C0::/28, 2345:00D0::/28 e 2345:000E::/32 respectivamente.

O C atribui ao "A" o prefixo 2345:00C1:CA00::/40, o "D" atribui ao "A" o prefixo 2345:00D2:DA00::/40 e o "E" atribui ao "B" o prefixo 2345:000E:EB00::/40.

O "A" atribui ao "X" o identificador "11" e o "B" atribui ao X o identificador "22". Como resultado o local "X" herda os três seguintes prefixos:

- 2345:00C1:CA11::/48 do "A" para rotas através do "C".
- 2345:00D2:DA11::/48 do "A" para rotas através do "D".
- 2345:000E:EB22::/48 do "B" para rotas através do "E".

Supondo agora que "Z" é um nó (um dispositivo de rede) pertencente à subrede 1 e que usa o identificador de *interface* "1234:5678:9ABC:DEF0". O nó ficará com os três seguintes endereços:

- 2345:00C1:CA11:0001:1234:5678:9ABC:DEF0

- 2345:00D2:DA11:0001:1234:5678:9ABC:DEF0
- 2345:000E:EB22:0001:1234:5678:9ABC:DEF0

Nota: O registo *DNAME* guarda toda a ramificação a que um endereço pertencente em vez de um único nó como é típico do registo *CNAME*.

Com os novos formatos A6 o número de registos de DNS irá aumentar, para alem disso as várias partes do endereço A6 irão estar espalhadas por varias áreas geográficas o que poderá trazer algumas complicações. Actualmente e enquanto novos teste têm lugar é aconselhado que os endereços A6 estejam distribuídos por uma ou duas zonas (níveis como os representados na Figura 11.3) no máximo de modo a não comprometer a performance deste novo formato. Um procedimento sensato seria o de começar sem delegações de prefixos (campo tamanho do prefixo a "0") e numa fase posterior, adoptar um nível de delegação para os prefixos. Posteriormente implementar-se-ia então uma delegação mais flexível de prefixos.

12 Tradução de endereços IP

Existem problemas de incompatibilidade ponto-a-ponto que surgem quando um nó IPv6 quer comunicar com um nó IPv4. Isto acontece se:

- Um nó IPv6-*only* tenta comunicar com um nó IPv4-*only*.
- Um nó de pilha dupla com a pilha IPv6 activa tenta comunicar com um nó IPv4-*only*.
- Um nó de pilha dupla com a pilha IPv6 activa tenta comunicar com um nó com características idênticas.

Para solucionar este problema é necessário implementar uma função algures entre os dois nós que transforme o formato de um cabeçalho para um outro formato. Isto permite que os nós IPv6 enviem e recebam somente pacotes IPv6, tal como os nós IPv4 enviem e recebam somente pacotes IPv4. No entanto esta solução gera outros problemas e deve, por isso, ser evitada se possível.

12.1 *Stateless IP ICMP Translator* (SIIT)

O mecanismo SIIT fornece um algoritmo *stateless* que pode ser usado para traduzir cabeçalhos IPv6 para cabeçalhos IPv4 e vice-versa. Tal como o nome sugere, este tradutor também funciona para ICMP. Esta função de tradução é implementada em routers com pilha dupla, que por sua vez são chamados de SIIT routers.

12.1.1 Funcionamento

O SIIT assume que num router SIIT seja alocado uma pool de endereços IPv4.

Quando uma máquina IPv6, que implementa este mecanismo de tradução, tenta comunicar com uma máquina IPv4, os pacotes enviados da máquina IPv6 têm de incluir no campo do endereço de destino o endereço IPv4 mapeado para o endereço IPv6 e o endereço IPv4 traduzido para IPv6 no campo do endereço de origem.

Todos os routers SIIT injectam rotas dentro das ilhas IPv6 para anunciar que consegue encaminhar pacotes destinados a endereços IPv4 mapeados para IPv6. Deste modo, os pacotes enviados de máquinas IPv6 para endereços IPv4 mapeados serão recebidos por um dos routers SIIT.

O cabeçalho IPv4 é formado usando a informação do cabeçalho IPv6. O endereço de origem e de destino são formados a partir dos últimos 32 bits dos endereços traduzidos e mapeados respectivamente. Os restantes campos do cabeçalho IPv4 são valores já conhecidos (por ex. o campo IP *version*) ou que têm o valor correspondente dos campos do cabeçalho IPv6. O contrário já não se aplica: alguns valores IPv6 não são traduzidos para o correspondente campo IPv4. Assim, algumas das funcionalidades IPv6 serão perdidas durante a tradução. Isto porque, como é lógico, o IPv6 tem mais funcionalidades que o IPv4.

Depois dos pacotes IPv6 serem traduzidos, são enviados para a rede IPv4 para o nó correspondente.

Os routers SIIT também injectam rotas para dentro das ilhas IPv4 para anunciar que conseguem encaminhar pacotes destinados para qualquer dos endereços IPv4 incluidos na sua *pool* de endereços IPv4.

