Instituto Politécnico de Leiria

Escola Superior de Tecnologia e Gestão

IPv6@ESTG-Leiria

Instalação de uma Rede Piloto

David Luís Santos Serafim

Vítor André Cordeiro dos Santos

Leiria

Julho de 2005

Instituto Politécnico de Leiria

Escola Superior de Tecnologia e Gestão

IPv6@ESTG-Leiria

Instalação de uma Rede Piloto

Relatório final da cadeira de Projecto I, do curso de Licenciatura em Engenharia Informática e Comunicações, ano lectivo 2004/2005

Realizado entre Março e Julho de 2005

Autores:

David Luís Santos Serafim, n.º 9035

Vítor André Cordeiro dos Santos, n.º 10451

Orientador: Prof. Mário Antunes

Co-orientador: Prof. Nuno Veiga

Leiria

Julho de 2005

Agradecimentos

Ao orientador do projecto, Prof. Mário Antunes, pela preciosa orientação e incentivo, e paciência quase sem limites.

Ao co-orientador do projecto, Prof. Nuno Veiga, pelas valiosíssimas sugestões e espírito crítico.

Ao Centro de Informática da ESTG-Leiria, nas pessoas do Eng. Carlos Canudo e Eng. Adaíl Ferreira, pela ajuda indispensável no estabelecimento da ligação ao exterior.

Ao Carlos Friaças, um dos responsáveis na área de IPv6 da FCCN, pela inigualável e indispensável ajuda ao nível da configuração e estabelecimento da ligação ao exterior.

Ao David Malone e ao Geoff Huston, pela disponibilidade imediata no esclarecimento de dúvidas.

A todos os que contribuíram directa ou indirectamente para a realização deste projecto.

Ao pessoal de Projecto I de EIC.

Às nossas namoradas, Laura e Marlene, pelo apoio, incentivo e dedicação incondicionais.

Resumo

O transporte dos dados na Internet é assegurado pelo protocolo IP (versão 4). Embora a sua utilização se tenha inicialmente revelado adequada, tornou-se necessário implementar medidas de ajuste à evolução da Internet e ao seu crescimento exponencial. Por exemplo, o NAT, o VLSM e o sub-endereçamento surgiram para minimizar o desperdício de endereços. Também o CIDR se revelou indispensável no encurtamento das tabelas de encaminhamento e consequente melhoria de processamento pelos *routers*. O aparecimento de novos paradigmas de comunicação assentes na mobilidade efectiva mostrou também inadequação do protocolo IPv4 e necessidade de mudança.

O objectivo da versão 6 do protocolo IP (IPv6), desenvolvido pelo IETF, consiste em resolver algumas inadaptações do IPv4 face ao cenário actual da Internet. Destacam-se como pontos fortes a estrutura e dimensão do espaço de endereçamento, a auto-configuração dos terminais, a simplificação do processamento nos *routers* e os cuidados relacionados com a segurança e a mobilidade. O IPv6 encontra-se actualmente numa fase madura de desenvolvimento, como o comprovam as implementações estáveis na maioria das plataformas de sistemas operativos. Também os principais serviços da Internet se encontram implementados em ambas as versões.

Este projecto pretende descrever a implementação de uma rede piloto IPv6 no Departamento de Engenharia Informática (DEI) da Escola Superior de Tecnologia e Gestão de Leiria (ESTG-Leiria). A rede é heterogénea, constituída por terminais com sistemas operativos distintos: Microsoft Windows XP Professional SP2, Linux Fedora Core 3, Open BSD, Apple Mac OS X e Cisco IOS.

O projecto desenvolvido divide-se em duas partes: estudo abrangente da tecnologia IPv6 e a implementação da rede IPv6. A primeira parte apresenta os aspectos diferenciadores do protocolo, recorrendo a testes de funcionalidade numa rede piloto. A segunda parte descreve as principais acções a desenvolver na migração de uma rede para IPv6. Este ponto contempla a configuração do acesso à Internet em IPv6, através da FCCN, usando duas soluções distintas: recorrendo a um túnel IPv6 sobre IPv4 e através de acesso nativo em IPv6. Evidenciam-se ainda as alterações a efectuar nos principais serviços (HTTP, DNS, DHCP), bem como na configuração dos terminais da rede.

É um facto que o protocolo IPv6 será adoptado na Internet, havendo necessidade de estudar a tecnologia e os processos de migração adequados. Assim, a implementação desta rede piloto pretende aprofundar o conhecimento da tecnologia e contribuir com boas práticas para a migração de uma rede para IPv6.

Índice

Lista de Siglas e Acrónimos

A	Address
AAAA	Quad A
AfriNIC	African Internet Numbers Registry IP Addresses
AH	Authentication Header
AIX	Advanced IBM Unix
ALICE	America Latina Interconectada Con Europa
APNIC	Asia Pacific Network Information Centre
APT	Advanced Package Tool
ARIN	American Registry for Internet Numbers
ARP	Address Resolution Protocol
ARPA	Advanced Research Projects Agency
ASCII	American Standard Code for Information Interchange
ATM	Asynchronous Transfer Mode
BGP	Border Gateway Protocol
BGP4+	Border Gateway Protocol 4+
BIND	Berkeley Internet Name Domain
BOOTP	Bootstrap Protocol
BSD	Berkeley Software Distribution
CD	Compact Disc
CDMA	Code Division Multiple Access
CE	Cabeçalho de Extensão
CE	Compact Edition (Windows)
CI	Centro de Informática
CIDR	Classless Inter-Domain Routing
CIFS	Common Internet File System
DAD	Duplicate Address Detection
DEC	Digital Equipment Corporation
DEI	Departamento de Engenharia Informática
DHCP	Dynamic Host Configuration Protocol
DHCPv4	Dynamic Host Configuration Protocol for IPv4
DHCPv6	Dynamic Host Configuration Protocol for IPv6
DiffServ	Differentiated Services
DNS	Domain Name System

DoD	Department of Defense
DoS	Denial of Service
DS	Differentiated Services
ESP	Encapsulating Security Payload
ESTG	Escola Superior de Tecnologia e Gestão de Leiria
ESTG-Leiria	Escola Superior de Tecnologia e Gestão de Leiria
EUA	Estados Unidos da América
EUI-64	64-bit Extended Unique Identifier
FCCN	Fundação para a Computação Científica Nacional
FDDI	Fiber Distributed Data Interface
FTP	File Transfer Protocol
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HP-UX	Hewlett-Packard Unix
HTTP	Hypertext Transfer Protocol
HTTPD	Hypertext Transfer Protocol Daemon
IA	Identify Association
IANA	Internet Assigned Numbers Authority
IBM	International Business Machines Corporation
ICANN	Internet Corporation for Assigned Names and Numbers
ICMP	Internet Control Message Protocol
ICMPv4	Internet Control Message Protocol for IPv4
ICMPv6	Internet Control Message Protocol for IPv6
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IGMPv3	Internet Group Management Protocol Version 3
IHL	Internet Header Length
IIS	Internet Information Server
IMAP	Internet Message Access Protocol
IntServ	Integrated Services
IP	Internet Protocol
IPng	Internet Protocol Next Generation
IPsec	Internet Protocol 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

IS-ISv6	IS-IS for IPv6
ISP	Internet Service Provider
KNOPPIX	Knopper Linux
LACNIC	Latin American and Caribbean Internet Addresses Registry
Linux	Linus Torvalds' UNIX
LIR	Local Internet Registry
Mac	Macintosh
MAC	Media Access Control
Mac OS X	Macintosh Operating System X
ME	Millennium Edition (Windows)
MIP	Mobile Internet Protocol
MLD	Multicast Listener Discovery
MLDv2	Multicast Listener Discovery Version 2
MPLS	Multi Protocol Label Switching
MTU	Maximum Transmission Unit
NAT	Network Address Translator
NAT-PT	Network Address Translation-Protocol Translation
ND	Neighbor Discovery
NNTP	Network News Transfer Protocol
NUD	Neighbor Unreachability Detection
OpenVMS	Open Virtual Memory System
OS	Operating System
OSI	Open Systems Interconnect
OSPF	Open Shortest Path First
OSPFv3	OSPF for IPv6
PDA	Personal Digital Assistant
POP	Post Office Protocol
PPP	Point-to-Point Protocol
PTR	Pointer
PVC	Permanent Virtual Circuit
QoS	Quality of Service
RCTS	Rede de Investigação e Ensino Nacional
RD	Router Discovery
RFC	Request For Comments
RFID	Radio Frequency Identification
RIP	Routing Information Protocol
RIPE	Réseaux IP Européens
RIPng	Routing Information Protocol Next Generation
RIR	Regional Internet Registry

RPC	Remote Procedure Call
RPM	Red Hat Package Manager
SIIT	Stateless IP/ICMP Translation
SMB	Server Message Block
SMTP	Simple Mail Transfer Protocol
SP	Service Pack
SSH	Secure Socket Shell
TCP	Transmission Control Protocol
TELNET	Network Virtual Terminal Protocol
TLS	Transport Layer Security
ToS	Type of Service
TRT	Transport Relay Translation
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
VoIP	Voice over Internet Protocol
WebDAV	Web-based Distributed Authoring and Versioning
WINS	Windows Internet Naming Service
Winsock	Windows Sockets
WLAN	Wireless Local Area Network
XP	Experience (Windows)
Yum	Yellow dog Updater, Modified

Lista de Figuras

Lista de Tabelas

1. Introdução

O IPv6 é um novo protocolo que abrange diversos assuntos relacionados com diferentes ramos das tecnologias de rede. Além do endereçamento, que é directamente afectado, encaminhamento, qualidade de serviço, mobilidade, segurança e mesmo alguns protocolos, são alguns dos temas que vêem alguns dos seus conceitos modificados. Sendo assim, além de uma nova linguagem de camada de rede, irá existir uma reacção em cadeia para todo um conjunto relacionado com as redes. A reacção terá sempre como objectivo o melhorar de cada tecnologia. Outro objectivo não menos importante é garantir facilidades a todo um processo de administração de redes, assegurando também o melhor desempenho da rede em questão.

O nascimento do IPv6 deve-se a motivos que constituem factos absolutamente necessários, protagonizados principalmente pelo crescimento da Internet e inclusão da mesma em diversos meios, de modo a facilitar qualquer forma de acesso à mesma. Diversas instituições contribuíram e continuam a contribuir, directa ou indirectamente, em todo o desenvolvimento e adopção do protocolo, pelo que uma referência a todo esse trabalho se revela essencial. Espera-se um futuro risonho, onde o IPv6 se tornará bastante popular, devendo atingir o nível de sucesso do seu antecessor. As diferenças entre os dois são variadas, sendo fundamental uma comparação clara tanto a nível de protocolos, como de implicações que o IPv6 terá em outros ramos de redes. Enquadrar o IPv6 em tudo o que existe actualmente no mundo das redes é também preponderante, com a necessidade acrescida de ser motivador.

Alguns dos pontos de passagem obrigatória no estudo desta nova tecnologia são a compreensão dos diversos mecanismos e protocolos envolventes que possui, o tipo e estrutura de mensagens que usa e o endereçamento. Toda esta abordagem será dada com um sentido prático e claro, explicando e exemplificando. Não serão esquecidas algumas referências obrigatórias ao antigo protocolo.

O sentido prático deverá, então, ser aplicado à própria prática na tecnologia IPv6, num processo de implementação cuidado, não esquecendo a diversidade de suporte IPv6 proporcionada pelos diversos sistemas operativos, aplicações e outros projectos. Deste modo, serão então criados cenários com o maior grau de heterogeneidade entre os diversos sistemas operativos com suporte IPv6, disponibilizando também diferentes serviços IPv6 nas respectivas redes. A abordagem será sempre feita num processo faseado de fácil compreensão, retirando-se as devidas conclusões em cada passo. Ligações à rede exterior (Internet) serão também consideradas, sendo explicadas diferentes perspectivas das formas de efectuar essa ligação.

É fundamental contribuir para a maior adopção deste protocolo por parte das instituições, pois ele irá ser absolutamente necessário num futuro próximo. Não existe melhor forma de contribuir para isso, do que explicar e implementar o protocolo na instituição. Incentivar explicando, é o principal objectivo deste projecto.

2. Enquadramento

Uma nova tecnologia traz sempre consigo bastantes vantagens. Resoluções de antigos problemas, facilidade de utilização e melhorias de desempenho são normalmente algumas delas. Mas, apesar disso, existe sempre um lado que torna a sua aceitação/implementação num processo mais demorado do que se poderia imaginar. Situações como a desconfiança sobre a tecnologia, os custos, a não necessidade/vontade de mudar e principalmente a falta de conhecimentos sobre a nova tecnologia, contribuem sempre para um abrandamento no processo de actualização.

O IPv6 não foge à regra, e é principalmente devido a estas últimas situações que um enquadramento correcto nesta nova tecnologia se revela importante, com a necessidade acrescida de ser claro e motivador.

2.1. Necessidade de um novo Protocolo da Internet

Imagine-se um mundo móvel, com uma lista extensa de dispositivos móveis ou não, mas todos a necessitar de um endereço unívoco, desde computadores portáteis, PDAs, telemóveis, electrodomésticos, automóveis, sensores inteligentes, dispositivos bio-electrónicos, robôs, ou até alguns dispositivos que ainda nem sequer existem. Com o número limite de endereços associado ao protocolo actual a aproximar-se do fim, dia após dia, a necessidade de um novo protocolo emerge. Esta utopia de interligação de múltiplos dispositivos numa rede global vai um dia deixar de o ser, tornando-se um hábito dos nossos dias.

Um novo paradigma de computação está cada vez mais a emergir, o conceito de computação ubíqua (*ubiquitous computing*). Este conceito é o contrário da realidade virtual. No caso da realidade virtual nós entramos no mundo computacional, mas no caso da computação ubíqua é o computador que entra no nosso mundo, adquire os nossos hábitos, faz parte da nossa vida mais rotineira.

Ligar o fogão para aquecer o jantar, quando se está prestes a chegar a casa, abrir a porta ao homem da luz quando se está em viagem. A imaginação poderá ser da mais variada, mas nada disto será plausível sem a respectiva segurança e qualidade de serviço associada, e com isso voltamos à necessidade de um novo protocolo.

Juntando então, às necessidades acima referidas, alguma falta de eficiência do IPv4 (ver Secção 2.4), os problemas do NAT, a extrema necessidade em ter ligações ponto-a-ponto e outros diversos problemas e vantagens que irão sendo referenciados, e temos a razão para o novo protocolo IPv6.

O IPv6 é bem mais do que mais endereços, ao contrário do que muitas vezes se pensa. A necessidade existe também pelo número de endereços, mas não apenas por eles.

O cenário utópico de dispositivos a que o IPv6 se propõe poderá não ser assim tão quimérico como possa parecer, e isso compreende-se facilmente acompanhando certas afirmações proferidas há alguns anos atrás por algumas personalidades no que diz respeito à tecnologia:

- "*It would appear that we have reached the limits of what it is possible to achieve with computer technology, although one should be careful with such statements, as they tend to sound pretty silly in 5 years.*" – John Von Neumann, 1949
- "*I think there is a world market for maybe five computers.*" – Thomas Watson, IBM, 1943
- "*640k should be enough for anybody.*" – Bill Gates, Microsoft Corporation, 1981
- "*32 bits ought to be enough address space.*" – Vint Cerf, 1977

- "*There is no reason for any individual to have a computer in his home.*" – Ken Olsen, Digital Equipment Corporation (DEC), 1977

Muitas das personalidades acima referidas, são consideradas com as mentalidades mais avançadas e desafiadoras que se podia imaginar para a época em questão, o que cria uma sensação de desproporção de ideias maior do que o cenário em que "tudo" no futuro irá ter um endereço próprio associado. Em 1981 era praticamente impensável que cerca de 4 mil milhões de endereços não fossem suficientes. A tecnologia já chegou, ficou e agora alastra-se cada vez mais a um ritmo estonteante, de forma que consegue tornar obsoleta qualquer afirmação menos arriscada que se faça sobre ela.

O IPv6 é um pouco controverso no que diz respeito à sua necessidade actual. Opções a curto prazo têm sido escolhidas em detrimento do IPv6, mas o desenvolvimento dessas estratégias tende a tornar-se desmotivador. É como fugir sabendo que mais tarde ou mais cedo se vai ser apanhado.

A nova versão do protocolo não é uma necessidade urgente, como o mediático *bug* do ano 2000. O IPv6 implicará uma movimentação bem mais vagarosa por parte do mundo empresarial, ao contrário do que aconteceu com o *bug*. Mas esta necessidade é real e a força que mais a impulsiona é o rápido crescimento da Internet [23].

Lista de necessidades

As necessidades associadas ao IPv6 podem definir-se sucintamente como:

- Protocolo com mais capacidade de endereços.

  O crescimento exponencial da Internet, junto com a maior abrangência geográfica da mesma, a isso implica.

- Protocolo auto-configurável (ver Secção 3.8).

  O facto de tudo no mundo informático poder ser *Plug-and-Play*[1] parece ser o caminho correcto.

- Protocolo mais seguro (ver Subsecção 2.4.4).

  Mais Internet, mais *e-business* (negócios na Internet), *e-shopping* (compras na Internet), *e-banking* (acesso aos bancos pela Internet), logo necessariamente mais responsabilidade e segurança.

- Protocolo com mais responsabilidade a nível de QoS (ver Subsecção 2.4.5).

  O tráfego não pode ser tratado de igual forma, devendo definir-se prioridades de tratamento.

- Protocolo com vista à mobilidade (ver Subsecção 2.4.6).

  Telemóveis e Internet tendem cada vez mais a convergir.

- Protocolo mais eficiente (ver Subsecção 2.4.1).

  Aprender com os erros do IPv4.

- Protocolo virado para o modelo ponto-a-ponto (ver Subsecção 2.4.4).

  Porque é essencial não existir alterações intermédias numa ligação.

- Mecanismos de transição entre os dois protocolos (ver Secção 3.10).

  A transição não vai ser repentina, pelo que a coexistência entre os dois protocolos é de importância extrema.

Em toda esta lista, está sempre presente no horizonte do IPv6, a facilidade e simplicidade de qualquer tipo de configuração necessária. Assim sendo, tanto o leigo, como o especialista do ramo, têm as suas tarefas facilitadas.

[1] Conceito em que o *hardware* pode ser ligado ao computador que é reconhecido automaticamente. O computador procederá de seguida às alterações necessárias para o seu correcto funcionamento.

Números da necessidade

A seguir são apresentados alguns números que estão de certa forma associados à nova tecnologia e à sua necessidade.

- O limite teórico de endereços IPv4 é de 4 mil milhões (4.294.967.296 – 2^{32}), mas na prática apenas cerca de 250 milhões podem ser alocados por utilizadores.
- $3{,}4 \times 10^{38}$ (340.282.366.920.938.463.463.374.607.431.768.211.456 – 2^{128}) é o número teórico de endereços associado ao IPv6. $6{,}7 \times 10^{23}$ (665.570.793.348.666.943.898.599) é aproximadamente o número de endereços possíveis de atribuir por metro quadrado do Planeta Terra. Considerando as hierarquias, no pior caso podem ser usados 1500 endereços por cada metro quadrado.
- Considerando que a quantidade de endereços IPv4 irá toda migrar para IPv6, ainda assim 85% do espaço de endereçamento sobra para uso futuro [40].
- "As projecções apontam para 1000 milhões de automóveis em 2010. Considerando que apenas 15% desses automóveis terão necessidade de IP, isso significa 150 milhões de endereços IP." [24]

É bastante visível a grande diferença no que diz respeito à quantidade de endereços entre os dois protocolos. Muitos pensam ainda hoje da mesma forma que se pensou em 1981 com o IPv4: "4 mil milhões de endereços chegam perfeitamente", mas a aprendizagem com os erros do passado tornou possível esta quantidade gigantesca de endereços.

- Apenas 5% da população mundial tem acesso à Internet [183].

É impressionante como esta minoria mundial conseguiu quase esgotar todos os endereços IPv4. Isto não se deve tanto à quantidade de computadores por pessoa nos países mais desenvolvidos, mas sim a más políticas associadas à alocação inicial dos endereços.

Em relação aos países menos desenvolvidos, existem governos completamente contra qualquer avanço da Internet no seu país. Isto acontece devido à abertura de ideias que a Internet pode proporcionar. Muitos governos conservadores e até manipuladores vêem na Internet uma forma de libertação intelectual que pode acontecer em desfavor deles. Mas, à medida que os anos passam, são menos os governos que conseguem suprimir a Internet do seu território, ficando então também os países menos desenvolvidos a precisar de endereços IP, aumentando assim a necessidade do IPv6.

- "A indústria de telemóveis aponta para 1000 milhões de utilizadores no final de 2005. Isto significa 2000 milhões de endereços IP, caso seja atribuído 2 endereços por utilizador." [24]

Foram considerados dois endereços por utilizador, pois o conceito de mobilidade associado ao IPv6 propõe a criação de um endereço de casa (*home address*) e outro para quando se está noutra rede que não seja a de casa (*roaming*). Isto acontece devido ao facto de o utilizador, quando sai da sua rede, necessitar imediatamente de estabelecer contacto com outra rede (a mais próxima possível), e o conjunto de endereços dessa rede será diferente dos da sua antiga rede (a de casa) (ver Subsecção 2.4.6).

O uso de IPv6 é mandatário para o IMS[2] [92], o que significa que o UMTS, e mesmo o GPRS, deviam usar IPv6. É absolutamente necessário o uso do IPv6 na indústria das telecomunicações móveis. Qualquer tecnologia móvel das novas gerações irá sempre ter o IPv6 como uma especificação mandatária.

Na Figura 2.1 verificamos a relação entre as subscrições de telemóveis e Internet móvel no passado recente, e as previsões para o futuro próximo.

[2] Tecnologia IP Multimédia e Telefónica, baseada nos protocolos existentes do IETF, independente do método de acesso, podendo ser usada em diversas tecnologias (UMTS, GPRS, WiMAX, etc.).

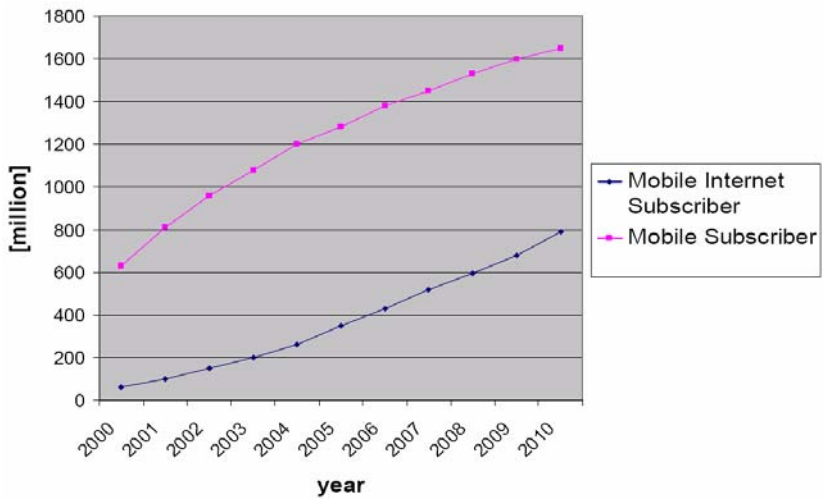


Figura 2.1 – Relação entre subscrições de telemóveis e Internet móvel (Extraído de [172]).

- Poucas empresas têm servidores para auto-configuração de endereços [23].
- Um estudo sugeriu que a configuração automática do IPv6 paga-se a ela própria em 12 meses comparando com a configuração manual do IPv4.

A maioria das empresas não tem os endereços IP atribuídos automaticamente, o que adivinha carga de trabalho adicional a cada modificação efectuada na rede. Com o IPv6 poupa-se em todo este trabalho, pois o conceito de auto-configuração já está definido à partida. Na Figura 2.2 tem-se a relação entre os custos de manutenção do IPv4 e do IPv6 e o custo de transição entre os dois protocolos.

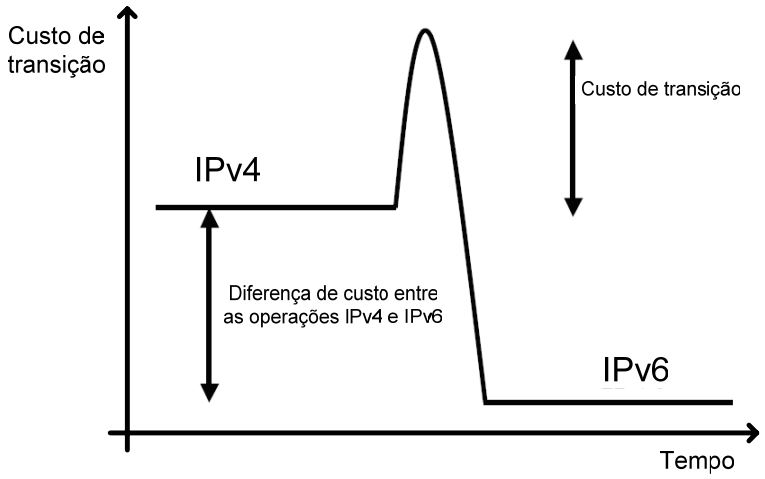


Figura 2.2 – Custo de transição do IPv6 (Adaptado de [13]).

- Estudo sugere que 60% das companhias têm endereços ilegais[3] algures na sua rede [23].

Devido à escassez de endereços IPv4, muitas empresas recorrem a endereços ilegais. Esta situação tende a piorar à medida que a escassez de endereços IPv4 aumenta. Existiu pelo menos um caso documentado, em que uma pequena empresa usava endereços completamente aleatórios causando sérios problemas a uma grande multinacional, pelo simples facto de se ligar à Internet.

- Dois programadores do Canadá demoraram apenas 32 horas para recuperar as fontes de código de um servidor público de Quake, descobrir onde fazer as modificações para a aplicação funcionar com IPv6, fazer as modificações, configurar o servidor e jogar a primeira vez com esse servidor com IPv6 [31].

[3] Endereços que não estão atribuídos pelas entidades competentes; poderão ser duplicados ou não.

Esta é a prova de que programar as aplicações para suportar o novo protocolo, não é assim tão complicado. É claro que a dificuldade diverge de código para código, conforme a sua boa estruturação.

- Existe uma universidade nos EUA que tem uma classe A de endereços IPv4 inteira atribuída, o que dá cerca de 18 milhões de endereços, para uma população de 18 milhares de indivíduos. Em contrapartida, a China, o país com maior densidade populacional, tem cerca de 20 milhões de endereços IP atribuídos [25]. Na Figura 2.3 pode ver-se precisamente um gráfico onde a desproporção entre a quantidade de endereços e a população é evidente.

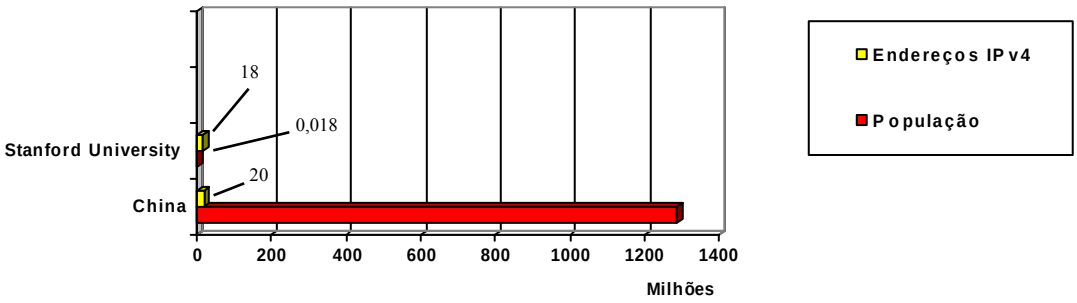


Figura 2.3 – Comparação da atribuição de endereços entre a Stanford University e a China.

- Cerca de 50% das ligações à Internet são ponto-a-ponto [40].

Para a computação ubíqua começar a fazer sentido, é essencial o uso de ligações ponto-a-ponto. Para que os dispositivos possam inserir-se na vida quotidiana das pessoas, não poderão existir intermediários nas ligações estabelecidas. O conceito terá de ser obrigatoriamente de origem para destino, com o máximo de privacidade.

2.2. Desenvolvimento

O desenvolvimento do IPv6, tal como qualquer outra tecnologia que se pretende afirmar, teve vários pontos essenciais, uns mais do que os outros, mas todos eles devem ser lembrados. Fala-se tanto no futuro do IPv6, que por vezes até se esquece o seu passado de igual importância. As especificações do IPv6 do IETF, o RFC 1883 e o mais actual RFC 2460, são o maior passo na normalização do IPv6, mas, por exemplo, o projecto 6Bone é o maior passo na igualmente importante parte prática deste novo protocolo. Depois disso seguem-se inúmeros projectos que conjugam de certa forma, a parte teórica, prática e principalmente a conversação entre os diversos países.

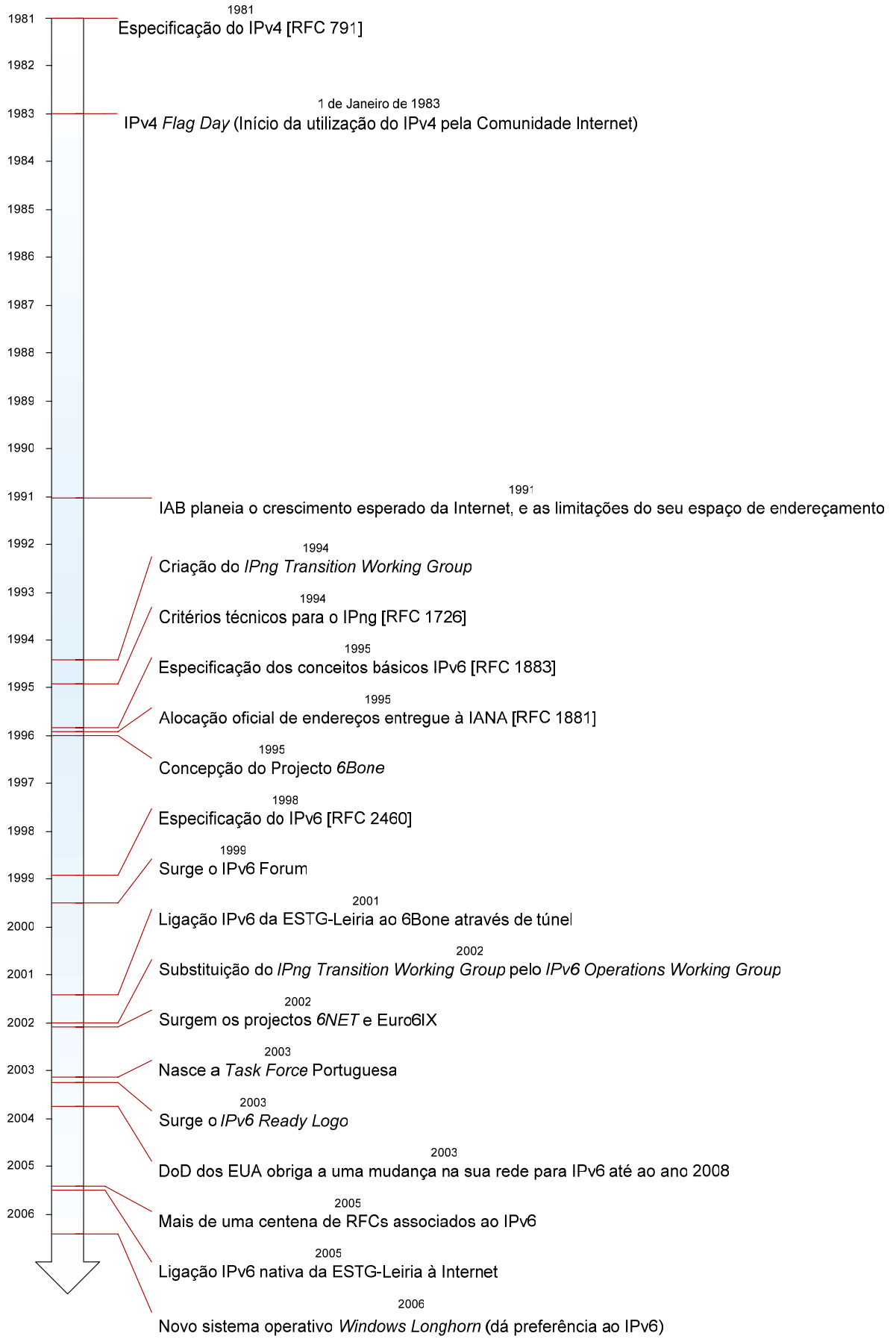


Figura 2.4 – Cronograma com alguns dos marcos do IPv6.

No cronograma da Figura 2.4 podem ver-se alguns dos marcos mais importantes do IPv6. Pelo facto de este protocolo ainda continuar em fase de adopção e desenvolvimento, muitos mais marcos existirão no futuro. Por agora irão sendo referidas as entidades mais importantes, por ordem cronológica.

2.2.1. IETF

O IETF (*Internet Engineering Task Force*), com os seus grupos de trabalho e respectivos RFCs (ver [103]), revelou-se fulcral ao longo de todo o tempo de desenvolvimento do protocolo. Mesmo estando o novo protocolo bem especificado, o IETF continua a ter uma grande importância, nomeadamente ao nível da interacção do IPv6 com outras tecnologias, e ao nível de outras informações sobre o funcionamento detalhado de algumas partes mais específicas do IPv6. Os RFCs começaram por ser meramente informativos sobre o que era necessário, tendo depois uma fase de espera, verificando o que estava a ser feito. Do que foi feito e discutido, e mais se aproximava do necessário, surgiu a especificação. Curiosamente, verifica-se que grande parte dos RFCs mais recentes está relacionada com o IPv6: a relação de várias tecnologias com o IPv6, determinados campos ainda em estudo do IPv6, mecanismos de transição, etc.

Dados:

- Início: 1986.
- Término: Provavelmente nunca.
- Objectivos:
  - Promove e desenvolve os *standards* da Internet.
- Outras Notas:
  - Organização baseada no voluntariado, não possui qualquer tipo de inscrição formal para membros;
  - Sem fins lucrativos.
- Mais informação: *http://www.ietf.org/*

O IPv4 foi especificado pelo IETF. As razões do sucesso que teve este protocolo, são sérias razões a analisar para o correcto caminho do IPv6 no domínio da sua adopção. Algumas razões são [6]:

- A simplicidade arquitectónica;
- Disponibilidade de documentos técnicos de um modo bastante simples, e praticamente todos eles grátis;
- Implementações práticas, à medida do ritmo teórico;
- A excelente qualidade das implementações;
- O suporte de inúmeras tecnologias de comunicação.

O IPv6 não se esqueceu de tudo o que se passou com o IPv4, pelo que foram mantidas muitas das abordagens feitas com o antigo protocolo. O objectivo não é destruir e esquecer tudo o que foi feito, mas sim aproveitar tudo o que de melhor se fez e, se possível, melhorar.

Não foi aqui que tudo começou, é certo. Provavelmente o protocolo IPv4 aprendeu também com o seu antecessor, mas foi aqui que se deu o salto. É com base em 25 anos de experiência neste protocolo que nasce o IPv6.

Dados:

- Início: 1 de Janeiro de 1981.
- Término: Provavelmente nunca.
- Objectivos:
  - Mover os pacotes num conjunto de redes distintas (p. ex., Internet);
  - Ser o protocolo principal da camada de rede;
  - Implementar algumas funcionalidades como a fragmentação, algumas opções (ver Subsecção 2.4.2) e alguma QoS (ver Subsecção 2.4.5);
  - Definir um modelo de endereçamento que identifica univocamente redes e máquinas.

2.2.2. 6Bone

O 6Bone foi criado para que o IETF pudesse passar da teoria à prática. Em termos de IPv6, foi sem dúvida o primeiro passo rumo à implementação do novo protocolo. Apesar de funcionar um pouco à parte do IETF, serviu como complemento, sendo o *IPng Transition working group* do IETF (ver [105]) o criador da rede 6Bone. Depois disso este grupo sofreu modificações a nível de critérios, passando a intitular-se *IPng Operations* (ver [106]). Este novo grupo não incluía o suporte ao 6Bone efectuado pelo antigo.

Outra motivação para o 6Bone era o facto de os endereços públicos serem grátis. Desta forma o 6Bone era como um projecto formal a que qualquer pessoa podia pertencer. Para isso, foi disponibilizado um prefixo temporário pela IANA (*3FFE::/16*) para ser utilizado por qualquer tipo de utilizador ligado ao projecto, funcionando o 6Bone como uma empresa de endereços grátis para testes experimentais de qualquer utilizador mais curioso em relação ao IPv6. Por enquanto, ainda se consegue ter acesso a um endereço desses através de algumas companhias (p. ex., a Hexago com a rede FreeNet), mas como tudo o que é experimental tem o seu término, o final dos endereços de teste 6Bone tem a data marcada para Junho de 2006 [80]. O projecto 6Bone irá, então, funcionar noutra perspectiva já com endereços públicos oficiais.

O 6Bone conseguiu reunir milhares de máquinas em mais de 50 países diferentes, provando assim, o sucesso do IPv6.

Dados:

- Início: 1996.
- Processo de transição: Junho de 2006.
- Término: desconhecido.
- Objectivos:
  - Ser a primeira plataforma de testes (*testbed*) IPv6;
  - Começar como uma rede virtual, usando túneis IPv6 sobre IPv4 (ver Subsecção 3.10.2) e transitar para endereços públicos IPv6 nativos.
- Mais informação: *http://www.6Bone.net/*

2.2.3. IPv6 Forum

Desde a sua criação, o *IPv6 Forum* tem-se revelado extremamente importante pelo efeito da troca de ideias que tem proporcionado, devido principalmente às conferências organizadas por todo o mundo. É praticamente o gestor de todas as conferências mundiais existentes sobre IPv6, intitulando-se as principais conferências "*IPv6 Global Summit*". Este tipo de conferências realiza-se mensalmente, sempre num país diferente. A ideia é incentivar o IPv6 em todo o Globo, e o conteúdo das conferências refere-se também ao estado nacional da tecnologia, mas nunca esquece o cariz global da conferência. A primeira conferência foi a "*IPv6 Global Summit Paris 1999*", em Agosto (ver [97]), e desde aí já passou por diversos países e continentes (ver [96]).

É um consórcio que junta todos os peritos no novo protocolo, sendo presidido pelo fundador Vint Cerf[4], pelo que não pretende esclarecer leigos, mas sim estabelecer estratégias de desenvolvimento.

Dados:

- Início: Julho de 1999.
- Término: Provavelmente quando o IPv6 conseguir atingir a popularidade do IPv4.
- Objectivos:
  - Estabelecer um fórum destinado principalmente a todos os especialistas da nova tecnologia;
  - Promover o uso das aplicações com suporte para IPv6;
  - Resolver os problemas relacionados com todas as barreiras existentes para a adopção do IPv6.
- Mais informação: http://www.ipv6forum.com/

2.2.4. 6NET e Euro6IX

São projectos com a mesma base prática do 6Bone mas funcionam mais como uma forma de reunião da Europa em torno do IPv6. No início usavam endereços do 6Bone, mas depois adoptaram endereços públicos próprios. Apenas estão acessíveis a entidades comerciais e académicas. A diferença entre estes dois projectos é que o 6NET é mais vocacionado para o mundo académico e o Euro6IX mais para o mundo empresarial. Em termos nacionais, a entidade portuguesa ligada ao 6NET é a FCCN e a entidade ligada ao Euro6IX é a PTInovação.

Dados:

- Início: 1 de Janeiro de 2002.
- Término: Inicialmente previsto para 31 de Dezembro de 2004, duração de 3 anos, mas foi alargado até 30 de Junho de 2005.

[4] Conhecido por "Pai da Internet", foi co-autor do protocolo TCP/IP e orientou o projecto que iria ser o primeiro serviço comercial de e-mail ligado à Internet. Autor também de diversos RFCs, foram-lhe atribuídos diversos prémios de mérito pelo seu grande contributo na vertente tecnológica mundial.

- Objectivos:
  - Criar uma rede piloto IPv6 europeia;
  - Testar estratégias de migração da actual estrutura IPv4;
  - Testar vários tipos de serviços, avaliar a alocação de endereços, o encaminhamento e o DNS;
  - Promover o IPv6 na Europa.
- Mais informação: *http://www.6net.org/* e *http://www.euro6ix.org/*

O término de projectos com endereços de teste, ao contrário do que se possa pensar, não é mau sinal de todo. Aliás, quanto mais cedo terminarem, melhor será para o estabelecimento a curto prazo do IPv6. Isto acontece pois os respectivos endereços terão efectivamente de voltar a ser entregues à IANA e esta, por fim, irá atribuir de forma definitiva esses endereços. Enquanto existirem estas redes de teste mantém-se o carácter experimental do IPv6, por isso é bom sinal o seu término.

2.2.5. FCCN e Task Force Portuguesa de IPv6

A *Task Force* Portuguesa é o movimento nacional de IPv6. Tem como orientação lembrar a Portugal que existe IPv6 e tentar contagiar entidades para o seu desenvolvimento. Foi criada pela FCCN, sendo esta que a controla. A FCCN por sua vez está envolvida em variados projectos IPv6: 6NET (ver Subsecção 2.2.4), 6DISS e ALICE (ver [112]). O 6DISS é um projecto que vem na continuação do 6NET mas que se irá alargar para fora da Europa; o ALICE é um projecto que tem como objectivo a ligação IPv6 entre a Europa e a América Latina. A rede RCTS[5] ligada ao projecto GÉANT2 (ver [111]) também já possui suporte IPv6 nativo.

A FCCN também organiza conferências e seminários acerca do estado do IPv6 no país, revelando-se esses eventos num dos principais incentivos à implementação do IPv6.

Dados:

- Início: Fevereiro de 2003
- Término: Provavelmente quando em Portugal, o IPv6 tenha a popularidade do IPv4.
- Objectivos:
  - Criação de um *website* em português, devidamente actualizado com informações sobre a *Task Force* e o IPv6 em geral;
  - Identificar os serviços e as aplicações onde o IPv6 será determinante;
  - Identificação das barreiras portuguesas ao processo de transição para IPv6;
  - Divulgação do IPv6;
  - Criação de uma base de dados em português, com informação relevante sobre o tema.
- Mais informação: *http://www.fccn.pt/* e *http://www.ipv6-tf.com.pt/*

[5] Rede de investigação e ensino nacional, de alta largura de banda e que interliga as principais Universidades e instituições do país.

2.2.6. IPv6 Ready Logo

O *IPv6 Ready Logo* foi criado pelo *IPv6 Forum*, e é especialmente importante na medida em que transmite uma motivação acrescida para o suporte IPv6 por parte dos dispositivos e aplicações. Funciona como um selo de qualidade IPv6, e atribui até agora duas categorias de emblemas, conforme o nível de suporte IPv6 associado, fase 1 ou fase 2. As entidades que desejem usufruir do emblema para os seus dispositivos ou aplicações poderão executar um auto-teste fornecido, e depois enviar as provas do teste à instituição reguladora que verificará, consoante os critérios, se tudo foi estabelecido correctamente. Caso tudo esteja certo, de acordo com o pedido (fase 1 ou fase 2), o emblema poderá ser usado.

Dados:

- Início: Abril de 2003.
- Término: Provavelmente quando o IPv6 atingir uma popularidade bem mais aceitável.
- Objectivos:
  - Promover o suporte de IPv6 nos dispositivos e aplicações.
- Mais informação: *http://www.ipv6ready.org/*

2.2.7. DoD

Este departamento é o mais antigo de todas as instituições até agora referidas. Porém, além do papel importantíssimo que desempenhou para o crescimento da Internet, a parte que nos interessa é o facto de ter incentivado o uso de IPv6 em todo o mundo, através de um anúncio feito em Outubro de 2003.

A força do Departamento de Defesa dos Estados Unidos em relação ao desenvolvimento IPv6 é uma força histórica, pelo seu significado na origem do que se chama hoje a Internet. A sua atitude de exigir uma mudança a prazo para IPv6 de toda a sua rede até 2008, como que acordou meio mundo para a novidade protocolar.

Dados:

- Início: 10 de Agosto de 1949.
- Data relativa ao IPv6: Outubro de 2003.
- Término: Provavelmente nunca.
- Objectivo indirecto: Promover o uso do IPv6 em todo o mundo.

Todo o crescimento do protocolo se deve grande parte a estas instituições. Porém, existem mais que não foram referenciadas, e que também contribuíram na medida das suas possibilidades. Prevê-se que,

cada vez mais, os grandes passos que se darão rumo ao IPv6, se prendam com o uso da nova versão do protocolo por instituições que funcionam como ícones, como o *Google*, *Yahoo!*, *Amazon.com*, *eBay*, etc., levando o IPv6 a atingir um nível de popularidade mais elevado.

2.3. Popularidade

A popularidade do IPv6 está muito relacionada com o desenvolvimento, na medida em que proporcionar projectos-piloto, conferenciar e incentivar, aumenta o índice de interesse e, por consequência, torna o IPv6 mais popular.

Ao contrário do que sucedeu com o IPv4, não vai existir um "Dia D". A pequena rede que passou para IPv4 em 1983 é hoje um mundo gigante, que abrange as instituições dos mais diversos tipos e os utilizadores pessoais. O IPv6 não vai ser um *sprint* tecnológico, mas sim uma maratona [27]. As vantagens do IPv4, tão importantes há alguns anos atrás, tornaram-se agora numa barreira para a adopção deste novo protocolo. O velho ditado "não tocar num sistema funcional" é adoptado muitas vezes, principalmente no mundo empresarial. Este mundo será o mais difícil de conquistar, e é provavelmente o mais importante. É preciso mostrar vantagens. Que trabalho se poupa? Que segurança a mais? Que velocidade a mais? É necessário mostrar as falhas do IPv4 e as melhorias do IPv6, e depois deixar esse mundo contagiar-se. É preciso substituir o velho ditado "não tocar num sistema funcional" pelo "se ele tem porque é que eu não devo ter?".

Protocolos velhos demoram tempo a desaparecer [26], por isso a tarefa não se adivinha fácil se o objectivo consistir em fazer desaparecer o velho protocolo. Sendo assim, os dois protocolos deverão coexistir lado a lado num processo de transição cuidado, contribuindo o crescimento de um para o decrescimento do outro. A situação acerca da popularidade do IPv6 também varia de continente para continente, conforme a escassez de endereços IPv4 associada. Na Figura 2.5 pode-se ver o estado da adopção do IPv6 no Japão, Europa e Estados Unidos, e os tempos previstos para essa adopção nos diversos campos. Importante na figura é reparar no adiantamento do Japão em relação à Europa e desta em relação aos Estados Unidos no campo da adopção do IPv6, concluindo-se que a nova versão do IP é muito mais popular no Japão e em outros países asiáticos que nos Estados Unidos.

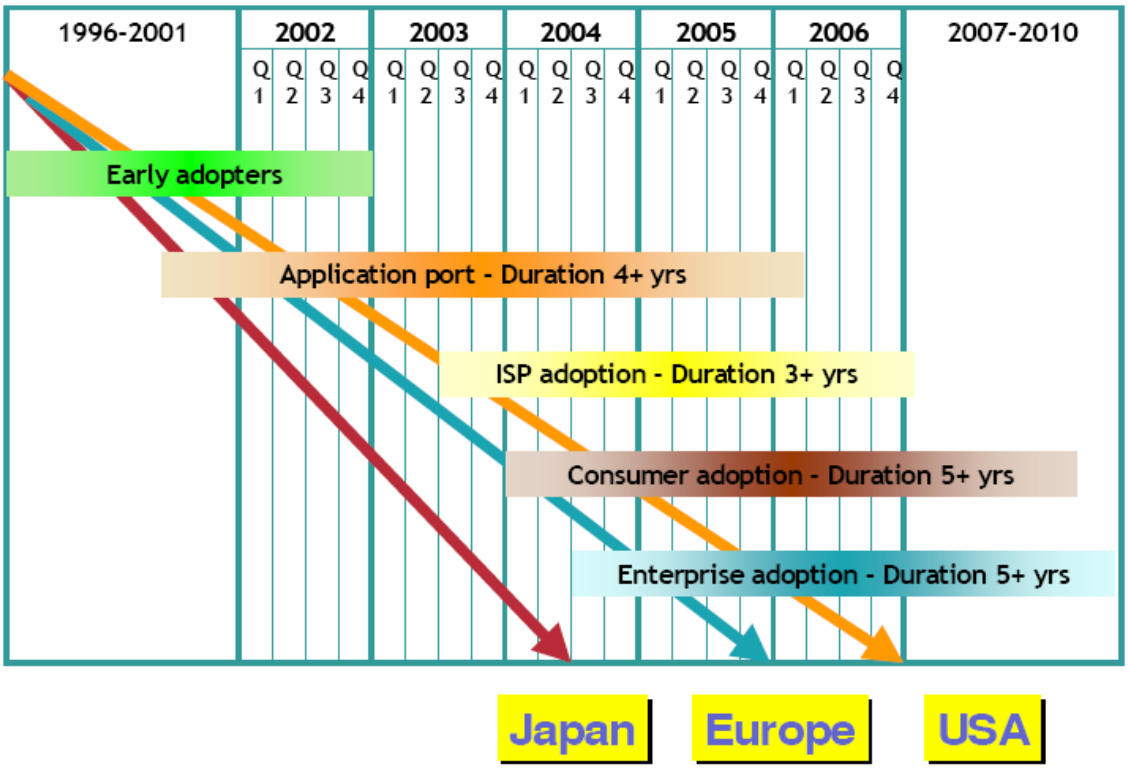


Figura 2.5 – Estado da adopção do IPv6 (Extraído de [13]).

2.4. Comparação IPv4-IPv6

Tal como qualquer versão superior, o IPv6 baseia-se essencialmente na resolução de antigos problemas e implementação de novas funcionalidades. Isso não quer dizer que o protocolo IPv4 esteja mal implementado, muito pelo contrário. O IPv4, especificado já desde 1981, obteve um grande sucesso e continua a tê-lo, por isso mesmo se adivinha ainda difícil a curto prazo a tarefa da sua substituição pelo IPv6.

2.4.1. Cabeçalhos

Geralmente, quando se fala em problemas do IPv4, o problema do limite de endereçamento é, sem dúvida, o tema mais abordado, mas não é o único. Apesar de esse ser sem dúvida um assunto chave para a especificação de uma nova versão do protocolo, existem mais alguns problemas que podem ser minorados ou eliminados.

O cabeçalho IPv4, apesar de na altura ser considerado bastante simples (uma das principais razões do seu sucesso), pode ser ainda mais simplificado. A ideia é simples e resume-se à remoção e alteração de alguns campos.

Na Figura 2.6 apresentam-se os cabeçalhos dos protocolos IPv4 e IPv6. Pode-se ver a relação entre o tamanho de ambos e as alterações ao nível dos campos dos cabeçalhos.

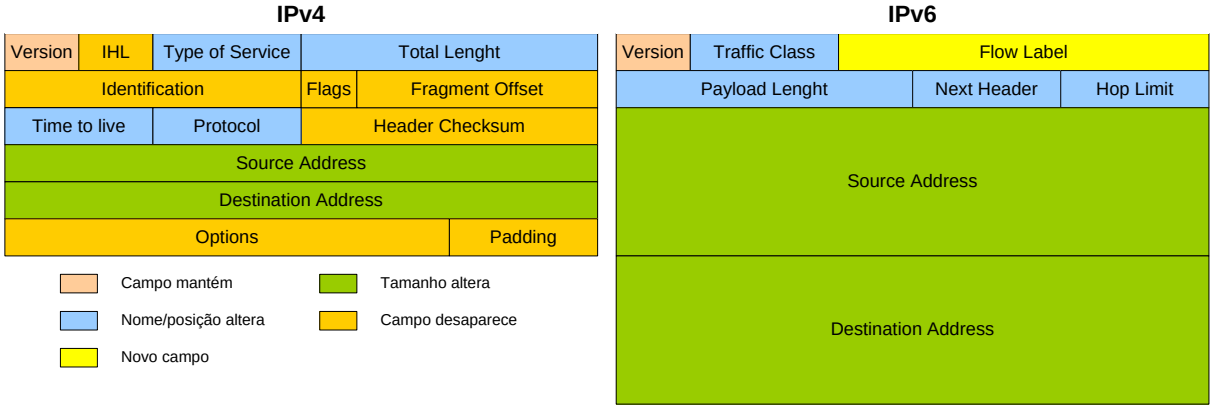


Figura 2.6 – Os cabeçalhos IPv4 [41] e IPv6 [53].

- **Remoção do campo IHL**

A remoção do campo IHL (*Internet Header Length*) acontece devido ao facto de este campo essencialmente especificar o *offset* para a porção de dados do cabeçalho IP [121]. Isto era necessário devido ao facto de existir o conceito de opções no IPv4, o que fazia com que o endereço dos dados[6], conforme o número de opções, fosse mudando de pacote para pacote.

Este conceito, que obrigava a uma informação adicional acerca do endereço dos dados, foi removido no IPv6.

- **Remoção dos campos *Identification, Flags* e *Fragment Offset***

Estes três campos dizem respeito ao processo de fragmentação que poderá ser necessário utilizar ao longo de todo o caminho do pacote. São campos de controlo e identificação dos fragmentos.

No IPv6, a fragmentação é feita usando os Cabeçalhos de Extensão (ver Secção 3.2) e apenas no início da comunicação, nunca em trânsito. Isso faz com que não seja desperdiçado processamento em campos desnecessários quando não ocorre fragmentação.

[6] Registo que identifica a parte específica onde os dados estão armazenados.

- **Remoção do campo *Header Checksum***

Este campo serve para verificação de erros apenas no cabeçalho IP. Foi retirado devido a um conjunto de situações:

- Geralmente os pacotes são corrompidos na camada 2, e essa camada tem geralmente melhores mecanismos para detecção de erros;
- Se mesmo assim existir algum problema, a camada acima poderá solucionar esse problema. No IPv6 a ideia é deixar o TCP encarregar-se da maior parte do controlo de erros;
- Se o pacote estiver corrompido, então o nó destino deverá estar preparado para o recusar, apesar de algum impacto a nível de processamento;
- Os meios de transmissão existentes (p. ex., fibra óptica) são mais fiáveis a assegurar uma comunicação com menos falhas.

- **Remoção dos campos *Options* e *Padding***

O campo *Padding* serve apenas para completar o pacote (tamanho múltiplo de 32 *bits*). Em relação ao campo *Options,* tem alguma utilidade a nível de encaminhamento, mas raramente é utilizado. O IPv6 junta um pouco a funcionalidade deste campo à do campo *Protocol* e utiliza-os no conceito dos Cabeçalhos de Extensão. De certa forma algumas funcionalidades do campo *Options* são passadas para os Cabeçalhos de Extensão. O campo *Protocol* é substituído pelo *Next Header*, parecido com o campo *Protocol* do IPv4, que interliga todos os cabeçalhos de extensão.

- **Substituição do campo *Type of Service***

Este campo era o responsável por algum QoS que pudesse existir no IPv4, no entanto nunca foi adoptado pelos *routers*, principalmente por ser muito pequeno. No IPv6 a QoS divide-se em dois campos: o campo *Traffic Class* e o campo *Flow Label*; o primeiro é direccionado ao catálogo de tráfego (*DiffServ*) e o segundo à geração de fluxos prioritários (*IntServ*) (ver Subsecção 2.4.4).

- **Substituição do campo *Total Length***

Indica o tamanho total do pacote, incluindo os dados. Foi substituído pelo campo *Payload Length* que representa o tamanho do resto do pacote que se segue ao cabeçalho IPv6 [53], ou seja, o tamanho do cabeçalho IPv6 não é neste caso necessário, pois o cabeçalho tem tamanho fixo.

- **Substituição do campo *Time to Live***

A substituição desde campo prende-se apenas com questões de interpretação. Ao contrário do que o nome indica, este campo é medido em saltos e não em tempo, pelo que a designação *Hop Limit* vem corrigir esta pequena confusão.

- **Substituição do campo *Protocol***

Apesar de o nome ser diferente, o novo campo *Next Header* funciona de uma forma semelhante ao campo *Protocol*. No IPv4 indicava o protocolo da camada acima que ia ser encapsulado. No IPv6 indica o protocolo de camada superior ou o *Extension Header* seguinte. É como que um campo que interliga todos os *Extension Headers* [123]. Quando não se verificarem mais *Extension Headers*, então indicará o protocolo da camada acima correspondente, tal como o campo *Protocol*.

- **Alteração dos campos *Source Address* e *Destination Address***

Esta alteração é a mais simples e a mais importante e consiste na passagem dos endereços de 32 *bits* para 128 *bits*.

Verifica-se então, nas alterações efectuadas, uma maior preocupação em relação a questões como a eficiência e a simplicidade. Eficiência na medida em que só é processado aquilo que é realmente

obrigatório. Assim, com o novo protocolo, o processamento desnecessário por parte dos *routers* é mínimo. Exemplos disso são os *Extension Headers* (ver Secção 3.2), a fragmentação (ver Subsecção 3.2.3 e Secção 3.5), a remoção do campo *checksum* e algumas outras mudanças relativas aos protocolos envolventes do IPv6 (ver Secção 3.4 e seguintes). A simplicidade é notada logo pelo facto de o tamanho do endereço IPv6 ser 4 vezes superior ao endereço IPv4, sendo o cabeçalho apenas 2 vezes maior. Concretamente, o tamanho do cabeçalho duplica passando de 20 *bytes* para 40 *bytes*, devendo-se isto principalmente à inclusão de endereços de origem e destino de 128 *bits*.

É importante não esquecer que os *routers* são sempre os dispositivos a que o IP se destina, pelo que as principais preocupações do cabeçalho são dirigidas principalmente ao tempo de processamento que lhes é associado. Neste tipo de preocupação, o IPv6 ganha claramente vantagem. A Figura 2.7 pretende ilustrar a clara diferença entre o tempo de processamento do cabeçalho IPv4 e do cabeçalho IPv6 por parte dos *routers*.

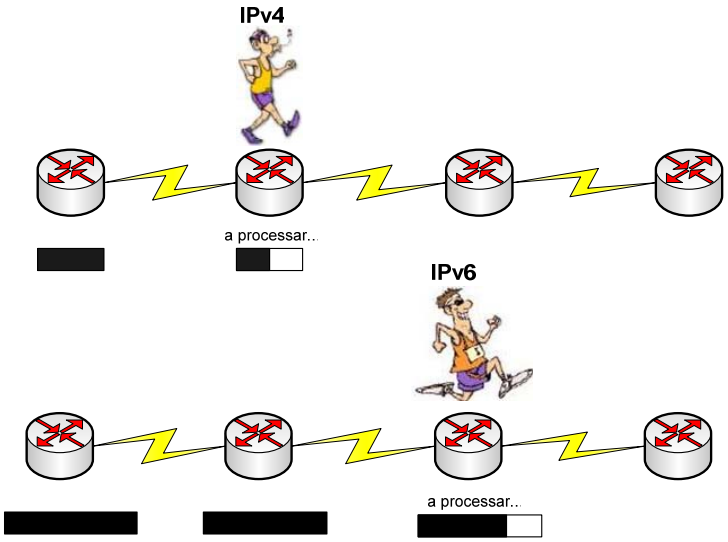


Figura 2.7 – Relação entre a velocidade de processamento dos cabeçalhos IPv4 e IPv6.

2.4.2. Cabeçalhos de Extensão vs. Opções

No IPv4, existe o conceito de opções incorporadas no cabeçalho IP. É um conceito simples que obriga todos os *routers* a analisar todas as opções, especificadas no campo *Options*, digam-lhes estas respeito ou não. No IPv6, por outro lado, existe um novo conceito de uma lista ordenada de Cabeçalhos de Extensão (*Extension Headers*) que permite que os *routers* apenas analisem aquilo que é necessário para o normal funcionamento da rede. A Figura 2.8 mostra como todos os *routers* de um caminho analisam o campo *Options* do cabeçalho IPv4, e a Figura 2.9 mostra como os *Extension Headers* do IPv6 apenas são analisados por alguns *routers*.

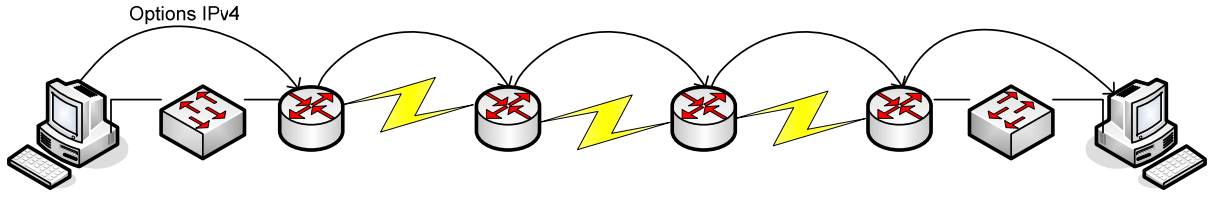


Figura 2.8 – Leitura do campo *Options* do cabeçalho IPv4 por parte de todos os *routers*.

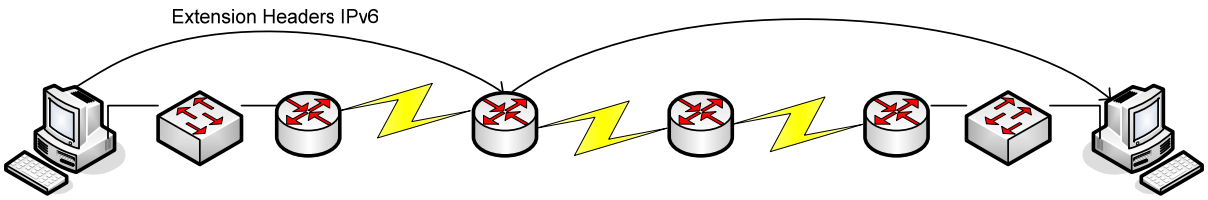


Figura 2.9 – Leitura dos *Extension Headers* IPv6 apenas por alguns *routers*.

O *router* saberá, então, através do campo *Next Header* dos cabeçalhos, se deve ou não processar o próximo cabeçalho, poupando assim tempo de processamento.

É este processo que faz claramente o conceito de opções tornar-se obsoleto, pois da forma que o IPv4 está implementado, o *router* perde tempo de processamento ao analisar sem excepção o campo *Options*, pois basta existir um *router* no caminho do pacote e determinada opção não lhe dizer respeito, para que seja desperdiçado processamento. A lista de cabeçalhos de extensão vai ser abordada mais à frente (ver Secção 3.2).

2.4.3. Endereçamento

As diferenças de endereçamento entre os dois protocolos verificam-se principalmente a nível da representação. O protocolo IPv4 é representado em *dotted decimal*, o IPv6 em *column hexadecimal*. Na Tabela 2.1 podem ver-se as diferenças entre as duas representações.

	Dotted decimal	***Column hexadecimal***
Definição	Representação sintáctica de um inteiro de 32 *bits*, que consiste em quatro partes de 8 *bits* em formato decimal separadas por um ponto.	Representação sintáctica de um inteiro de 128 *bits*, que consiste em oito partes de 16 *bits* em formato hexadecimal separadas por dois pontos.
Exemplo	*123.234.123.234*	*CAFE:1234:5678:90AB:CDEF:1234:5678:90AB*

Tabela 2.1 – Os formatos *dotted decimal* e *column hexadecimal*.

Em relação aos diversos tipos de endereços, existem algumas modificações no IPv6 ao comparar com o IPv4. O endereço de *loopback* e os *unicast* não sofreram qualquer tipo de modificação significativa. Os endereços *anycast* e *multicast*, duas promessas antigas já existentes no IPv4, têm agora uma importância bem mais destacada neste novo protocolo. A especificação dos endereços *anycast*[7] (ver Subsecção 3.3.4.3) foi efectuada em 1993 [45] e, ao contrário do que se possa pensar, estes não são exclusivos do IPv6, já existindo no IPv4. Em relação ao *multicast*, todos os mecanismos que funcionavam em *broadcast* no IPv4 foram alterados para este paradigma. Assim, o endereço *broadcast* deixou de existir no IPv6, tendo sido substituído pelo *multicast*. A Tabela 2.2 apresenta os diferentes tipos de endereços no IPv4 e no IPv6.

[7] Tipo de endereçamento em que o pacote é encaminhado para o destino mais próximo, tendo em conta as métricas de encaminhamento. Para mais informação acerca deste tipo de endereços, consultar o RFC 1546.

	IPv4	IPv6	Outras considerações
Unspecified	Utilizado. (*0.0.0.0*)	Utilizado. (*::*)	Endereço *unicast* usado quando a interface não tem endereço ou não é usado o endereço configurado.
Loopback	Utilizado apenas como teste à interface local da própria máquina. (*127.0.0.1*)	Utilizado apenas como teste à interface local da própria máquina. (*::1*)	Poderá também considerar-se um endereço *unicast*.
Unicast	Muito utilizado.	Muito utilizado.	Existem variados tipos de endereços dentro do *unicast*.
Anycast	Muito pouco utilizado.	Utilizado.	No IPv4 existe uma gama própria para endereços *anycast*, no IPv6 não.
Multicast	Muito pouco utilizado.	Muito utilizado.	Tudo o que era feito em *broadcast* no IPv4 é agora feito em *multicast* no IPv6.
Broadcast	Muito utilizado, abusivamente.	Não é utilizado.	O *multicast* no IPv6 poderá facilmente fazer o papel do *broadcast* no IPv4.

Tabela 2.2 – Os diversos tipos de endereços no IPv4 e no IPv6.

As razões para o desaparecimento do *broadcast* dividem-se em duas partes essenciais:

- A nível da camada de ligação, o *broadcast* é sempre prejudicial de alguma forma. Os computadores terão de processar o pacote caso este lhe interesse ou não, e isto faz com que se perca eficiência.

Exemplo 1:

O Windows geralmente utiliza imenso os *broadcasts* em aplicações/protocolos proprietários, e qualquer computador que esteja na mesma rede (considerando a camada de ligação de dados), mesmo que utilize um sistema operativo diferente, será "bombardeado" com todos os pacotes *broadcast* que sejam gerados pelo sistema operativo Windows. Logo, se por exemplo o computador usar o sistema operativo Linux, vai processar o pacote gerado pelo Windows, vai achar o pacote completamente desconhecido, e irá descartá-lo. A Figura 2.10 ilustra precisamente esse problema.

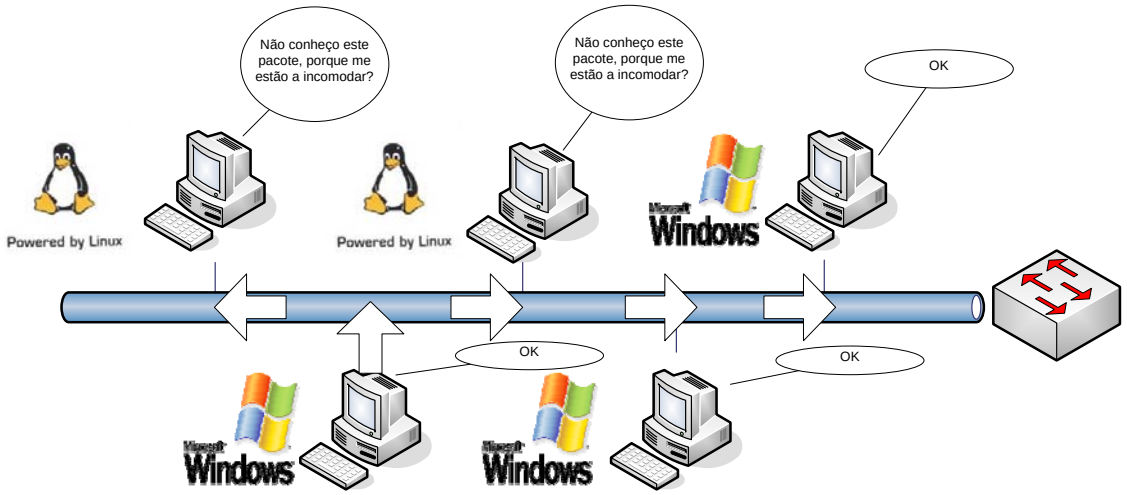


Figura 2.10 – Problema do *broadcast* numa rede heterogénea.

Com *multicast*, esta situação não acontece, sendo os pacotes processados apenas pelos computadores a que realmente esse pacote interessa.

Exemplo 2:

A resolução de endereços pelo ARP[8], utilizada no IPv4, é efectuada enviando os pedidos por *broadcast*. Este processo faz com que se pergunte a todos os dispositivos se determinado endereço IP é o dele, e no caso de ser, esse dispositivo irá enviar-lhe o seu MAC. Isso gera tráfego desnecessário para todos os nós a quem o endereço IP não pertença.

No IPv6 não se utiliza o protocolo ARP mas sim o *Neighbor Discovery*, que utiliza *multicast* (ver Secção 3.6).

- Subindo um nível, parando na camada de rede, *broadcast* era uma palavra "apetecível" para exploradores de falhas de segurança, principalmente usando este tipo de endereços em ataques DoS (*Denial of Service*)[9]. Sendo o *broadcast* bastante apetecível para gerar tráfego desnecessário para toda a rede, percebe-se bem o porquê destes ataques.

Exemplo:

Ao imaginar gerar o máximo tráfego possível em determinada rede, torna-se óbvio para os atacantes que usar um endereço que abranja o maior número possível de dispositivos será o melhor método. Os endereços de *broadcast* servem perfeitamente essa definição, daí serem tão utilizados. Os endereços *multicast* vêm diminuir a capacidade dos atacantes de sobrecarregar a rede com tráfego inútil, pois podem diminuir em muito o número de dispositivos alvo.

2.4.4. Segurança

A segurança é a parte menos apreciada da Internet. Geralmente quando se critica a Internet, a segurança é sempre o alvo favorito dos críticos. A crítica vai de encontro a diferentes níveis de segurança, desde a segurança pessoal, à autenticidade e confidencialidade da informação.

Quando foi criado o IPv4, não havia qualquer tipo de preocupação com a segurança, e nunca se pensou que ela pudesse vir a ter um papel tão crucial na Internet. Na pequena rede criada na altura, o ambiente não era nada malicioso, mas sim completamente cooperativo e harmonioso. Era praticamente a rede de sonho onde se conhecia a maioria dos utilizadores. O ânimo que existia na criação da potencial rede gigantesca de partilha de informação era o principal inimigo. Ninguém pensava em restrições de informação. Pelo contrário, quanto mais fosse partilhada melhor. A Internet era o acesso à informação, não a restrição a ela.

Infelizmente, apesar de a Internet ainda ter como significado o acesso à informação de forma rápida e simples, ela cresceu, e junto com todas as vantagens vieram as desvantagens.

A Rede é composta hoje por muitos conteúdos, dos mais variados tipos, havendo nalguns necessidade de restrições, para que apenas algum público mais restrito tenha acesso a essa informação. É nesta base que entra a necessidade de segurança, mas é bem mais do que isso. Hoje em dia compra-se na Internet, gerem-se contas bancárias, e tanto pessoas como empresas têm uma identidade própria na Internet. E é claro que existe quem indevidamente se queira apoderar dessas identidades.

O IPv6 já sabe disso tudo. Ao contrário do IPv4, que ainda tem a mentalidade de uma "falsa harmonia" existente na partilha de informação, o IPv6 já sabe do perigo existente e por isso tem a obrigação de se saber defender dele.

Será então que a Internet ficou estes anos todos à espera sem ter direito a qualquer tipo de segurança?

[8] Protocolo utilizado no IPv4 que tem como objectivo, através de um endereço IP de um dispositivo, obter o respectivo endereço MAC.
[9] Tipo de ataque a uma rede, em que um dos objectivos é "inundar a rede", "enchendo-a" de tráfego sem qualquer tipo de utilidade.

A resposta é não. Essa segurança pode ser conseguida de várias formas:

- Através de segurança baseada em mecanismos da Internet (o tradicional nome de utilizador e respectiva palavra-passe). Este tipo de mecanismos está presente praticamente em qualquer sítio da rede que envolva qualquer tipo de restrição;
- Utilizando segurança em outras camadas que não a de rede. Usando, por exemplo, SSH na camada de aplicação, TLS na camada de transporte ou 802.1x na camada de ligação de dados;
- Firewalls;
- IPsec;
- NAT.

Apenas os dois últimos pontos (IPsec e NAT) dizem respeito directamente ao IPv6, o primeiro pela sua inclusão, o segundo pela exclusão.

IPsec

Destes tipos de segurança, o que diz mais respeito directamente ao IPv6 é o IPsec. Este conjunto de protocolos, ao contrário do que se possa pensar, nasceu no IPv6 tendo depois migrado para o IPv4 [32] devido à necessidade de segurança na camada de rede.

No IPv6, não é falado em IPsec por boas razões; é que ele está incluído no novo protocolo através dos *Extension Headers*. Poderá parecer uma fraca vantagem, já que a novidade não é assim tão motivadora, sendo que o IPv4 poderia ter praticamente a mesma segurança do novo protocolo, bastando para isso ser adicionado o IPsec.

Olhando agora de outra forma, o mundo empresarial actualmente já deu um grande passo. A maioria das empresas já tem a preocupação com os ataques vindos do exterior. O NAT contribuiu de forma indirecta para o efeito de protecção, mas a maioria das empresas também utiliza *firewalls*. O problema verifica-se essencialmente com a falta de protecção a nível interno. As empresas formam um castelo, sem pensar que o perigo pode estar dentro dele, não se preocupando com outras implementações de segurança como o IPsec. É aqui que entra o IPv6, através da inclusão do IPsec na sua especificação. A protecção interna nas empresas torna-se automática, sem ser necessário recorrer a qualquer tipo de protocolos extra de segurança [29]. Sem mudar mentalidades, a segurança interna é totalmente implementada. A Figura 2.11 mostra como os ataques que mais podem prejudicar são os provenientes de dentro e não os de fora.

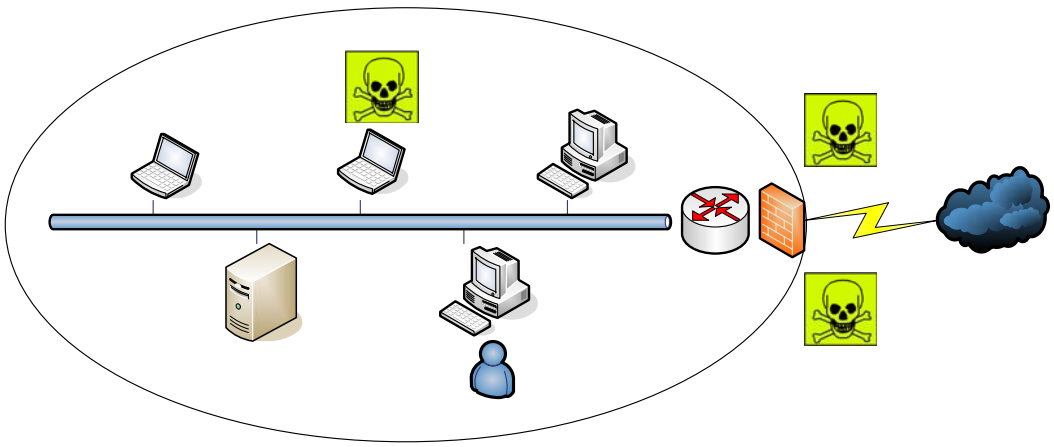


Figura 2.11 – Os perigos dentro e fora de uma rede.

É de referir também que para garantir segurança no IPv4 é preciso mais trabalho e investimento monetário do que fazer a migração directa para IPv6 [23].

NAT

A finalidade principal do NAT é sem dúvida poupar endereços IPv4, mas também lhe eram atribuídas funções adicionais de segurança, nomeadamente ao nível básico de filtragem de pacotes que vinham do exterior da rede. É claro que o NAT por si só não bastava para ter uma rede segura, pois não restringia qualquer tipo de ligação efectuada ao exterior se ela fosse iniciada pela parte interior da rede; para isso já era necessária uma *firewall*.

O NAT pode não ser a segurança ideal, mas de certo modo ajuda. É certo que o IPv6 faz com que o NAT tenha os dias praticamente contados, pelo que pode parecer que alguma segurança possa por isso ser perdida, mas isso é puro engano. Existe uma abordagem completamente distinta, que transforma de certa forma o NAT apenas num transtorno para a segurança. Algumas das razões para a despreocupação acerca da perca do NAT são:

- A maioria dos ataques não se prende com a ligação directa no sentido exterior-interior; envolve sim ligações do interior da própria rede.

  Pode-se tomar como exemplo os vírus em *e-mails* ou páginas *web* que estabelecem uma ligação ao exterior praticamente sem o próprio utilizador notar, podendo expor totalmente a rede interna.

  Infelizmente, nas empresas, a ideia de estar protegido do exterior causa uma falsa sensação de segurança que pode muitas vezes ser prejudicial para a sua rede.

- Privacidade, independentemente se é desejada ou não.

  A privacidade pode ser um requisito do utilizador, mas também pode não o ser. Pode até ser um entrave à segurança, impedindo de se detectar a origem de um ataque.

  Imagine-se que existe um ataque proveniente da Instituição X, por exemplo, à rede interna do Governo: o Governo faz uma queixa à entidade competente, descobre-se que o ataque foi proveniente da Instituição X, que é notificada para informar à entidade competente o invasor. A questão agora é a seguinte:

  Sabe a Instituição X quem foi o invasor?

  A resposta é: normalmente não. Poderá saber caso mantenha um ficheiro de *logs* cuidado acerca de todas as transições para endereços públicos efectuados por todos os computadores da sua rede interna, mas isso é muito raro.

- Com NAT nunca poderá existir segurança ponto-a-ponto.

  O NAT não permite a implementação de técnicas de segurança ponto-a-ponto. Essas técnicas, onde se inclui o IPsec, encriptam e marcam todos os pacotes enviados por determinada máquina. Os pacotes ao passarem pela transição de mudança para endereço público são modificados, o que faz com que a assinatura (marcação) esperada pelo destino possa não ser igual. A Figura 2.12 ilustra o referido.

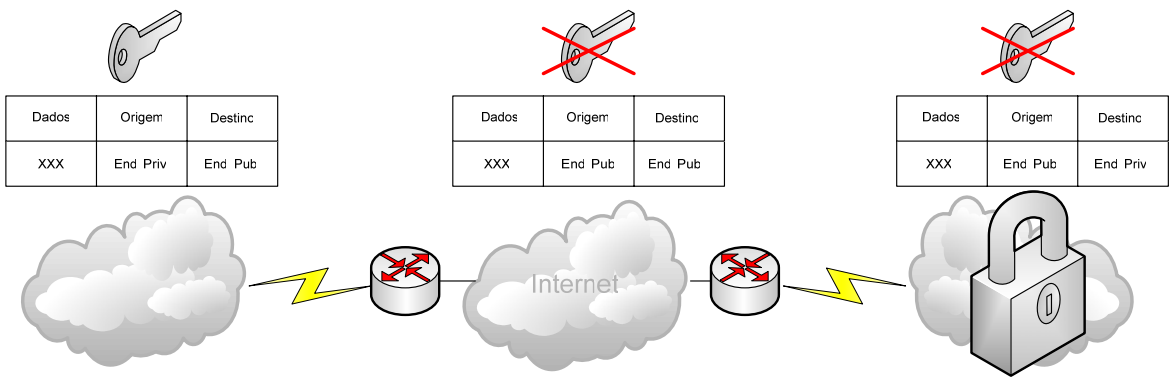


Figura 2.12 – Problemas de segurança ponto-a-ponto.

Existem também outras considerações a ter a nível de segurança entre IPv6 e IPv4, que se prendem principalmente com o tamanho do endereço. À primeira vista, poderá parecer que o facto de o endereço IPv6 ser bem mais longo e bem mais difícil de memorizar será uma desvantagem mas, pensando em segurança, não é isso que acontece. Ter um endereço difícil de memorizar é uma simples e boa forma de obter segurança. O endereço IPv4 decorava-se praticamente com a mesma facilidade com que se decora um endereço telefónico. Caso o nosso computador usasse um endereço IPv4 estático, qualquer pessoa que tivesse acesso a ele, facilmente o decorava e o ligava ao nosso computador sem precisar de recorrer a qualquer tipo de "cábula". O problema é que se esta associação for feita por alguém que se queira aproveitar das nossas falhas de segurança para qualquer tipo de ataque, é sem dúvida a primeira vantagem para o atacante, que de certa forma memoriza o local onde estamos. Este tipo de problema a nível do utilizador comum em IPv4 não se põe, devido ao facto de os ISPs nos fornecerem sempre endereços dinâmicos da sua *pool* de endereços. Embora exista o protocolo DHCPv6, o IPv6 não está pensado para ser assim. Está pensado para funcionar com o endereço estático de qualquer dispositivo.

De qualquer forma, poderá parecer que o endereço IPv6, por ser desenhado para ser estático, fornece poucas opções de privacidade ao utilizador, mas isto não acontece (ver Subsecção 3.3.4.1), pois os endereços IPv6 podem também ser gerados aleatoriamente cada vez que existe ligação à rede.

Não é apenas a dificuldade de memorização o maior indicador de segurança do endereço IPv6, até porque uma boa "cábula" e alguma atenção podem reverter a situação, mas pode-se imaginar então a comum situação [29]:

Um atacante, através de um analisador de portas, tenta verificar se existem máquinas com alguma porta aberta interessante, para usufruir das suas fragilidades de segurança.

- Exemplo de uma análise a uma rede de classe C em IPv4 (cenário típico):

  $2^8 = 256$ endereços a analisar.

- Exemplo de uma rede de 64 *bits* de prefixo em IPv6 (cenário típico):

  $2^{64} = 18.446.744.073.709.551.616$ endereços a analisar.

- Considerando que se demora cerca de um segundo a analisar cada endereço, situação com IPv4:

  $2^8 / 60 = 4{,}27$ minutos [10].

- Situação com IPv6:

  $2^{64} / 31536000 = 58.494.217.355$ anos [11].

Os números falam por si. O tempo necessário para analisar todos os endereços torna a tarefa inconcebível, e mesmo apesar de algumas técnicas simples utilizadas por atacantes, por exemplo, analisar apenas aqueles endereços que realmente fazem mais sentido, ou começar pelos mais óbvios acabando nos endereços menos prováveis de ter portas abertas (é certo que os administradores começam quase sempre de forma crescente a sua numeração de máquinas), a análise temporal não mudará muito.

2.4.5. Qualidade de Serviço

A qualidade de serviço é muito importante, na medida em que as tecnologias que hoje a requerem são as que cada vez mais irão ser utilizadas. São as chamadas tecnologias do futuro. Neste lote inserem-se a multi-conferenciação, VoIP, os fluxos de banda larga de áudio/vídeo e a mobilidade.

Existem várias formas de melhorar a qualidade de serviço, e curiosamente já foram descritas algumas delas, tais como:

- Suporte apenas de *multicast*, eliminando o *broadcast* (ver Subsecção 2.4.3);

[10] 60 são os segundos de um minuto.
[11] 31536000 são os segundos de um ano.

- Retirar o campo *checksum* (ver Subsecção 2.4.1);
- *Extension Headers* (ver Subsecção 2.4.2);
- Menos campos no cabeçalho IPv6 (ver Subsecção 2.4.1).

Por outro lado, alguns dos problemas do IPv4 são:

- A fragmentação é efectuada nos *routers* com o pacote em trânsito, o que produz congestão, consumo de largura de banda e processador;
- O ICMP [42] tem demasiadas opções que o tornam por vezes pouco eficiente;
- O suporte para QoS (*Quality of Service*)[12] é mínimo. O campo ToS (*Type of Service*) é demasiado curto e fixo e por isso não foi adoptado pelos *routers*. Apenas mais tarde foi adoptado o campo ToS como DS (*Differentiated Services*) [57], esse sim já adoptado pelos *routers* [33].

DiffServ e IntServ

A nível de diferenciação de serviços, o campo responsável pela marcação dos pacotes (*Traffic Class*) mudou de nome, mas o funcionamento é basicamente o mesmo.

Em relação à integração de serviços, o objectivo é tentar aumentar a eficiência dos fluxos prioritários, nomeadamente ao nível de anular o problema de violação de camada. A violação de camada, ilustrada na Figura 2.13, acontece quando algum dispositivo opera numa camada que não seja a sua, o que se traduz num decréscimo a nível de desempenho. O que se passa neste caso, é que o *router*, para identificar os fluxos, utiliza a camada de rede e a camada de transporte, apesar de estar especificado no que apenas a camada de rede lhe diz respeito.