Quando os routers SIIT recebem pacotes da sua rede, sabem que estes terão de ser traduzidos (pois os pacotes têm incluído no endereço de destino um dos endereços IPv4 da pool de endereços do router) e executa o algoritmo SIIT para construir o cabeçalho IPv6 para posteriormente o encaminhar para a máquina IPv6 correspondente. Se o router SIIT estiver no mesmo *link* que a máquina IPv6 não existe qualquer problema. No entanto se o router SIIT estiver a mais que um salto de distância da máquina de destino, este terá de determinar o endereço IPv6 *Unicast* da máquina de destino para encapsular o pacote para essa máquina. Este problema não é solucionado por nenhuma RFC. No entanto pode ser resolvido de duas maneiras:

- Colocar sempre os routers SIIT no mesmo *link* que a máquina IPv6, ou seja, activar o mecanismo SIIT em todos os routers por defeito e atribuir a cada *link* uma pool de endereços IPv4 dados através de DHCPv6.
- Encontrar um novo protocolo que informe todos os routers SIIT do endereço IPv6 *Unicast* da máquina IPv6 à qual foi atribuído um endereço IPv4 para este propósito.

A vantagem da segunda opção é que permite uma utilização mais eficiente dos endereços IPv4. No entanto a primeira opção é muito mais fácil de implementar.

12.1.2 Vantagens

A maior vantagem do mecanismo SIIT é o facto deste ser *stateless*. Deste modo, as falhas dos routers SIIT não são diferentes das falhas de qualquer outro router. Além disso, uma vez que a tradução é feita na camada IP, é independente da aplicação a ser usada.

Algumas aplicações enviam o endereço IP no seu *payload*. Embora este comportamento viole o modelo das camadas, este facto é comum acontecer em várias aplicações (ex. FTP). Quando estas aplicações comunicam através do SIIT, estas devem incluir somente a parte do endereço IPv4 do endereço traduzido quando enviam o endereço IP da máquina para o seu correspondente. Deste modo, o receptor (máquina IPv4) não sabe que está a comunicar com uma máquina IPv6.

12.1.3 Desvantagens

A tradução de endereços transforma o conteúdo de um cabeçalho IP. No entanto a integridade ponto-a-ponto não é mantida. Deste modo, o campo *Authentication Header* não pode ser usado para proteger a comunicação entre máquinas IPv4 e IPv6 quando estão a comunicar através de um tradutor de endereços pois o Authentication Header cobre tanto o cabeçalho IP como o *payload*.

Por outro lado, o modo de transporte ESP pode ser usado pois o conteúdo do cabeçalho IP é-lhe transparente. No entanto, quando ao negociar uma SA, a implementação IKE da máquina IPv6 deve enviar o seu endereço IPv4.

Quando é utilizado ESP, uma máquina IPv6 não pode incluir o cabeçalho *Destination Options* depois do cabeçalho IPv6 pois o cabeçalho *Destination Options* não pode ser traduzida (uma vez que não é visível para um router SIIT) e não será tratada pelo destino final IPv4.

12.2 Network Address Translator and Protocol Translator NAT-PT

NAT-PT utiliza o algoritmo SIIT e uma função de alocação centralizada de endereços para permitir a comunicação entre máquinas IPv6 e IPv4.

O NAT-PT é um mecanismo de tradução *stateful* centralizado enquanto que o SIIT é um mecanismo de tradução *stateless* distribuído.

O objectivo do NAT-PT é o de fornecer um tradutor de endereço e de protocolo (IP e ICMP respectivamente) que seja transparente para o destino final. Ao contrário das máquinas SIIT, as máquinas IPv6 ás quais o tráfego é traduzido pelo NAT-PT estão completamente inconscientes da existência do tradutor.

As funções NAT-PT estão implementadas em routers tipicamente localizados na periferia duma rede informática.

Com o mecanismo bidireccional NAT-PT, as sessões podem ser iniciadas tanto de computadores de uma rede IPv4 como de computadores de uma rede IPv6. Os endereços de uma rede IPv6 são traduzidos estaticamente ou dinamicamente para endereços IPv4 à medida que as ligações são estabelecidas para cada direcção. Os computadores de uma rede IPv4 acedem a computadores de uma rede IPv6 utilizando o DNS para resolução de endereços. Tem de ser implementado um servidor DNS-ALG em conjunto com este mecanismo para facilitar o mapeamento de nomes para endereços.

Combinado com o protocolo SIIT, este mecanismo oferece a possibilidade de um vasto número de aplicações comunicar entre nós IPv6-*only* e IPv4-*only*.