A identificação de fluxos no IPv4 é conseguida através de 5 campos: *Source Address*, *Destination Address*, *Source Port*, *Destination Port* e *Protocol*.

A identificação no IPv6 é feita com 3 campos: *Source Address*, *Destination Address* e *Flow Label*.

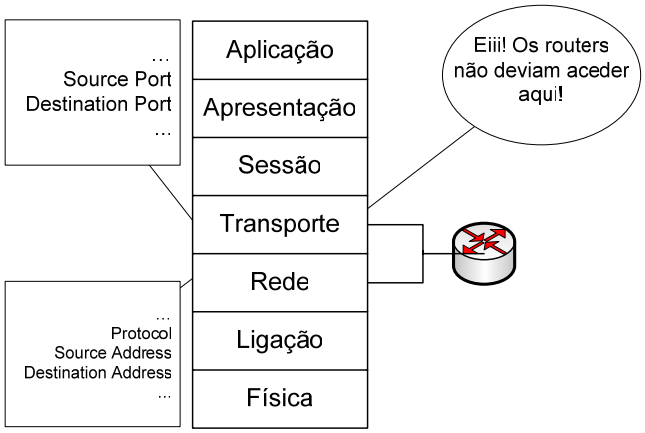


Figura 2.13 – Violação de camada por parte dos *routers* no IPv4.

Existem outros problemas adicionais associados à violação de camada. Esses problemas verificam-se quando existe encriptação na camada de rede (caso do IPsec), ou quando existe fragmentação. No primeiro caso, o que acontece é que existindo encriptação na camada de rede, todas as camadas acima da de rede estarão também encriptadas. Desta forma não é possível ao *router* verificar os portos e, consequentemente, identificar os fluxos. Existem formas de resolver este problema, mas não são muito simples. Este problema é ilustrado na Figura 2.14.

[12] QoS, apesar de as siglas significarem qualidade de serviço, diz respeito a uma parte mais específica dela, onde é abordado a diferenciação de tráfego (*DiffServ*) e a integração de serviços (*IntServ*).

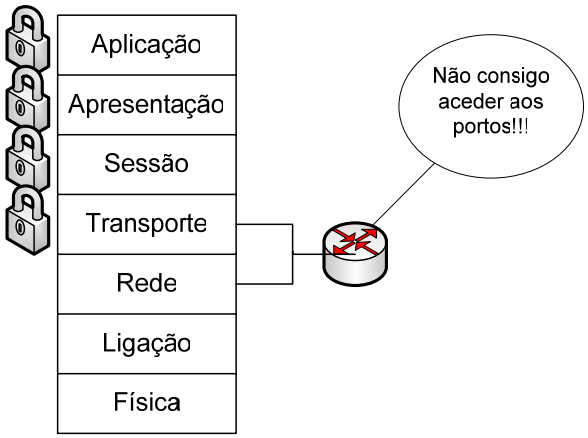


Figura 2.14 – Encriptação ao nível da camada 3, pelo que o *router* não consegue aceder à camada 4.

No caso da fragmentação, o caso é bastante simples. É que nem sempre nos fragmentos de pacotes fragmentados existe o cabeçalho TCP, logo existem pacotes em que o fluxo não conseguirá ser identificado. Na Figura 2.15 pode ver-se o exemplo de um pacote fragmentado em dois fragmentos, em que o cabeçalho TCP só aparece num deles.

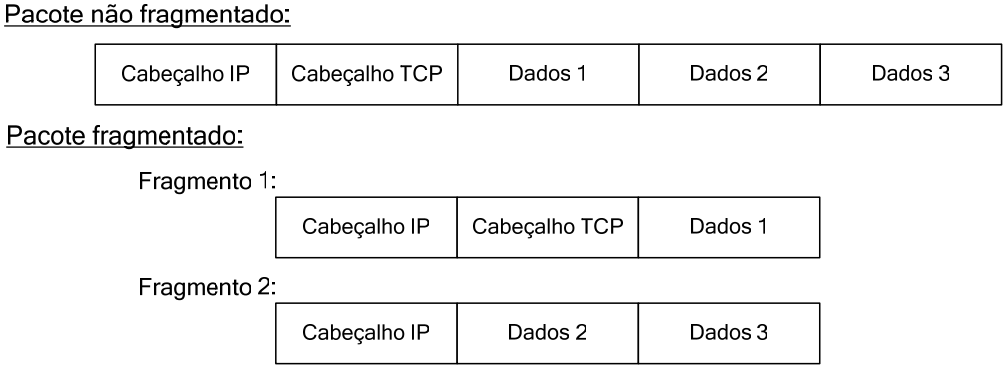


Figura 2.15 – Exemplo de fragmentação de um pacote.

É de referir também que a qualidade de serviço é a parte de IPv6 que está ainda na fase mais experimental, principalmente o campo *Flow Label*. É, assim, imprescindível uma boa leitura dos respectivos RFCs e *drafts* associados.

2.4.6. Mobilidade

Tal como na segurança, o IPv4 quando foi especificado, não tinha qualquer ideia de que poderia ser associado à mobilidade. A própria ideia de qualquer pessoa poder ter acesso a um telemóvel era já bastante estranha, quanto mais juntar a isso computadores portáteis, PDAs ou *smartphones*[13]. Desta forma, o IPv6 volta a não ter justificação para não possuir suporte, desta vez no que diz respeito a todo o conceito de mobilidade. Voltando à ideia do maior número de endereços disponibilizado pelo protocolo, esta é por si só, uma vantagem principal no que diz respeito à mobilidade. Tornava-se praticamente impossível construir um projecto credível de mobilidade mundial a longo prazo, apenas com o resto dos endereços IPv4 disponíveis. Para além desta vantagem, existem muitas outras que abrangem diversos domínios desde a segurança ao encaminhamento.

Quando se fala em mobilidade, pensa-se logo em redes sem fios, directamente associadas à norma IEEE 802.11. Esta norma situa-se na camada de ligação, tendo por isso apenas significado local.

[13] Uma fusão de telemóvel com PDA; estes dispositivos têm já sistema operativo próprio e armazenamento local, bem como podem aceder à Internet e instalar variadas aplicações.

Passando para a camada de rede, esta norma deixa de existir. Pensar em mudar de rede apenas com IEEE 802.11 torna-se assim impossível. Para ser possível esta mudança de rede transparente para o utilizador, é necessário utilizar MIP (*Mobile Internet Protocol*) seja ele MIPv4 ou MIPv6. A Figura 2.16 mostra a relação daqueles protocolos com as camadas do modelo OSI.

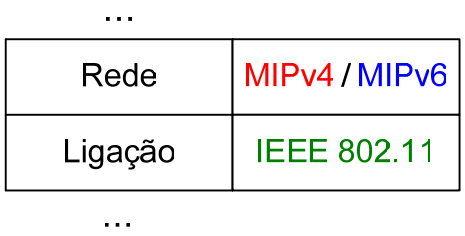


Figura 2.16 – Relação entre os protocolos e o modelo OSI.

Tanto o MIPv4 como o MIPv6 têm a capacidade de permitir dois endereços distintos: o 1.º será um endereço fixo, que se considera como endereço de raiz; o 2.º será um endereço dinâmico conforme a nova conexão que seja estabelecida com o dispositivo móvel. Estes endereços chamam-se, respectivamente, *home address* e *care-of-address*.

No IPv4, o responsável pelo *home address* é *o home agent* (HA), sendo o *foreign agent* (FA) responsável pelo *care-of-address*. A ideia básica no MIPv4 é o *home agent* servir de localizador do dispositivo móvel. O dispositivo da rede X, ao comunicar com o dispositivo móvel, irá sempre ter como referência de destino o *home address*, só que antes de chegar a esse destino ele é interceptado pelo *home agent*, que o irá encaminhar então para o destino certo. De seguida, a resposta do dispositivo móvel para o dispositivo da rede X é efectuada de forma normal. Com isto, obtém-se uma comunicação triangular. Este processo é ilustrado na Figura 2.17.

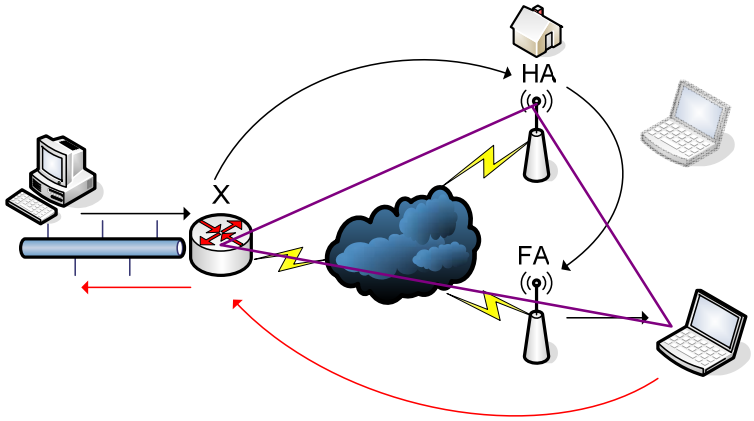


Figura 2.17 – Processo de comunicação entre dispositivos fixos e móveis com MIPv4.

Este procedimento já resolve o problema da transparência na troca de redes, mas será totalmente adequado e eficiente? Não, este procedimento tem alguns problemas, como a sobrecarga do *home agent*, ou a distância a que ele poderá estar do dispositivo móvel.

Geralmente só existe um *home agent* numa rede com vários dispositivos. Esse *home agent* terá obrigatoriamente de encaminhar de forma correcta todo o tráfego associado aos vários dispositivos móveis, por isso é muito provável o esgotamento dos seus recursos. Por outro lado, o caminho em que os pacotes seguem pode ser de uma ineficiência tremenda. O dispositivo da rede X poderá enviar pacotes para o dispositivo móvel que está numa rede imediatamente a seguir à sua, mas se o *home agent* estiver a uma distância bastante considerável, os pacotes primeiro irão em direcção ao longínquo *home agent*, voltando para trás, para perto do dispositivo que efectuou o pedido de ligação.

Assim, o que se pede ao IPv6 é que melhore consideravelmente esta falta de eficiência no encaminhamento dos pacotes. É isso mesmo que acontece, o IPv6 optimiza as rotas. Os dispositivos

que pretendem estabelecer ligação ao dispositivo móvel saberão onde se encontra o dispositivo móvel antes de enviar o pacote com destino ao *home address*, consequentemente enviam os pacotes directamente para o sítio correcto. A diferença é que, ao contrário do IPv4, em que o dispositivo móvel, cada vez que mudava de rede, informava o *home agent* da sua nova localização, no IPv6 informa-se o próprio dispositivo que lhe irá mandar a mensagem. Essa informação é dada imediatamente antes do dispositivo enviar para o dispositivo móvel. O dispositivo irá então solicitar ao móvel para que lhe dê a sua localização temporária.

Também foram testadas técnicas de optimização de rotas no MIPv4, mas nunca passaram de *drafts* e deixaram de fazer parte da preocupação da IETF.

Em relação à sobrecarga sofrida pelo *home agent*, no IPv6 ela vai ser diminuída drasticamente, pois o tráfego que irá passar pelo *home agent* é mínimo.

A Figura 2.18 mostra a melhoria na comunicação entre dispositivos fixos e móveis usando MIPv6.

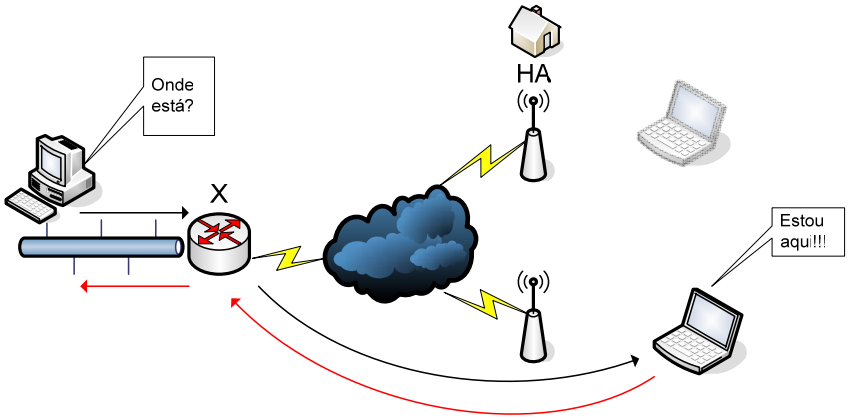


Figura 2.18 – Melhoria na comunicação utilizando MIPv6.

Para se executar esta optimização de rotas, é absolutamente necessário o uso de MIPv6 em todos os dispositivos (nomeadamente no computador que envia para o dispositivo móvel), caso contrário, se isto não se verificar, o funcionamento será igual ao MIPv4.

Para além dos problemas já referenciados em relação ao MIPv4, existem outros mais ocasionais, mas que não deixam de ter a sua importância. Um caso desses é o *ingress-filtering* [64], ilustrado na Figura 2.19. Esta técnica anti-*spoofing*[14] analisa a rede do computador que enviou os respectivos pacotes e compara-a ao endereço de origem do pacote. Se verificar que a rede do respectivo computador é diferente da rede origem do pacote, descarta esse pacote automaticamente. A relação desta técnica (muito utilizada nas *firewalls*) com o MIPv4 é um grande problema, pois o dispositivo móvel manda na sua origem o seu endereço *home address*, para que o dispositivo responda para o endereço de raiz, e consequentemente o *home agent* o reencaminhe para o endereço certo.

Existe uma forma de resolver este problema, que é fazer o dispositivo móvel enviar os pacotes via *home agent*. Desta forma o endereço de rede já não seria falso. Este procedimento implica ainda mais sobrecarga do *home agent*.

[14] Método com o objectivo de se fingir alguém que não é, para ter direito a atributos especiais.

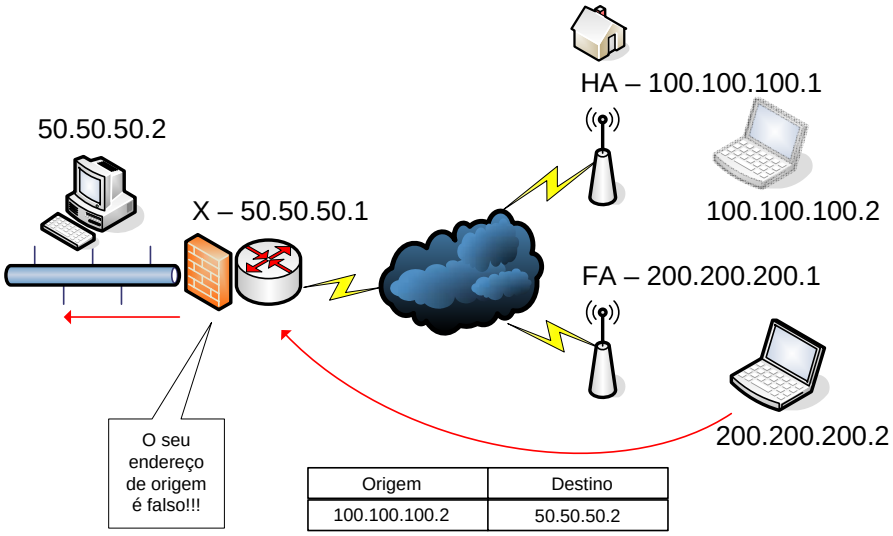


Figura 2.19 – O *ingress-filtering* no MIPv4.

O NAT também poderá causar problemas parecidos, devido à mudança de endereços em relação à origem. Com MIPv6, estes problemas são imediatamente eliminados. O endereço que vai no pacote do dispositivo móvel é o seu endereço verdadeiro.

Existem outras vantagens do MIPv6 obtidas de forma indirecta devido às próprias características do IPv6. Algumas delas são:

- Não é necessário *foreign agents*, devido à auto-configuração do IPv6;
- O *home agent* é descoberto dinamicamente usando *anycast* e não *broadcast*, como com IPv4;
- O IPsec é importantíssimo na mobilidade devido aos vários registos efectuados no *home agent* e *foreign agent*, e no IPv6 este tipo de segurança é mandatário;
- O IPv6, através do ND (ver Secção 3.6), torna-se independente da camada física e de ligação, ao contrário do que acontecia com o ARP. Assim poderão utilizar-se tecnologias tão diferentes como: *Ethernet*, WLAN, CDMA, GSM, *Bluetooth*, etc.

2.5. Mitos

Cada vez que é introduzida uma nova tecnologia, ela é sempre acompanhada de mitos. Esta situação não é nova, e ao longo da história da evolução da tecnologia, verificaram-se situações semelhantes. O IPv6 não foge à regra, e os mitos abrangem os mais diversos domínios do IPv6, desde questões políticas a questões essencialmente técnicas. Alguns dos mitos são a favor do IPv6, outros menos a favor e outros contra.

Mito: O IPv6 não é necessário, pois existem endereços IPv4 suficientes e ainda por cima ainda existe o NAT para ajudar.

Situação real: Falso.

Estas afirmações vieram em tempos a partir do governo, universidades e outras instituições dos EUA. Os EUA detêm praticamente 60% de todos os endereços IPv4 do mundo (ver [16]) e já tiveram mais. Só para se ter uma ideia, existe um ISP nos EUA com 3 classes A de endereços. Este tipo de mito tende cada vez mais a desaparecer, pois o departamento de defesa norte-americano já alertou para a importância do protocolo (ver Subsecção 2.2.7). Em relação ao NAT apenas se poderá pensar como opção a curto prazo. Os principais problemas do NAT são a nível de segurança (ver Subsecção 2.4.4), e como, por exemplo, o comércio electrónico é cada vez mais popular, é necessária por isso cada vez mais segurança [25].

Mito: Os endereços IPv6 dão para tudo o que existe na Terra.

Situação real: Verdadeiro.

A quantidade de endereços IPv6 é mesmo astronómica (ver Secção 2.1). Existe quem fale no IPv6 quase como endereços infinitos, existe quem o associe a praticamente todos os objectos, existe quem faça cálculos com estes números que aparecem absolutamente absurdos. Este é um tema que pode despertar diversas "imaginações". Bons exemplos disso são algumas ideias que foram surgindo para a utilização de endereços IPv6 em vez de RFID[15], e dos normais códigos de barras [30].

Mito: As tecnologias nas quais o IPv6 se baseia começam a ficar obsoletas.

Situação real: Por enquanto, é falso.

Este mito baseia-se na ideia do que aconteceu, por exemplo, com o UMTS. A tecnologia demorou tanto tempo a ser implementada que agora, quando finalmente o começou a ser, já está um pouco ultrapassada no domínio da largura de banda e outras características.

Mito: A arquitectura fundamental do IPv6 é subdividida em partes que não serão fixas a longo prazo.

Situação real: Verdadeiro, mas tende a tornar-se falso futuramente.

Existem sim, algumas partes do IPv6 ainda em fase de estudo, apesar de já estar especificado o que se quer fazer por parte da IETF (ver Subsecção 2.2.1). Por essa razão, é normal ocorrerem algumas alterações até que o IPv6 se fixe mais. Um outro exemplo é o caso do prefixo de teste para 6Bone (ver Subsecção 2.2.2) que está atribuído temporariamente, e terá de ser alocado mais tarde de forma definitiva.

Mito: Todos os dispositivos necessitam de efectuar o *upgrade* para IPv6, se não ele nunca poderá ser implementado [10].

Situação real: Falso.

Existem os mecanismos de transição (ver Secção 3.10) que servem exactamente para que nem todos os dispositivos necessitem do suporte IPv6. É claro que terão de existir também dispositivos com suporte IPv6. Para se poder ter acesso a redes IPv6, é necessário pelo menos um dispositivo IPv6, como no exemplo do *tunneling* em que um *relay agent* faz a passagem para uma rede IPv6, mas daí a generalizar que todos precisam do suporte para IPv6 é errado.

Mito: Fazer um *upgrade* ao *backbone* da Internet é extremamente dispendioso e difícil de efectuar [10].

Situação real: Verdadeiro.

Pensar em fazer um *upgrade* ao *backbone* da Internet, é uma tarefa complicada, pelo que deverá ser pensada a longo prazo. Mudar todo o *backbone* em pouco tempo é uma proposta pouco atingível. Deverá começar pela utilização de processos de transição, e ao longo do tempo deverá começar-se a atribuir prioridade ao IPv6. O conceito de *backbone* IPv4 com ilhas IPv6 deverá transformar-se em ilhas IPv4 num *backbone* IPv6. A Figura 2.20 ilustra a dificuldade de um *upgrade* ao *backbone* da Internet.

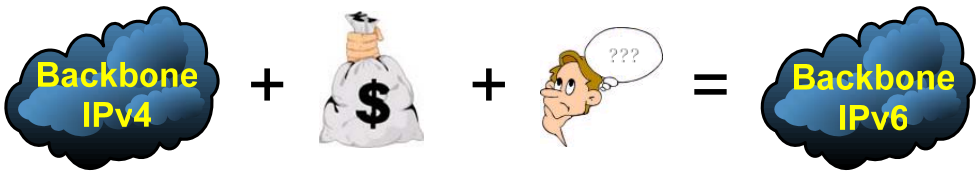


Figura 2.20 – A dificuldade de um *upgrade* ao *backbone*.

[15] Identificação de objectos por radiofrequência. Tem bastantes vantagens em relação ao normal código de barras tais como a maior capacidade de armazenamento e a menor sensibilidade a erros.

Mito: Vai ser difícil fazer o *upgrade* das aplicações para IPv6 [10].

Situação real: Verdadeiro.

Apesar de ser verdadeiro, a dificuldade irá variar muito de aplicação para aplicação; tudo depende da estrutura do código estabelecida em cada aplicação. Poderão existir aplicações em que o código já tenha sido pensado para mais tarde suportar IPv6 mas, por outro lado, existirão também aplicações com estruturas muito complicadas para alterar, e que nunca foram pensadas para IPv6.

Mito: O IPv6 é muito mais seguro do que o IPv4.

Situação real: Exagerado.

Não existe uma diferença assim tão grande, para se poder fazer uma afirmação deste tipo. A parte mais importante da segurança no IPv6 é o IPsec estar incluído nos E*xtension Headers*. Diz-se também que o IPsec é mandatário no IPv6, no entanto este "mandatário" engana um pouco, pois o protocolo IPsec necessita também de ser configurado. O IPv6 a nível de segurança é igual ao IPv4 com IPsec: não existe qualquer diferença.

Existe outro ponto de vista que tem a ver com os protocolos envolventes do IPv6, como por exemplo o ND (ver Secção 3.6). Este protocolo, por substituir o ARP, elimina algumas falhas de segurança existentes neste último. Por outro lado, provavelmente devido aos novos mecanismos, poderá introduzir novas falhas de segurança. Por isso de certa forma o balanço ao nível de segurança nos dois protocolos fica muito equilibrado, talvez com alguma vantagem para o IPv6.

Mito: Com IPv6, usando os *Extension Headers* de segurança (IPsec), as empresas não precisarão sequer de *firewall* [30].

Situação real: Falso.

A *firewall* tem funcionalidades completamente diferentes. Um exemplo é o bloqueio a determinada rede, ou mesmo a determinado dispositivo. Porém, a situação de ter uma *firewall* e o mecanismo de segurança do IPv6 é complicada, na medida em que, devido ao uso deste tipo de segurança em que a informação irá encriptada, a *firewall* não fará a mínima ideia do conteúdo dela, não sabendo assim se a informação será maliciosa ou não. As políticas de segurança neste caso, dentro e fora da empresa devem ser bem cuidadas, para que se consiga descobrir as possíveis fontes com código malicioso.

Mito: Para existir mobilidade é necessário IPv6.

Situação real: Falso a curto prazo, verdadeiro a longo prazo.

Existe também mobilidade para IPv4 (ver Subsecção 2.4.6), mas a tecnologia da mobilidade IPv6 é melhor a longo prazo devido à quantidade de endereços, pelo que apenas a mobilidade IPv6 deve ser considerada [28].

Mito: Este novo protocolo é bem melhor em termos de detecção e correcção de erros para redes sem fios.

Situação real: Falso.

Não, não é. Os mecanismos de detecção e correcção de erros localizam-se na camada de transporte, não tendo o IPv6 qualquer responsabilidade nessa camada [28].

Mito: IPv6 oferece uma QoS bastante melhor.

Situação real: Espera-se que seja verdadeiro.

O objectivo é realmente esse, mas actualmente ainda está em fase de estudo e praticamente não existem implementações [28].

Mito: Apenas o IPv6 suporta auto-configuração [28].

Situação real: Falso.

É bom não esquecer o DHCP do IPv4, que não deixa de ser também importante no IPv6. O IPv6 suporta outro tipo de auto-configuração (ver Secção 3.8), diferente do DHCP, mas convém lembrar que também existe DHCP para IPv6 (DHCPv6).

Mito: Com o IPv6, os *routers* vão ser bastante "aliviados" em relação à quantidade de redes das suas tabelas de encaminhamento.

Situação real: Verdadeiro.

Apesar de esta afirmação ser verdadeira, os mecanismos "aliviadores" dessas listas de rotas já se utilizam em IPv4 com o CIDR, porém nem todas as redes se preocuparam em o utilizar. O IPv6 tem uma estrutura hierárquica, logo este mecanismo é implementado de raiz, pelo que todas as redes o deverão utilizar, diminuindo assim ainda mais as longas listas de rotas, armazenadas nas tabelas dos *routers*.

Mito: Devido ao aumento do cabeçalho, o IPv6 vem tornar a Internet ainda mais lenta.

Situação real: Falso.

O que importa não é o tamanho do cabeçalho, mas sim a atenção que o *router* necessita de despender para cada campo dele. O IPv6 tem características que permitem maximizar a velocidade com que os *routers* processam os pacotes (ver Secção 2.4). Outro ponto importante é o facto do cabeçalho IPv6 ser fixo, o que vem facilitar em muito as implementações em *hardware* [30]. Também importante é considerar que o seu endereçamento é baseado já no CIDR aplicado ao IPv4.

2.6. Ponto de Situação

Até quando o IPv4 vai aguentar? É uma pergunta pertinente. O Ocidente está praticamente todo conectado; basta uma explosão da Internet nos países menos desenvolvidos e os endereços IPv4 esgotam-se rapidamente.

O principal erro do IPv4 começou logo no início das atribuições de endereços a entidades. Como na altura não se fazia a menor ideia da explosão que a Internet iria ter, atribuíram-se endereços IPv4 em quantidades absurdas a certas instituições. Devido ao facto de os EUA terem sido praticamente os obreiros de toda a estrutura da Internet, instituições académicas, governamentais e comerciais do país têm a maior parte da percentagem de endereços IPv4 (ver Secção 2.1).

A IANA, entidade responsável pela correcta atribuição dos endereços IP, atribui endereços aos RIRs, que por sua vez atribuem regionalmente os endereços IP aos respectivos ISPs. A IANA apesar de ser considerada uma entidade internacional, é controlada pela ICANN, que é uma entidade americana. Poderá pensar-se que se deve a isso o monopólio de endereços IP verificado nos Estados Unidos, mas não é esse o caso. O problema é anterior à IANA, que ainda tentou minimizar os estragos na medida do possível. Apesar disso muitas tentativas se têm feito para retirar da ICANN o controlo da IANA, sem a respectiva cooperação do governo dos EUA [122].

Em relação aos endereços IPv6, já não se vai repetir o mesmo erro no que diz respeito a atribuições monopolizadas, pois já existe a IANA para os regular e dividir correctamente pelos variados RIRs.

Considerando a parte de desenvolvimento IPv6, a maioria das organizações pertence aos EUA, mas é justamente esse o país que menos necessita de endereços IPv6 e onde eles são menos populares, devido à quantidade enorme de endereços IPv4 de que usufrui. Verifica-se assim um contra-senso, que só traz mais dificuldades ao progresso do novo protocolo.

Em relação aos prefixos atribuídos por parte da IANA aos diversos RIRs, a situação foi mudando ao longo dos anos, e provavelmente vai continuar a mudar, sendo atribuídos cada vez mais prefixos por região. Actualmente, o RIPE comanda a alocação de prefixos tendo 17 prefixos; segue-se o ARIN com 8 e o APNIC com 7; por último surge o LACNIC só com 1 prefixo atribuído. O AfriNIC ainda não

tem qualquer prefixo atribuído. Na Figura 2.21 podem ver-se estes números. Existe também um prefixo já atribuído para cenários de *tunneling* de IPv6 para IPv4 e o velho prefixo da 6Bone que em breve expirará (ver Subsecção 2.2.2).

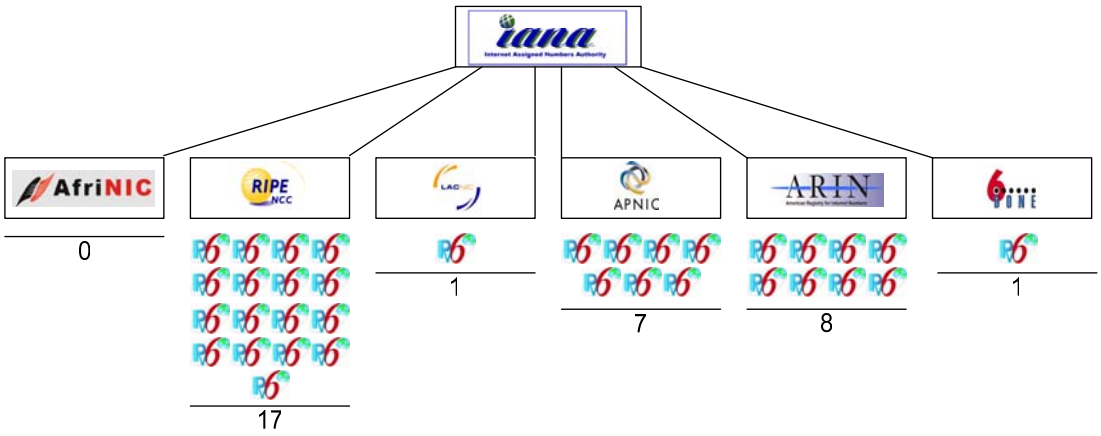


Figura 2.21 – A atribuição de prefixos pelos vários RIRs.

A máscara dos prefixos nem sempre é igual. A mais comum é */23*, mas existem outras. É de referir que a alteração duma máscara para um *bit* a menos contribui para uma quantidade de endereços muito significativa por continente, pelo que se o objectivo é uma análise detalhada, convém uma verificação na documentação da IANA (ver [178]).

Os RIRs por sua vez irão utilizar, com base nos prefixos atribuídos pela IANA, novos prefixos para os ISPs. Por fim o ISP irá atribuir um prefixo ou apenas um endereço para o utilizador final. O mecanismo é bastante semelhante ao IPv4, só que neste novo protocolo não serão atribuídos endereços dinâmicos[16] ao utilizador final, sendo que cada utilizador final deverá ter um endereço fixo.

Para mais informação acerca da situação actual do IPv6 consultar [13] e [14].

2.6.1. Ásia

A Ásia é, de longe, o continente que mais necessita do IPv6. Isto acontece devido ao facto de ser o continente com mais índice populacional que, juntamente com a menor quantidade de endereços IPv4 atribuídos (menos de 20%), o faz tornar na região em que o IPv6 é mais apetecível. O NAT é também bastante usado neste continente, sendo até usadas múltiplas camadas de NAT [10], onde máquinas para se conectarem à Internet têm de passar por vários processos de transição de endereços até na própria rede interna.

China e Índia são os principais responsáveis pela grande densidade populacional do continente asiático. Poderia pensar-se, então, que a maior parte da iniciativa asiática IPv6 se deve a estes dois países, mas não é isso que acontece, sendo o Japão o principal impulsionador de toda a tecnologia IPv6. A Índia, apesar de ter acordado mais tarde para esta tecnologia, segue logo a seguir. A China tem a grande desvantagem de ter um governo pouco incentivador à Internet. O IPv6 está depois presente em diversos países asiáticos, desde a Coreia do Sul ao Afeganistão, onde alguns testes já foram efectuados.

Se existe sítio para encontrar de momento boas implementações de IPv6, desde o nível empresarial até ao educacional, a Ásia é o sítio certo, sendo o Japão o expoente máximo.

Nem todas as organizações responsáveis pela promoção e esclarecimento do IPv6 na Ásia se intitulam de *task forces*, optando alguns países por variadas nomenclaturas, nomeadamente, *IPv6 Promotion Council* (Japão) e *IPv6 Forum* (Índia e Coreia do Sul). Estes casos não significam que sejam

[16] Tipo de endereço que muda, cada vez que o dispositivo correspondente acede à rede.

organizações à parte da *Task Force* Internacional, sendo o regime de cooperação absolutamente igual às outras *task forces*.

2.6.2. Europa

O velho continente é provavelmente o continente onde existe o número mais variado de países já com implementações IPv6, sendo os projectos Euro6IX e 6NET sem dúvida importantíssimos para a interligação de todos eles. Não existe nenhum país pertencente à União Europeia que não tenha a sua própria *task force* IPv6. Estes organismos são essenciais para a divulgação nacional do IPv6 em toda a organização empresarial e servem também como esclarecedores de qualquer dúvida a nível da tecnologia IPv6 no país.

Infelizmente, as implementações na Europa, comparando por exemplo com a Ásia, são ainda muito a nível promocional e de testes. O IPv6 como que ainda não aprendeu a "andar sozinho", precisando sempre de entidades a motivar e promover o novo protocolo.

2.6.3. América do Norte

Canadá e EUA estão ligados pela mesma entidade, a *task force* norte-americana. A quantidade de endereços IPv4 disponível só nos EUA, ultrapassa a população de ambos os países, pelo que se a pretensão é encontrar um local onde o IPv6 faça pouco sentido, então chegou-se ao sítio certo. Sendo a escassez de endereços IPv4 uma força motivadora que é posta fora de questão nestes países, terão de ser encontradas novas formas para motivar o uso do IPv6. É preciso criar a mentalidade de que o IPv6 não é apenas mais endereços. A *task force* norte-americana será, sem dúvida, a que terá um papel mais difícil no meio de todas as *task forces* existentes.

Inicialmente era esperado que a América do Norte fosse das partes mais difíceis de "acordar" para o mundo IPv6 [10], devido às razões apontadas acima, e de certo modo foi mesmo difícil de acordar, só que quando "acordou" foi de uma forma repentina e com uma intensidade até superior à verificada no continente Asiático, superando de longe as fracas expectativas esperadas para o Norte da América a nível de tecnologia IPv6. Só para se ter uma ideia, os EUA são já o país com a maior quantidade de endereços IPv6. O responsável por toda esta movimentação IPv6 repentina foi sem dúvida o DoD (ver Subsecção 2.2.7). Este organismo é praticamente considerado o principal ícone de toda a infra-estrutura da Internet nos EUA, e é mesmo uma entidade de referência a nível mundial.

Figura 2.22 – Responsabilidade do DoD dos Estados Unidos no desenvolvimento do IPv6.

2.6.4. América do Sul

Existem 3 *task forces* na América do Sul: Brasil, México e Cuba. É curioso como um país tão conservador como é Cuba, esteja já um passo à frente no que diz respeito ao IPv6. É de referir também que o LACNIC foi um dos últimos RIRs a ser criado, o que demonstra um certo atraso da América Latina e Central em relação a políticas de atribuição de endereços IPv4 e posteriormente IPv6.

2.6.5. Oceânia

Acções preliminares já desenvolvidas na Nova Zelândia e uma *task force* Australiana, são os acontecimentos mais significativos. O IPv6 na Austrália move-se de forma muito parecida aos EUA. De forma um pouco mais vagarosa, o DoD movimentou também muito a Austrália para o IPv6.

2.6.6. África

A Tunísia é o país mais avançado, tendo já sidas organizadas conferências a nível local e existindo já algumas redes IPv6 no país (ver [117]).

2.6.7. Portugal

Até agora, o grande passo dado no que diz respeito ao IPv6 em Portugal, foi a criação da *Task Force* Portuguesa (ver Subsecção 2.2.5). A situação nacional não é muito diferente da existente na Europa, principalmente considerando os países da União Europeia. O principal problema nem é a falta de vontade de mudar para uma nova tecnologia, mas sim a aquisição de dispositivos capazes de a suportar. Isto é um grande problema verificado por exemplo no mundo académico, que se sabe ser sempre um dos primeiros a chegar às mais recentes tecnologias. Mesmo assim, já existe um número bastante aceitável de instituições académicas com acesso IPv6 nativo:

- Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa;
- Instituto Superior Técnico, Universidade Técnica de Lisboa;
- Universidade de Évora;
- Universidade do Porto;
- Instituto Politécnico da Guarda;
- Instituto Politécnico de Bragança;
- Instituto Superior de Engenharia de Lisboa, Instituto Politécnico de Lisboa;

A próxima instituição a ligar-se ao *backbone* IPv6 de forma nativa será a Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria, durante a execução deste projecto.

Falando em termos comerciais, uma grande parte dos ISPs e das principais empresas de telecomunicações já alocou endereços IPv6, agora basta apenas disponibilizá-los. As empresas são:

- FCCN (*/35*, */35*, */34*, */33*);
- PT Comunicações (*/32*);
- Oni (*/32*);
- Novis (*/32*);
- KPNQwest Portugal (*/32*);
- NFSi (*/32*);
- Vodafone (*/32*);
- Telepac (*/32*).

Os tipos de endereços disponibilizados podem ser obtidos de forma automática, da mesma forma que o normal endereço IPv4, bastando para isso que o sistema operativo suporte IPv6. Infelizmente, o número de utilizadores a utilizarem endereços IPv6 é escasso, o que é compreensível, já que a maioria dos utilizadores nem sabe que tem um endereço IPv4.

3. Internet Protocol version 6

Estamos perante outro tipo de linguagem, referente à camada de rede. Os *routers* vão passar a comunicar de forma diferente e mais precisa, e vão ser também mais exigentes com a gestão do seu próprio tempo de processamento.

Desde o cabeçalho (presente em qualquer comunicação IPv6), até ao ICMPv6 (protocolo responsável por todo o esforço de manutenção de uma rede IPv6 saudável), passando ainda por todos os cabeçalhos de extensão, pelo endereçamento e os vários tipos de endereços; tudo isto é a nova linguagem de camada de rede.

3.1. Cabeçalho

O cabeçalho IPv6, independentemente das diferenças com o seu antecessor (ver Secção 2.4), está especificado [53], mas tem ainda alguns campos em fase de desenvolvimento. Porém, esses campos já têm os objectivos definidos, sendo apenas as formas de como os atingir que surgem na fase de experimentação. Na Figura 3.1 é mostrado o cabeçalho IPv6, e na Figura 3.2 uma captura real.

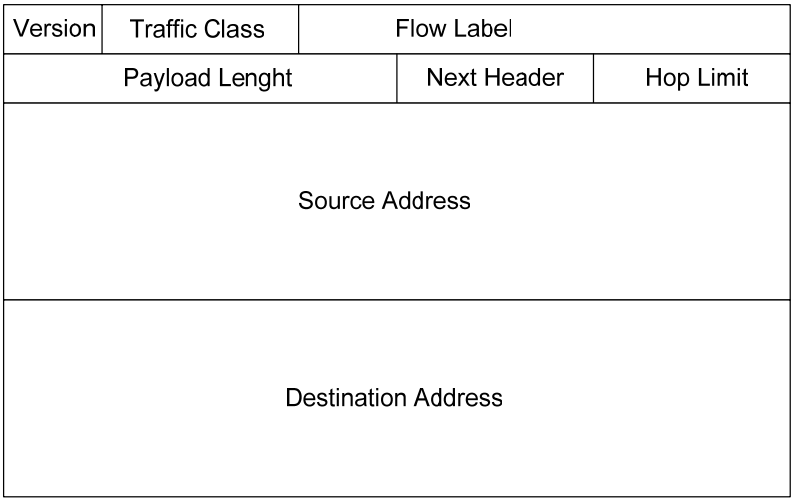


Figura 3.1 – Cabeçalho IPv6.

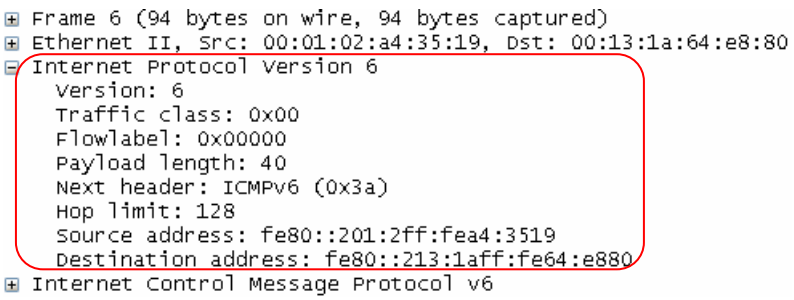


Figura 3.2 – Captura do cabeçalho IPv6.

Apresenta-se a seguir uma breve descrição dos campos do cabeçalho IPv6.

- ***Version***

A identificação do protocolo perante os *routers*. Como se pode verificar através da captura, a identificação é de IP versão 6.

- ***Traffic Class* e *Flow Label***

Os representantes da Qualidade de Serviço no cabeçalho IPv6. Tudo o que seja nestes dois campos diferente de 0 indicará uma diferenciação de tráfego (campo *Traffic Class*), ou diferenciação de fluxo (campo *Flow Label*). Na captura não foi utilizada qualquer tipo de diferenciação.

O uso do campo *Flow Label* é uma das maiores novidades do IPv6 e, talvez por isso, tem uma especificação própria [79].

- ***Payload Length***

Indicação para o tamanho, que imediatamente se segue, ao normal cabeçalho IPv6.

Verifica-se, na Figura 3.3, que o valor do *Payload Length* é igual a 40 (medida em *bytes*). Esse valor será igual ao tamanho total do cabeçalho seguinte, neste caso ICMPv6.

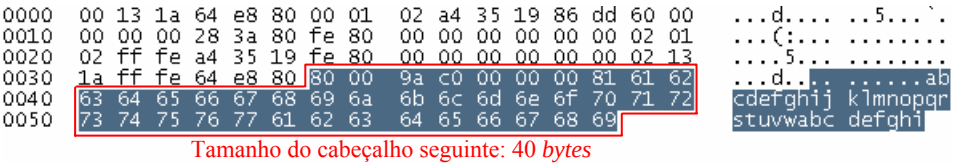


Figura 3.3 – Captura dos *bytes* do cabeçalho seguinte.