Uma suposição fundamental é de que o mecanismo NAT-PT só deve ser usado em último caso para comunicação com redes nativas IPv6.[28]

12.2.1 Detalhes do protocolo

Os cabeçalhos do IPv4 e do ICMPv4 são similares aos seus correspondentes da versão 6 com excepções de faltarem alguns campos, terem significados diferentes ou tamanho diferente. O mecanismo NAT-PT traduz todos os cabeçalhos IP/ICMP da versão 4 para a versão 6 e vice-versa de modo a que torne possível a comunicação entre nós IPv6 e IPv4. Devido à função de tradução de endereços e possível multiplexagem de portas, o mecanismo NAT-PT deve fazer ajustes nos cabeçalhos da camada acima (TCP/UDP). [28]

12.2.2 Tradução de endereços IPv4 para endereços IPv6

Quando o mecanismo NAT-PT recebe um datagrama IPv4 que tenha como destino uma ilha IPv6, o cabeçalho IPv4 é traduzido para um cabeçalho IPv6. Seguidamente o pacote é encaminhado baseado no endereço IPv6 de destino. O cabeçalho original do pacote

IPv4 é eliminado e substituído por um cabeçalho IPv6.

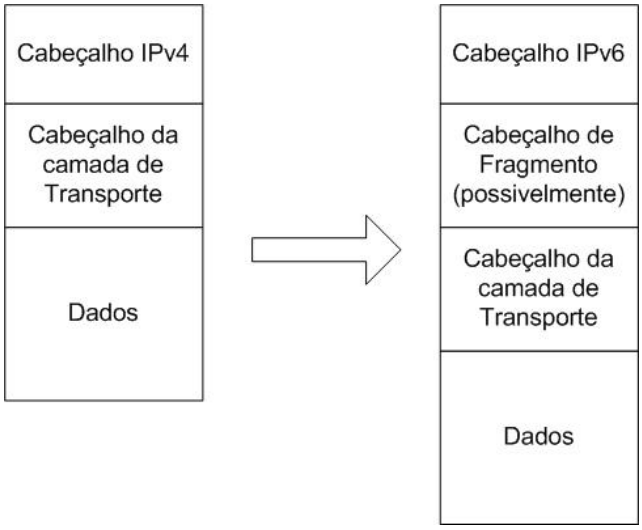


Figura 12.1 – Tradução de um pacote IPv4 para IPv6.

Uma das diferenças entre IPv4 e IPv6 é que em IPv6 o *Path MTU Discovery* é obrigatório enquanto que em IPv4 é opcional. Isto implica que os routers IPv6 nunca *Fragment*em um pacote (somente o emissor pode *Fragmentar* o pacote).

Quando um nó IPv4 executa o *Path MTU Discovery*, este pode operar ponto-a-ponto pelo mecanismo de tradução. Neste caso quer os routers IPv4 como os routers IPv6 poderão enviar de volta mensagens ICMP ("*Packet Too Big*") para o emissor. Quando estes erros ICMP são enviados pelos routers IPv6, estes passam pelo mecanismo de tradução de endereços que traduzirá o erro ICMP de forma a que o emissor IPv4 possa perceber o pacote recebido. No entanto, quando o emissor IPv4 não executa o *Path MTU Discovery*, o NAT-PT tem de garantir que o pacote não excede o path MTU do lado IPv6. Isto é feito fragmentando o pacote IPv4 de modo a que caiba num pacote IPv6 de 1280 bytes não sendo assim possível os pacotes serem fragmentados.

Fora as excepções acima mencionadas, a tradução do cabeçalho de um pacote consiste no simples mapeamento de endereços.[31]

12.2.2.1 Tradução de cabeçalhos IPv4 para cabeçalhos IPv6

Relativamente ao endereço de origem, os 32 bits menos significativos correspondem ao endereço IPv4 de origem. Os restantes 96 bits mais significativos correspondem ao prefixo para todas as comunicações da versão 4 do protocolo IP. Os endereços que utilizem esse prefixo serão encaminhados pelo *Gateway* NAT-PT (PREFIXO::/96).

Em relação ao endereço de destino, o mecanismo NAT-PT guarda o mapeamento entre o endereço IPv4 de destino e o endereço IPv6 de origem. O destino do endereço IPv4 é substituído pelo endereço IPv6 guardado pelo NAT-PT.[28]

12.2.3 Tradução de IPv6 para IPv4

Quando o mecanismo NAT-PT recebe um pacote IPv6 endereçado para um endereço IPv6 baseado em IPv4, traduz o cabeçalho IPv6 desse pacote para um cabeçalho IPv4. De seguida encaminha o pacote baseado no seu endereço IPv4. O cabeçalho do pacote original é removido e substituído pelo cabeçalho IPv4. Existe uma excepção para o caso dos pacotes ICMP, os quais o cabeçalho da camada de transporte e os dados do pacote são deixados intactos.

Existem algumas diferenças entre IPv6 e IPv4 na área da fragmentação de pacotes e do MTU mínimo aquando da tradução. Uma ligação IPv6 tem de ter um MTU mínimo de 1280 Bytes. O correspondente limite para IPv4 é 68 Bytes. Assim, não deve ser possível fazer *Path MTU Discovery* ponto a ponto quando existe o mecanismo NAT-PT no caminho pois o nó IPv6 poderá receber mensagens ICMP "*Packet Too Big*" originadas por um router IPv4 que reporta um MTU menor que 1280. No entanto o protocolo IPv6 trata estas mensagens reduzindo o *Path MTU Discovery* para 1280 Bytes e incluir um fragmento de cabeçalho IPv6 em cada um dos pacotes, o que faz com que seja possível fazer *Path MTU Discovery* ponto a ponto quando existe o mecanismo NAT-PT desde que o path MTU tenha no mínimo 1280 bytes de tamanho.[31]

12.2.3.1 Tradução de cabeçalhos IPv6 para cabeçalhos IPv4

Relativamente ao endereço de origem, o mecanismo NAT-PT guarda o mapeamento efectuado entre o endereço de origem IPv6 e um endereço de destino IPv4 de uma gama de endereços disponiveis. O endereço de origem IPv6 é substituído pelo endereço IPv4 guardado no mapeamento.

Em relação ao endereço de destino, os pacotes IPv6 que são traduzidos têm como destino o endereço na forma PREFIXO ::IPv4/96. Assim, os 32 bits menos significativos do endereço IPv6 de destino são copiados para o endereço de destino IPv4. [28]

12.2.4 Tradução de mensagens de erro ICMPv4 em mensagens ICMPv6

Existem algumas diferenças entre o formato das mensagens de erro ICMPv4 e ICMPv6.

As mensagens de erro ICMP têm um cabeçalho IP no pacote que será traduzido. Assim, a tradução deste pacote para um pacote v6 consiste em alterar o tamanho do datagrama. Deste modo, o campo *Payload Length* do cabeçalho IPv6 poderá ter de ser actualizado.

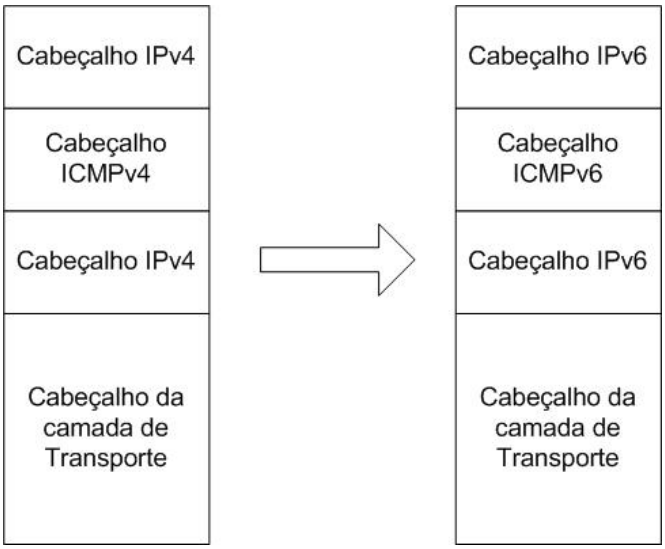


Figura 12.2 – Tradução de uma mensagem ICMPv4 para ICMPv6.

[31]

12.2.5 Operação NAT-PT Básica

Para explicar este ponto, será dado o seguinte exemplo e a sua respectiva análise para se tornar mais perceptível:

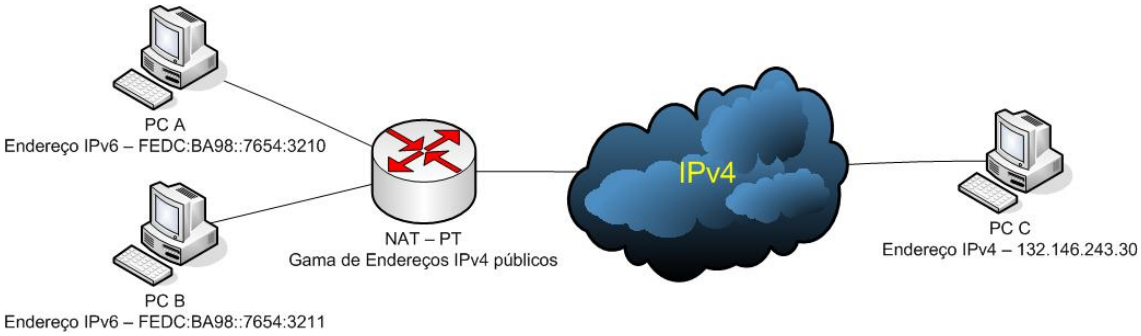


Figura 12.3 – Exemplo de funcionamento do mecanismo NAT.

Neste exemplo, NAT-PT tem uma gama de endereços que inclui a subrede 120.130.26/24. É também assumido que existem menos endereços IPv6 que endereços IPv4.

Se o PC A quiser comunicar com o PC C, o PC A cria um pacote com:

Endereço de origem: FEDC:BA98::7654:3210

Endereço de destino: PREFIXO::132.146.243.30

O prefixo PREFIXO ::/96 é anunciado no domínio *stub* pelo NAT-PT e os pacotes endereçados para este prefixo serão encaminhados para o NAT-PT.

O pacote é encaminhado pelo *Gateway* NAT-PT onde é traduzido para IPv4.