- ***Next Header***

Depois de ser indicada a carga do pacote, é indicado que protocolo ou cabeçalho de extensão vem a seguir ao cabeçalho IPv6. Devido ao facto de ter sido efectuado um *ping* (utiliza ICMP), o que se segue é um cabeçalho ICMPv6. Foi elaborada uma lista [182], que define uma numeração para cada protocolo. Neste caso, a identificação é 0x3a (formato hexadecimal), que equivale a 38 (formato decimal), o que indica ICMPv6. É de realçar que, na lista, se constata já uma distinção entre ICMPv6 e o normal ICMP (ICMPv4). Poderá então verificar-se já aqui, a distinção entre os dois protocolos que apesar dos mesmos objectivos, têm diferentes métodos.

- ***Hop Limit***

O alcance do pacote. Até que fronteiras determinado pacote poderá chegar (em termos de passagens pelos *routers*), ou seja, o número de saltos que um pacote pode dar. Cada valor unitário do *hop limit* significa uma passagem por um *router*, um salto.

O valor especificado nesta captura não tem qualquer tipo de significado especial. O valor varia de 0 a 255, devido aos 8 *bits* reservados a este campo. O valor de *hop limit* mais significativo nos mecanismos de IPv6 é o 1, pois limita a nível da mesma rede a propagação do pacote. Outros valores, como o 128 e o 64, também são bastante utilizados em outros mecanismos. O valor de *hop limit* pode também ser mudado nos respectivos *routers*.

- ***Source Address* e *Destination Address***

Os endereços de origem e destino. Neste caso, foram utilizados endereços *link-local*. (ver Subsecção 3.3.4.1). Como se vai perceber mais à frente, são os mais simples de configurar.

3.2. Cabeçalhos de Extensão (Extension Headers)

Os Cabeçalhos de Extensão (*Extension Headers*) [53] são os responsáveis por grande parte da eficiência do IPv6. Os cabeçalhos de extensão podem ser vistos numa localização intermédia, existente entre a camada de rede e a camada de transporte, não fazendo parte do normal cabeçalho IPv6, apesar de o complementarem. Utilizam níveis hierárquicos como forma de atingir a máxima eficiência, e esse nível é baseado no número de dispositivos que o precisam de processar. Quanto maior o número de dispositivos, maior a prioridade do cabeçalho de extensão. Sendo assim, a razão da hierarquia está associada à frequência com que é necessário o processamento do cabeçalho. A Figura 3.4 mostra a ordem hierárquica que determina a ordem dos cabeçalhos de extensão.

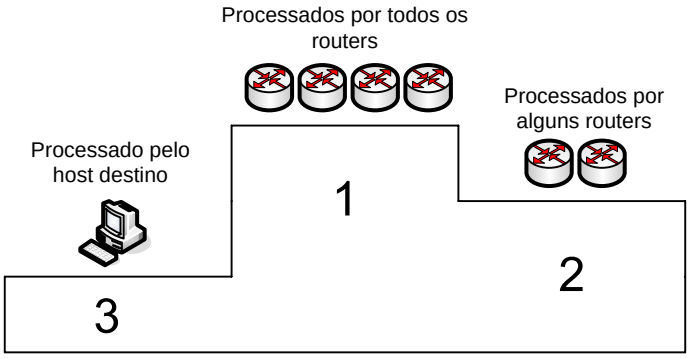


Figura 3.4 – Ordem hierárquica determina a ordem dos cabeçalhos de extensão.

Podem existir vários cabeçalhos de extensão (CE), ou pode não existir nenhum. No caso de existirem vários, interligam-se todos uns com os outros através do campo *Next Header*. Estes cabeçalhos formam, assim, uma corrente, até que o último cabeçalho de extensão aponte, por exemplo para um protocolo da camada acima. A Figura 3.5 exemplifica este processo de interligação.

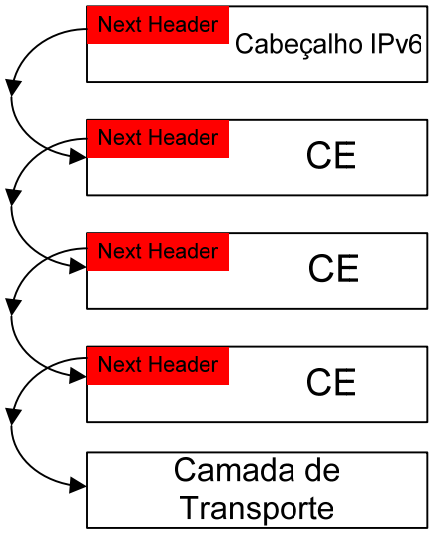


Figura 3.5 – Forma de como os cabeçalhos de extensão se interligam.

Os cabeçalhos de extensão não são examinados nem processados por todos os nós, a não ser pelo nó que contém o endereço do campo *Destination Address* do pacote. A única excepção é o cabeçalho *Hop-by-Hop Options*.

Alguns cabeçalhos de extensão são bastantes parecidos e completamente novos (*Hop-by-Hop Options header*, *Destination Options header*), outros não trazem novidades (*Authentication header*, *Encapsulating Security Payload*), e outros são um aproveitamento de alguns conceitos anteriores, misturados com algumas novidades (*Routing header*, *Fragment header*).

3.2.1. Hop-by-Hop Options Header

A excepção de todos os outros *extension headers*. É o cabeçalho utilizado, quando não existe qualquer estratégia possível de poupança de processamento a alguns *routers*, para se obter a função pretendida. De certa forma é um cabeçalho que nos faz relembrar a forma como todo o processamento do IPv4 funcionava, ou seja, *router* a *router*.

Todo o tipo de informação adicional, para além do normal cabeçalho IP, que precise de ser analisada por todos os *routers*, é transportada neste cabeçalho. Na Figura 3.6 e Figura 3.7 é apresentado o cabeçalho de extensão *hop-by-hop options* e a respectiva captura.

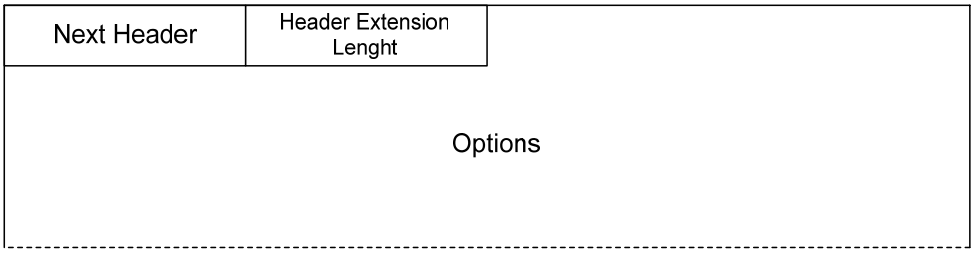


Figura 3.6 – Cabeçalho *Hop-by-Hop Options*.

```
⊞ Frame 169 (86 bytes on wire, 86 bytes captured)
⊞ Ethernet II, Src: 00:01:02:a4:35:19, Dst: 33:33:ff:a4:35:19
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 1
    Source address: fe80::201:2ff:fea4:3519
    Destination address: ff02::1:ffa4:3519
⊞ Hop-by-hop Option Header
    Next header: ICMPv6 (0x3a)
    Length: 0 (8 bytes)
    Router alert: MLD (4 bytes)
    PadN: 2 bytes
⊞ Internet Control Message Protocol v6
```

Figura 3.7 – Captura do cabeçalho *Hop-by-Hop Options*.

Existem vários mecanismos que necessitam deste cabeçalho de extensão. Neste caso trata-se do *Multicast Listener Discovery* (MLD) (ver Secção 3.7), e é dos exemplos mais fáceis de implementar, pois é automático, sem qualquer tipo de configuração adicional.

A estrutura do cabeçalho é bastante simples. Existe o já conhecido *next header*, depois um campo responsável pelo tamanho da opção e a respectiva opção. No final, poderá ser necessário um campo de *padding*, para completar o tamanho do pacote. Este tipo de campo era também utilizado no IPv4 (ver Subsecção 2.4.1).

O campo *next header* do cabeçalho IPv6 contém o valor 0, correspondente ao *hop-by-hop options*, pela numeração definida. De seguida, o *next header* do cabeçalho *hop-by-hop options* contém o valor para o próximo cabeçalho ICMPv6.

A opção escolhida neste caso foi a *Router Alert* [60]. Esta opção exclusiva do IPv6 permite dar informações adicionais aos *routers* de como tratar o pacote. Este tipo de informação adicional refere-se à informação que não pode ser transportada, no normal cabeçalho IP. No IPv4 onde não existia esta possibilidade, essa informação era forçada a ir parar às camadas superiores, voltando os *routers* a efectuar a violação de camada (ver Subsecção 2.4.5), sendo, desta forma, o processamento ainda mais demorado. Felizmente o cabeçalho de extensão *hop-by-hop options* veio resolver esse problema. A Figura 3.8 exemplifica um funcionamento da mensagem *Router Alert*.

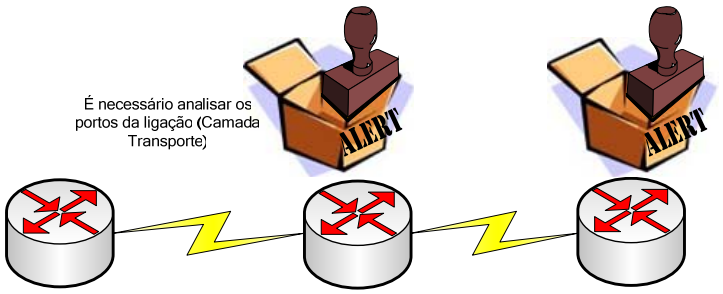


Figura 3.8 – Exemplo de um funcionamento da mensagem *Router Alert*.

3.2.2. Routing Header

Responsável pelo estabelecimento de itinerários para o pacote. O cabeçalho corresponde praticamente a um mapa, por onde os *routers* se orientam para o correcto encaminhamento do pacote. O conceito não é novo e era chamado de *Loose Source* e *Record Route Option* na versão 4 protocolo. Na Figura 3.9 é representado o cabeçalho de extensão de *routing*, com o único *Routing Type* definido, o "0". Na Figura 3.10 é apresentada a respectiva captura.

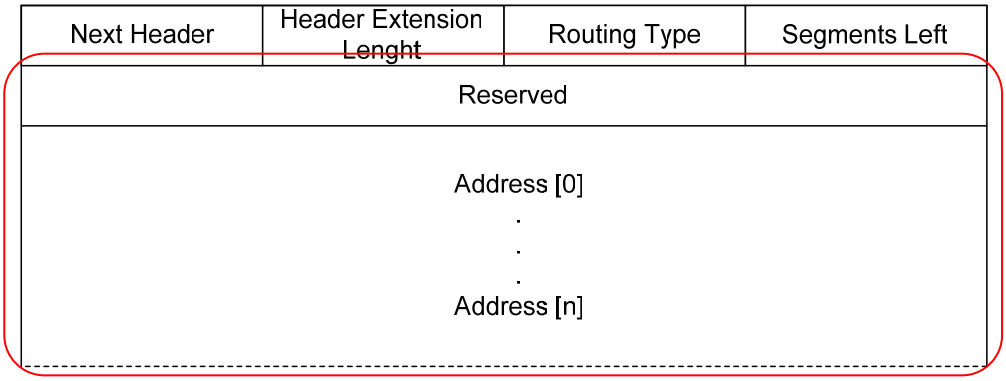


Figura 3.9 – Cabeçalho de extensão de *Routing*.

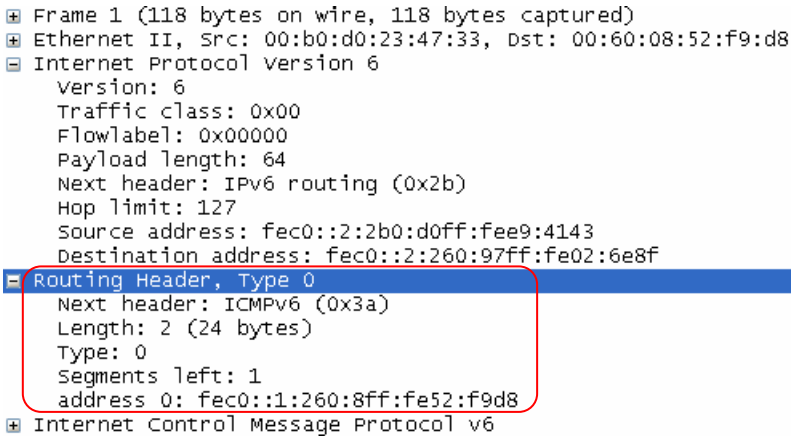


Figura 3.10 – Captura do cabeçalho de extensão de *Routing*.

As novidades neste cabeçalho são os campos *Routing Type*, *Segments Left*, *Reserved* e os vários campos *Address*.

O campo *Routing Type* só tem especificado o valor 0. Não existe, de momento, nenhuma proposta para especificação de outro. Poderá existir futuramente, pelo que o cabeçalho poderá mudar a partir do campo *Segments Left*. Nesta abordagem, apenas foi considerado o uso do *Routing Type* 0.

O número de segmentos que falta no itinerário do pacote é informação dada pelo campo *Segments Left*. Neste caso, sendo o *Segments Left* de valor 1, significa que ainda falta a passagem por mais um nó, até que se atinja o destino pretendido.

O campo *Reserved* não tem ainda qualquer utilidade prática. Este campo representa o carácter ainda experimental deste cabeçalho de extensão. Deve ser inicializado a 0 para a transmissão, e é ignorado na recepção; por essa razão não aparece na captura.

Por fim, verificam-se os endereços. No mínimo, precisará de existir um endereço no cabeçalho de *routing*, pois só assim o uso deste cabeçalho começa a fazer sentido. Se apenas um destino for especificado, então o normal cabeçalho IP trata do encaminhamento sozinho.

O funcionamento de todo este processo de encaminhamento mapeado não diz respeito apenas ao cabeçalho de *routing*, mas sim a um processo de interligação, existente entre o cabeçalho IP e o cabeçalho de *routing*. Este processo consiste em trocas consecutivas de endereços, conforme a sua quantidade. Mais especificamente, as trocas são efectuadas entre os campos *Address* do cabeçalho de *routing* e o campo *Destination Address* do IP. O campo do IP, neste caso, torna-se temporário até o destino final.

O mecanismo baseia-se essencialmente nos seguintes passos:

- Chegada ao 1.º destino (campo *destination address* do IP);
- É efectuada a primeira troca, entre o 1.º *address* (*routing*) com o *destination address* (IP);
- Decrementa-se o número de segmentos em 1 unidade (*segments left* – 1);
- Chegada ao 2.º destino.

Na captura, é exactamente este mecanismo que se irá suceder, sendo o 2.º destino, já o destino final. Ao chegar ao 1.º destino, o endereço *FEC0::2:260:87FF:FE02:6E8F*, irá ser substituído pelo *address* 0: *FEC0::1:260:8FF:FE52:F9D8* (destino final). Em seguida, o *segments left* 1, ao ser decrementado passará a 0, e atinge-se o destino final. O mecanismo vai-se repetindo, conforme o número de endereços, especificados no cabeçalho de *routing*.

Na Figura 3.11 apresenta-se um esquema do mecanismo em maior detalhe, verificando-se todas as trocas de endereços.

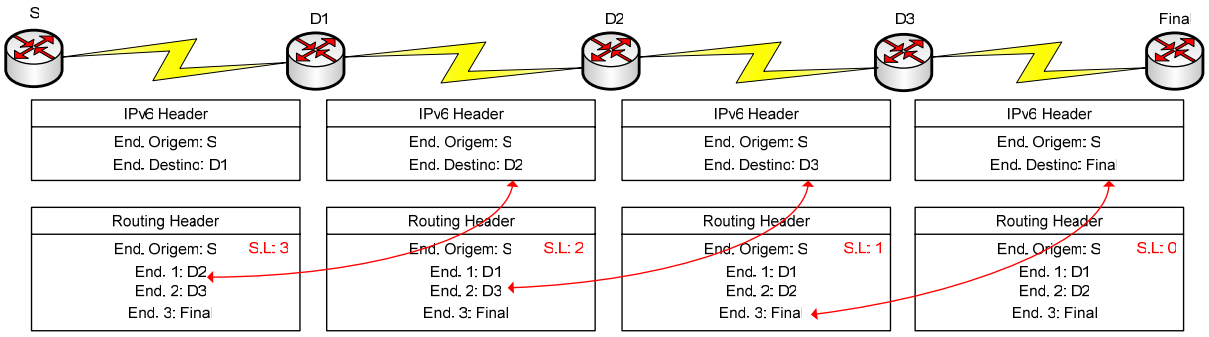


Figura 3.11 – Mecanismo utilizado pelo cabeçalho de *routing* para estabelecimento de rotas.

3.2.3. Fragment Header

Tal como sucede com o conceito de *routing*, o conceito de fragmentação também não é novo. O mecanismo deste cabeçalho de extensão é basicamente o mesmo da fragmentação existente no IPv4. A Figura 3.12 e a Figura 3.13 representam o cabeçalho *fragment* e as respectivas capturas.

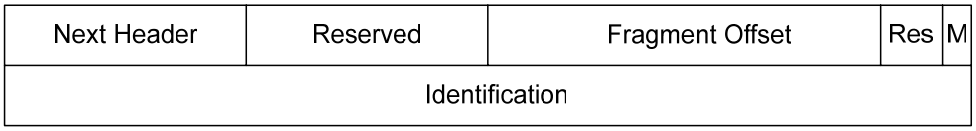


Figura 3.12 – Cabeçalho de extensão *fragment*.

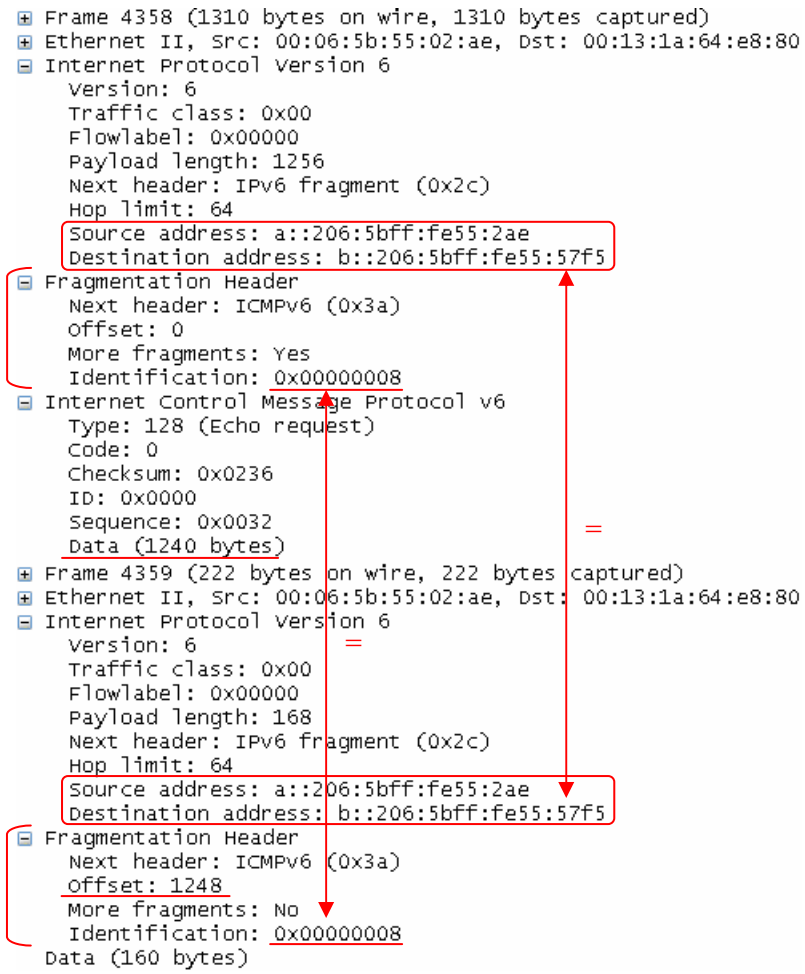


Figura 3.13 – Capturas dos cabeçalhos *fragment*.

Curiosamente, existem dois campos *Reserved* (*Reserved* e *Res*), que funcionam de forma igual ao *Reserved* do *routing header*; verifica-se então, também aqui, alguma experimentação. Depois surgem os três campos essenciais em todo o processo de fragmentação: *Fragment Offset*, *M flag* e *Identification*. Estes campos são bem conhecidos do anterior cabeçalho IPv4 (ver Subsecção 2.4.1).

O endereço, em termos de *bits*, do próximo fragmento, ou outro cabeçalho, aparece assim ao *Fragment Offset*. A notificação de ser ou não, último fragmento é da responsabilidade da *M flag*. Por fim, a identificação do fragmento, para correcta recepção de todo o pacote pelo receptor, é feita pelo campo *Identification*.

Para esta captura foi efectuado um *ping* de 1400 *bytes*.

Verifica-se então, na Figura 3.13, a fragmentação a ocorrer, sendo o normal pacote dividido em dois pacotes mais pequenos (fragmentos). A soma dos dados dos dois fragmentos irá dar o tamanho dos dados do pacote original: 1400 *bytes* (pacote original) = 1240 *bytes* (1.º fragmento) + 160 *bytes* (2.º fragmento).

Em relação ao *offset* do segundo pacote, consegue verificar-se o que valor de 1248 *bytes* é todo o tamanho do 1.º fragmento.

A *M flag* indica, no 1.º fragmento, que ainda falta outro. Isto é essencial para a correcta recepção dos pacotes. Essencial é também o facto de os campos *Identification*, *Source Address* e *Destination*

Address serem necessariamente iguais nos vários fragmentos, conseguindo-se assim a correcta identificação dos pacotes.

Em todo este processo de fragmentação terá de existir sempre uma parte não fragmentável, para que toda a parte fragmentável possa ser correctamente identificada pelo receptor. Assim sendo, a parte que não é fragmentável corresponde a todos os cabeçalhos existentes, e a parte fragmentável corresponde a todos os dados. Esta estrutura é representada na Figura 3.14.

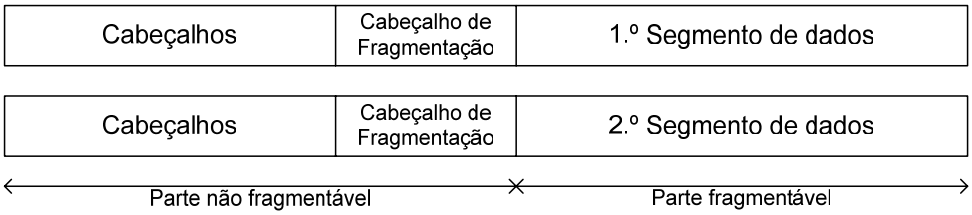


Figura 3.14 – Estrutura de um pacote fragmentado.

3.2.4. Authentication Header e Encapsulating Security Payload

Nada de novo, exactamente aquilo que existia em IPv4 com o adicionar do IPsec, mas agora em formato de cabeçalhos de extensão [51] [52]. Na Figura 3.15 e Figura 3.16 é mostrado um exemplo com o cabeçalho de extensão *Authentication* do IPv6, comparando com o do IPv4.

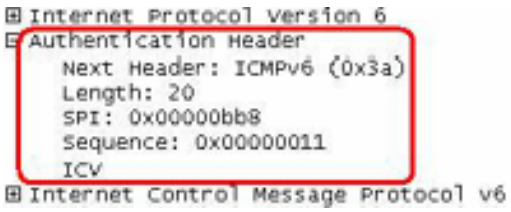


Figura 3.15 – Captura do *Authentication Header* no IPv6.

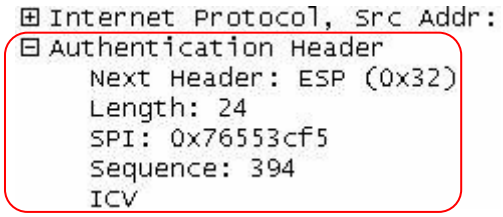


Figura 3.16 – Captura do *Authentication Header* no IPv4.

Como se verifica nas duas capturas, o cabeçalho é exactamente igual ao do IPSec existente no IPv4. Em relação ao Encapsulating Security Payload, a situação é igual.

3.2.5. Destination Options Header

O cabeçalho *Destination Options* é igual ao *Hop-by-Hop Options*, mas o objectivo deste cabeçalho é totalmente diferente. O objectivo é transportar informação adicional, que apenas será lida pelos *routers* destino. Existem duas situações, em que poderá existir mais do que um *router* destino: a primeira é a simples situação de o endereço destino ser um endereço *multicast*, a segunda é quando o cabeçalho *destination options* surge imediatamente antes do cabeçalho de *routing*. Neste último caso, o cabeçalho irá ser lido por todos os endereços de destino pertencentes ao itinerário do pacote.

Um caso concreto da utilidade desde cabeçalho é, por exemplo, o seu uso na mobilidade. O nó móvel no IPv6 envia os pacotes, com o seu *care-of-address* como origem (ver Subsecção 2.4.6), mas envia também o seu *home address* no *destination options*. Este mecanismo é essencial, pois fornece a

informação de controlo da sessão estabelecida, entre o nó móvel e outro dispositivo. Em IPv4, este tipo de informação teria de ser enviada separadamente em pacotes UDP [1].

Tanto este cabeçalho como o *hop-by-hop options* funcionam como cabeçalhos de *piggybacking*[17] para todo o tipo de informação, adicional ao normal cabeçalho IP. A principal diferença é que um é pouco incomodativo (apenas os *routers* destino), e o outro é bastante incomodativo (todos os *routers*).

Tendo sido referida a ordem hierárquica dos cabeçalhos de extensão (ver Figura 3.4), e eles sido apresentados, a ordem torna-se até fácil de adivinhar. É de realçar que o cabeçalho de extensão *destination options* é o único cabeçalho a poder estar numa ordem diferente. Isto no caso de ser especificado que a leitura desse cabeçalho de extensão é também feita pelos *routers* intermédios da rota do pacote.

Ordem de aparecimento dos cabeçalhos de extensão:

1. *Hop-by-Hop Options header*;
2. *Destination Options header* (1.ª ordem);
3. *Routing header*;
4. *Fragment header*;
5. *Authentication header*;
6. *Encapsulating Security Payload*;
7. *Destination Options header* (2.ª ordem).

Na Figura 3.17 poderá verificar-se em maior detalhe a importância de toda esta hierarquia de cabeçalhos. Foi considerado, neste exemplo, o uso específico de 4 cabeçalhos de extensão na transmissão (*Hop-by-Hop Options header*, *Authentication header, Encapsulating Security Payload* e *Destination Options header*). Ao existir a correcta hierarquia, os *routers* processarão primeiro, os que são mais utilizados pela maioria dos *routers*. Desta forma, eles não vão sequer analisar os cabeçalhos de extensão que não lhe dizem respeito. Pelo contrário, se uma hierarquia correcta não fosse definida, a maioria dos *routers* teria obrigatoriamente de passar por uma fase de análise de todos os cabeçalhos de extensão, até chegar aos que lhe dizem respeito e então finalmente os processar. Desta forma, iria ser desperdiçado tempo de análise nos cabeçalhos de extensão que não são importantes para o *router*.

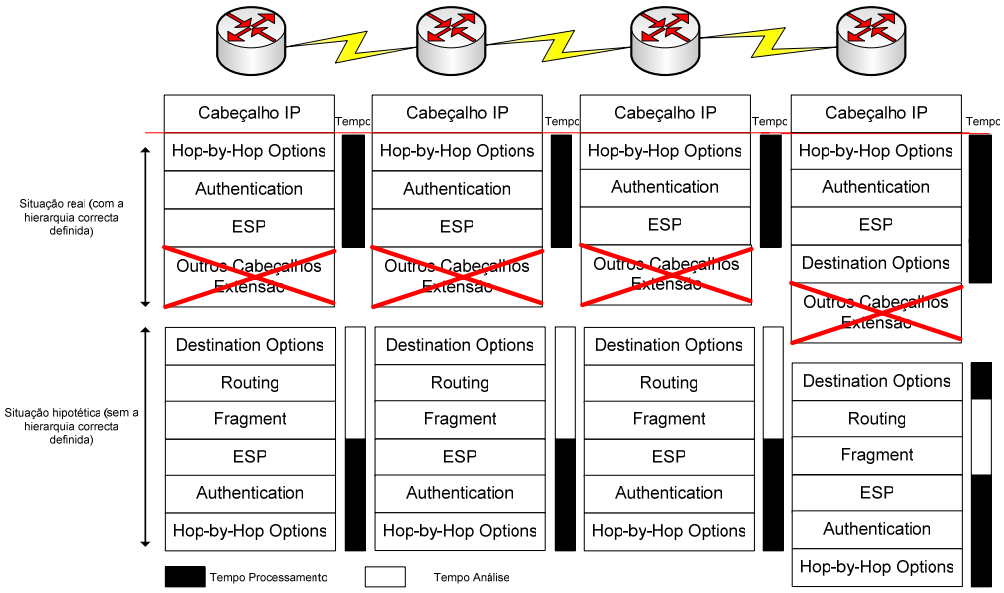


Figura 3.17 – O funcionamento da hierarquia nos cabeçalhos de extensão.

[17] Mecanismo no qual, certa informação vai "à boleia" de um pacote que transporta outro tipo de informação.

3.3. Endereçamento

O endereçamento [77] é a face mais visível do IPv6. Num primeiro olhar, ao verificar qualquer vestígio de IPv6, é por ele que procuramos. O endereçamento em IPv6 é bem diferente do IPv4 (ver Subsecção 2.4.3). A nova notação *column hexadecimal* vem alterar um pouco a nossa noção de endereço decimal. Começando pela quantidade de números, passando pela introdução de letras, e acabando nos sistemas de compressão de endereços, assim se diferencia o endereço IPv6.

3.3.1. Representação

A representação escolhida foi de 8 grupos de 4 dígitos hexadecimais cada. O método hexadecimal foi o escolhido, devido à facilidade de conversão entre binário e hexadecimal (4 dígitos em binário, correspondem directamente a 1 dígito hexadecimal).

Na Figura 3.18 é exemplificado o processo de conversão.

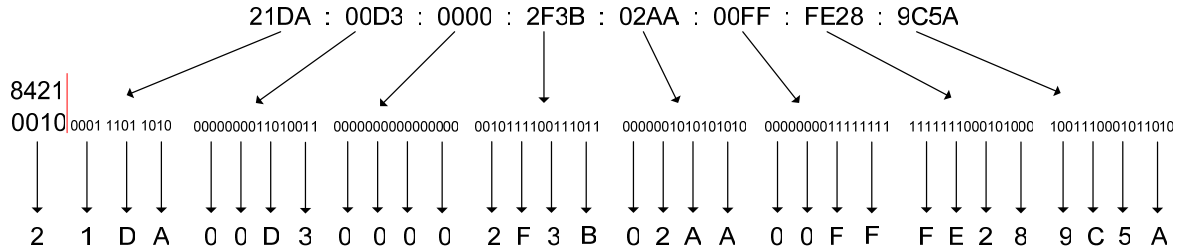


Figura 3.18 – Endereçamento Hexadecimal/Binário e Binário/Hexadecimal [2].

Curiosamente, foi especificada [47] outra representação possível para os endereços IPv6. Este tipo de representação tinha como principal objectivo diminuir a extensão de números para apenas 20 dígitos, no máximo. Intitula-se base 85, devido ao facto de serem necessários 85 símbolos para representar o endereço IPv6, apenas com 20 dígitos. Estes 85 símbolos são símbolos da normal tabela ASCII.

Este possível tipo de representação já foi especificado há bastante tempo (1996). Apesar de nada indicar estar obsoleta, raramente se fala ou utiliza esta representação, pelo que está condenada a desaparecer naturalmente. A Figura 3.19 representa uma tradução de um endereço IPv6 para este tipo de representação.

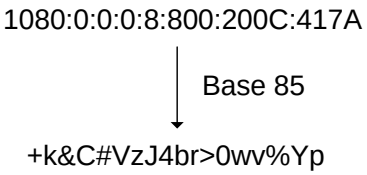


Figura 3.19 – Representação de um endereço IPv6 em base 85 [3].

Poderá pensar-se o porquê de tanta preocupação com a representação dos endereços IPv6, já que é praticamente impensável o seu uso directo, sem recorrer ao DNS. Esta ideia realmente faz algum sentido, se começarmos a pensar no normal utilizador de uma rede, mas perde-o se pensarmos nos administradores das mesmas. O tempo que um administrador demora a configurar uma rede, e principalmente a facilidade na resolução de algum problema ocorrente na mesma, é extremamente importante. Apesar da auto-configuração (ver Secção 3.8) do IPv6 ajudar imenso, situações como a definição de hierarquias (ver Subsecção 3.3.2) dentro da própria rede, atribuição por DHCPv6, ou até avarias no servidor de DNS, estão sempre sujeitas a ocorrer, devendo-se, por isso, facilitar a tarefa ao administrador.

Compressão de endereços

Devido a alguns métodos de atribuir endereços IPv6 (principalmente o simples método de começar a atribuir endereços de forma crescente), os endereços estão bastante sujeitos a ter extensas sequências do dígito zero. Devido a este facto, foram elaboradas duas técnicas de compressão destas sequências, para a consequente diminuição da extensão dos endereços IPv6.

A primeira técnica é bastante simples, e de certa forma já existia no IPv4. Consiste em eliminar todos os zeros à esquerda de um bloco de 16 *bits*. A excepção acontece quando existe um bloco de 16 *bits* apenas com zeros, nesse caso um zero terá de ficar no bloco. Esta primeira técnica é exemplificada na Figura 3.20.

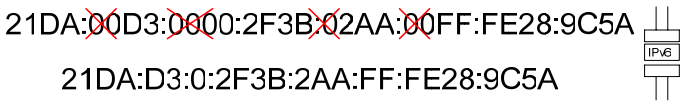


Figura 3.20 – Primeira técnica de compressão de zeros.

A segunda técnica consiste em agrupar as variadas sequências de zeros existentes nos blocos de 16 *bits*. Este procedimento é efectuado com dois "dois pontos" ("*::*") e é exemplificado na Figura 3.21.

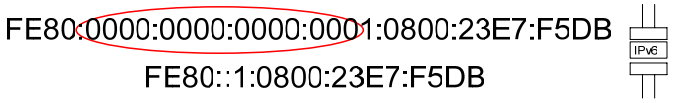


Figura 3.21 – Segunda técnica de compressão de zeros.

O uso desta segunda abreviação só pode ser efectuado uma vez. A razão para esta restrição é que o computador usa sempre uma representação de 128 *bits* do endereço IPv6, ou seja, ele quando encontra "*::*" expande automaticamente os zeros até chegar aos normais 128 *bits*. Na situação de existirem duas abreviações, o computador nunca sabe quantos zeros há-de adicionar a cada uma [1].

Por vezes pode parecer um pouco complicado saber quantos *bits* a 0 existem em determinada abreviação, mas existe uma simples fórmula que facilita imenso essa dúvida [2]:

- Número de *bits* a zero compactados em bloco = (8 – [número de blocos visíveis]) × 16
- Aplicando ao exemplo da Figura 3.21: (8 – 5) × 16 = 48 zeros compactados

O uso destas duas técnicas dá origem por vezes a alguma confusão. Alguns exemplos típicos estão representados na Figura 3.22.

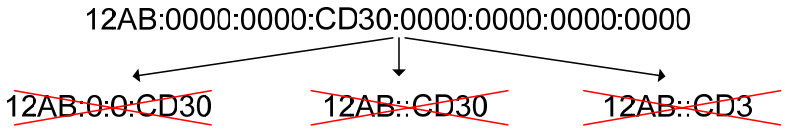


Figura 3.22 – Algumas confusões, devido às técnicas de compressão de zeros.

Nota: O endereço não é *case sensitive*. Utilizaram-se apenas letras maiúsculas para uma melhor visualização.

Prefixos

Em relação aos prefixos, também existem algumas confusões no que diz respeito à sua representação. Olhando a Figura 3.23, a partir do primeiro endereço, repare-se que só se pretende o prefixo desse endereço, por isso o que interessa são os primeiros 64 *bits* a contar da esquerda (os *bits* mais significativos).

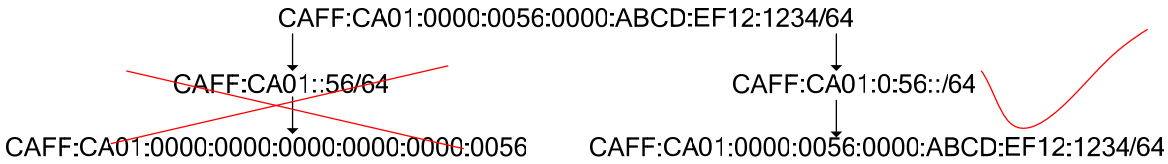


Figura 3.23 – Possível confusão no uso de prefixos [1].

Baseado ainda na figura acima (mas na parte correcta), podemos verificar também que, ao invés de estar a escrever o endereço do nó, e depois também o seu prefixo, ou seja *CAFF:CA01:0000:0056:0000:ABCD:EF12:1234* e *CAFF:CA01:0:56::/64*, podemos abreviar e escrever o endereço e o prefixo tudo num só: *CAFF:CA01:0000:0056:0000:ABCD:EF12:1234/64* [77]. De certa forma, o que se verifica neste caso é um funcionamento muito idêntico a um endereço IPv4 com a sua respectiva máscara (p. ex., *192.168.0.1/24*).

3.3.2. Hierarquia

Uma das principais características do IPv6, principalmente para a organização do *backbone* da Internet e também para a organização interna e externa de entidades, é o facto de o endereço IPv6 ser completamente hierárquico, baseado no CIDR do IPv4.

Na Figura 3.24 é mostrado como está organizado um normal endereço global *unicast* (ver Subsecção 3.3.4.1). Será este o tipo de endereço a que se poderá associar uma hierarquia.

Figura 3.24 – Estrutura de um endereço global.

Este endereço tem um prefixo global, dado pelas entidades competentes, como o RIPE na Europa. Esse endereço poderá já conter alguma hierarquia, mas ela é completamente transparente para a entidade a que o prefixo foi atribuído. É necessário que a Interface ID tenha pelo menos 64 *bits* [77], assim a entidade a quem é entregue o prefixo, controla sempre mais de 50% do endereço (biliões de endereços), pois para além dos 64 *bits* da Interface ID, controla ainda a Sub-rede ID.

Existem vários tipos de prefixos que se podem atribuir (ver Subsecção 3.3.5).

Imagine-se que é atribuído um prefixo de 48 *bits*, e que uma empresa pretende, por exemplo, fazer uma hierarquização usando 4 *bits* para os níveis geográficos (2^4 = 16 regiões) e 6 *bits* para o nível departamental (2^6 = 64 departamentos). Resumindo, existirão até 16 regiões, e cada uma dessas regiões está dividida em até 64 departamentos. A Figura 3.25 exemplifica a constituição acima enunciada.

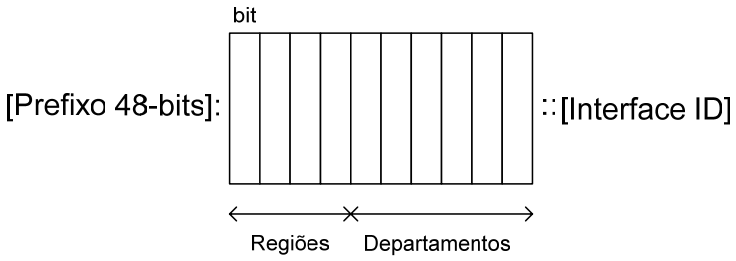


Figura 3.25 – Estrutura de uma hierarquia.

Agora poderá imaginar-se um caso mais completo. Um caso em que é dado um prefixo específico a uma entidade, e a partir desse prefixo a entidade constitui novos prefixos para a sua hierarquia.

Para criar uma lista numerada de sub-redes, e respectivos prefixos é necessário seguir essencialmente os seguintes passos [17]:

- Calcular o número de *bits* da hierarquia imediatamente anterior à que irá ser criada (A).

  A = m – 48, onde m = tamanho do prefixo da rede onde se vai efectuar a hierarquia.

  O valor 48 poderá ser diferente, consoante o prefixo que a entidade competente forneça; foi utilizado neste caso, porque é o cenário típico.

- Número de prefixos que se irá obter (N).

  $N = 2^s$, onde s é igual ao número de *bits* escolhidos para criar sub-redes.

- O valor incremental entre as sub-redes (i, valor hexadecimal).

  $i = 2^{16-(A+s)}$

- Criar duas colunas, com N linhas. A primeira irá conter os números dos prefixos, ou alguma outra informação sobre eles; a segunda irá conter os prefixos das sub-redes.

  Na primeira linha, segunda coluna, colocar o prefixo original com a nova extensão.

  [Prefixo 48-bits]:F::/[tamanho do novo prefixo], onde F é o valor hexadecimal, do último conjunto de 16 *bits*.

  Na linha imediatamente abaixo, adicionar em hexadecimal, o valor do incremento ao último conjunto de 16 *bits*, e continuar a incrementá-lo até chegar a última linha.

  [Prefixo 48-bits]:F+i::/[tamanho do novo prefixo]

Basta então, aplicar toda esta teoria. Considere-se o seguinte exemplo:

A FCCN pediu ao RIPE para que lhe fosse atribuído um prefixo */48*, efectuando a divisão em sub-redes com prefixos */51* (ou seja, utilizando 3 *bits* para a divisão). A Instituição X, que tem a FCCN como seu ISP, pediu agora um prefixo para que pudesse usufruir de uma rede IPv6. Assim sendo, a FCCN disponibilizou à Instituição X um prefixo */51*, mais concretamente o *2001:690:FFFF:C000::/51*.

A Instituição X quer agora dividir em sub-redes todos os seus departamentos. Como tem oito, o número de *bits* que disponibiliza para a divisão será 3 ($2^3 = 8$ departamentos).

Sendo assim, para começar temos os valores: F = C000, s = 3, m = 51.

- A = 51 – 48 = 3
- $N = 2^3 = 8$
- $i = 2^{16-(3+3)} = 1024 = 0x400$ (hexadecimal)

Departamento	Prefixo de sub-rede
A	*2001:690:FFFF:C000::/54*
B	*2001:690:FFFF:C400::/54*
C	*2001:690:FFFF:C800::/54*
D	*2001:690:FFFF:CC00::/54*
E	*2001:690:FFFF:D000::/54*
F	*2001:690:FFFF:D400::/54*
G	*2001:690:FFFF:D800::/54*
H	*2001:690:FFFF:DC00::/54*

Tabela 3.1 – Exemplo de uma hierarquia efectuada.