O pacote IPv4 resultante ao sair da rede IPv6 tem o endereço de origem 120.130.26.10 e endereço de destino 132.146.243.30. Qualquer tráfego de retorno será identificado como pertencente à mesma sessão pelo NAT-PT. O NAT-PT usará o estado da sessão para traduzir o pacote. Assim sendo, os endereços de retorno resultantes serão:

Endereço de origem: PREFIXO::132.146.243.30

Endereço de destino: FEDC:BA98::7654:3210

[28]

12.2.6 Limitações

Existem limitações no uso do mecanismo de tradução de endereços NAT-PT.

O protocolo IPv6 foi concebido para que o *checksum* dos cabeçalhos TCP e UDP não fossem afectados pelas traduções de endereços. Assim, quando os endereços são traduzidos, não é necessário modificar os cabeçalhos TCP e UDP. A única excepção deve-se aos pacotes UDP IPv4 fragmentados, os quais necessitam de ter um *checksum* UDP desde que foi necessário o *checksum* para pacotes UDP para IPv6. Como o protocolo ICMPv6 inclui um cabeçalho *checksum*, o *checksum* das mensagens ICMPv4 necessitam de ser modificadas pelo mecanismo de tradução.

As mensagens de erro do protocolo ICMP contêm um cabeçalho IP como parte do *payload*, o que faz com que o mecanismo de tradução necessite de reescrever essas partes dos pacotes para que o receptor perceba o cabeçalho IP recebido.

No entanto, todas as operações de tradução de endereços (incluindo o path MTU deicovery) é *stateless* na medida em que o NAT-PT opera independentemente em cada pacote e não guarda nenhum estado de um pacote para outro.

O mecanismo NAT-PT não traduz as opções IPv4 nem os cabeçalhos IPv6 de encaminhamento, nem os cabeçalhos de estensão Hop-by-Hop, nem os cabeçalhos de opções do destino. [28]

A utilidade do encaminhamento ao passar por um mecanismo de tradução de endereços pode ser limitado pois todos os routers IPv6 precisam de ter um endereço IPv4 traduzido para IPv6 devido à eventualidade do nó de origem ser IPv4-only.

È necessário que todos os pedidos e respostas pertencentes a uma sessão sejam encaminhados para o mesmo router NAT-PT. Uma maneira de garantir isto é ter o router NAT-PT colocado num ponto da rede em que todo o tráfego IP tenha como destino ou seja originário desse domínio que o router engloba. De salientar o facto desta limitação não se aplicar a nós com pilha dupla que se comuniquem entre si pois um endereço IPv6 contém o endereço IPv4 com quem quer comunicar (PREFIXO ::x.y.z.w). Também não afecta a comunicação entre nós IPv6.

Resumindo, deve-se evitar o uso deste mecanismo de tradução de endereços. O seu uso deverá ser efectuado quando não existir nenhuma outra alternativa.

O uso deste mecanismo de tradução de endereços necessita de um caminho (túnel) entre dois nós IPv6 para que estes se comuniquem entre si. Para isso existem vários tipos de túneis que implementam esse “caminho”.[31]

12.2.6.1 Impacto da tradução de endereços

Uma vez que o NAT-PT efectua tradução de endereços, as aplicações que transportam o endereço IP para as camadas superiores não funcionam. Neste caso têm de ser incorporadas ALG's (*Application Layer Gateway*s) para darem suporte a essas aplicações. [28]

12.2.6.2 Falta de segurança

Uma das limitações mais importantes do NAT-PT é o facto de não ser possível garantir segurança na camada de rede. A segurança na camada de transporte e na camada de aplicação pode não ser possível em aplicações que transportem o endereço IP para as camadas superiores. Esta é uma limitação inerente ao próprio mecanismo NAT. [28]

13 Conclusão

Actualmente existe um grande esforço em permitir uma migração do protocolo IPv4 para o protocolo IPv6 sem grandes complicações, no entanto, ainda existem falhas e soluções que nem sempre são simples de implementar. Alguns mecanismos de transição como os túneis 6over4 estão já obsoletos pois assentam em pressupostos (*Multicast*) que nem sempre são os mais apropriados para o sucesso da solução.

Os principais e mais importantes mecanismos de transição são os túneis 6to4 e ISATAP. Estes dois tipos de túneis são actualmente a melhor solução para quem deseja migrar para uma rede em IPv6. Os restantes tipos de túneis nomeadamente os manuais, GRE e Teredo são soluções um pouco mais específicas que não irão ter grande relevância uma vez que não serão vastamente implementados.