Caso seja necessário subdividir determinado departamento, basta aplicar os mesmos passos. Apenas se terá de ter em atenção que não se pode ultrapassar o */64*, pois nesse caso a Interface ID será abrangida.

A hierarquia proporcionada pelas sub-redes nem sempre é necessária. Não faz o menor sentido fazer este tipo de divisões em pequenas empresas.

3.3.3. A Interface ID e a Norma EUI-64

Esta parte do endereçamento está directamente ligada à auto-configuração (ver Subsecção 3.8.1) do endereço IPv6. Atrás, já existe uma referência sobre o facto de a Interface ID nunca poder ser menor que 64 *bits*. A razão de ser desta situação prende-se com a utilização da norma EUI-64, que é utilizada num processo de auto-configuração do IPv6.

A norma EUI-64 (ver [93]) baseia-se geralmente no endereço MAC, mas poderá não se basear. Exemplos de tecnologias que usam o MAC, ou seja que se baseiam no *standard* IEEE 802, são a *Ethernet*, o *Token Ring* e o FDDI. Existem, por outro lado, tecnologias como o ATM e o *Frame Relay*, que apesar de não terem qualquer endereço MAC, também utilizam a norma EUI-64.

O endereço MAC é já sobejamente conhecido e abrangente, mas existem dois pormenores (ver Figura 3.26) que podem ter passado despercebidos:

- ***Universal/Local bit* (U/L)** – Indica se o endereço foi atribuído universalmente ou localmente. Se for 0 o IEEE é que o atribuiu, se for 1 o endereço foi atribuído localmente. Este último caso, significa que o administrador substituiu o normal endereço de empresa por outro.
- ***Individual/Group bit* (I/G)** – Indica se o endereço é *unicast* ou *multicast*. O valor 0 representa o *unicast*, o valor 1 o *multicast*.

Estes dois *bits* por omissão estão a 0, correspondendo respectivamente a um endereço universal *unicast*.

Devido à imensa popularidade da *Ethernet*, esta foi a escolhida para o exemplo de como é efectuado todo o processo, desde o endereço MAC até ao endereço final da Interface ID. Assim sendo, o processo divide-se em 2 passos:

1.º Passo – Adição de 16 *bits*, ao normal endereço MAC, a partir do final dos primeiros 24 *bits*, respectivamente 11111111 11111110 (FFFE em hexadecimal).

2.º Passo – Tendo o endereço EUI-64, irá existir uma inversão no *bit* "u" para que a Interface ID fique correctamente atribuída.

Estes passos estão representados na Figura 3.26.

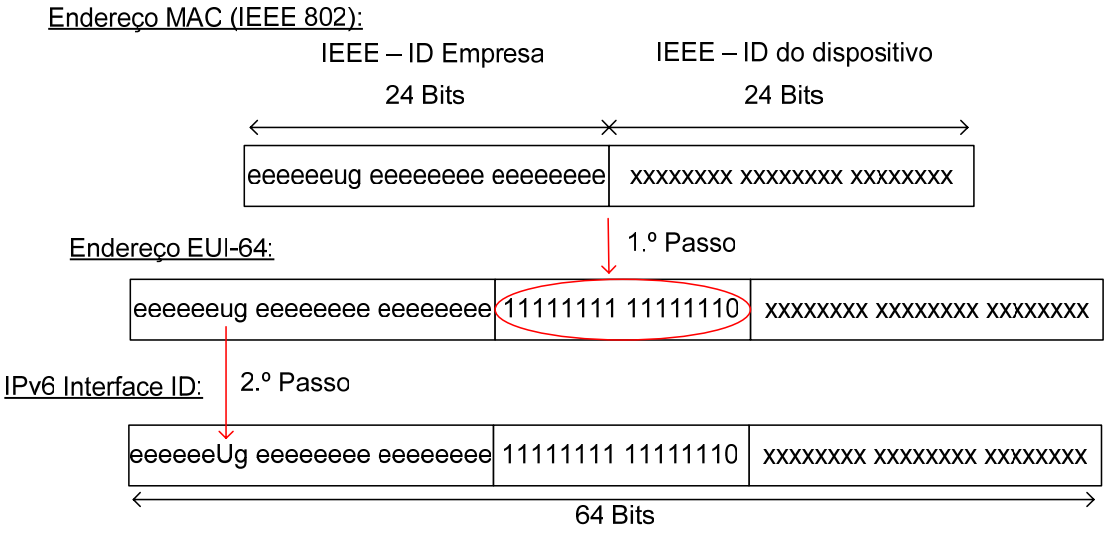


Figura 3.26 – Configuração da Interface ID.

Esta inversão de *bit*, não tem qualquer utilidade no exemplo específico da *Ethernet*. A sua utilidade só se verifica em outro tipo de tecnologia que não siga o *standard* IEEE 802. Nesse caso não existirá MAC, podendo o endereço em que se baseia a norma EUI-64 ser um simples 1 ou 2.

Considerando, então, este último caso, irá demonstrar-se o que pode acontecer se não existir inversão:

> Utilizando o endereço 1, na passagem para EUI-64 o endereço passava de 1 para *00.00.00.00.00.00.00.01*, e a Interface ID (sem inversão do *bit*), ficaria *0000:0000:0000:0001*. Então, inserindo agora a Interface ID num endereço completo, por exemplo o *link-local*, o resultado final seria *FE80::1*. Este endereço é o *loopback link-local*, o que geraria conflitos.

Situação, agora com inversão:

> A passagem do endereço EUI-64 para Interface ID ficaria *0200:0000:0000:0001*, e consequentemente o endereço final *link-local* ficaria *FE80::200:0:0:1*, não existindo assim qualquer conflito.

Este foi apenas um exemplo, mas o objectivo essencial desta inversão é facilitar aos administradores a atribuição de endereços que não tenham endereço MAC atribuído. Assim, os administradores poderão continuar a atribuir manualmente endereços bastante simples, e de forma crescente (0, 1 ,2 ,3, ...).

Como já foi mencionado, o funcionamento da norma EUI-64 varia de tecnologia para tecnologia, pelo que uma boa leitura sobre o funcionamento de determinada tecnologia com o IPv6 se revela essencial, nomeadamente ao nível de RFCs:

- *Ethernet* – RFC 2464;
- FDDI – RFC 2467;
- *Token Ring* – RFC 2470;
- PPP – RFC 2472;
- ATM – RFC 2492;
- *Frame Relay* – RFC 2492.

3.3.4. Tipos de endereços e abrangência

Esta caracterização dos endereços IPv6 revela-se temporária, na medida em que é das partes mais discutidas do IPv6. Já existiram variados tipos de endereços que ficaram obsoletos à medida que as discussões se foram sucedendo. A quantidade de endereços especiais que existiu já chegou a ser mais do dobro do que é agora. Agora existe um número relativamente curto de endereços especiais (com prefixo já definido).

Os prefixos até agora mencionados podem-se chamar de prefixos normais, ou seja, aqueles que são atribuídos a entidades, para a partir daí elas organizarem os seus endereços de forma hierárquica ou não. O que se vai falar agora é de prefixos especiais, que foram atribuídos pela IANA, e têm funções específicas, seja em abrangência geográfica (*link*, *global*, …) ou em tipo (*unicast, multicast*, *anycast*). A Tabela 3.2 pretende elucidar sobre os principais tipos de prefixos especiais.

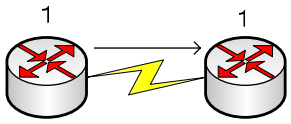
Atribuição	Prefixo em binário	Prefixo em hexadecimal
Global Unicast	*001*	*2000::/3*
Unique Local Unicast	*1111 110*	*FC00::/7*
Multicast	*1111 1111*	*FF00::/8*
Link-Local	*1111 1110 10*	*FE80::/10*

Tabela 3.2 – Principais prefixos especiais [178].

Verifica-se, então, que existe um prefixo específico para cada tipo de endereçamento, *Global Unicast*, *Unique Local Unicast*, *Multicast* e *Link-Local*. Poderá parecer que falta nesta tabela o *Anycast*, pois também é um endereço especial, mas esse facto será explicado adiante. Por enquanto fica-se a saber que existem mais endereços especiais para além destes.

3.3.4.1. Unicast

Tipo: 1 para 1



Todos os endereços da Tabela 3.2 são *unicast*, excepto o *multicast*. Os endereços *unicast* geralmente têm associado um prefixo. Existem variados tipos de endereços *unicast*, e futuramente é muito possível a existência ainda de mais [179].

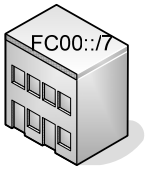
- ***Global Unicast***

Abrangência: Universal

Os endereços *Global Unicast* [78] são equivalentes aos normais endereços públicos IPv4 [3]. Têm necessariamente um prefixo associado, e a partir desse prefixo poderá ser estabelecida uma hierarquia (ver Subsecção 3.3.2). Alguns já estão atribuídos aos respectivos RIRs (ver Figura 2.21), e prevê-se que a sua atribuição venha a aumentar cada vez mais [179]. A estrutura deste tipo de endereçamento já foi demonstrada (ver Figura 3.24), quando foi explicado o estabelecimento de uma hierarquia.

- ***Unique Local Unicast***

Abrangência: Departamental / Regional



Este tipo de endereços [88] é o que mais assemelha com o conceito de endereço privado existente no IPv4 [47]. As características são muito parecidas:

- Tem um prefixo previamente conhecido, para a correcta filtragem efectuada nos *routers* de fronteira, nomeadamente entre a passagem de uma rede privada para a Internet;
- Não tem qualquer tipo de significado na Internet, e mesmo se acidentalmente passar os *routers* de fronteira, é muito raro existir conflito de endereços.

Assim, como as vantagens:

- Providencia prefixos locais que são usados de forma completamente independente dos globais; desta forma estes endereços podem ser utilizados por redes internas, que não estejam ligadas à Internet e pretendam a sua ligação mais tarde;
- As aplicações tratam estes endereços como se fossem normais endereços globais;
- Redes privadas podem mudar de ISP e continuar com a sua actual numeração *unique local*.

Da mesma forma se assemelham algumas desvantagens:

- Não é possível usar este tipo de endereços na Internet, pelo que é necessário um esforço adicional aos administradores pela preocupação da sua filtragem, enquanto os próprios dispositivos (*routers*) ainda não tenham essa filtragem previamente definida, o que sucede para os endereços privados do IPv4;
- Como se irá explicar de seguida, existe uma mínima probabilidade de não ser único, mas existe.

O endereço *unique local* é dividido como se verifica na Figura 3.27.

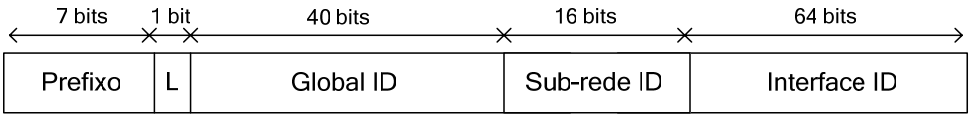


Figura 3.27 – Estrutura do endereço *unique local*.

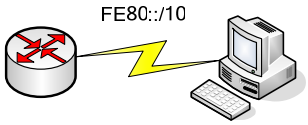
O prefixo já foi conhecido acima, o resto das novidades são:

- L *bit* – só existe definição para o valor 1, que significa que o endereço foi atribuído localmente. Para o valor 0 ainda está em estudo o seu significado.
- Global ID – Parte do endereço que é gerada de forma aleatória, de forma a garantir o máximo de unicidade.
- Existiu um tipo de endereço, o endereço *site-local*, que se assemelhava a este tipo de endereço, principalmente no objectivo. O problema principal desse tipo de endereço era a falta de garantia de unicidade dos endereços, principalmente se existissem redes privadas diferentes interligadas. Nesse caso, cada administrador configurava a sua rede com os endereços o mais simples possível, e com o mesmo prefixo de sub-rede. Este prefixo, por sua vez, era igual à sub-rede do endereço global da rede [77]. Essa situação aumentava em muito a probabilidade de existir conflito de endereços. As diversas desvantagens que existiam com os endereços *site-local* estão pormenorizadas em [82], que torna obsoleto este tipo de endereço.
- A unicidade deste novo tipo de endereço é garantida pelo campo Global ID. Poderá dizer-se que, na prática não existe probabilidade de ser único, mas na teoria sim.

  Este tipo de endereço está numa fase muito experimental, pelo que implementações a fundo com este tipo de endereços não é aconselhado; a melhor forma de abordar por agora este tipo de endereços, é esperar para ver.

- ***Link-Local***

Abrangência: Local



É, provavelmente, um dos primeiros indicadores de que determinado nó suporta IPv6. Não necessita de qualquer *router* (caso dos endereços globais) ou servidor adicional para ser configurado (caso do DHCPv6) (ver Secção 3.8).

Este endereço só tem significado na ligação até ao primeiro *router*, ou seja, no *link* local. O *router* funciona, então, como o delimitador dos endereços *link-local*. Na Figura 3.28 é explicada a forma como a abrangência dos endereços *link-local* funciona, notando-se que os endereços da 1.ª região perdem todo o significado na 2.ª região, assim como os da 2.ª região perdem o significado na 1.ª.

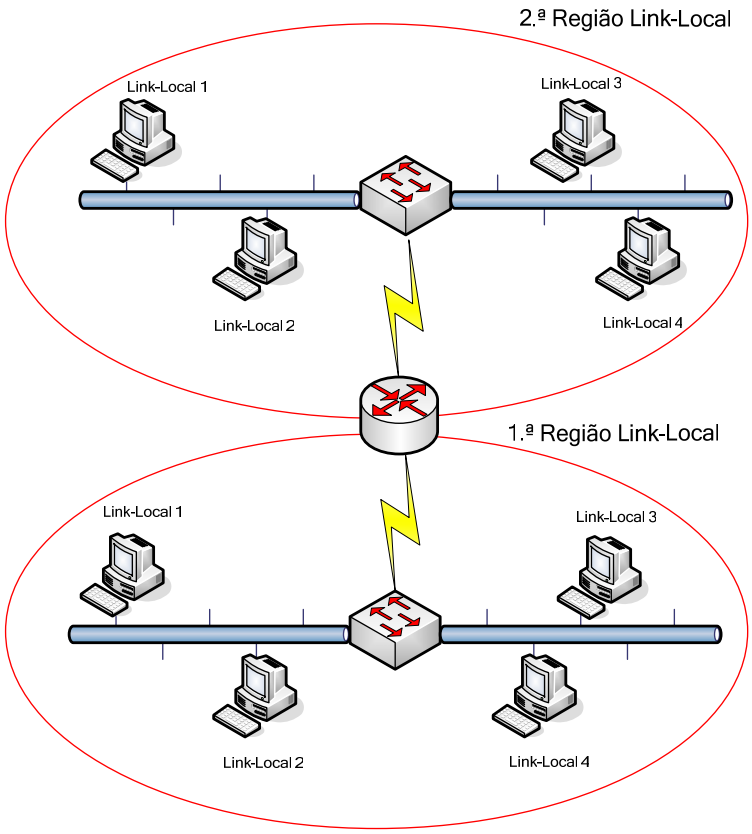


Figura 3.28 – Abrangência geográfica dos endereços *link-local*.

Não existe qualquer novidade a nível de novos campos neste tipo de endereço, mas existe a eliminação de um bastante importante, a Sub-rede ID. Por não existir esta divisão, significa que o prefixo deste endereço é sempre igual (*FE80::/10*, como já se tinha verificado acima). O facto de este prefixo ser estático faz com que uma notação de prefixo neste tipo de endereço perca todo o significado e, por isso mesmo, o prefixo nunca é referenciado neste tipo de endereços. Na Figura 3.29 é mostrada a disposição deste endereço.

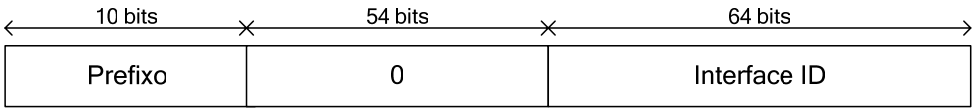


Figura 3.29 – Estrutura do endereço *link-local*.

Este endereço é o endereço mais simples e mais rápido de obter numa ligação à rede, com vantagem notória.

Um conceito bastante ligado a estes endereços é o conceito de *Zone ID*. Este poderia também ser utilizado nos obsoletos endereços *site-local*, e provavelmente será também utilizado nos novos *unique local*, nos mesmos parâmetros. Porém, devido ao carácter experimental deste último tipo de endereços, e ao carácter completamente definido do *link-local*, na prática, a maioria das vezes que se utiliza *Zone ID* é nos *link-local.*

- **O conceito de *Zone ID***

O conceito de *Zone ID* é um conceito que veio resolver um problema que se pode verificar nos endereços *link-local*. A situação referida é explicada na Figura 3.30.

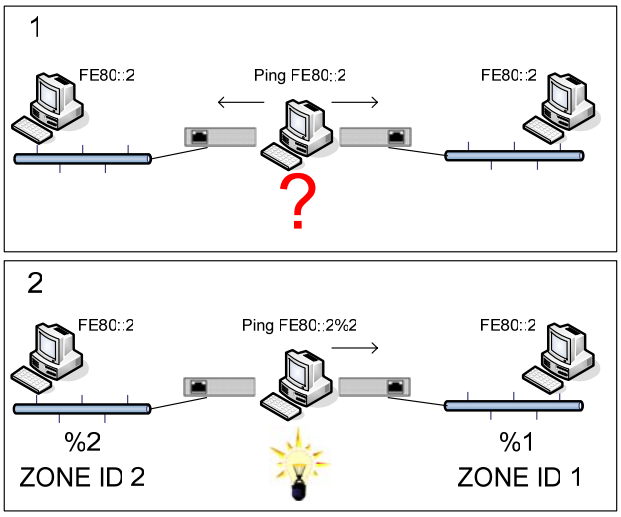


Figura 3.30 – Funcionamento do *Zone ID*.

Devido ao facto de não existir o conceito de sub-redes como nos endereços privados do IPv4, é completamente impossível ao computador "saber" neste caso para que rede mandar o pacote. Então, existe o *Zone ID* que no caso específico do *link-local* é tipicamente igual ao número (índice) da interface. No caso da figura, para aceder ao computador da *Zone ID* 2, utiliza-se o endereço *FE80::2%2*.

No caso dos obsoletos *site-local*, a *Zone ID* era representativa de uma região da topologia da rede. Não existe ainda definição de como funcionará com os *unique local*, mas prevê-se que seja igual. Este tipo de endereços pode também funcionar sem qualquer tipo de referência à *Zone ID*. Nesse caso, por omissão, todos estarão na mesma *Zone ID* (*Zone ID* 1) [85].

- **Endereços IPv6 com endereços IPv4 embutidos**

Existem dois tipos de endereços IPv6 muito utilizados que têm endereços IPv4 embutidos. Estes dois tipos de endereços estão associados aos mecanismos de transição existentes entre IPv4 e IPv6 (ver Secção 3.10). O objectivo de existir distinção entre estes dois tipos endereços é saber informação acerca da sua origem, nomeadamente se existe suporte IPv6 no dispositivo ou não.

IPv4-compatible IPv6 address

Este tipo de endereço é apenas usado por dispositivos que tenham suporte IPv6; sendo assim, o dispositivo destino saberá que quem lhe enviou o pacote também suporta IPv6. A Figura 3.31 mostra o formato deste tipo de endereço, que será todo a zeros, excepto os 32 *bits* finais, que serão o endereço IPv4, por exemplo, *::192.168:59:62*.

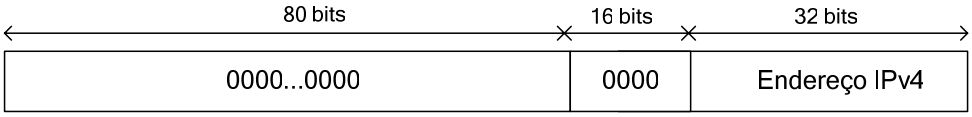


Figura 3.31 – Estrutura do endereço *IPv4-compatible IPv6 address.*

IPv4-mapped IPv6 address

Este endereço sucede quando a origem não tem qualquer tipo de suporte IPv6. Então nesse caso, o endereço é mapeado para IPv6, por um mecanismo intermédio como, por exemplo, o DNS. A Figura 3.32 mostra a estrutura deste tipo de endereço.

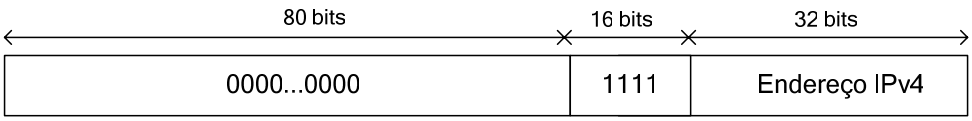


Figura 3.32 – Estrutura do endereço *IPv4-mapped IPv6 address*.

- **Unspecified e Loopback**

O endereço *unspecified 0:0:0:0:0:0:0:0* (*::*) representa a ausência de endereço IPv6 e nunca deve ser atribuído a um nó. É utilizado, por exemplo, no mecanismo de *Duplicate Address Detection* (DAD) (ver Subsecção 3.6.6).

O endereço *0:0:0:0:0:0:0:1* (*::1*) é chamado *loopback* e pode ser usado por um nó para enviar um pacote para ele próprio.

- **Endereço Privativo**

Este tipo de endereços tem uma razão específica da sua existência: a falta de privacidade que poderia ser proporcionada pelo mecanismo de auto-configuração. O mecanismo de auto-configuração (ver Subsecção 3.8.1), baseia-se em informação única do dispositivo (ver Subsecção 3.3.3) para configurar o seu respectivo endereço. Isto permite que se consiga localizar o dispositivo na Internet a qualquer altura e, consequentemente, o seu utilizador. Este facto poderá ser considerado uma vantagem ou uma desvantagem, já que o utilizador poderá querer ser identificado, ou não. É claro que tem de ser garantida a privacidade do utilizador, caso esta seja o seu desejo, e é aqui que entra o conceito de endereço privativo. A forma como funciona este tipo de endereço é bastante simples, pois em vez de os últimos 64 *bits* (Interface ID) serem gerados com informação do dispositivo, são gerados de forma completamente aleatória. Na Figura 3.33 está um esquema de como, apesar de o endereço mudar de sítio (devido ao prefixo dos *routers*), os últimos 64 *bits* permanecem inalteráveis, pelo que é possível a localização do dispositivo em questão. No caso do endereço ser aleatório, não há qualquer probabilidade de localizar esse mesmo dispositivo.

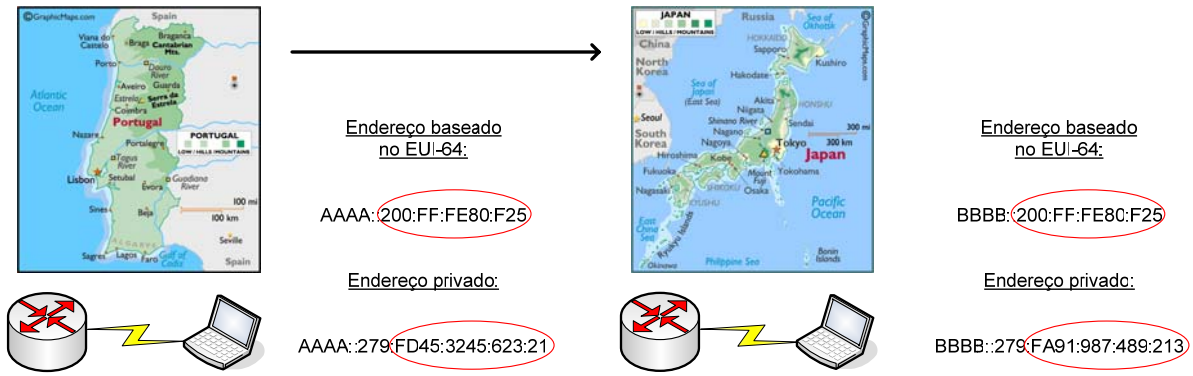


Figura 3.33 – Exemplo representativo de um endereço privativo, e de um não privativo.

A necessidade deste tipo de endereço anónimo torna-se ainda mais necessária devido ao facto do crescimento exponencial dos dispositivos portáteis, em que as pessoas são imediatamente associadas ao dispositivo.

- **Endereço temporário**

Já foi referido atrás que os endereços divergem em abrangência e tipo. Neste caso específico os endereços divergem em tempo de vida. Este tempo de vida é definido na mensagem *Router Advertisement* (ver Subsecção 3.6.3). Existem dois tipos de tempo definidos: o tempo de vida válido (*valid lifetime*) e o tempo de vida preferido (*prefered lifetime*): o primeiro é o tempo de vida propriamente dito do endereço; o segundo é o tempo em que esse endereço é preferido em relação a outros, também definidos nessa interface. Estes tempos só existem a partir do momento em que o

endereço é aceite, e só terminam quando o endereço ficar obsoleto. Na Figura 3.34 mostram-se os diversos estados por que podem passar os endereços.

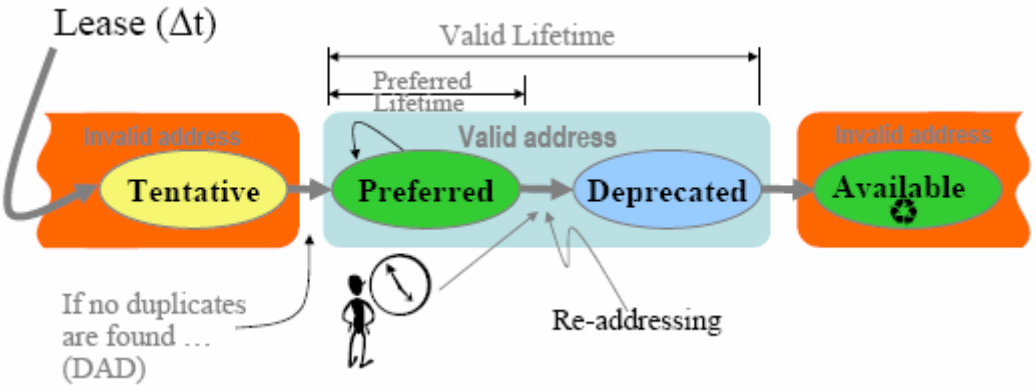
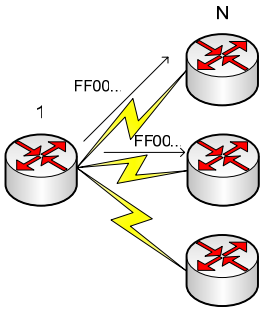


Figura 3.34 – Os diversos estados de um endereço ao longo do tempo (Extraído de [40]).

Sendo assim, é utilizado um mecanismo para verificar se o endereço é válido ou não, o DAD (*Duplicate Address Detection*, ver Subsecção 3.6.6). De seguida se o endereço for aceite, irá usufruir dos seus tempos de vida válido e tempo de vida preferido, até ficar obsoleto.

3.3.4.2. Multicast

Tipo: 1 para N



O conceito essencial deste tipo de endereço é basicamente igual ao do IPv4, mas o uso no IPv6 vai bastante mais além do que o exemplo da vídeo-conferência no IPv4. Ele é utilizado em diversos mecanismos que formam toda a base do IPv6. Porém, mesmo o próprio endereço foi adaptado à forma com que o IPv6 lida com todo o endereçamento, tendo sido dividido (tal como o *unicast*) por abrangência:

- *Node-Local*;
- *Link-Local*;
- *Site-Local* (igual à abrangência do *Unique Local*).

As novidades são o aparecimento do *Node-Local*, que só tem significado no próprio nó, e o desaparecimento do *Global*.

Para além dos prefixos especiais associados aos endereços *Node-Local*, *Link-Local* e *Site-Local*, existem endereços predefinidos, que têm significado em qualquer tipo de abrangência. Esse tipo de endereços pode ser requisitado por empresas, para algumas implementações de protocolos proprietários.

É de referir também que alguns endereços são automaticamente configurados, conforme o cenário existente. A Tabela 3.3 referencia-os:

Endereço	Abrangência	Definição
FF01::1	*Node-Local*	Endereço *multicast* de todos os nós.
FF02::1	*Link-Local*	Endereço *multicast* de todos os nós.
FF01::2	*Node-Local*	Endereço *multicast* de todos os *routers*.
FF02::2	*Link-Local*	Endereço *multicast* de todos os *routers*.
FF05::2	*Site-Local*	Endereço *multicast* de todos os *routers*.

Tabela 3.3 – Alguns endereços *multicast* que são configurados automaticamente [180].

Outro endereço *multicast* de grande importância é o *Solicited-Node*, que funciona como um pseudo endereço *unicast*. Este endereço é da forma *FF02::1:FF:FFFF* e deriva do endereço IPv6 *unicast* do nó. O conjunto *FF:FFFF* corresponde aos últimos 24 *bits* do endereço do nó, o que faz com que seja muito improvável existirem duas máquinas pertencentes ao grupo *multicast* do mesmo endereço. É este facto que torna este endereço muito útil na substituição do *broadcast* do ARP na resolução de endereços do IPv6 (ver Subsecção 3.6.2).

Existem três novidades neste tipo de endereço: as *flags*, o *scope* e o Grupo ID. No que diz respeito às flags, apenas o primeiro *bit* interessa: se for 0 então estamos perante um endereço atribuído de forma permanente (um endereçamento previamente conhecido); se for 1, indica um endereço que não é permanente (que não é previamente conhecido, pode ser por exemplo um endereço *multicast* definido manualmente por um programa de gestão de grupos *multicast*). O *scope* representa a abrangência do endereço. Estão definidas algumas abrangências para além das especificadas acima [77]; porém, na prática, não existe qualquer tipo de implementação das mesmas. O Grupo ID é o identificador do grupo *multicast*. A estrutura dos endereços *multicast* pode ser vista na Figura 3.35.

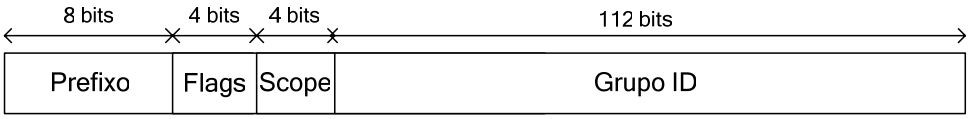


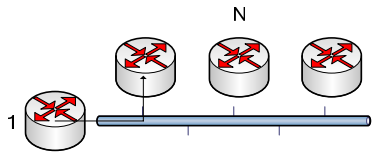
Figura 3.35 – Representação dos endereços *multicast*.

Não é necessário qualquer configuração adicional no que diz respeito à ligação local (*link-local*); porém quando se atravessa um *router*, é necessária configuração adicional para cooperação entre os diversos *routers*.

Este tipo de endereços nunca poderá ser usado como endereço de origem de qualquer nó, nem como endereço intermédio.

3.3.4.3. Anycast

Tipo: 1 para N



Este tipo de endereço é uma mistura de *multicast*, com *unicast* [3]. O mesmo endereço é atribuído a múltiplas interfaces, mas só uma delas é que receberá o pacote enviado para esse determinado endereço. Geralmente, se não existir qualquer problema nos dispositivos associados às interfaces, a escolhida será a interface mais próxima em relação ao dispositivo que continha o endereço de origem. Para o nó origem do pacote enviado para um endereço *anycast* saber qual a interface mais próxima, recorre ao protocolo de encaminhamento existente, trabalhando assim num regime de cooperação com ele. As aplicações práticas deste tipo de endereço são, assim, óbvias: maior rapidez em qualquer tipo

de prestação de serviços. Na Figura 3.35 é mostrado um exemplo de como pode funcionar este tipo de endereços.

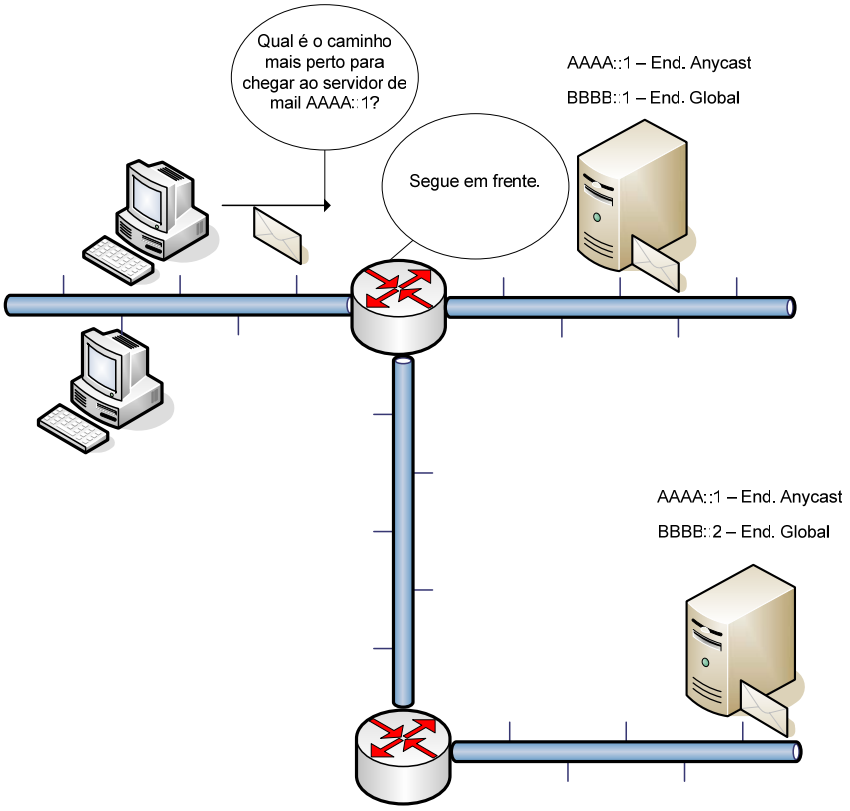


Figura 3.36 – Exemplo de um cenário usando endereços *anycast*, com um servidor de *mail*.

É de referir que a descoberta do caminho mais rápido é sempre com base nas tabelas de encaminhamento e respectivas métricas associadas, ou seja, sempre a nível da camada 3.

Os endereços *anycast* são impossíveis de distinguir de um normal endereço *unicast*. Devido a esse facto, também pode usufruir de hierarquias, tendo um formato específico para a criação de sub-redes, que consiste em ter os últimos 16 *bits* do endereço a 0.

É o tipo de endereço que ainda está mais em fase de estudo, e por isso é fulcral uma verificação do seu estado [89].

Tal como o *multicast,* este tipo de endereço também não pode ser usado como endereço de origem.

3.3.5. Políticas de Atribuição

As políticas de atribuição [15] não divergem muito das aplicadas no IPv4. Em termos da estrutura hierárquica, ela é exactamente a mesma: a IANA atribui os endereços aos RIRs, que por sua vez atribuem aos ISPs e, a partir dos ISPs, finalmente o utilizador final tem o seu endereço. Esta é a hierarquia mais simples, mas poderão existir LIRs e NIRs intermédios, entre ISPs e RIRs. Sendo assim, a hierarquia existente será toda aproveitada, podendo por exemplo os ISPs requisitarem prefixos IPv6, tendo por base o número já existente de utilizadores IPv4 da sua rede.

As atribuições dividem-se em dois passos de maior importância: atribuição inicial e atribuição subsequente.

Atribuição inicial

Para determinada instituição se qualificar a ter uma atribuição inicial de um prefixo IPv6, ela tem de assumir certos compromissos:

- Ser um LIR;
- Não ser um utilizador final;
- Após uma hierarquia previamente definida com base no prefixo fornecido, preveja fornecer conectividade a instituições, ligando-as através das sub-redes definidas;
- Num plano de dois anos preveja no mínimo atribuir 200 prefixos */48* a organizações.

Este último ponto parece sem dúvida ser o mais restritivo, porém terá de ser visto como uma forma de aumentar a popularidade do IPv6.

As organizações que assumam todos estes compromissos estão automaticamente elegíveis para receber no mínimo um prefixo */32*, mas com justificação bastante válida pode até requisitar um prefixo com maior capacidade de endereços.

Atribuição subsequente

Este tipo de atribuição diz respeito directamente à atribuição feita para os ISPs. Neste caso os requisitos para a atribuição, podem ser definidos por exemplo pelo próprio LIR. Os prefixos que se devem atribuir estão especificados [72]:

- */48* no caso mais comum, excepto para quem tenha largos assinantes;
- */64* quando se sabe que apenas uma sub-rede é necessária (não é necessário estabelecer hierarquias);
- */128* quando se sabe que apenas um dispositivo será ligado à rede.

De momento, ainda não existiu quem requisitasse mais do que um prefixo */48*, o que é perfeitamente justificável, dada a quantidade de endereços que um prefixo desses disponibiliza, mesmo contando com o sub-endereçamento.

3.4. Internet Control Message Protocol for IPv6 (ICMPv6)

O ICMPv6 [56] é o protocolo usado pelo IPv6 para executar diversas tarefas de camada 3, entre elas relatório de erros e funções de diagnóstico. Este protocolo serve de base aos mecanismos de *Path MTU Discovery*, ND (*Neighbor Discovery*) e MLD (*Multicast Listener Discovery*), indispensáveis ao funcionamento do IPv6, na medida em que as suas mensagens são usadas nesses mecanismos.

O ICMPv6 combina muitas das funcionalidades que, na versão 4 do protocolo IP, estavam divididas por vários protocolos, como o ICMP (*Internet Control Message Protocol*) [42], o IGMP (*Internet Group Management Protocol*) [76] e o ARP (*Address Resolution Protocol*) [43].

Como parte integrante do IPv6 que é, o ICMPv6 tem que ser totalmente implementado por todos os nós IPv6.

3.4.1. Mensagens ICMPv6

As mensagens ICMPv6 estão divididas em duas classes, as de erro e as informativas. Em relação ao valor do tipo de mensagem, as primeiras são caracterizadas por possuir o *bit* de maior ordem igual a 0, podendo assumir valores de 0 a 127, e as segundas por o *bit* de maior ordem ser igual a 1, assumindo valores de 128 a 255. As mensagens de erro englobam as *Destination Unreachable*, *Packet Too Big*, *Time Exceeded* e *Parameter Problem*. Nas mensagens informativas incluem-se as *Echo Request* e *Echo Reply*, assim como as mensagens de protocolos complementares como o ND e o MLD. Para uma

lista actualizada dos tipos de mensagens ICMPv6, consultar [181]. As mensagens ICMPv6 básicas são:

- ***Destination Unreachable*:** mensagem de erro que informa o nó emissor que um pacote não pode ser entregue.
- ***Packet Too Big*:** mensagem de erro que informa o nó emissor que um pacote não pode ser reencaminhado devido ao facto de o seu tamanho ser maior que o MTU (*Maximun Transmission Unit*) do *link* de saída (ver Secção 3.5).
- ***Time Exceeded*:** mensagem de erro que informa o nó emissor que o limite de saltos do pacote IPv6 chegou ao fim.
- ***Parameter Problem*:** mensagem de erro que informa o nó emissor de um pacote IPv6 que foram detectados erros no cabeçalho ou cabeçalhos de extensão do pacote enquanto este era processado.
- ***Echo Request*:** mensagem informativa usada para verificar a disponibilidade e atingibilidade de um nó, e que obriga a um *Echo Reply* imediato por parte do destino.
- ***Echo Reply*:** mensagem informativa usada em resposta a um *Echo Request*.

O formato geral das mensagens ICMPv6 é apresentado na Figura 3.37.

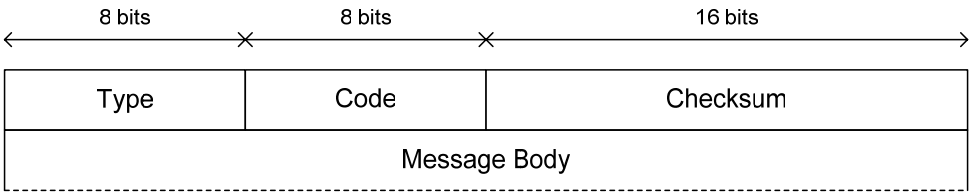


Figura 3.37 – Formato geral das mensagens ICMPv6.

Para demonstrar o formato geral das mensagens ICMPv6 utilizou-se o cenário apresentado na Figura 3.38. A figura representa uma rede IPv6 constituída por duas máquinas com conectividade através do seu endereço *link-local*.

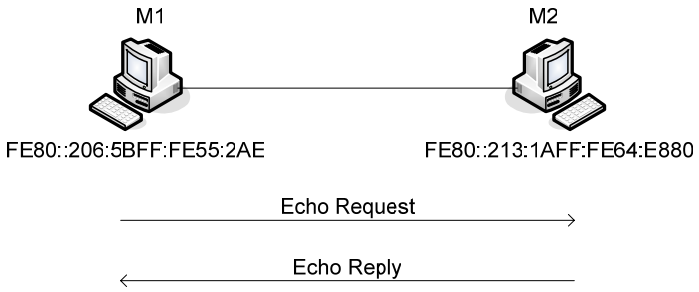


Figura 3.38 – Cenário de teste utilizado para verificar o formato das mensagens ICMPv6.

As imagens seguintes demonstram o formato de uma mensagem ICMPv6 *Echo Request* e respectivo *Echo Reply* no analisador de protocolos de rede Ethereal. Estas mensagens são o resultado de um *ping* entre as duas máquinas IPv6.

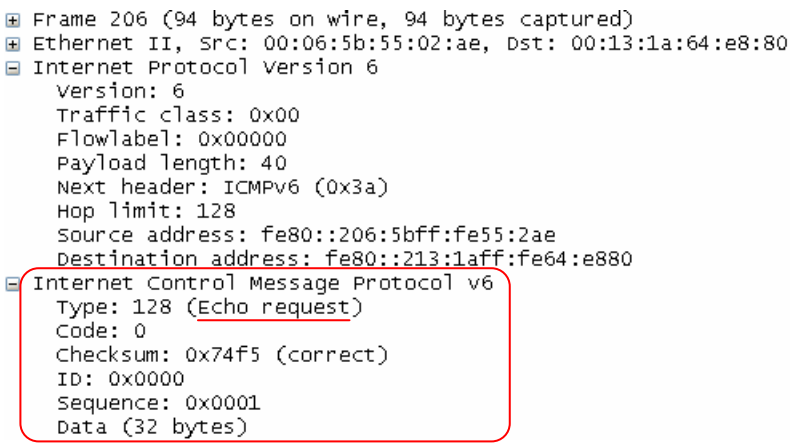


Figura 3.39 – Captura de uma mensagem ICMPv6 *Echo Request*.