Ao elaborar este projecto, deparámo-nos com um facto interessante: existe muita informação disponível sobre mecanismos de transição e de como implementar os mecanismos necessários para transitar, no entanto a informação relativamente à aplicação prática, ou seja, à implementação dos mecanismos nem sempre estava completa e correcta. Na teoria existe informação concreta e abundante, mas para partir para a implementação de uma solução para uma empresa por exemplo, existe muita informação contraditória, errada, incompleta e desactualizada, ganhando este documento uma relevância importante para quem está interessando em perceber e implementar mecanismos de transição. Um bom exemplo das dificuldades que uma pessoa poderia ter em implementar uma solução de transição, seria o cenário 1 do capítulo 6to4 + ISATAP do documento Cenários em que se pretende garantir conectividade de toda uma intranet (endereços privados) IPv4 ao *Backbone* IPv6. Este cenário é de grande importância pois é um cenário que simula o que se passa em muitas organizações actualmente como a ESTG por exemplo. Apesar de ser possível implementar este cenário chegamos à conclusão que a implementação não é propriamente simples envolvendo muitos passos intermédios. Neste aspecto não nos parece que haja muito a fazer uma vez que a transição é mesmo o que o nome indica, um processo de transição, não sendo por isso esperado encontrar soluções rápidas e fáceis. Mas nem tudo é complexo, os túneis 6to4 garantem a qualquer utilizador (desde que não se encontre numa intranet) conectividade ao 6bone de um modo bastante simples! Os túneis ISATAP garantem também conectividade IPv6 dentro de uma intranet IPv4 de modo fácil e rápido.

Um outro facto interessante é o de não haver uma boa solução ISATAP para máquinas Linux. Foi criado um projecto no Japão denominado “Usagi” que tem como objectivo principal o desenvolvimento de soluções IPv6 para sistemas Linux. De facto foi implementado um kit que teria o suporte para os túneis e endereços ISATAP, no entanto esse kit apenas foi disponibilizado para a versão Kernel 2.4 (versão bastante antiga) não havendo suporte para kernels mais recentes. O cenário atrás referido será portanto impossível de implementar com máquinas Linux actuais numa Intranet, algo que nos parece grave e a exigir uma solução rápida!

Em relação a outros pontos da transição como o DNS existe ainda muito a fazer em relação aos novos formatos A6. Este novo tipo de registo promete mas de momento ainda está em fase de testes, sendo progressivamente corrigido e implementado em

maior número. Para já usam-se os registos AAAA que funcionam do mesmo modo que os seus homólogos de 32 bits.

Com o protocolo IPv6 surgem também alterações nos principais protocolos. A grande maioria foi já actualizada para o suporte do novo tipo de endereços.

Um outro ponto de vista poderá ser o seguinte: com o surgimento de novos ISP a disponibilizarem endereços IPv6 será fácil mudar os endereços de uma intranet para endereços IPv6, ou seja, uma intranet exclusivamente IPv6, neste caso os problemas atrás referidos não se colocam, e é um passo em frente no processo de transição.

Esperamos que este documento possa ajudar quem estiver interessado em migrar para IPv6. Tentámos abordar as questões de modo completo, ou seja, abrangendo a parte teórica e prática da questão filtrando toda a informação errada e incompleta existente de modo a que um técnico de redes por exemplo, já com alguns conhecimentos de IPv6 pudesse implementar algum mecanismo de transição numa rede a seu cargo sem o auxílio de mais documentos.

Referências

RFC´s:

[1] C. HUITEMA; R. AUSTEIN; S. SATAPATI; R. VAN DER POL - *Evaluation of IPv6 Transition Mechanisms for Unmanaged Networks (RFC 3904)*, Setembro 2004, ftp://ftp.rfc-editor.org/in-notes/rfc3904.txt

[2] R. GILLIGAN; E. NORDMARK - *Transition Mechanisms for IPv6 Hosts and Routers (RFC 1933)*, Abril 1996, ftp://ftp.rfc-editor.org/in-notes/rfc1933.txt

[3] G. KEENI; K. KOIDE; K. NAGAMI; S. GUNDAVELLI - *Mobile IPv6 Management Information Base (RFC 4295),* Abril 2006, ftp://ftp.rfc-editor.org/in-notes/rfc4295.txt

[4] R. GILLIGAN; E. NORDMARK - *Transition Mechanisms for IPv6 Hosts and Routers (RFC 2893)*, Agosto 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2893.txt

[5] M-K. SHIN, ED.; Y-G. HONG; J. HAGINO; P. SAVOLA; E. M. CASTRO - *Application Aspects of IPv6 Transition (RFC 4038)*, Março 2005, ftp://ftp.rfc-editor.org/in-notes/rfc4038.txt

[6] R. HINDEN; S. DEERING - *IP Version 6 Addressing Architecture (RFC 4291)*, Fevereiro 2006, ftp://ftp.rfc-editor.org/in-notes/rfc4291.txt

[7] S. DEERING; R. HINDEN - *Internet Protocol, Version 6 (IPv6) Specification (RFC 2460)*, Dezembro 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2460.txt

[8] P. CHRISTIAN - *Generic Routing Encapsulation over CLNS Networks (RFC 3147),* Julho 2001, ftp://ftp.rfc-editor.org/in-notes/rfc3147.txt

[9] F. TEMPLIN; T. GLEESON; M. TALWAR; D. THALER - *Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) (RFC 4214)*, Outubro 2005, ftp://ftp.rfc-editor.org/in-notes/rfc4214.txt

[10] S. HANKS; T. LI; D. FARINACCI; P. TRAINA - *Generic Routing Encapsulation (GRE) (RFC 1701),* Outubro 1994, ftp://ftp.rfc-editor.org/in-notes/rfc1701.txt

[11] J. REYNOLDS; J. POSTEL – *ASSIGNED NUMBERS (RFC 1700),* Outubro 1994, ftp://ftp.rfc-editor.org/in-notes/rfc1700.txt

[12] C. HUITEMA - Teredo*: Tunneling IPv6 over UDP through Network Address Translations (NATs) (RFC 4380),* Fevereiro 2006, ftp://ftp.rfc-editor.org/in-notes/rfc4380.txt

[13] S. THOMSON; C. HUITEMA - *DNS Extensions to support IP version 6 (RFC 1886)*, Dezembro 1995, ftp://ftp.rfc-editor.org/in-notes/rfc1886.txt

[14] M. CRAWFORD; C. HUITEMA - *DNS Extensions to Support IPv6 Address Aggregation and Renumbering (RFC 2874),* Julho 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2874.txt

[15] R. DROMS - *Dynamic Host Configuration Protocol (RFC 2131)*, Março 1997, ftp://ftp.rfc-editor.org/in-notes/rfc2131.txt

[16] S. ALEXANDER – *DHCP Options and BOOTP Vendor Extensions (RFC 2132),* Março 1997, ftp://ftp.rfc-editor.org/in-notes/rfc2132.txt

[17] R. DROMS, ED.; J. BOUND; B. VOLZ; T. LEMON; C. PERKINS; M. CARNEY - *Dynamic Host Configuration Protocol for IPv6 (DHCPv6) (RFC 3315),* Julho 2003, ftp://ftp.rfc-editor.org/in-notes/rfc3315.txt

[18] T. LEMON; B. SOMMERFELD - *Node-specific Client Identifiers for Dynamic Host Configuration Protocol Version Four (DHCPv4) (RFC 4361)*, Fevereiro 2006, ftp://ftp.rfc-editor.org/in-notes/rfc4361.txt

[19] A. CONTA; S. DEERING; M. GUPTA, ED. - *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification (RFC 4443)*, Março 2006, ftp://ftp.rfc-editor.org/in-notes/rfc4443.txt

[20] Y. REKHTER; T. LI - *A Border Gateway Protocol 4 (BGP-4) (RFC 1771)*, Março 2005, ftp://ftp.rfc-editor.org/in-notes/rfc1771.txt

[21] Y. REKHTER, ED.; S. HARES, ED. - *A Border Gateway Protocol 4 (BGP-4) (RFC 4271)*, Janeiro 2006, ftp://ftp.rfc-editor.org/in-notes/rfc4271.txt

[22] J. MOY – *OSPF Version 2 (RFC 2328)*, Abril 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2328.txt

[23] R. DRAVES - *Default Address Selection for Internet Protocol version 6 (IPv6) (rfc 3484)*, Fevereiro 2003, ftp://ftp.rfc-editor.org/in-notes/rfc3484.txt

[24] B. CARPENTER - *Connection of IPv6 Domains via IPv4 Clouds (RFC 3056)*, Fevereiro 2001, ftp://ftp.rfc-editor.org/in-notes/rfc3056.txt

[25] A. CONTA; S. DEERING - *Generic Packet Tunneling in IPv6 Generic Packet Tunneling in IPv6 (RFC 2473)*, Dezembro 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2473.txt

[26] D. FARINACCI; S. HANKS; D. MEYER; P. TRAINA - *Generic Routing Encapsulation (GRE) (RFC 2784)*, Março 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2784.txt

[27] G. MALKIN, R. MINNEAR *RIPng for IPv6 (RFC 2080)*, Janeiro 1997, ftp://ftp.rfc-editor.org/in-notes/rfc2080.txt

[28] G. TSIRTIS, P. SRISURESH - *Network Address Translation - Protocol Translation (NAT-PT) (RFC 2766)*, Fevereiro 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2766.txt

[29] B. HABERMAN, D.THALER - *Unicast-Prefix IPv6 Multicast Addresses (RFC 3306))*, Agosto 2002, *ftp://ftp.rfc-editor.org/in-notes/rfc3306.txt*

[30] R. HINDEN, S. DEERING – I*P Version 6 Addressing Architecture (RFC 2373)*, Julho 1998, *ftp://ftp.rfc-editor.org/in-notes/rfc2373.txt*

[31] E. NORDMARK - *Stateless IP/ICMP Translation Algorithm (SIIT) (RFC 2765)*, Fevereiro 2000, *ftp://ftp.rfc-editor.org/in-notes/rfc2765.txt*

Websites:

[32] The IPv6 Portal, *http://www.ipv6tf.org/*

[33] IPv6 Fórum, *http://www.ipv6forum.com/*

[34] Internet da próxima geração prioridades de acção na migração para o novo protocolo Internet IPv6, *http://www.pt.ipv6tf.org/documentos/geral/Europa/PlanoAccao%20IPv6.pdf*.