Como se pode verificar na Figura 3.39, a mensagem ICMPv6 é encapsulada num pacote IPv6, com o campo *Next Header* com o valor 0x3A (58 em decimal) correspondente ao protocolo ICMPv6. Note-se também a constituição da mensagem ICMPv6 com os campos *Type*, *Code* e *Checksum*, um campo de identificação e um campo de sequência próprios do *Echo Reques*t, e um campo de dados.

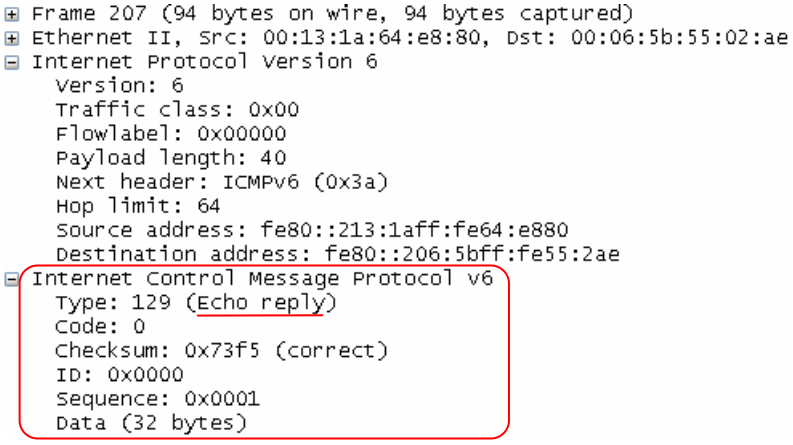


Figura 3.40 – Captura da mensagem ICMPv6 *Echo Reply*, resposta ao anterior *Echo Request*.

Na Figura 3.40 pode-se analisar a resposta à mensagem anterior, correspondendo ao pacote com o número de sequência 1.

3.5. Path MTU Discovery

Os *routers* IPv6 não efectuam fragmentação de pacotes em trânsito, sendo esta feita, quando necessária, pelo nó origem. Um nó IPv6 tem de efectuar fragmentação, ou seja, dividir os dados a enviar por vários fragmentos IPv6, quando necessita de enviar uma quantidade de dados, cujo tamanho é maior que o mínimo MTU de todos os *links* do caminho. Ao tamanho máximo que um pacote pode ter ao longo de todo o caminho, desde o nó origem até ao nó destino, chama-se *path* MTU (MTU do caminho), e o mecanismo através do qual um nó descobre dinamicamente esse valor é o *Path MTU Discovery* [49]. Este processo já existia no IPv4.

A descoberta do *path* MTU por parte do nó origem, processa-se da seguinte forma:

1. O nó origem assume que o *path* MTU para o nó destino é o MTU do *link* a que está ligada a interface pela qual vai enviar os dados;
2. O nó origem envia os pacotes com o tamanho do *path* MTU assumido;

3. Se um *router* do caminho é incapaz de reencaminhar um pacote, por o seu tamanho ser superior ao *link* MTU da interface pela qual ia reencaminhar o pacote, envia uma mensagem ICMPv6 *Packet Too Big* para o nó origem, com o *link* MTU da interface na qual o reencaminhamento falhou;
4. O nó origem assume como novo *path* MTU para o destino, o valor do campo MTU do pacote ICMPv6 *Packet Too Big*.

O nó origem recomeça no passo 2 e repete os passos 2, 3 e 4, tantas vezes quanto as necessárias para descobrir o *path* MTU até à máquina destino. O *path* MTU está determinado, quando não forem recebidas mais mensagens ICMPv6 *Packet Too Big*, ou quando for recebida uma confirmação, ou uma resposta por parte do nó destino.

Para visualizar o funcionamento deste mecanismo, utilizou-se o cenário apresentado na Figura 3.41, e a aplicação Ethereal para efectuar a captura de pacotes. A rede apresentada é composta por duas máquinas Windows XP e dois *routers* Cisco IOS, com o protocolo de encaminhamento RIPng. Por uma questão de simplicidade de implementação, na exemplificação dos mecanismos complementares ao IPv6, utilizam-se máquinas Windows XP e *routers* Cisco IOS, mas tal poderia ser feito com equipamentos de outros fabricantes.

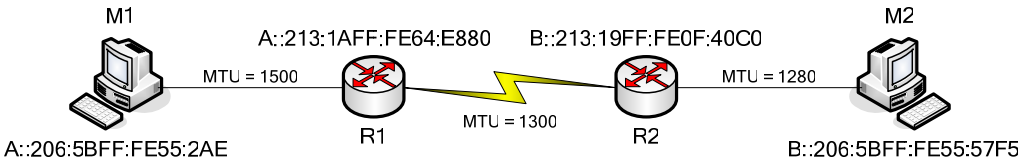


Figura 3.41 - Cenário utilizado para testar o *Path MTU Discovery*.

Enviou-se, então, um pacote com 1400 *bytes*, utilizando o utilitário *ping*, da máquina M1 para a máquina M2. A mensagem ICMPv6 E*cho Request* gerada apresenta-se na Figura 3.42.

```
⊞ Frame 4356 (1462 bytes on wire, 1462 bytes captured)
⊞ Ethernet II, Src: 00:06:5b:55:02:ae, Dst: 00:13:1a:64:e8:80
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 1408
    Next header: ICMPv6 (0x3a)
    Hop limit: 64
    Source address: a::206:5bff:fe55:2ae
    Destination address: b::206:5bff:fe55:57f5
⊟ Internet Control Message Protocol v6
    Type: 128 (Echo request)
    Code: 0
    Checksum: 0x0237 (correct)
    ID: 0x0000
    Sequence: 0x0031
    Data (1400 bytes)
```

Figura 3.42 – Captura do *ping* da máquina M1 para a máquina M2.

Quando o *router* R1 tenta reencaminhar o pacote para o segundo *link*, verifica que o MTU desse *link* é menor que o tamanho do pacote, e envia para o nó origem do pacote, a máquina M1, uma mensagem ICMPv6 *Packet Too Big* a informar o MTU do *link* seguinte, conforme se pode verificar na Figura 3.43.

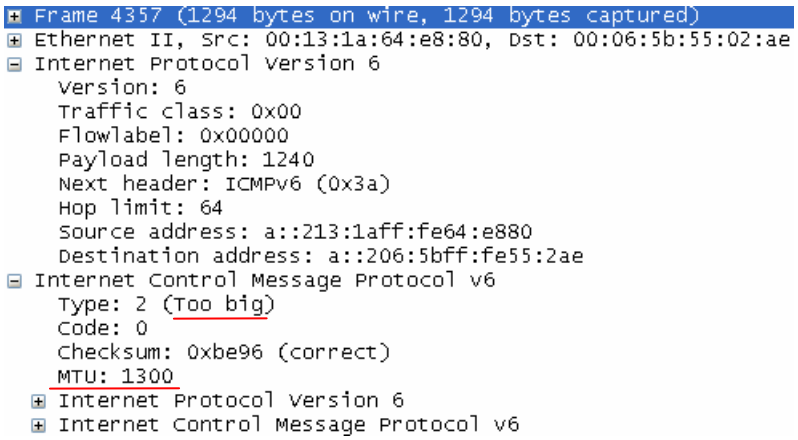


Figura 3.43 – Captura da mensagem *Packet Too Big* do *router* R1 para a máquina M1.

A máquina M1, ao receber a mensagem ICMPv6 *Packet Too Big*, fragmenta então o pacote inicial, tendo em conta o MTU de 1300 indicado na mensagem, e reenvia o pacote fragmentado. Na Figura 3.44 pode constatar-se a existência de dois fragmentos IPv6, identificados pela existência de um *Fragment Header* em cada um deles.

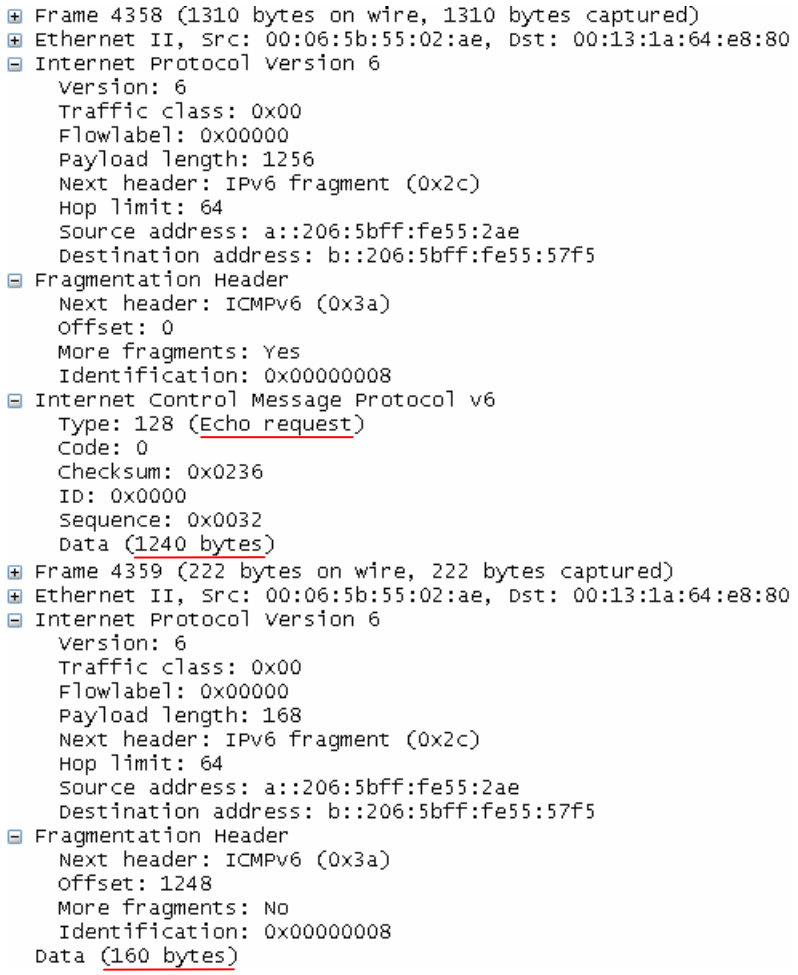


Figura 3.44 – Captura do pacote fragmentado em dois da máquina M1 para a máquina M2.

O pacote fragmentado passa no *router* R1 e no segundo *link* sem problemas, mas quando o *router* R2 o tenta reencaminhar para o terceiro *link*, envia uma mensagem *Packet Too Big*, apresentada na Figura 3.45, para a máquina M1, devido ao facto de o MTU do *link* (1280 *bytes*) ser inferior ao tamanho de um fragmento do pacote (1300 *bytes*).

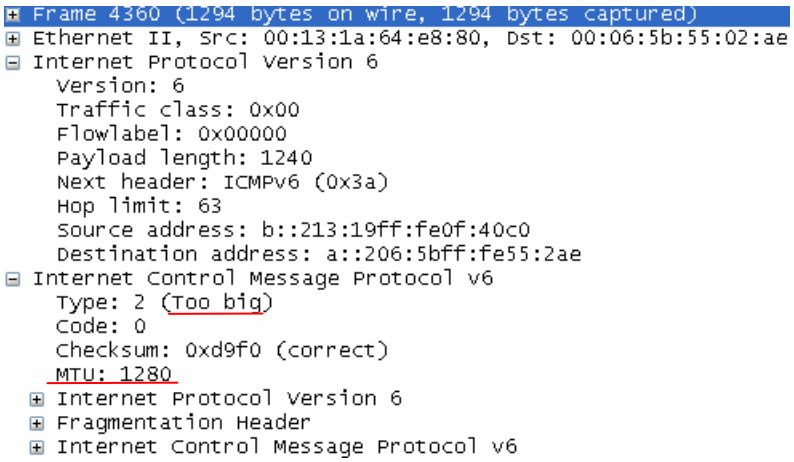


Figura 3.45 – Captura da mensagem *Packet Too Big* do *router* R2 para a máquina M1.

A máquina M1 volta a fragmentar o pacote inicial, agora tendo em conta o MTU de 1280 do terceiro *link*, e reenvia o pacote para a máquina M2. Tal pode verificar-se na Figura 3.46.

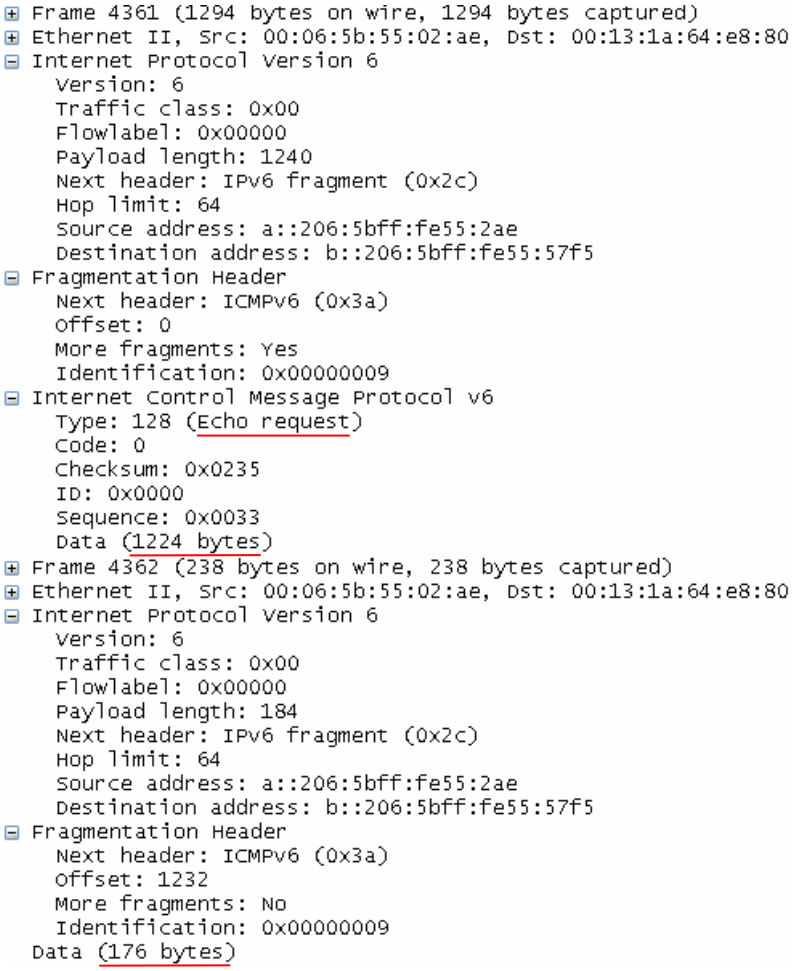


Figura 3.46 – Captura do pacote re-fragmentado em dois da máquina M1 para a máquina M2.

A máquina M2 recebe o pacote da máquina M1, e envia-lhe a respectiva resposta *Echo Reply* também fragmentada em dois pacotes. Esta mensagem é visível na Figura 3.47.

```
⊞ Frame 4363 (1294 bytes on wire, 1294 bytes captured)
⊞ Ethernet II, Src: 00:13:1a:64:e8:80, Dst: 00:06:5b:55:02:ae
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 1240
    Next header: IPv6 fragment (0x2c)
    Hop limit: 62
    Source address: b::206:5bff:fe55:57f5
    Destination address: a::206:5bff:fe55:2ae
⊟ Fragmentation Header
    Next header: ICMPv6 (0x3a)
    Offset: 0
    More fragments: Yes
    Identification: 0x00000007
⊟ Internet Control Message Protocol v6
    Type: 129 (Echo reply)
    Code: 0
    Checksum: 0x0135
    ID: 0x0000
    Sequence: 0x0033
    Data (1224 bytes)
⊞ Frame 4364 (238 bytes on wire, 238 bytes captured)
⊞ Ethernet II, Src: 00:13:1a:64:e8:80, Dst: 00:06:5b:55:02:ae
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 184
    Next header: IPv6 fragment (0x2c)
    Hop limit: 62
    Source address: b::206:5bff:fe55:57f5
    Destination address: a::206:5bff:fe55:2ae
⊟ Fragmentation Header
    Next header: ICMPv6 (0x3a)
    Offset: 1232
    More fragments: No
    Identification: 0x00000007
  Data (176 bytes)
```

Figura 3.47 – Captura do *Echo Reply* da máquina M2 para a máquina M1.

A partir deste momento, a máquina M1 sabe que o *path* MTU, de si até M2, é de 1280 *bytes* e assim sendo, os próximos pacotes serão enviados tendo em conta esse *path* MTU, ou seja, serão logo fragmentados, se necessário, com um tamanho máximo de 1280 *bytes*.

A Figura 3.48 sintetiza o processo verificado.

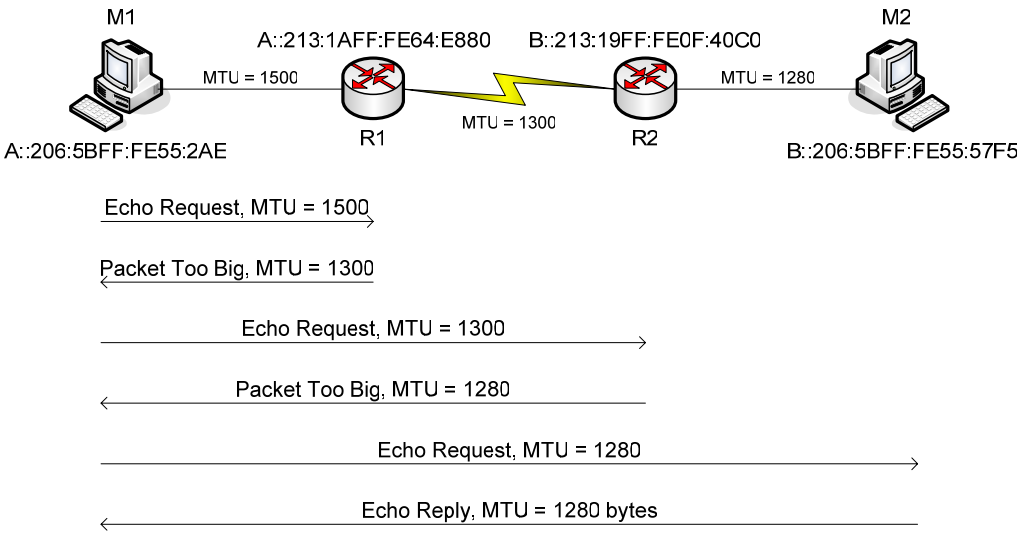


Figura 3.48 – Síntese do processo de descoberta do *path* MTU.

3.6. Neighbor Discovery (ND)

O IPv6 *Neighbor Discovery* (ND) [54] é um protocolo que especifica um conjunto de mensagens e processos, que determinam as relações entre nós vizinhos de uma rede IPv6. O ND substitui o ARP, o ICMP *Router Discovery* [44] e o ICMP *Redirect* [42], utilizados no IPv4, disponibilizando ainda funcionalidades adicionais.

O ND é usado por todos os nós da rede (*routers* e terminais) para:

- Resolver endereços físicos de nós vizinhos;
- Determinar quando o endereço físico de um nó vizinho muda;
- Determinar se um nó vizinho ainda está atingível.

O ND é usado apenas pelos nós terminais para:

- Descobrir *routers* vizinhos;
- Auto-configurar endereços, prefixos de endereços, rotas de encaminhamento, e outros parâmetros de configuração.

O ND é usado apenas pelos *routers* para:

- Anunciarem a sua presença, parâmetros de configuração de terminais, etc.;
- Informarem os terminais do melhor *router* para onde encaminhar o tráfego.

Os principais processos do ND são:

- Resolução de Endereços;
- *Router Discovery* (RD);
- *Redirect*;
- *Neighbor Unreachability Detection* (NUD);
- *Duplicate Address Detection* (DAD).

3.6.1. Mensagens ND

Todas as funções ND são efectuadas utilizando as seguintes mensagens:

- ***Neighbor Solicitation*:** mensagem enviada por um nó IPv6, para descobrir o endereço físico de um nó vizinho.
- ***Neighbor Advertisement*:** mensagem enviada em resposta a uma mensagem *Neighbor Solicitation*. Também pode ser enviada de forma não solicitada, para anunciar uma mudança de endereço físico.
- ***Router Solicitation*:** mensagem enviada por terminais IPv6, para descobrir a presença de *routers* vizinhos no *link*.
- ***Router Advertisement*:** mensagem enviada pelos *routers* IPv6, para anunciarem a sua presença, assim como configurações para os terminais do *link*. Estas mensagens são enviadas periodicamente, ou em resposta a mensagens *Router Solicitation*.
- ***Redirect*:** mensagem enviada por um *router* IPv6, para informar um nó emissor de uma melhor rota para um determinado destino.

Como as mensagens utilizadas nos mecanismos do ND são mensagens ICMPv6, o formato geral das primeiras é o mesmo das últimas.

3.6.2. Resolução de Endereços

É o processo pelo qual um nó da rede resolve o endereço IPv6 de um vizinho no seu endereço físico. Este processo é o equivalente ao ARP no IPv4, e consiste na troca de mensagens *Neighbor Solicitation* e *Neighbor Advertisement*, por parte dos nós vizinhos.

Se, ao tentar enviar um pacote, o nó origem não tiver informação sobre o endereço físico do nó destino nas suas tabelas locais, tem de resolver o seu endereço IP no respectivo endereço físico. Verifica-se então o seguinte processo:

- O nó origem envia então uma mensagem *Neighbor Solicitation* para o endereço *multicast solicited-node* (*FF02::1:FFXX:XXXX*), derivado do endereço IP destino, com o seu endereço físico.
- Quando o nó destino recebe a mensagem *Neighbor Solicitation*, actualiza a sua própria tabela de vizinhos com o endereço físico do nó origem, e envia uma mensagem *Neighbor Advertisement* para o emissor da mensagem *Neighbor Solicitation* com o seu endereço físico.
- Depois de receber a *Neighbor Advertisement*, o nó origem actualiza a sua tabela de vizinhos e o processo de resolução de endereços está terminado.

A partir daqui pode ser enviado tráfego entre o nó origem e o nó destino da mensagem *Neighbor Solicitation*.

Para se testar a resolução de endereços utilizou-se o cenário da Figura 3.49.

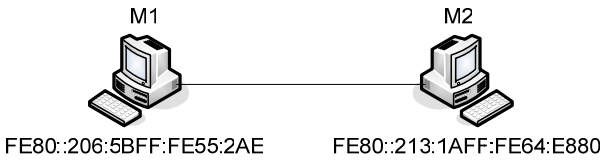


Figura 3.49 – Cenário utilizado para testar a resolução de endereços.

Com as tabelas de vizinhos e de destinos (*neighbor cache* e *destination cache*) limpas, tenta-se enviar um *Echo Request* da máquina M1 para a máquina M2.

Como a máquina M1 não dispõe do endereço físico da máquina M2, precisa de resolver o seu endereço IP no endereço físico, e envia para o *link* uma mensagem ICMPv6 *Neighbor Solicitation* contendo o seu endereço físico. A mensagem *Neighbor Solicitation* é enviada para o endereço *multicast solicited-node* derivado do endereço IP da máquina M2. A estrutura desta mensagem pode ser analisada na Figura 3.50.

```
⊞ Frame 204 (86 bytes on wire, 86 bytes captured)
⊞ Ethernet II, Src: 00:06:5b:55:02:ae, Dst: 33:33:ff:64:e8:80
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: ICMPv6 (0x3a)
    Hop limit: 255
    Source address: fe80::206:5bff:fe55:2ae
    Destination address: ff02::1:ff64:e880
⊟ Internet Control Message Protocol v6
    Type: 135 (Neighbor solicitation)
    Code: 0
    Checksum: 0xd2ad (correct)
    Target: fe80::213:1aff:fe64:e880
  ⊟ ICMPv6 options
      Type: 1 (Source link-layer address)
      Length: 8 bytes (1)
      Link-layer address: 00:06:5b:55:02:ae
```

Figura 3.50 – Captura da mensagem *Neighbor Solicitation* da máquina M1.

Ao receber a mensagem *Neighbor Solicitation* da máquina M1, a máquina M2 envia uma *Neighbor Advertisement*, visível na Figura 3.51, com o seu endereço físico para a máquina M1.

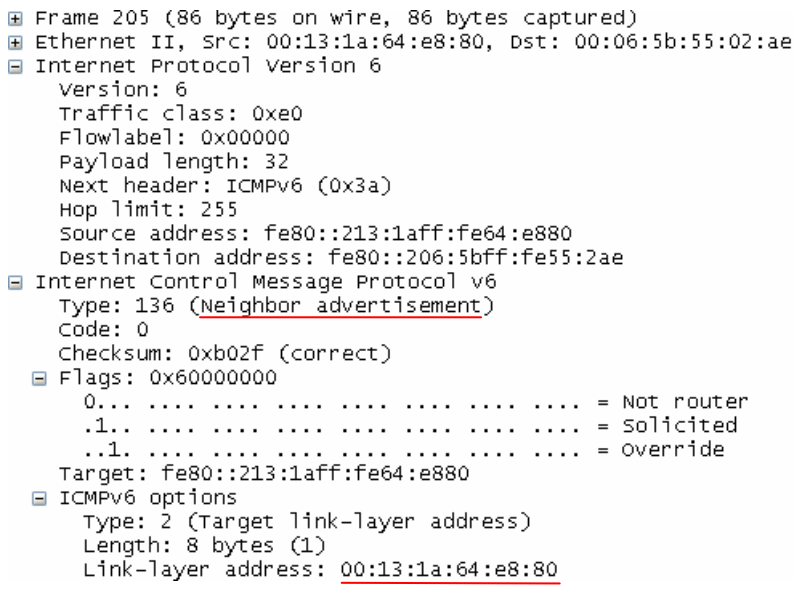


Figura 3.51 – Captura da mensagem *Neighbor Advertisement* da máquina M2.

A partir do momento em que ambas as máquinas conhecem os endereços físicos de cada uma, a comunicação efectua-se sem problemas. A máquina M1 envia o *Echo Request*, e a máquina M2 o respectivo *Echo Reply*. Estas duas mensagens podem-se verificar, respectivamente, na Figura 3.52 e Figura 3.53.

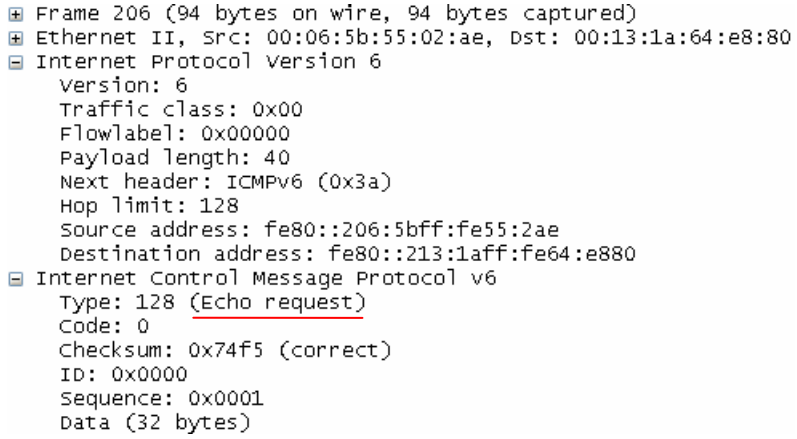


Figura 3.52 – Captura do *Echo Request* da máquina M1 para a máquina M2.

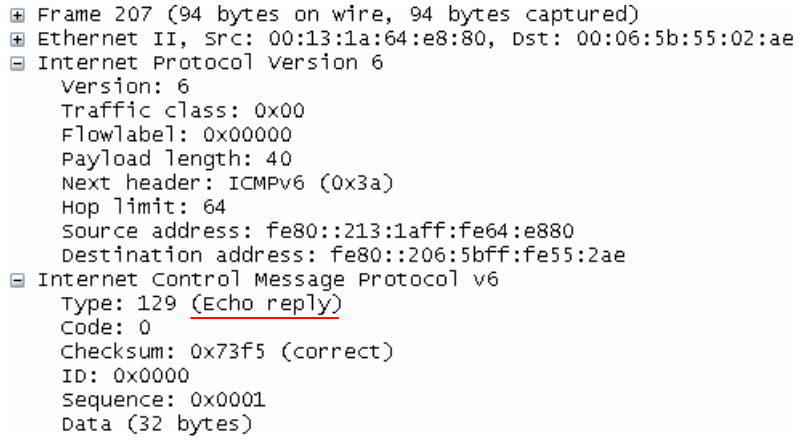


Figura 3.53 – Captura do *Echo Reply* da máquina M2 para a máquina M1.

A Figura 3.54 sintetiza o que se efectuou.

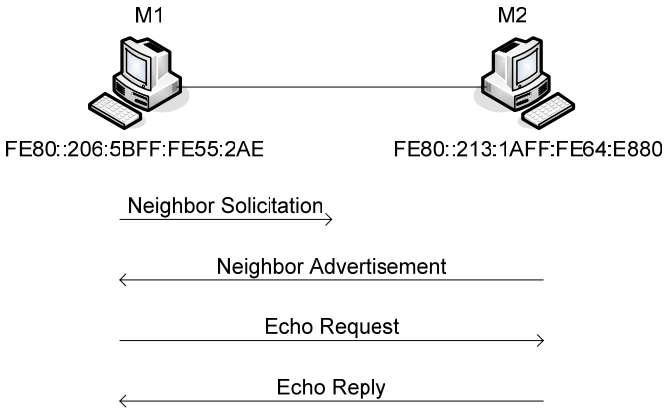


Figura 3.54 – Síntese do processo de resolução de endereços.

3.6.3. Router Discovery (RD)

O *Router Discovery* (RD) é o processo através do qual um terminal descobre os *routers* locais no mesmo *link*, ou seja, os *routers* vizinhos. Este processo é equivalente ao ICMP *Router Discovery* do IPv4, e usa as mensagens ICMPv6 *Router Solicitation* e *Router Advertisement* para executar as suas funções.

Para além de servir para configurar o *default router* de uma máquina, o RD serve para configurar os seguintes parâmetros: o *hop limit* por omissão; o método de auto-configuração que deve ser usado pela máquina, *stateful ou stateless* (ver Secção 3.8); a lista de prefixos definida para o *link* e os seus tempos de vida; o MTU do *link*; rotas específicas a ser adicionadas à tabela de encaminhamento; temporizadores para mecanismos diversos; etc.

Os *routers* IPv6 enviam periodicamente mensagens *Router Advertisement* para o *link* local, de forma a anunciar a sua presença, e fornecer parâmetros de configuração aos terminais da rede. Em vez de esperarem por aquelas mensagens, os terminais IPv6 podem enviar mensagens *Router Solicitation* para o *link* de forma a solicitar as respectivas *Router Advertisement* por parte dos *routers*.

Em relação aos endereços para os quais as mensagens são enviadas, as *Router Advertisement* são enviadas para o endereço *multicast all nodes* (*FF02::1*) (endereço *multicast* a cujo grupo pertencem todos os nós IPv6), e as *Router Solicitation* para o endereço *multicast all routers* (*FF02::2*) (endereço *multicast* a cujo grupo pertencem todos os *routers* IPv6).

Para ilustrar o funcionamento do RD utilizou-se o cenário constante na Figura 3.55.

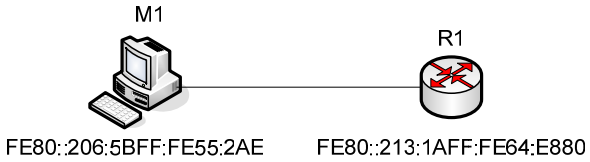


Figura 3.55 – Cenário utilizado para ilustrar o funcionamento do RD.

Ao activar-se a ligação da máquina M1 ao *router* R1, a máquina envia uma mensagem *Router Solicitation* para o endereço *multicast all routers*, por forma a que caso exista algum *router* no *link*, este lhe anuncie a sua presença e lhe forneça configurações. Na Figura 3.56 apresenta-se esta mensagem.

```
⊞ Frame 558 (70 bytes on wire, 70 bytes captured)
⊞ Ethernet II, Src: 00:06:5b:55:02:ae, Dst: 33:33:00:00:00:02
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 16
    Next header: ICMPv6 (0x3a)
    Hop limit: 255
    Source address: fe80::206:5bff:fe55:2ae
    Destination address: ff02::2
⊟ Internet Control Message Protocol v6
    Type: 133 (Router solicitation)
    Code: 0
    Checksum: 0xbf1b (correct)
  ⊟ ICMPv6 options
      Type: 1 (Source link-layer address)
      Length: 8 bytes (1)
      Link-layer address: 00:06:5b:55:02:ae
```

Figura 3.56 – Captura da mensagem *Router Solicitation* enviada pela máquina M1.

Ao receber a mensagem *Router Solicitation*, o *router* R1 envia para o endereço *multicast all nodes* uma *Router Advertisement*.

```
⊞ Frame 562 (118 bytes on wire, 118 bytes captured)
⊞ Ethernet II, Src: 00:13:1a:64:e8:80, Dst: 33:33:00:00:00:01
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0xe0
    Flowlabel: 0x00000
    Payload length: 64
    Next header: ICMPv6 (0x3a)
    Hop limit: 255
    Source address: fe80::213:1aff:fe64:e880
    Destination address: ff02::1
⊟ Internet Control Message Protocol v6
    Type: 134 (Router advertisement)
    Code: 0
    Checksum: 0x16ab (correct)
    Cur hop limit: 64
  ⊟ Flags: 0x00
      0... .... = Not managed
      .0.. .... = Not other
      ..0. .... = Not Home Agent
      ...0 0... = Router preference: Medium
    Router lifetime: 1800
    Reachable time: 0
    Retrans time: 0
  ⊟ ICMPv6 options
      Type: 1 (Source link-layer address)
      Length: 8 bytes (1)
      Link-layer address: 00:13:1a:64:e8:80
  ⊟ ICMPv6 options
      Type: 5 (MTU)
      Length: 8 bytes (1)
      MTU: 1500
  ⊟ ICMPv6 options
      Type: 3 (Prefix information)
      Length: 32 bytes (4)
      Prefix length: 64
    ⊟ Flags: 0xc0
        1... .... = Onlink
        .1.. .... = Auto
        ..0. .... = Not router address
        ...0 .... = Not site prefix
      Valid lifetime: 0x00278d00
      Preferred lifetime: 0x00093a80
      Prefix: a::
```

Figura 3.57 – Captura da mensagem *Router Advertisement* enviada pelo *router* R1.

A mensagem *Router Advertisement* enviada contém, entre outras informações, o MTU do *link*, e o prefixo e seus tempos de vida (válido e preferido), a ser usado pela máquina M1 na auto-configuração do seu endereço IPv6. Esta mensagem pode-se analisar na Figura 3.57.

Na Figura 3.58 ilustra-se o processo verificado.

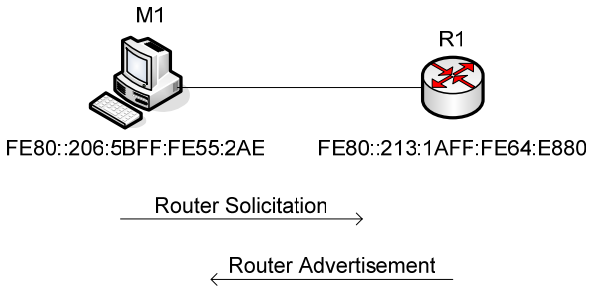


Figura 3.58 – Síntese do processo de *Router Discovery*.

3.6.4. Redirect

O *Redirect* (redireccionamento) é o processo através do qual os *default routers* informam os terminais IPv6, de um melhor endereço de primeiro salto para atingir um determinado destino. Este mecanismo já existia no IPv4, através da utilização das mensagens ICMP *Redirect*. Existem duas situações em que o redireccionamento pode ser usado:

- Quando o *default router* de uma máquina não é o melhor para atingir o destino, ou seja, quando existe um outro *router* "mais perto" (tendo em conta as métricas de encaminhamento) desse destino;
- Quando a máquina destino de determinado tráfego está no mesmo *link* da máquina origem, mas a máquina origem não sabe disso, e encaminha o tráfego para o *default router*.

As mensagens ICMPv6 *Redirect* apenas são enviadas pelo primeiro *router* do caminho entre o nó origem e destino. Os terminais IPv6 nunca enviam mensagens *Redirect*, e os *routers* nunca actualizam as suas tabelas de encaminhamento tendo em conta a sua recepção.

Para exemplificar o funcionamento do processo de *Redirect*, utilizou-se o cenário da Figura 3.59. Na rede de testes montada, os dois *routers* correm o protocolo de encaminhamento RIPng.

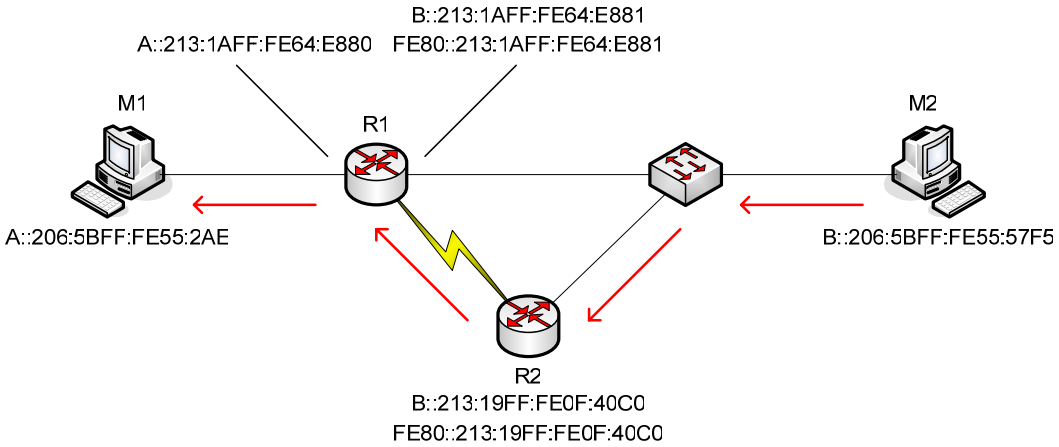


Figura 3.59 – Cenário utilizado para exemplificar o processo de *Redirect*.

No cenário anterior o *default router* da máquina M2 é o *router* R2, pelo que a comunicação entre a máquina M2 e a máquina M1 se processa como indicado pelas setas (M2 → R2 → R1 → M1). Na figura seguinte (Figura 3.60), que mostra a captura de um *ping* da máquina M2 para a máquina M1, tendo como endereço de próximo salto o endereço do *router* R2 (endereço físico *00:13:19:0F:40:C0*), verifica-se isso mesmo.

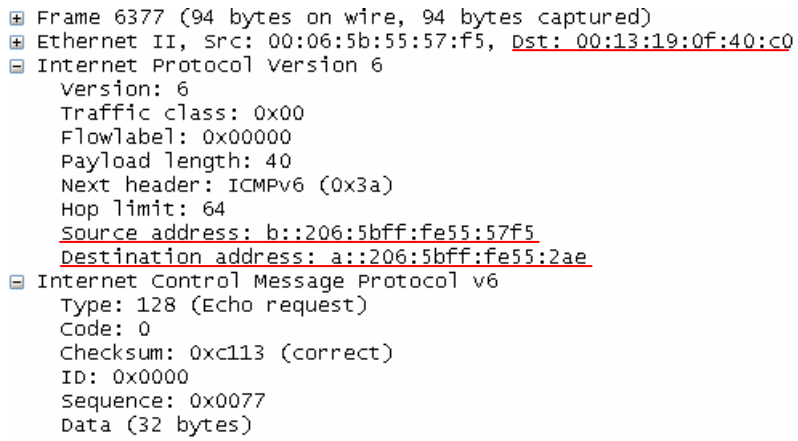


Figura 3.60 – Captura de um *ping* da máquina M2 para a máquina M1.

Mas o melhor caminho entre as máquinas M2 e M1 é aquele que os pacotes seguem, se a máquina M2 encaminhar o tráfego com destino a M1 para o *router* R1. Sendo assim, quando o *router* R2 tenta encaminhar o *ping* anterior para o *router* R1, apercebe-se desta situação e envia para a máquina M2 uma mensagem ICMPv6 *Redirect* (Figura 3.61), para que ela altere a sua tabela de encaminhamento e da próxima vez que quiser enviar tráfego para M1 o faça através de R1.

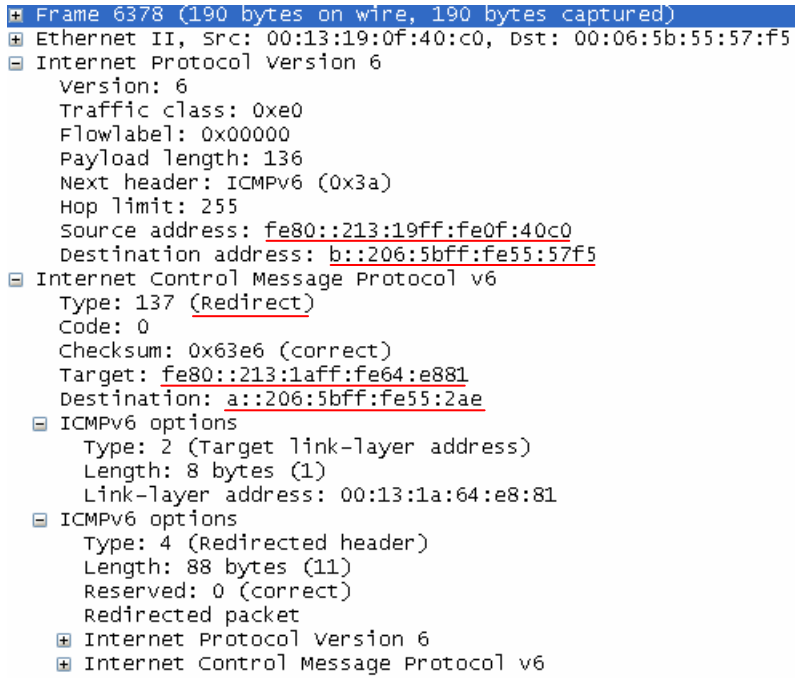


Figura 3.61 – Captura da mensagem ICMPv6 *Redirect* enviada pelo *router* R2 para a máquina M2.