[35] Wikipedia – IPv6, *http://pt.wikipedia.org/wiki/IPv6*.

[36] Wikipedia – BGP, *http://en.wikipedia.org/wiki/BGP*.

[37] Wikipedia - 6to4, *http://en.wikipedia.org/wiki/6to4*.

[38] Wikipedia - ISATAP, *http://en.wikipedia.org/wiki/ISATAP*.

[39] Wikipedia - TEREDO, *http://en.wikipedia.org/wiki/Teredo_tunneling*.

[40] Wikipedia - 6over4, *http://en.wikipedia.org/wiki/6over4*.

[41] Debian, *http://people.debian.org/~csmall/ipv6/*.

[42] Swiss IPv6 Task Force, *http://www.ch.ipv6tf.org/?cid=105*.

[43] Redes móveis baseadas nos protocolos IPv4 e IPv6 - uma visão geral do MIP e MIPv6, *http://www.rnp.br/newsgen/0301/mip.html*.

[44] Implementing Tunnelling for IPv6, *http://www.cisco.com/en/US/products/sw/iosswrel/ps5187/products_configuration_guide_chapter09186a00801d6604.html*.

[45] Windows Server TechCenter, http://technet2.microsoft.com/windowsserver/en/library//b2c271bf-abd1-4218-87a9-176dcdd83b1b1033.mspx.

[46] Microsoft TechNet, http://www.microsoft.com/technet/community/columns/cableguy/cg1005.msp

x.

[47] How GRE Keepalives Work, http://www.cisco.com/en/US/tech/tk827/tk369/technologies_tech_note09186a008040a17c.shtml.

[48] IPSec Solution Troubleshooting Tips, http://www.cisco.com/univercd/cc/td/doc/product/rtrmgmt/vpnsc/ipsec/2_1/prov_gd/ipsecpga.htm.

[49] PoP – RS, http://www.pop-rs.rnp.br/index.php?i=dns6.

[50] Manageable Transition Technologies, http://www.ipv6.net.cn/2006/images/zhanghaofeng.pdf#search=%22teredo%20advantages%22

[51] The TCP/IP Guide, http://tcpipguide.com/free/t_DNSChangesToSupportIPVersion6.htm.

Livros:

[52] DAVIES, J. - *Undertstanding IPv6*, Microsoft Press, ISBN 0735612455, 2002

[53] HAGEN, S. - *Ipv6 Essencials, O´Reilly*, ISBN 0596001258, 2002

[54] CIPRIAN P. POPOVICIU, ERIC LEVY-ABEGNOLI, PATRICK GROSSETETE - D*eploying IPv6 networks,* Fevereiro 2006

[55] PETER BIERINGER - *Linux IPv6 Howto*, 2005

[56] MCGRAW-HILL - *Internetworking IPv6 with cisco Routers*

[57] DAVID MALONE; NIALL MURPHY - *Ipv6 Network Administration*, Março 2005

[58] SILVIA HAGEN - *Ipv6 Essentials*, Julho 2002

[59] SAM BROWN; BRIAN BROWNE; NEAL CHEN; PAULJ. FONG; ROBBIE HARRELL; ERIC KNIPP; BART SAYLORS; ROB WEBBER; EDGAR PARENTI JR. - *Configuring IPv6 for Cisco IOS*, 2002

[60] HESHAM SOLIMAN. - *Mobile IPv6: Mobility in a Wireless Internet*, Abril 2004

Apresentações:

[61] JOSEPH DAVIES - *Internet protocol version 6 transition technologies*

[62] FLORENT PARENT; RÉGIS DESMEULES - *IPv6 Tutorial*, Março 2000

[63] Konstantin.Kabassanov@lip6.fr, Vlaidmir.ksinant@6wind.com, Vincent.Jardin@6wind.com - *Euronetlab Implementation of Teredo*

Pdf´s e doc´s:

[64] FLORENT PARENT; MARK BLANCHET - *IPv6 Transition Mechanisms*

[65] DAVID SERAFIM; VITOR SANTOS - *IPv6@ESTG-Leiria_Relatorio_Final_Projecto*

[66] MICROSOFT - *Intra-site Automatic Tunnel Addressing Protocol Deployment Guide*, Maio 2006

[67] CISCO SYSTEMS INC - *IPv6 routing*

[68] *Implementing IPv6 for Cisco IOS software*

[69] *DNS in an IPv6 World*

[70] CISCO SYSTEMS INC - *Implementing Multiprotocol BGP for IPv6*

[71] CISCO SYSTEMS INC – Ipv6: *Connecting to the 6bone Using 6to4 Tunnels,* Maio 2003

[72] MICROSOFT – *centro de ajuda e suporte*

[73] 6WIND, *the New Internet*

[74] JÁNOS MOHÁCSI – IPv6 host Configuration

[75] MÍCHEÁL Ó FOGHLÚ – M-Zones: Infrastructure Requirements for Smart Spaces and Managed Zones