Na Figura 3.62 verifica-se o envio da mensagem *Redirect* por parte do *router* R2 para máquina M2 e o novo percurso dos pacotes com origem na máquina M2 e destino a M1.

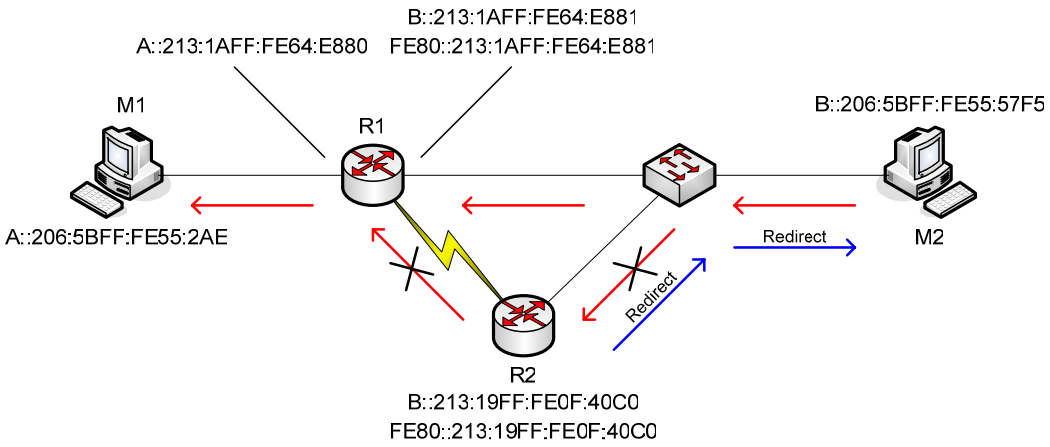


Figura 3.62 – Síntese do processo de *Redirect*.

A partir deste momento, o tráfego tendo como destino a máquina M1 proveniente da máquina M2 é encaminhado directamente por esta para o *router* R1 (endereço físico *00:13:1A:64:E8:81*), conforme se pode ver na Figura 3.63.

```
⊞ Frame 6381 (94 bytes on wire, 94 bytes captured)
⊞ Ethernet II, Src: 00:06:5b:55:57:f5, Dst: 00:13:1a:64:e8:81
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 40
    Next header: ICMPv6 (0x3a)
    Hop limit: 64
    Source address: b::206:5bff:fe55:57f5
    Destination address: a::206:5bff:fe55:2ae
⊟ Internet Control Message Protocol v6
    Type: 128 (Echo request)
    Code: 0
    Checksum: 0xc112 (correct)
    ID: 0x0000
    Sequence: 0x0078
    Data (32 bytes)
```

Figura 3.63 – Captura de um *Echo Request* de M2 para M1 já encaminhado pelo *router* R1.

3.6.5. Neighbor Unreachability Detection (NUD)

O processo de *Neighbor Unreachability Detection* (NUD) é o mecanismo através do qual um nó determina que a camada IPv6 de um vizinho já não está a receber pacotes, ou seja, não está atingível. Um vizinho está atingível se tiver existido uma confirmação recente de que pacotes IPv6 enviados para esse nó vizinho foram recebidos e processados. Este mecanismo não verifica necessariamente a conectividade ponto-a-ponto entre origem e destino, visto que, devido ao facto de o nó vizinho poder ser uma máquina ou um *router*, o nó vizinho poderá não ser o destino final do pacote. O NUD apenas verifica a atingibilidade do primeiro salto para o destino.

Uma das formas da atingibilidade ser confirmada é através do envio de uma mensagem *unicast Neighbor Solicitation* e da recepção de uma mensagem *Neighbor Advertisement* solicitada. As mensagens *Neighbor Advertisement* solicitadas, com as suas *Solicited flags* a 1, apenas são enviadas em resposta a mensagens *Neighbor Solicitation*, pelo que as *Neighbor Advertisement* não solicitadas não são prova de atingibilidade. A troca destas mensagens apenas confirma a atingibilidade do nó que envia a *Neighbor Advertisement* ao nó que envia a *Neighbor Solicitation*.

Tome-se como exemplo o cenário da Figura 3.64.

Figura 3.64 – Cenário usado para explicar o NUD.

Se a máquina M1 envia uma mensagem *unicast Neighbor Solicitation* para a máquina M2, e a M2 envia uma *Neighbor Advertisement* solicitada para a M1, a máquina M1 considera que a M2 está atingível. Como não há confirmação de que a M1 recebeu a *Neighbor Advertisement* da M2, esta não considera a M1 atingível. Para confirmar a atingibilidade da máquina M1, a máquina M2 deve enviar a sua mensagem *unicast Neighbor Solicitation* para a M1 e receber a respectiva *Neighbor Advertisement unicast* solicitada.

A Figura 3.65 apresenta um sumário das mensagens trocadas entre as duas máquinas durante o processo descrito.

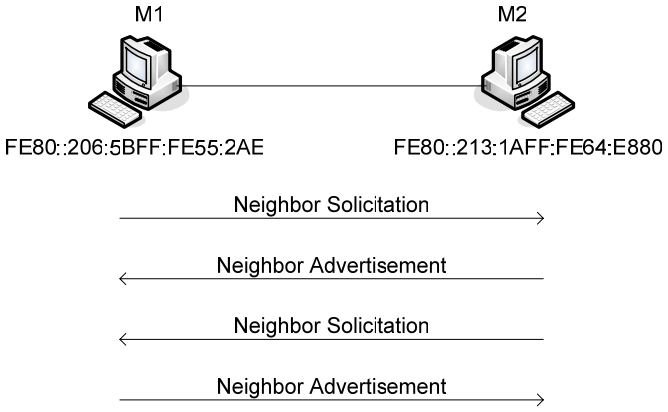


Figura 3.65 – Sumário das mensagens trocadas pelas duas máquinas durante o processo de NUD.

Na Figura 3.66 podem ver-se as mensagens trocadas no processo de *Neighbor Unreachability Detection*, capturadas no analisador de protocolos de rede Ethereal.

No.	Time	Source	Destination	Protocol	Info
3995	2010	fe80::206:5bff:fe55:2ae	fe80::213:1aff:fe64:e880	ICMPv6	Neighbor solicitation
3996	2010	fe80::213:1aff:fe64:e880	fe80::206:5bff:fe55:2ae	ICMPv6	Neighbor advertisement
3997	2011	fe80::213:1aff:fe64:e880	fe80::206:5bff:fe55:2ae	ICMPv6	Neighbor solicitation
3998	2011	fe80::206:5bff:fe55:2ae	fe80::213:1aff:fe64:e880	ICMPv6	Neighbor advertisement

Figura 3.66 – Resumo das mensagens capturadas durante o NUD.

Na Figura 3.67 pode constatar-se a *flag Solicited* da mensagem *Neighbor Advertisement*, solicitada da máquina M1 para a máquina M2 colocada a 1.

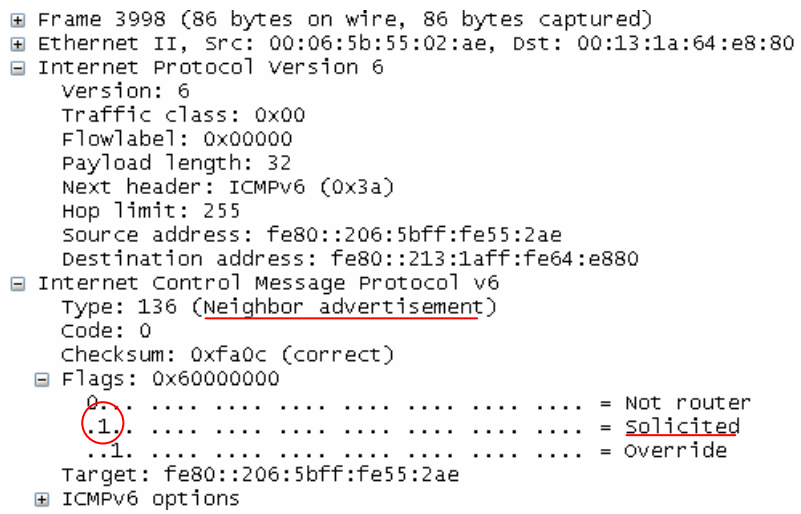


Figura 3.67 – Captura de uma mensagem *Neighbor Advertisement* solicitada.

Outro método de determinar a atingibilidade de um nó é quando são recebidas as confirmações de dados enviados por parte dos protocolos de camada superior, como por exemplo o TCP. Neste caso, confirma-se a atingibilidade ponto-a-ponto. Outros protocolos, como por exemplo o UDP, não têm um método de determinar ou indicar o sucesso da comunicação, e aqui, a troca de mensagens *Neighbor Solicitation* e *Neighbor Advertisement* torna-se importante para confirmar a atingibilidade dos nós.

3.6.6. Duplicate Address Detection (DAD)

No IPv4 os nós utilizam mensagens ARP *Request* e o método *Gratuitous* ARP para detectar endereços duplicados no *link* local. No IPv6, o equivalente a este mecanismo é o processo de *Duplicate Address Detection* (DAD) [55]. Durante este processo, um nó verifica precisamente se o endereço que quer usar não está já a ser usado por um nó vizinho, ou seja, a sua unicidade no *link*.

O mecanismo de DAD utiliza as mensagens *Neighbor Solicitation* e *Neighbor Advertisement*, e é muito parecido com a resolução de endereços, mas difere dela na medida em que o endereço origem da mensagem *Neighbor Solicitation* é o endereço *unspecified* (*::*), e o endereço destino da mensagem *Neighbor Advertisement* é o endereço *link local multicast all nodes* (*FF02::1*). É usado o endereço *unspecified* como endereço origem da mensagem *Neighbor Solicitation*, pois o endereço para o qual está a ser feito o DAD, só pode ser usado quando estiver determinado que não existem duplicados. Para o caso de haver uma resposta à *Neighbor Solicitation*, essa resposta *Neighbor Advertisement* é enviada para o endereço *multicast all nodes*, pois o nó que está a executar o DAD pode ainda não ter qualquer endereço IPv6 configurado, e por isso não pode receber tráfego *unicast*.

Para se explicar o funcionamento do mecanismo de DAD, utilizou-se a rede de testes ilustrada na Figura 3.68.

Figura 3.68 – Cenário usado na explicação do DAD.

A máquina M1 tem o endereço IPv6 *A::1* configurado na sua interface. Ao se tentar configurar o mesmo endereço na máquina M2, esta vai iniciar o mecanismo de DAD, marcando o endereço *A::1* como *tentative* e enviando para o endereço *multicast solicited-node FF02::1:FF00::1*, derivado daquele endereço, uma mensagem *Neighbor Solicitation*. Como se pode ver na Figura 3.69, o

endereço origem desta mensagem é o endereço *unspecified*, e no campo *Target* vai o endereço para o qual se está a fazer o DAD.

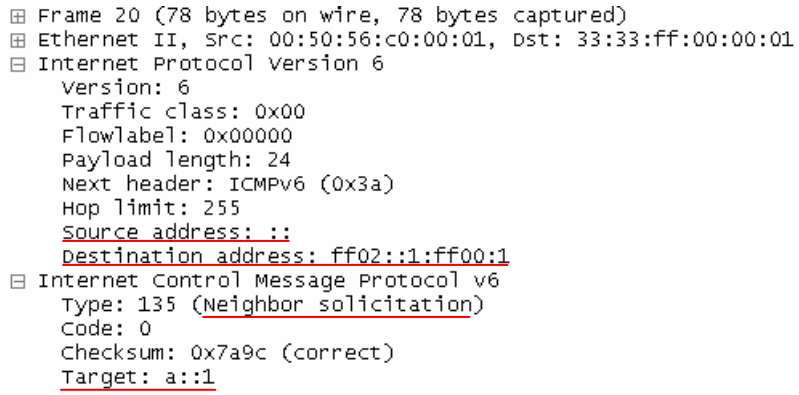


Figura 3.69 – Captura da mensagem *Neighbor Solicitation* do mecanismo de DAD.

Como a máquina M1 está no mesmo *link* da máquina M2, e tem configurado o endereço constante no campo *Target* da mensagem *Neighbor Solicitation*, vai enviar para o *link* uma mensagem *Neighbor Advertisement*, para o endereço *multicast all nodes*, para que se saiba que aquele endereço já está em uso, e que não pode ser usado por outra máquina do *link* local. A captura desta mensagem pode ser vista na Figura 3.70.

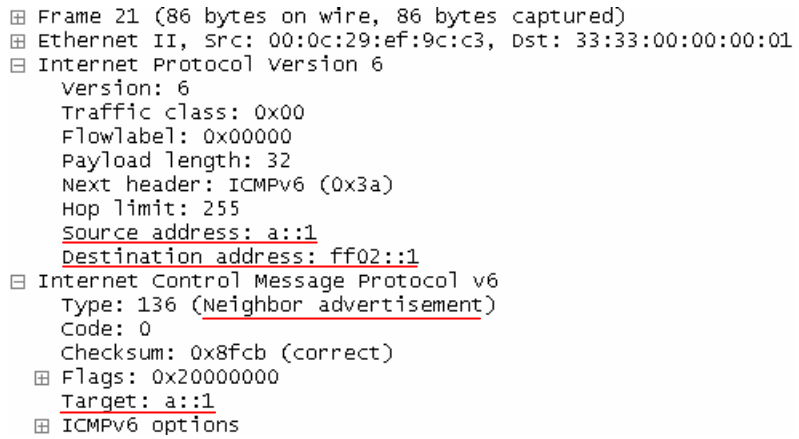


Figura 3.70 – Captura da mensagem *Neighbor Advertisement* do mecanismo de DAD.

Como a máquina M2 recebeu uma mensagem *Neighbor Advertisement* em resposta à sua *Neighbor Solicitation*, o mecanismo de DAD falha e o endereço marcado como *tentative* será marcado como duplicado, não podendo assim ser usado pela interface da máquina M2. Caso não tivesse sido recebida nenhuma *Neighbor Advertisement* em resposta à *Neighbor Solicitation*, o endereço seria único no *link* e poderia ser usado pela interface da máquina M2.

3.7. Multicast Listener Discovery (MLD)

O *Multicast Listener Discovery* (MLD) [59] é o protocolo usado pelos *routers* IPv6 para descobrir a presença de ouvintes *multicast* (isto é, nós que desejam receber pacotes *multicast*) nos *links* aos quais estão directamente ligados, e para saber especificamente que endereços *multicast* são do interesse desses nós vizinhos. A informação adquirida pelo MLD é usada pelos protocolos de encaminhamento em uso pelos *routers*, para que o tráfego *multicast* possa ser distribuído por todos os *links* onde existam nós interessados nesse tráfego *multicast*.

O MLD é o equivalente no IPv6 à versão 2 do IGMP no IPv4. Em complemento ao MLD está especificado o MLDv2 (*Multicast Listener Discovery Version 2*) [81] que é a tradução do IGMPv3 do IPv4 para IPv6.

3.7.1. Mensagens MLD

O MLD é um sub-protocolo do ICMPv6 e, assim sendo, as suas mensagens seguem o formato geral das mensagens ICMPv6. O formato das mensagens MLD é apresentado na Figura 3.71.

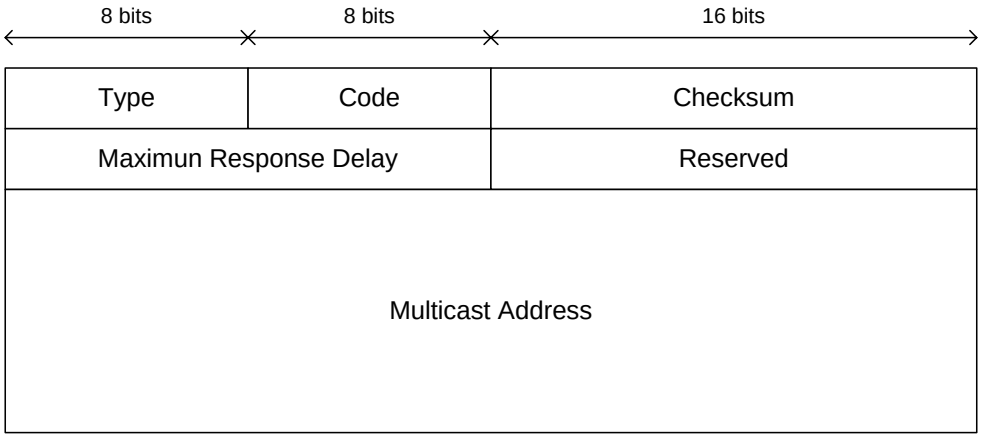


Figura 3.71 – Formato das mensagens MLD.

Todas as mensagens MLD são enviadas com um endereço origem *link-local* e um *Hop Limit* igual a 1 para que não passem da rede local. Para que os *routers* não ignorem as mensagens MLD, mesmo que não estejam interessados no tráfego *multicast* em questão, as mensagens MLD são enviadas com uma opção *Router Alert* num *Hop-by-Hop Options header*.

Existem três tipos de mensagens MLD:

- ***Multicast Listener Query*:** mensagem usada pelos *routers* para periodicamente saber se existem nós pertencentes a grupos *multicast* de qualquer endereço, e para determinar que nós pertencem ao grupo de um dado endereço *multicast*.
- ***Multicast Listener Report*:** mensagem usada pelos nós IPv6 para demonstrarem interesse em receber tráfego *multicast* num determinado endereço, ou para responderem a uma mensagem *Multicast Listener Query*.
- ***Multicast Listener Done*:** mensagem usada pelos nós IPv6 para abandonar grupos *multicast*, ou seja, para informar os *routers* que já não estão interessados em receber tráfego *multicast* num determinado endereço.

Na Figura 3.72 e na Figura 3.73 podem-se analisar duas mensagens, *Multicast Listener Report* e *Multicast Listener Done*, respectivamente, capturadas pelo analisador de protocolos de rede Ethereal. É de notar a presença do *Hop-by-Hop Options header* com a opção *Router Alert* em ambas as mensagens e do *Hop Limit* do cabeçalho IPv6 igual a 1.

Com a mensagem *Multicast Listener Report* da Figura 3.72, o nó identificado pelo endereço IPv6 *link-local FE80::206:5BFF:FE55:2AE* manifesta o seu interesse em pertencer ao grupo *multicast* do endereço *FF02::1:FFAF:BFE5*.

```
⊞ Frame 4576 (86 bytes on wire, 86 bytes captured)
⊞ Ethernet II, Src: 00:06:5b:55:02:ae, Dst: 33:33:ff:af:bf:e5
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 1
    Source address: fe80::206:5bff:fe55:2ae
    Destination address: ff02::1:ffaf:bfe5
⊟ Hop-by-hop Option Header
    Next header: ICMPv6 (0x3a)
    Length: 0 (8 bytes)
    Router alert: MLD (4 bytes)
    PadN: 2 bytes
⊟ Internet Control Message Protocol v6
    Type: 131 (Multicast listener report)
    Code: 0
    Checksum: 0xa1ef (correct)
    Maximum response delay: 0
    Multicast Address: ff02::1:ffaf:bfe5
```

Figura 3.72 – Mensagem *Multicast Listener Report*.

Com a mensagem *Multicast Listener Done* da Figura 3.73, o nó referido anteriormente deseja abandonar o grupo daquele endereço *multicast*.

```
⊞ Frame 4584 (86 bytes on wire, 86 bytes captured)
⊞ Ethernet II, Src: 00:06:5b:55:02:ae, Dst: 33:33:00:00:00:02
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 32
    Next header: IPv6 hop-by-hop option (0x00)
    Hop limit: 1
    Source address: fe80::206:5bff:fe55:2ae
    Destination address: ff02::2
⊟ Hop-by-hop Option Header
    Next header: ICMPv6 (0x3a)
    Length: 0 (8 bytes)
    Router alert: MLD (4 bytes)
    PadN: 2 bytes
⊟ Internet Control Message Protocol v6
    Type: 132 (Multicast listener done)
    Code: 0
    Checksum: 0x6084 (correct)
    Maximum response delay: 0
    Multicast Address: ff02::1:ffaf:bfe5
```

Figura 3.73 – Mensagem *Multicast Listener Done*.

3.8. Auto-configuração

Uma das características chave do IPv6, e uma das que mais tempo poupa aos administradores de rede, é a auto-configuração de endereços [55]. Esta foi desenhada para assegurar que uma máquina tenha conectividade à rede sem necessidade de configurações manuais. Até mesmo redes de maior dimensão, com múltiplas sub-redes e diversos *routers*, se poderão auto-configurar sem necessidade de recorrer a servidores de DHCP (*Dynamic Host Configuration Protocol*). A auto-configuração do IPv6 torna-se muito importante quando se pensa que numa rede podem existir todo o tipo de dispositivos, desde televisores a frigoríficos, passando por fogões e leitores de vídeo, e não se quer depender de um servidor de DHCP para que esses dispositivos domésticos funcionem.

O IPv6 define dois modos de auto-configuração: o modo *stateful* e o modo *stateless*. A aproximação *stateless* é usada quando não é muito importante quais os endereços utilizados na rede, desde que sejam únicos e devidamente encaminhados. A aproximação *stateful* é usada quando a rede exige um maior controlo sobre a atribuição de endereços, e quando é necessário configurar outros parâmetros e informações (p. ex., servidores de DNS). O administrador da rede especifica que modo de auto-configuração deve ser usado fazendo as devidas configurações nos *routers*, que por sua vez fazem uso

dos respectivos campos nas mensagens *Router Advertisement* (*flags Managed Address Configuration* e *Other Stateful Configuration*) [54].

Para assegurar que todos os endereços configurados são únicos num dado *link*, todos os nós deverão executar o mecanismo de *Duplicate Address Detection* (DAD) (ver Subsecção 3.6.6) antes de atribuir os endereços às suas interfaces. Este processo tem de ser realizado para todos os endereços, independentemente se foram adquiridos através do modo *stateful* ou do modo *stateless*.

3.8.1. Stateless

A auto-configuração *stateless* baseia-se no mecanismo de *Router Discovery* (RD) já explicado (ver Subsecção 3.6.3), e não requer qualquer configuração manual dos terminais, nem servidores adicionais, apenas configuração mínima dos *routers*. Este modo de auto-configuração permite às máquinas gerar os seus próprios endereços usando uma combinação de informação disponível localmente (p. ex., endereço MAC) e informação anunciada pelos *routers*, através de mensagens *Router Advertisement*. Os *routers* anunciam prefixos para cada *link*, enquanto que os terminais geram um identificador de interface que a identifica unicamente na rede, e o endereço IPv6 é formado combinando estes dois elementos. Na ausência de *routers*, um terminal gera apenas (e sempre) um endereço *link-local*, que é suficiente para permitir a comunicação entre nós do mesmo *link*. O endereço *link-local* (ver Subsecção 3.3.4.1) é formado juntando ao prefixo *link-local FE80::/10* o identificador da interface baseado no modelo EUI-64 (ver Subsecção 3.3.3).

Este modo de auto-configuração facilita muito a renumeração de máquinas de uma rede, porque basta mudar o prefixo da rede no *router* e o processo de RD tratará do resto. Durante a renumeração, existe um período de transição em que a interface tem atribuído mais que um endereço, no mínimo um usando o prefixo antigo e outro usando o novo prefixo. Durante o processo de renumeração, os *routers*, através das mensagens *Router Advertisement*, baixam o valor do tempo de vida do antigo prefixo e divulgam o novo. Passado algum tempo, o antigo prefixo deixa de poder ser utilizado pelos terminais e estes passam apenas a usar os endereços formados usando o novo prefixo.

3.8.2. Stateful

A auto-configuração *stateful* permite aos terminais obterem os seus endereços e/ou informações e parâmetros de configuração de um servidor, normalmente, um servidor de DHCPv6 [73]. Os servidores mantêm bases de dados com os endereços que são atribuídos a cada máquina. Os modos *stateless* e *stateful* complementam-se um ao outro, na medida em que podem ser usados simultaneamente por um terminal. Por exemplo, um terminal pode usar a auto-configuração *stateless* para configurar o seu próprio endereço, mas usar a auto-configuração *stateful* para obter outra informação necessária para a sua conectividade à rede (p. ex., o endereço de um servidor de DNS). Na Tabela 3.4 podemos ver uma comparação entre os modos de auto-configuração *stateless* e *stateful* e as suas principais diferenças.

Stateless	***Stateful***
Usa *Neighbor Discovery*	Usa DHCPv6
Leve, incluído no IPv6	Requer servidores e clientes DHCPv6
Configuração e tempo de administração mínimos	Requer alguma administração do servidor de DHCP
Renumeração gradual	Renumeração provocada
Endereços gerados automaticamente	Endereços mais controlados
Endereços fixos configurados manualmente	Endereços fixos podem ser atribuídos
Não fornece (ainda) parâmetros DNS	Fornece parâmetros DNS

Tabela 3.4 – Comparação entre a auto-configuração *stateless* e *stateful* (Adaptado de [37]).

3.9. Encaminhamento

Encaminhamento é o processo de encaminhar pacotes entre segmentos de rede, ou seja, o mecanismo através do qual nós localizados em segmentos de rede diferentes conseguem comunicar e trocar pacotes entre si. O encaminhamento é uma parte fundamental do IP, independentemente da sua versão.

A arquitectura hierárquica dos endereços IPv6 permite a sua agregação por parte dos ISPs, para que o encaminhamento seja escalável e eficiente. Esta estrutura hierárquica foi desenhada para limitar o crescimento das tabelas de encaminhamento da Internet pois, sem um bom esquema de endereçamento hierárquico, os *routers* têm de guardar tabelas de encaminhamento gigantescas, o que traz implicações significativas para o encaminhamento interno e externo. O CIDR (*Classless Inter-Domain Routing*) no IPv4 já resolve este problema com agregação de rotas, mas não da melhor forma. O encaminhamento no IPv6 é feito, como já era no IPv4, tendo em conta o *longest prefix match*, ou seja o maior prefixo cujos *bits* coincidem com o endereço a encaminhar.

O encaminhamento pode ser estático ou dinâmico. O encaminhamento estático é baseado em entradas nas tabelas de encaminhamento configuradas manualmente que não se alteram com a alteração da topologia da rede. O encaminhamento dinâmico, sensível às alterações da rede, baseia-se na actualização automática das tabelas de encaminhamento através do uso de protocolos de encaminhamento. Os quatro protocolos mais comuns: RIP (*Routing Information Protocol*), OSPF (*Open Shortest Path First*), IS-IS (*Intermediate System to Intermediate System*) e BGP (*Border Gateway Protocol*), já têm extensões para IPv6. A grande parte dos maiores fabricantes de *routers* já implementa estas extensões, conhecidas como RIPng [50], OSPFv3 [61], IS-ISv6 [91] [90] e BGP4+ [58] [65].

3.10. Mecanismos de Transição

O sucesso da adopção de qualquer nova tecnologia depende da sua fácil integração com a infra-estrutura já existente. Logo, aquando de uma mudança, a transição tem de ser discutida, e é geralmente muito importante (normalmente é para a transição que vai grande parte do orçamento). Muitas tecnologias não tiveram sucesso devido à falta de cenários e ferramentas de transição. O IPv6 foi desenhado, desde o início, pensando na transição: não existirá um novo dia "1 de Janeiro de 1983". Olhando para a Internet, e vendo que esta é composta actualmente por centenas de milhares de redes IPv4 e milhões de nós IPv4, o desafio está em fazer a integração e a transição tão transparentes quanto possível para os utilizadores finais.

A transição de protocolos não é fácil e a transição de IPv4 para IPv6 não é excepção. Os dois protocolos vão coexistir durante muitos anos e, tendo isso em conta, um vasto leque de técnicas foram definidas e desenvolvidas para tornar a coexistência possível e proporcionar uma fácil transição [67]. Os principais mecanismos envolvidos na migração de IPv4 para IPv6 são:

- Pilha Dupla (*Dual Stack*);
- Túneis IPv6 sobre IPv4;
- Tradução.

3.10.1. Pilha Dupla (Dual Stack)

As técnicas de pilha dupla, ou *dual stack*, permitem ao IPv4 e IPv6 coexistir nos mesmos dispositivos. Um nó com pilha dupla tem suporte para ambas as versões do protocolo, o que é o equivalente a dizer que tem ambas as *stacks* IPv4 e IPv6 instaladas e activas. Este tipo de nó é referido como um nó IPv6/IPv4. A Figura 3.74 mostra a arquitectura protocolar de um nó *dual stack*.

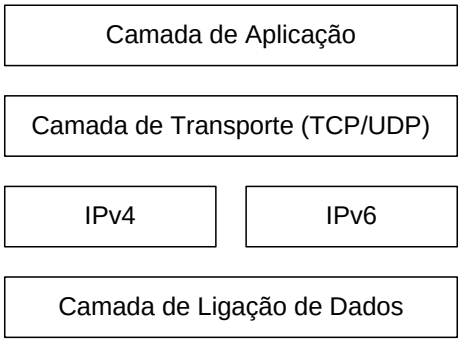


Figura 3.74 – Arquitectura protocolar de um nó *dual stack*.

Muito relacionado com o conceito de pilha dupla está o DNS (*Domain Name System*, ver Subsecção 3.11.1), que é usado com ambas as versões do IP para resolver nomes e endereços. Uma boa infra-estrutura DNS é necessária devido ao predominante uso de nomes (em vez de endereços) para referir recursos da rede. Um nó IPv6/IPv4 necessita de um servidor de DNS capaz de resolver os dois tipos de endereços. Para isso é necessário que o servidor de DNS possua registos que suportem as resoluções IPv4 e IPv6. O DNS passa, então, a poder devolver endereços IPv4, IPv6 ou ambos, e dessa forma a versão do IP a ser usada é baseada na resolução de nomes e na preferência da aplicação.

3.10.2. Túneis IPv6 sobre IPv4

Enquanto a conectividade IPv6 nativa não está disponível e não é suportada pela infra-estrutura de encaminhamento existente, é necessário usar mecanismos para fornecer conectividade IPv6 através das redes IPv4. Os túneis IPv6 sobre IPv4 encapsulam o tráfego IPv6 em pacotes IPv4 (campo *protocol* igual a 41), de modo a que esse tráfego possa ser transportado pelo *backbone* IPv4, permitindo a sistemas IPv6 isolados comunicar entre si, sem ser necessário actualizar a infra-estrutura IPv4 existente entre eles. Os túneis são uma das estratégias de desenvolvimento chave durante o período de coexistência do IPv4 e IPv6. A Figura 3.75 mostra o uso dos túneis IPv6 sobre IPv4. De notar que devido ao encapsulamento é acrescentado ao pacote IPv6 original o cabeçalho IPv4.

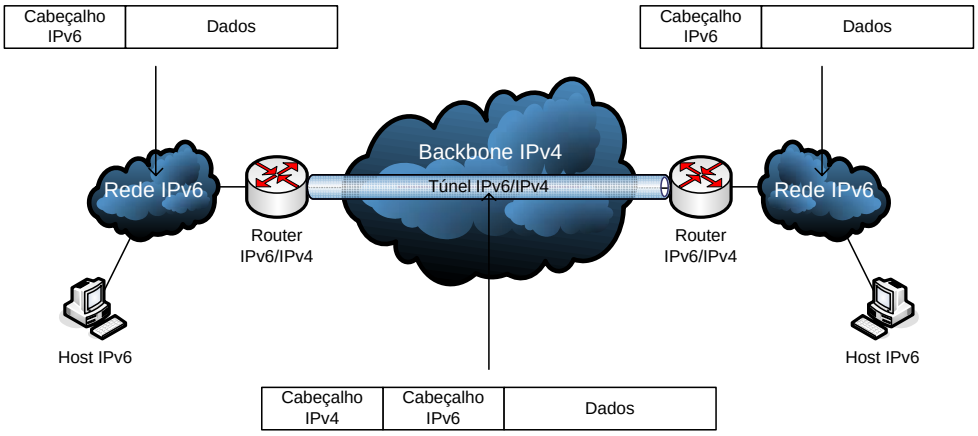


Figura 3.75 – Uso dos túneis IPv6 sobre IPv4.

Os nós extremidade do túnel tratam do encapsulamento, sendo este processo transparente para os nós intermédios. Como é óbvio, os nós que implementam os túneis, *routers* ou terminais, tem de ser *dual stack* de forma a suportarem IPv6 e IPv4 e poderem efectuar o encapsulamento. Os túneis podem ser usados em diferentes topologias: *Router-to-Router*, *Host-to-Router*, *Router-to-Host* e *Host-to-Host*. Existem dois tipos principais de túneis: os túneis automáticos e os túneis configurados.

Túneis automáticos

Os túneis automáticos permitem aos nós IPv6/IPv4 comunicar sobre uma infra-estrutura IPv4 sem a necessidade de efectuar configurações manuais. As suas extremidades são caracterizadas pelo uso de interfaces lógicas e endereços IPv6 que derivam dos endereços IPv4 configurados (*IPv4-compatible*, *IPv4-mapped*, *6to4*, *6over4* ou ISATAP). Este tipo de túnel é usado pelas topologias *Router-to-Host* e *Host-to-Host*.

Túneis configurados

Os túneis configurados são túneis que exigem que ambas as extremidades do túnel, entrada e saída, sejam explicitamente configuradas com os endereços IPv6 e IPv4 apropriados. No caso de um túnel configurado, os endereços IPv4 das extremidades do túnel não são codificados nos endereços IPv6 origem e destino. Este tipo de túnel é usado nas topologias *Router-to-Router* e *Host-to-Router*.

3.10.3. Tradução

As técnicas de tradução, base dos mecanismos de *proxy*, permitem a nós IPv6 comunicar com nós IPv4 e vice-versa, e baseiam-se na tradução dos protocolos. São técnicas que oferecem mecanismos de transição em adição aos de pilha dupla e de túneis. O objectivo é fornecer transparência na comunicação entre nós de redes IPv6 e nós de redes IPv4 e vice-versa. A ideia base destas técnicas é pegar num pacote IPv6 (ou IPv4), retirar os dados e inseri-los num pacote IPv4 (ou IPv6). Existem muitas aplicações de tradução, nomeadamente: SIIT (*Stateless IP/ICMP Translation*) [62], NAT-PT (*Network Address Translation-Protocol Translation*) [63], TRT (*Transport Relay Translation*) [71].

3.11. Protocolos de Suporte

Existem outros protocolos que, apesar de não estarem directamente ligados ao IPv6, se modificaram de forma a se adaptar a esta nova tecnologia. Os mais importantes são o DNS e o DHCP.

3.11.1. DNS

O serviço DNS [74] é um serviço indispensável para facilitar o acesso a qualquer nó da rede. Numa rede interna, são utilizados por vezes protocolos próprios do sistema operativo para a resolução de nomes (p. ex., o WINS no Windows), mas saindo dessa rede, o DNS é o tipo de serviço que mais facilita toda a navegação. Se já facilitava no IPv4, muito mais facilita no IPv6. É praticamente impensável fazer com que o utilizador comum da Internet tenha de aceder aos seus sítios, utilizando normais endereços IPv6 em hexadecimal. O DNS é o responsável pela parte mais visível da Internet e, provavelmente, com ele existirão utilizadores que nem notarão a passagem de IPv4 para IPv6.

Para resoluções IPv4 os registos já existentes são o "*A*" para endereços e o "*PTR*" no domínio *IN-ADDR.ARPA* para nomes. Para resoluções IPv6 foi criado o novo registo "*AAAA*" (*quad A*) [46], e mais recentemente o "*A6*" [66], para suportar a resolução de nomes em endereços IPv6. O registo "*PTR*" foi estendido para suportar a resolução inversa nos novos domínios *IP6.INT* (caso "*AAAA*") e *IP6.ARPA* (caso "*A6*"). Ambos os registos têm características distintivas.

"*AAAA*" vs. "*A6*"

O registo (*resource record*) funciona basicamente como o normal "*A*" associado ao IPv4, a única diferença será mesmo o maior número de bits.

O funcionamento do "*A6*" varia significativamente dos "*A*" e "*AAAA*". A ideia é fazer com que o DNS se torne ainda mais hierárquico do que é, e inserir também alguma auto-configuração em todo o procedimento de actualização de *resource records*. O "*A6*" funciona com segmentos. Por exemplo: o domínio *estg.ipv6* tem o segmento de endereço *CAFE:1234*, e o domínio *host.estg.ipv6* só terá *0001*

(ver Subsecção 4.2.2). Assim, podem-se construir as mais diversas hierarquias com variados segmentos. No caso do cliente DNS solicitar uma resolução, todo o processo começa no topo da hierarquia, descendo à medida da solicitação efectuada, até encontrar o endereço correcto. Baseado no mesmo processo funciona o processo de auto-configuração: quando existe alguma mudança de endereço no topo da hierarquia, automaticamente as hierarquias abaixo ficam correctamente configuradas. Na Tabela 3.5 apresentam-se algumas diferenças entre os dois registos [75].

"AAAA"	***"A6"***
Bastante simples de usar.	Um pouco mais trabalhoso a primeira vez que se configura. Processo de auto-configuração nas actualizações.
Adequado para pequenas redes, com poucas hierarquias.	Adequado para redes com diversas hierarquias, e mudanças de endereçamento constantes.
Optimização de leitura dos *resource records*: leitura directa, não necessita de procurar em diversos *resource records*.	Para construir um endereço precisa de várias leituras em diferentes *resource records*, o que pode piorar um pouco o desempenho das respostas aos pedidos.

Tabela 3.5 – Diferenças entre os registos "*AAAA*" e "*A6*".

O IETF específica o uso de "*AAAA*", existindo o "*A6*" ainda só de forma experimental. Apesar de o conceito ser interessante, os cerca de quinze anos de experiência com o outro tipo de registo, deixam por agora muito pouco espaço para o uso do "*A6*".

3.11.2. DHCPv6

Já foi referido que o uso de DHCPv6 [73] está directamente ligado com o modo de auto-configuração *stateful* (ver Subsecção 3.8.2). Devido ao facto de existir também a auto-configuração *stateless* no IPv6 (ver Subsecção 3.8.1), pode-se deduzir que no IPv6 seja menos necessária a auto-configuração por DHCP. No IPv4 a única forma de auto-configurar os endereços era usando este protocolo.

Este protocolo aproveitou a mudança necessária para suportar IPv6, para melhorar também o protocolo em si. Existem diversas mudanças, tanto a nível do cabeçalho do protocolo, como a nível de endereçamento.

As principais diferenças são:

- O DHCPv6 deixou de derivar do BOOTP, não existindo agora compatibilidade entre os dois protocolos, como acontecia no IPv4;
- O uso de *multicast*, utilizando os endereços *multicast all-dhcp-agents* (*FF02::1:2*) e *all-dhcp-servers* (*FF05::1:3*) [180] em vez de *broadcast*, nas diversas negociações, como é mostrado na Figura 3.76;

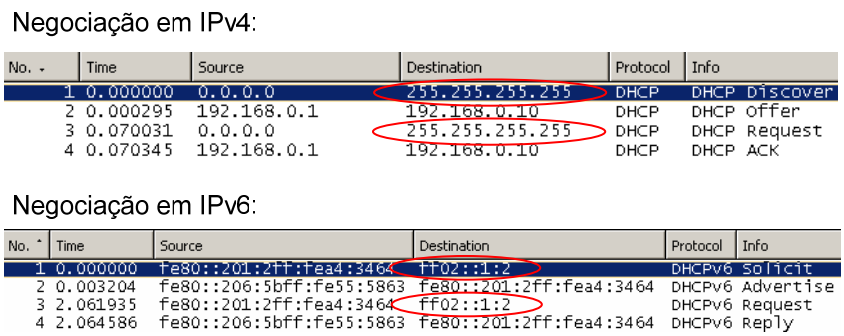


Figura 3.76 – Vizualização dos endereços usados nas negociações DHCPv4 e DHCPv6.

- Existem diversos tipos de mensagens, tanto no DHCP como no DHCPv6. O objectivo das mensagens principais não mudou significativamente. O *Solicit* do DHCPv6 equivale praticamente ao DHCP *Discover*, e o mesmo se passa com as outras três mensagens. Uma das mensagens mais importantes que não existe no DHCP é a mensagem DHCPv6 *Renew*. Esta mensagem é utilizada quando já se tem configurado o IP respectivo, mas apenas se pretende mudar qualquer configuração no endereço (p. ex., *prefered lifetime*, *valid lifetime*…).
- O cabeçalho ficou mais organizado, sendo a alocação de endereços feita com recurso a *message options*, como acontece, por exemplo, nos cabeçalhos de extensão (ver Secção 3.2). O aspecto dos cabeçalhos de ambos os protocolos pode ser visualizado na Figura 3.77.

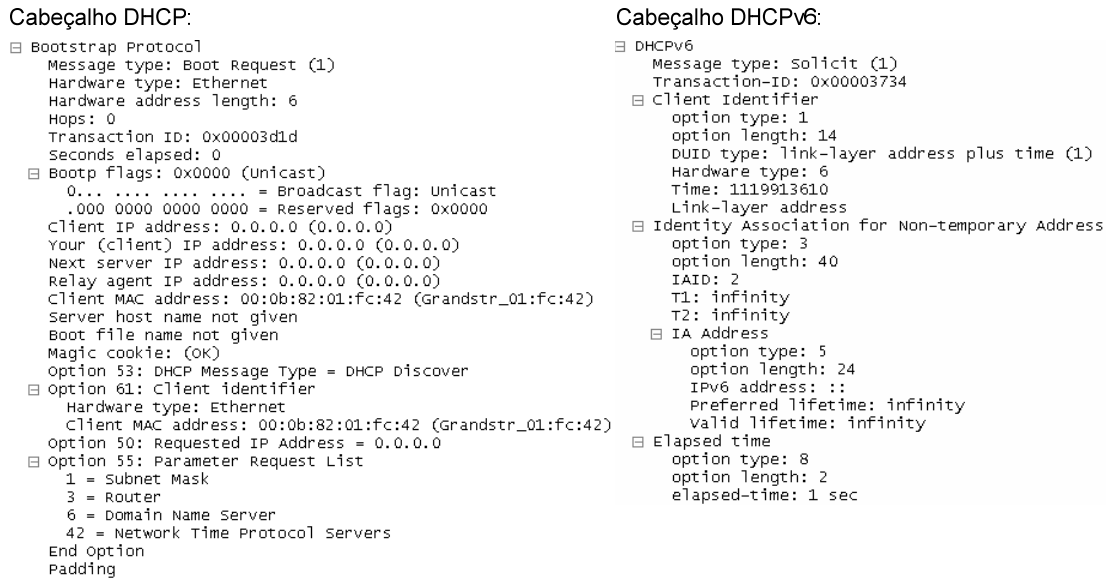


Figura 3.77 – Cabeçalho DHCP e DHCPv6.

Neste exemplo, foram capturadas as primeiras mensagens do início da negociação. No caso do DHCP, foi capturada a mensagem *DHCP Discover*, no DHCPv6 foi capturada a *DHCPv6 Solicit*. Numa segunda mensagem do DHCPv6 para além do *Client Identifier*, existirá outro conjunto que será a identificação do servidor (*Server Identifier*).

O *Identify Association* serve para que cliente e servidor formem um grupo entre si; existem diversas opções neste conjunto, muitas delas muito importantes, como o T1 e T2 que servem para especificar o tempo em segundos com que o cliente irá efectuar pedidos ao servidor para que lhe seja alterado o *Prefered Lifetime* e o *Valid Lifetime*. Nas próximas mensagens, o T1 e o T2 vão adquirir valores em segundos fornecidos pelo servidor.

Por fim existe o *Elapsed Lifetime* que serve para cronometrar o tempo de uma negociação. Neste caso, verifica-se o tempo de 1 segundo no *Elapsed Lifetime*, pois a negociação ainda está na primeira mensagem. Esta cronometragem do tempo é bastante útil, pois no caso do servidor de DNS ter um tempo de negociação muito alto, é sinal que está congestionado ou não está a funcionar nas devidas condições, pelo que deve avisar outro servidor para tratar dessa negociação.

4. Implementação

A tecnologia IPv6 não tem a menor utilidade se os diversos sistemas operativos, e respectivos serviços ligados às redes informáticas não a perceberem. Durante a fase de adopção deste novo protocolo, todas as implementações têm um papel fundamental no progresso do IPv6. Neste capítulo descrevem-se as implementações existentes ao nível dos sistemas operativos, serviços e diversas aplicações. Apresentam-se ainda diferentes formas de ligação a uma rede exterior (Internet), e acesso a partir de uma normal ligação doméstica proporcionada por um ISP.

4.1. Sistemas Operativos e Suporte IPv6

Felizmente, existem diversas aplicações já com suporte IPv6, seja de forma nativa, ou com recurso a alguma *patch*. As aplicações mais importantes, pela sua obrigatoriedade, são praticamente os sistemas operativos, e é por aqui que todo o suporte começa. Neste momento existe já suporte IPv6 em todos os sistemas operativos, nas suas versões mais recentes.

4.1.1. Desenvolvimento

A importância do contributo para o IPv6 proporcionada pelo sistema operativo prende-se bastante com a popularidade do mesmo. Hoje em dia, caso o sistema operativo não possua IPv6 de forma nativa, raro é o utilizador que o implementa, e mesmo trazendo de forma nativa, a maior parte dos utilizadores não altera qualquer opção para o seu suporte. Um exemplo bastante simples, é o caso do sistema operativo Windows XP, que apesar de apenas necessitar de uma linha de comando para activar o suporte IPv6, raro é o utilizador que depois de uma normal instalação do Windows, execute essa linha. Existem também autênticos manuais na Internet de como se desliga o suporte para IPv6, sob a justificação de por vezes poder tornar algumas pesquisas mais lentas. Este tipo de iniciativas vem piorar em muito a aceitação do IPv6 nos diversos sistemas operativos.

Microsoft Windows

Os primeiros passos dados pela Microsoft foram dados no Windows 2000. Nessa altura foi desenvolvido uma *patch* (ver [125]) que tinha como objectivo principal dar aos programadores algumas opções para programação em IPv6. Dava assim oportunidade de experimentar e aprender alguns conceitos. Nessa altura existia já no horizonte uma nova versão do Windows com suporte IPv6 nativo, pelo que o fornecido ao Windows 2000 foi uma versão preliminar do que seria a *stack* IPv6 no Windows XP.

De seguida surgiu o tão esperado Windows XP com o suporte nativo IPv6 (ver [126]), sendo necessário para o seu uso apenas um comando. O uso de Windows XP Home Edition ou Windows XP Professional Edition é indiferente em termos de IPv6. Por outro lado, com os *Service Packs* não acontece o mesmo, existindo diferenças significativas, principalmente na passagem para o *Service Pack 1*. Pode afirmar-se que só no *Service Pack 1* o IPv6 é considerado realmente um protocolo suportado. Em relação ao *Service Pack 2*, houve uma alteração significativa em termos de comandos.

Os comandos para IPv6 são agora iguais aos utilizados no Windows Server 2003 (ver [127]); outras alterações são: uma *firewall* IPv6 por defeito e o *Teredo tunneling* que permite IPv6 através de NAT.

No que diz respeito ao Windows Server 2003, surgiu logo de seguida, e o suporte IPv6 utiliza na sua maioria a base do Windows XP. São introduzidas algumas novidades devido principalmente à maior quantidade de serviços proporcionada por este sistema operativo em relação ao Windows XP (ver [128]).

Na Tabela 4.1 pode-se verificar os diversos serviços proporcionados pelos dois sistemas operativos nas suas mais recentes versões.

Windows XP SP2	**Windows Server 2003 SP1**
Internet Explorer	
Cliente Telnet	
Cliente FTP	
Cliente DNS	
Partilha de ficheiros e impressoras (SMB)	
Gestão de *Web Servers* remotos (WebDAV)	
Autenticação e Encriptação (IPsec)	
	Mobilidade
	Servidor de DNS
	Servidor de HTTP (IIS 6.0)
	Windows Media Services
	Gestor de rede
	Diversas ferramentas para programação (*Winsocks*, RPCs…)

Tabela 4.1 – Suporte IPv6 existente no Windows XP SP2 e Windows Server 2003 SP1.

Curiosamente também existem algumas funcionalidades IPv6 para Windows 95, 98 e ME, mas nada têm a ver com a própria Microsoft, tendo sido proporcionadas pela Trumphet (ver [129]) e algumas outras empresas.

Outro sistema operativo que ainda não foi falado e que é bastante importante, devido a todo o conceito de computação ubíqua a que o IPv6 pretende aderir, é o Windows CE. Versões do Windows CE a partir da .Net 4.1 já têm incluído o suporte nativo IPv6. É claro que o suporte é sempre a nível de cliente e nunca de servidor, mas o mais importante é os programadores já puderem programar aplicações para os PDAs, com suporte IPv6 nesta plataforma.

De futuro, esperam-se grandes desenvolvimentos IPv6 no Windows Longhorn [34]:

- IPv6 por omissão;
- IPv6 preferido em relação ao IPv4;
- Poder de opção para usar apenas IPv6;
- Todos os serviços com suporte IPv6;
- Todos os servidores com suporte IPv6 (versão Windows Longhorn Wave).

Outro ponto interessante será também a introdução do IPv6 no mundo das consolas, neste caso a Xbox.

Em termos de suporte oficial ao nível do *IPv6 Ready Logo* (ver Subsecção 2.2.6), a Microsoft tem dois sistemas já com aprovação de *Logo Phase-1*: o Windows Server 2003 e o Windows CE 4.2 (ver [99]). O *Logo* aplica-se também a todas as versões existentes superiores a estas.

Linux

O primeiro código foi adicionado à versão 2.1.8 do *kernel* em Novembro de 1996, por Pedro Roque (na altura na Faculdade de Ciências da Universidade de Lisboa). O desenvolvimento do IPv6 para Linux nunca conseguiu acompanhar o ritmo das especificações dos RFCs e *drafts* até nascer em 2002 o projecto japonês USAGI (ver [139]) [12]. Este projecto tinha, e continua a ter, o objectivo de pôr o Linux no rumo do IPv6. Ou seja, ter um nível de implementação no Linux à medida que vão sendo especificados os respectivos RFCs e *Internet Drafts.* O projecto aproveitou a ideia dos objectivos com o projecto KAME (ver [140]), este ligado ao sistema BSD (*Berkeley Software Distribution*).

Existem inúmeras distribuições de Linux, mas o suporte IPv6 das mesmas é sempre igual se for considerado o mesmo *kernel* (ver [144]). É como se o *kernel* fosse o destino, e as distribuições o caminho para chegar a esse destino. O *kernel* permite, por exemplo, mobilidade IPv6, mas conforme a distribuição, existe uma forma diferente de alcançar essa mobilidade IPv6. Por esse facto, o melhor a fazer é consultar o respectivo *site* da distribuição, e verificar se existe documentação acerca do suporte IPv6. Se não existir, verificará provavelmente alguma página sobre IPv6 nessa distribuição de forma não oficial. Alguns exemplos de distribuições populares são:

- Debian (ver [145]);
- Gentoo (ver [146]);
- Fedora Core (ver [147]).

Os serviços com suporte IPv6 são imensos (ver [148]), e variam conforme a distribuição (ver [149]) e respectivo *kernel* associado (ver [150]).

Em termos oficiais, a situação mais importante é a aprovação do *kernel* 2.6.11-rc2 como *Logo Phase-1*. Porém existem outras aprovações associadas ao Linux. Isso prende-se principalmente com distribuições a usar implementações USAGI, e implementações do próprio USAGI. Existe também referência a uma versão especial da distribuição KNOPPIX, direccionada especialmente para o IPv6, que também tem aprovação *IPv6 Ready Logo*.

BSD

O caso dos sistemas BSD, mais propriamente FreeBSD (ver [151]), OpenBSD, NetBSD (ver [152]) e BSDi, é bastante parecido com o que sucedeu com o Linux. Só que neste caso, o projecto (KAME) que deu o grande passo no desenvolvimento IPv6 para este tipo de sistema operativo, é antecessor do que surgiu no Linux. O projecto surgiu em Abril de 1998, e na altura estava previsto ter a duração de dois anos acabando em Março de 2000. O objectivo principal deste projecto era, tal como sucedeu depois com o projecto associado ao Linux, encaminhar os sistemas BSD para as implementações IPv6 especificadas pela IETF. No início pretendia-se, num espaço de dois anos, fazer renascer o IPv6 para o BSD, para depois cada variante evoluir de forma continuada em todas as implementações IPv6. Passados dois anos, os objectivos mudaram, sendo agora o KAME responsável por todo o IPv6 associado às diversas variantes BSD. Ainda assim, sendo um projecto várias vezes expandido na sua duração com incrementos de dois anos, tem o seu término por agora agendado para Março de 2006.

O suporte IPv6 nas seguintes plataformas foi criado pelo KAME:

- FreeBSD 4.0 e versões acima;
- OpenBSD 2.7 e versões acima;

- NetBSD 1.5;
- BSD/OS 4.2 e versões acima (também conhecido como BSDi, variante descontinuada).

Ao contrário do Linux, em que o *kernel* reunia de certa forma todas as distribuições de forma a terem o mesmo suporte IPv6, no caso das variantes BSD o caso é diferente pois cada variante tem o seu *kernel* próprio. Isso faz com que o suporte IPv6 possa variar significativamente de variante para variante. O aconselhável é verificar as respectivas páginas das variantes e a página do KAME (ver [141]) para verificação do suporte IPv6 respectivo.

O suporte oficial refere-se directamente ao KAME, e não às respectivas variantes. O KAME já garantiu então, através da sua *stack*, o *Logo Phase-1*.

KAME, USAGI e TAHI

O KAME e o USAGI já foram bastante falados, e estão intimamente ligados no que diz respeito a objectivos. O TAHI (ver [143]) é um projecto que surgiu praticamente em parceria com o KAME em 1998, e serve essencialmente para o desenvolvimento de diversos testes de conformidade IPv6. Pode-se fazer um pouco a comparação com o oficial *IPv6 Ready Logo*, mas não é bem a mesma coisa pois não certifica nada, apenas desenvolve para testes. Além disso, apenas trabalha para Linux e BSD, estando por isso ligado directamente aos dois projectos (KAME e USAGI).

Estas iniciativas são todas referenciadas como fundamentais para o desenvolvimento IPv6 nas plataformas Linux e BSD, mas na simples instalação de qualquer sistema operativo, mesmo usando todo o suporte IPv6 existente, não existe qualquer tipo de referência a elas. Pode parecer estranho, o porquê de tantas *patches* disponibilizadas por elas, suportando o sistema operativo de forma nativa praticamente todo suporte IPv6 para essa plataforma. Porém todo esse código nativo do sistema operativo foi desenvolvido pelos projectos. O que se passa é que a instituição responsável pelos diversos *kernels* dos sistemas operativos, espera sempre por uma versão bastante estável proporcionada por eles, e simplesmente adoptam-na. Isto faz com que existam inúmeras *patches* disponíveis, mas só algumas numa fase bastante final, sejam adoptadas pelos *kernels*. Na Figura 4.1 é explicado em maior detalhe o processo.

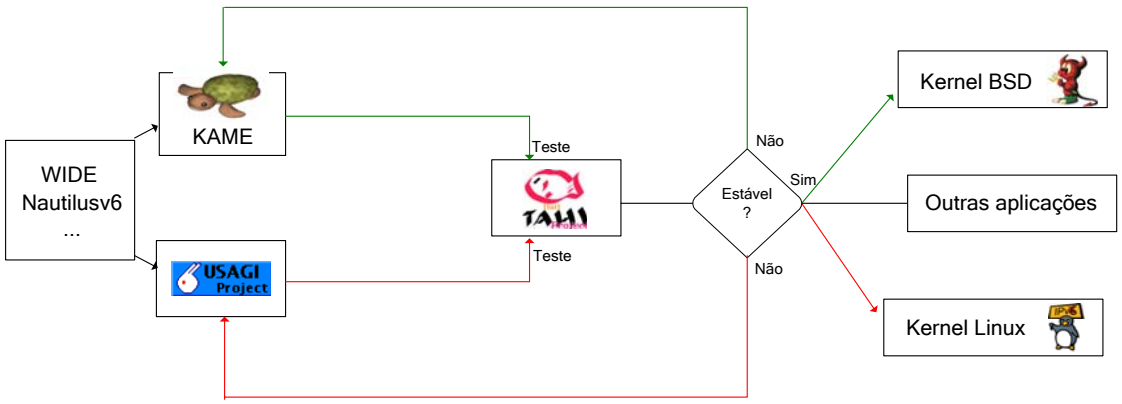


Figura 4.1 – Esquema de todo a implementação IPv6 *Open-Source*.

Até agora temos referido sempre estes projectos, direccionando-os para a base do sistema operativo, mas não é só isso a sua abrangência, trabalhando por vezes em regime de cooperação com certas aplicações.

De referir é também o facto de existirem outras iniciativas acopladas. Estas iniciativas são geralmente muito específicas, abordando temas como a mobilidade IPv6, a segurança IPv6, etc.

Sendo assim, podemos recorrer aos próprios projectos se quisermos necessariamente ficar actualizados o máximo possível (ver [18]), mesmo que as versões disponíveis não estejam ainda bastante estáveis.

Podemos também verificar essa estabilidade através do projecto USAGI, que está acessível a qualquer utilizador/programador.

Por vezes podemos ter de recorrer também a estes projectos, quando estamos perante aplicações com versões antigas e necessitamos algum suporte IPv6. Geralmente o que se passa com as aplicações, é que os projectos começam a desenvolver *patches* para a versão actual, mas todo o código desenvolvido por eles só vai ser implementado na próxima versão da aplicação. Como exemplo disso podemos olhar para a aplicação Apache (ver Subsecção 4.2.1): o código feito pelo KAME foi feito para as versões 1.3.x, mas este código só foi implementado de raiz na versão 2 do Apache, o que obriga a que alguém que queira implementar IPv6 com a versão 1.3.x tenha de recorrer à *patch* fornecida pelo projecto em questão (ver [142]), neste caso o KAME.

Existem já muitas aplicações com suporte nativo IPv6, tanto para Linux como para BSD (ver [19]). Muito se deve a todas estas iniciativas, que fizeram com que o leque de aplicações com suporte IPv6 para estes dois sistemas ultrapassasse qualquer outro.

Mac OS X e Darwin

Existe suporte IPv6 desde a versão Mac OS X 10.2 Jaguar. Existe de forma nativa e não é necessário qualquer configuração adicional para o seu suporte. Este tipo de sistema tem como *core*[18] a solução *open-source* Darwin, que por sua vez se baseia na plataforma FreeBSD. Como foi referido anteriormente, todo o suporte IPv6 para sistemas com base BSD é proporcionado na sua maioria pelo KAME, e este sistema utilizado nos Mac não foge a regra.

Nas versões Mac OS X 10.2.x Jaguar, o Darwin 6.x é baseado nas versões 4.x do FreeBSD, que coincide com o suporte mínimo para ter IPv6 neste tipo de sistema. Ou seja, foi aproveitado o código feito pelo KAME, tendo-lhe sido adicionada apenas a parte gráfica correspondente ao OS X. Porém só os mais básicos serviços tinham suporte IPv6: *ping6*, *traceroute6*, *telnet*.

Com a chegada do Mac OS X 10.3 Panther e Darwin 7.0 baseado nas versões 5.x do FreeBSD, o suporte IPv6 foi estendido, sendo suportados em IPv6 os seguintes serviços:

- DNS (Bind);
- *Firewall*;
- *Mail* (POP, IMAP, SMTP);
- SMB;
- HTTP (Apache 2).

Mais recentemente, com Mac OS X 10.4 e Darwin 8.0 foi adicionado além do serviço SMB, o CIFS.

O valor da atribuição *Logo Phase-1*, dado ao KAME nas suas implementações, devido à sua ligação ao FreeBSD e consequente ligação do FreeBSD ao Darwin, faz com que Mac OS X tenha também garantido toda a certificação de suporte IPv6.

Cisco IOS

Em Junho de 2001, começou a corrida da Cisco (ver [135]) ao IPv6. Estava-se perante o lançamento do IOS 12.2T; a letra "T" numa versão do IOS da Cisco representa uma introdução de uma nova tecnologia, neste caso foi o IPv6. O suporte IPv6 nesta versão servia mais como uma introdução do

[18] A parte mais importante de todo o sistema operativo.

novo protocolo nos *routers* Cisco. De seguida, o IPv6 já surge na linha 12.3. Isso é uma prova da estabilidade que o IPv6 já tinha atingido na Cisco.

É de referir que os *routers* Cisco se dividem em dois grupos distintos: os que se baseiam no *software* (*Software-Based*), e os que se baseiam no *hardware* (*Harware-Based)*. As versões acima referidas (12.2T e 12.3 e 12.4) são para *routers* baseados no *software* como as séries 2500, 2600 e 7200; as versões 12.0.(x)S são para *routers* de *hardware* como a série 12000.

Nas versões 12.0.(x)S o IPv6 chegou mais tarde, devido a duas razões: o facto de este tipo de sistema (*Hardware-Based*) ser necessariamente muito estável, devido à importância dos sítios onde estes *routers* geralmente são colocados (*backbone*); e o facto de um *upgrade* a estes *routers* não ser tão simples como o que acontece com os *routers* baseados no *software*.

Dentro das linhas mãe 12.0S, 12.2, 12.3, 12.4 são várias as versões com vários tipos de suporte IPv6 (ver [136]). O *upgrade* das versões nos *routers* baseados no *software* é extremamente simples, porém o espaço de armazenamento para um IOS com suporte IPv6 aumenta consideravelmente. Talvez devido a este problema, a versão 12.2S de 2004, permite implementação IPv6 num pacote de *software* bem mais pequeno.

A evolução da Cisco em todo o suporte IPv6 é absolutamente crucial para o aumento das implementações desta tecnologia. Para além disso tem também grande responsabilidade em todo o encaminhamento IPv6 (ver Secção 3.9): RIPng, OSPFv3 e BGP4+.

A compatibilidade do IPv6 com diversas tecnologias também não foi esquecida, recomendando a Cisco já alguns cenários em que engloba a tecnologia IPv6 com outras. Um caso específico é o cenário 6PE (ver [138]). Este cenário junta o IPv6 ao MPLS, usando IPv6 nas redes internas e um túnel IPv4 entre as extremidades dos *routers*.

Existem diversas versões destes sistemas operativos certificadas já com *Logo Phase-1*, tanto para *routers* baseados em *hardware*, como baseados em *software*.

Outros sistemas operativos

São diversos os sistemas operativos com suporte IPv6. Em termos de servidores a lista limita-se um pouco, existindo para além dos referidos, alguns sistemas baseados na sua maioria em Unix:

- A distribuição Solaris, da Sun (ver [153]);
- A distribuição IBM AIX (ver [154]);
- A distribuição HP-UX (ver [155]);
- As distribuições Tru64 (ver [156]) e OpenVMS (ver [157]), que antigamente pertenciam à Compaq e agora pertencem também à HP.

Falando em sistemas operativos de *routers* a lista é bastante grande. Deve-se assim consultar a página *web* do respectivo fabricante, ou então o próprio manual. No caso de o IPv6 ter um interesse extremo, deve-se consultar as listas de certificação do *IPv6 Ready Logo* (ver [99] e [100]), sendo uma certificação *Logo Phase-2* aconselhada.

Existe também aplicações, que permitem usar terminais como *routers*, utilizando um sistema operativo próprio, dentro do sistema operativo da própria máquina. Exemplo disso é o projecto Zebra (ver [158]), agora também associado ao IP Infusion (ver [159]) com o nome de sistema operativo ZebOS. Este sistema tem suporte *Logo Phase-1*.

4.1.2. Sistemas Operativos e Instalação

A instalação de um sistema operativo é geralmente o primeiro contacto que o utilizador tem com o *software*, pelo que poderá ser também o primeiro contacto com qualquer opção IPv6 existente. Ela pode existir ou não, variando conforme o sistema operativo. No caso de não existir, poderá vir o suporte já por omissão, ou poderá haver necessidade de recorrer a alguma *patch* depois da instalação concluída. Felizmente nos sistemas operativos mais recentes e testados, esta última situação nunca acontece.

Microsoft Windows XP SP2

A instalação é muito simples neste sistema operativo, bastando executar, após a instalação do Windows, o comando "`ipv6 install`". Na Figura 4.2 é mostrado como o processo sucede.

```
C:\Documents and Settings\David_S>ipv6 install
A instalar...
Êxito.
```

Figura 4.2 – Instalação do IPv6 no Windows XP.

De seguida, poderão visualizar-se os endereços IPv6 configurados através do comando "`ipconfig`".

Microsoft Windows Server 2003 SP1

Neste sistema operativo a instalação é feita através de uma interface gráfica, e não com recurso à linha de comandos anterior. Nas *Connection Properties* seleccionar *Install* > *Protocol* > *Microsoft TCP/IP version 6*. O processo é exemplificado na Figura 4.3:

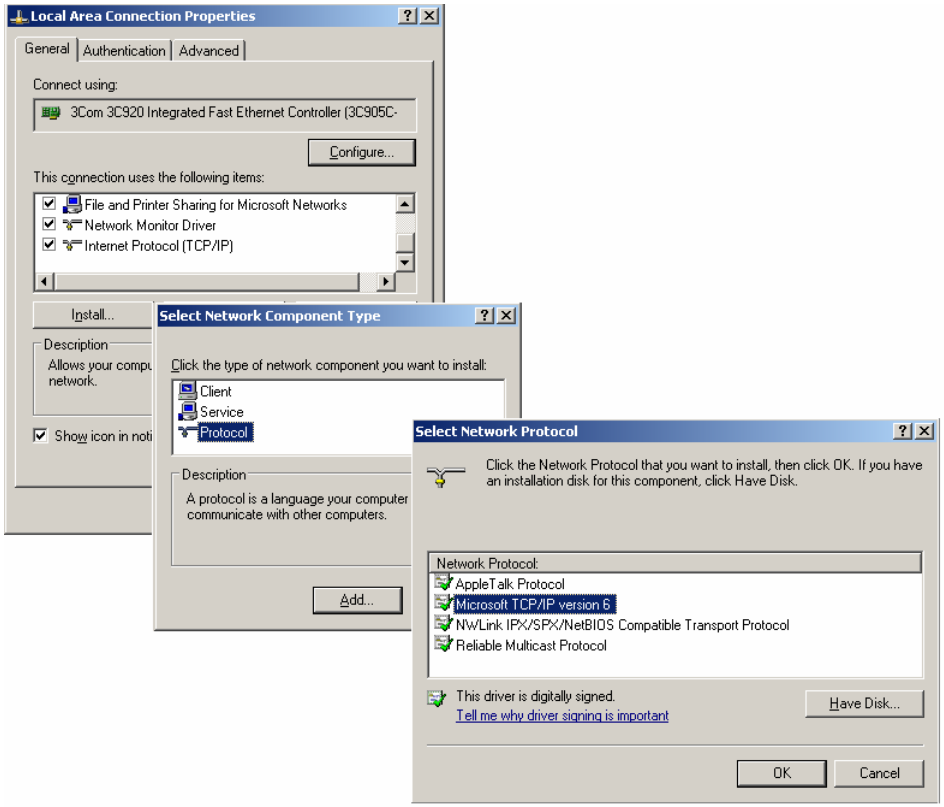


Figura 4.3 – Processo de instalação do IPv6 em Windows Server 2003.

Linux (Fedora Core 3)

A instalação no Linux também é simples. Porém, poderão ser necessários conhecimentos de instalação do *kernel* conforme a distribuição escolhida. Neste caso optou-se pela distribuição Fedora Core 3, e na instalação deste tipo de distribuição não é preciso qualquer configuração adicional, basta simplesmente instalar o sistema operativo para ter IPv6.

Ao optar por outras distribuições, poderá ser necessário configurar o seu *kernel*. Nesse caso, no menu de configuração terá que se aceder a *Device Drivers* > *Networking Support* > *Netwoking Options* > *The IPv6 protocol (EXPERIMENTAL)*, e seleccionar as opções que se pretende para o IPv6. Na Figura 4.4 está o menu final que se irá aceder, com todas as opções seleccionadas.

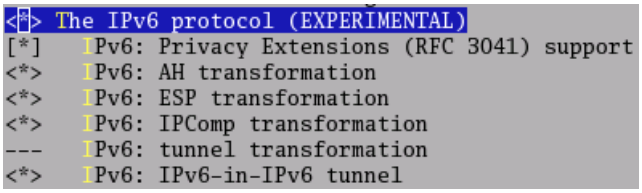


Figura 4.4 – Instalação de IPv6 no *kernel*.

A instalação pode também ser feita utilizando módulos[19], basta substituir o "*" pelo "M", e adicionar os ficheiros de módulos IPv6 no ficheiro correspondente à inicialização dos módulos. No caso do Fedora Core 3, o ficheiro é */etc/rc.modules*. Na Figura 4.5 é mostrado o aspecto que deve ter este ficheiro caso se pretendam utilizar todos os módulos IPv6.

```
modprobe ipcomp6
modprobe ipv6
modprobe xfrm6_tunnel
modprobe ah6
modprobe esp6
modprobe ip6_tunnel
```

Figura 4.5 – Módulos IPv6.

Estes tipos de configuração do *kernel* podem também ser bastante úteis no caso de se querer efectuar um *upgrade* ao *kernel*. Fazer um *upgrade* ao *kernel* em termos de IPv6 pode revelar-se extremamente útil, na medida em que o protocolo está em constante desenvolvimento de versão para versão.

O Fedora Core 3 por exemplo, traz de origem a versão do *kernel* 2.6.9. Esta versão já suportava IPv6 mas ainda não estava certificada pelo *IPv6 Ready Logo*. Assim, um *upgrade* para uma versão já certificada revela-se bastante importante para um suporte IPv6 mais estável. Como a versão que possui *Logo Phase-1* é a 2.6.11, ter um *kernel* 2.6.11 ou superior é aconselhável.

Foi escolhida a distribuição Fedora Core 3 por razões históricas. O projecto Fedora descende do Red Hat Linux. Na altura que se começou a implementar IPv6 era a distribuição mais popular. Devido a essa razão, a maioria das distribuições é baseada na abordagem IPv6 que foi feita com Red Hat. Optou-se também por efectuar o *upgrade* do *kernel* para a versão mais recente da altura.

OpenBSD 3.7

De todos os sistemas operativos testados, a instalação no OpenBSD revelou-se a mais trabalhosa, na medida em que é necessário activar diversas opções para o normal funcionamento dos mecanismos IPv6.

O *kernel* por omissão vem com as opções para IPv6 já seleccionadas, pelo que normalmente não é necessário efectuar qualquer modificação na sua configuração. Na Figura 4.6 é mostrado uma parte do ficheiro de configuração *GENERIC*, na sua configuração normal com suporte IPv6.

[19] Módulos em Linux são partes específicas do kernel que se podem executar ou não, conforme a opção do utilizador.

```
# for IPv6
pseudo-device   faith   1       # IPv[46] tcp relay translation i/f

pseudo-device   pppoe   1       # PPP over Ethernet (RFC 2516)
```

Figura 4.6 – Configuração *kernel* OpenBSD.

É necessário garantir que estas duas linhas não estão comentadas ("#" no início). A primeira linha é para suportar túneis (ver Subsecção 3.10.2), a segunda é o suporte IPv6 propriamente dito.

Após o *kernel* estar instalado, já existe algum suporte IPv6 mas apenas a nível local, ou seja apenas os endereços *link-local* estão configurados, como se poderá verificar com o comando "ifconfig".

É agora necessário proporcionar ao sistema operativo a possibilidade de adquirir endereços globais. Para isso é necessário permitir que o sistema operativo envie mensagens *Router Solicitation*, e receba as *Router Advertisement* (ver subsecção 3.6.3). Para enviar *Router Solicitations* é necessário usar o comando "rtsol", este comando envia *Router Solicitations* de forma a receber o respectivo *Advertisement*. É aconselhável executar o comando logo inicialmente, de forma que a negociação seja efectuada logo que o sistema operativo seja iniciado, não sendo necessário estar sempre a correr o comando. Basta adicionar este comando a um ficheiro de inicialização, neste caso o */etc/rc.local*, como é exemplificado na Figura 4.7.

```
#       $OpenBSD: rc.local,v 1.37 2005/02/07 06:08:10 david Exp $

# site-specific startup actions, daemons, and other things which
# can be done AFTER your system goes into securemode.  For actions
# which should be done BEFORE your system has gone into securemode
# please see /etc/rc.securelevel

# site-specific startup actions, daemons which can be run
# Add your local changes additions to this file

#mandar router solicitations para configuraçao do endereço global_
rtsol -D le2
```

Figura 4.7 – Activar *Router Solicitations* em OpenBSD (ficheiro */etc/rc.local*).

A opção "-D" permite que se mostre no ecrã todos os passos efectuados pelo comando. A opção "le2" é a interface seleccionada para o envio.

Agora só falta permitir que o sistema operativo aceite as *Router Advertisements*. Então acede-se ao ficheiro */etc/sysctl.conf* e retira-se o sinal de comentário "#" da opção "net.inet6.ip6.accept_rtadv=1", como exemplificado na Figura 4.8.

```
#       $OpenBSD: sysctl.conf,v 1.33 2004/09/22 17:49:39 hshoexer Exp $
#
# This file contains a list of sysctl options the user wants set at
# boot time.  See sysctl(3) and sysctl(8) for more information on
# the many available variables.
#
#net.inet.ip.forwarding=1       # 1=Permit forwarding (routing) of packets
#net.inet6.ip6.forwarding=1     # 1=Permit forwarding (routing) of packets
net.inet6.ip6.accept_rtadv=1    # 1=Permit IPv6 autoconf (forwarding must be 0)
```

Figura 4.8 – Permitir que o OpenBSD receba *Router Advertisements* (ficheito */etc/sysctl.conf*)

Agora está tudo pronto para funcionarem os endereços globais com os respectivos prefixos anunciados pelos *routers*.

A escolha do OpenBSD foi apenas devido ao facto de ser a variante que há menos tempo tinha sido alvo de uma actualização do suporte IPv6.

Mac OS X 10.3.7

Nada a assinalar, suporte completamente automático. Poderão verificar-se os endereços IPv6 tanto através do comando "ifconfig", como na respectiva interface gráfica.

4.1.3. Comandos Úteis

Existem alguns comandos importantes que se podem revelar de extrema utilidade em certas circunstâncias de implementação. Alguns deles são referidos a seguir.

Windows XP e Windows Server 2003

- Adicionar endereço IPv6 (*unicast* ou *anycast*):

  `"netsh inteface ipv6 add address <interface> <endereço>"`

- Remover endereço:

  `"netsh interface ipv6 delete address <interface> <endereço>"`

- Verificar grupos *multicast*:

  `"netsh interface ipv6 show joins"`

- Estatísticas de pacotes IPv4 e IPv6 trocados:

  `"netstat –s"`

- Verificar portas do protocolo TCPv6:

  `"netstat –p TCPv6 –a"`

- Verificar do UDPv6:

  `"netstat –p UDPv6 –a"`

Estes dois últimos comandos podem-se revelar bastante úteis para verificar configurações de serviços, pois pode mostrar se a porta utilizada por determinado serviço está à escuta para IPv6.

- IPsec

  `"ipsec6"` (ver [130] e [131])

- Cliente DNS

  `"nslookup –type=AAAA [site]"`

Linux

- Adicionar endereço IPv6 (*unicast* ou *anycast*):

  `"ip -6 addr add <endereço ipv6>/<prefixo> <interface>"`

- Remover endereço:

  `"ip -6 addr del <endereço ipv6>/<prefixo> <interface>"`

- Tempos de vida dos endereços:

  `"ip -6 addr show dev <interface>"`

- *Ping* ao endereço *link-local all nodes multicast* (ver Subsecção 3.3.4.2); permite saber o endereço *link-local* de todos os nós localmente ligados:

  `"ping6 –I <interface> ff02::1"`

- Verificar grupos *multicast*:

  `"netstat –g"`

- IPsec:

  daemon pluto (ver [160]), ou *daemon* racoon (ver [163]) e o comando `"setkey"`

- Cliente DNS:

  `"dig <servidor DNS> <resource record>"`

BSD

- Adicionar e remover endereço IPv6 (*unicast* ou *anycast*):

  É necessário aceder ao ficheiro */etc/hostname.[interface]*, e acrescentar a seguinte linha:

  `"inet 6 alias <endereço IPv6> <prefixo> <tipo: unicast/anycast>"`

  Para remover basta apagar a linha.

- Verificar portas:

  `"netstat -a"`

Poderá ser verificado com este comando se a porta está a escutar IPv6, basta para isso reparar na notação TCP46 ou UDP46 (no caso de escutar IPv6 e IPv4), ou também TCP6 ou UDP6 (no caso de apenas escutar IPv6.

- IPsec:

  daemon racoon (ver [164]) e o comando `"setkey"`

- Cliente DNS:

  Igual ao utilizado no Linux.

Nestas duas ultimas implementações (Linux e BSD), utilizando o comando `"sysctl -a"` poderão ser verificadas diversas opções relacionados com o IPv6. A gama de opções é bastante vasta podendo fornecer um controlo pormenorizado de todos os mecanismos associados a este novo protocolo.

4.2. Cenário 1: Serviços

Foi criada uma rede IPv6 heterogénea com vários sistemas operativos, que disponibilizar alguns serviços com suporte IPv6, de forma a testar as diversas compatibilidades entre terminais IPv6.

Foram utilizados cinco sistemas operativos diferentes:

- Windows XP Professional SP2;
- Windows Server 2003 Enterprise;
- Linux Fedora Core 3 (*kernel* 2.9.11);
- OpenBSD 3.7;
- Cisco IOS 12.3(13).

A Figura 4.9 apresenta o cenário elaborado.

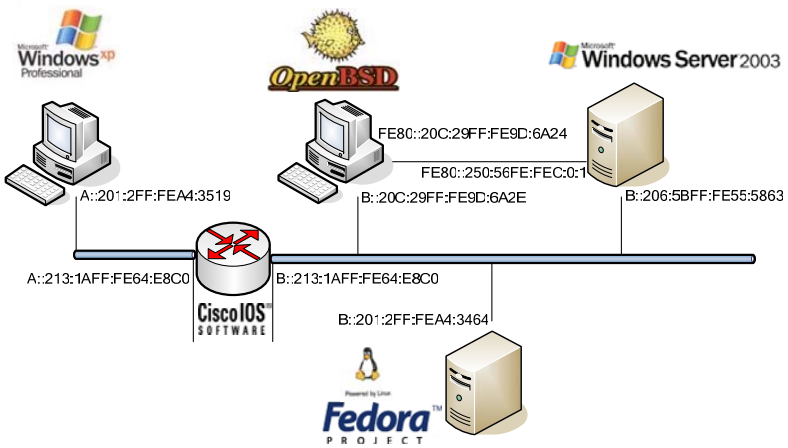


Figura 4.9 – Cenário 1 no seu estado inicial.

Todos os nós foram configurados com endereços globais (ver Subsecção 3.3.4.1), sendo os *link-local* criados automaticamente. Apenas existe uma ligação com endereços *link-local*. Essa ligação liga o Windows Server 2003 ao OpenBSD, e é uma ligação local visto o OpenBSD estar implementado numa máquina virtual no Windows Server 2003.

Ao longo de todo o percurso de construção do cenário foram aplicadas diversas configurações e testes para verificação de toda a implementação IPv6 nos variados sistemas operativos.

1.º Passo: Configuração das interfaces do *router*

- Aplicou-se o comando "`ipv6 unicast-routing`", sem este comando o *router* não encaminha tráfego *unicast*, logo não envia *Router Advertisements*, logo não configura endereços globais nos terminais.
- Aplicou-se o comando "`ipv6 enable`" em todas as interfaces necessárias. Neste caso *FastEthernet0/0* e *FastEthernet0/1*.
- Verificou-se o correcto funcionamento das interfaces IPv6, através do comando "`show ipv6 interface brief`".
- Configuraram-se os endereços globais. Aplicou-se o comando "`ipv6 address a::/64 eui-64`" na interface *FastEthernet0/0* e o comando "`ipv6 address b::/64 eui-64`" na interface *FastEthernet0/1*.
- Por fim, verificaram-se as rotas do *router*, através do comando "`show ipv6 route`". Este comando não é muito importante neste caso, pois apenas existe um *router*, o que faz com que não exista distribuição de rotas.

A Figura 4.10 demonstra a configuração efectuada.

```
R1(config)#ipv6 unicast-routing
R1(config)#
R1(config)#interface FastEthernet0/0
R1(config-if)#ipv6 enable
R1(config-if)#ipv6 address A::/64 eui-64
R1(config-if)#
R1(config-if)#exit
R1(config)#
R1(config)#interface FastEthernet0/1
R1(config-if)#ipv6 enable
R1(config-if)#ipv6 address B::/64 eui-64
R1(config-if)#
R1(config-if)#exit
```

Figura 4.10 – Configuração do *router* do cenário 1

A configuração completa deste *router* segue também no Anexo A.1.1.

2.º Passo: Verificação dos endereços nas máquinas existentes

- **Windows XP**

Efectuou-se a verificação do endereço *link-local* e do endereço global, utilizado o comando "`ipconfig`". A visualização apresenta-se na Figura 4.11.

```
Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        Description . . . . . . . . . . . : 3Com EtherLink XL 10/100 P
plete PC Management NIC (3C905C-TX)
        Physical Address. . . . . . . . . : 00-01-02-A4-35-19
        Dhcp Enabled. . . . . . . . . . . : No
        IP Address. . . . . . . . . . . . : 192.168.0.3
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        IP Address. . . . . . . . . . . . : a::3076:1b1c:abad:a1ba
        IP Address. . . . . . . . . . . . : a::201:2ff:fea4:3519
        IP Address. . . . . . . . . . . . : fe80::201:2ff:fea4:3519%6
        Default Gateway . . . . . . . . . : fe80::213:1aff:fe64:e8c0%6
```

Figura 4.11 – Visualização dos endereços IPv4 e IPv6 no Windows XP.

Neste caso verifica-se uma implementação *dual stack* IPv4/IPv6 (ver Subsecção 3.10.1). Constata-se então a criação de dois endereços globais, um baseado em informações da placa de rede (ver Subsecção 3.3.3), e outro gerado aleatoriamente (ver Subsecção 3.3.4.1). O *gateway* no IPv6 é configurado automaticamente, devido ao facto de ele considerar o nó por onde recebe *Router Advertisements*, o seu *gateway*.

Outra forma de verificar os endereços IPv6 é utilizando o utilitário `"netsh"` (NetShell). Nesse caso é necessário executar o comando `"netsh interface ipv6 show address"`, então o aspecto da consola será como a Figura 4.12.

```
Interface 6: Local Area Connection

Addr Type  DAD State  Valid Life   Pref. Life   Address
---------  ---------- ------------ ------------ ------------------------------
Temporary  Preferred   6d23h51m46s    23h48m59s a::3076:1b1c:abad:a1ba
Public     Preferred  29d23h57m19s  6d23h57m19s a::201:2ff:fea4:3519
Link       Preferred      infinite     infinite fe80::201:2ff:fea4:3519
```

Figura 4.12 – Visualização dos endereços IPv6 através do NetShell.

Aqui pode notar-se o tipo de endereço, bem como o tempo de vida válido do endereço e o seu tempo de vida como preferido (ver Subsecção 3.3.4.1).

- **OpenBSD**

Efectua-se o comando `"ifconfig"`, e neste caso visualizam-se duas interfaces com endereços IPv6, uma configurada com endereço global, e outra apenas com *link-local* como demonstra a Figura 4.13.

```
IPv6# ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 33224
        inet 127.0.0.1 netmask 0xff000000
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x6
le1: flags=8863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        address: 00:0c:29:9d:6a:24
        inet 192.168.1.4 netmask 0xffffff00 broadcast 192.168.1.255
        inet6 fe80::20c:29ff:fe9d:6a24%le1 prefixlen 64 scopeid 0x1
le2: flags=8863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        address: 00:0c:29:9d:6a:2e
        inet 192.168.0.6 netmask 0xffffff00 broadcast 192.168.0.255
        inet6 fe80::20c:29ff:fe9d:6a2e%le2 prefixlen 64 scopeid 0x2
        inet6 b::20c:29ff:fe9d:6a2e prefixlen 64 pltime 604766 vltime 2591966
pflog0: flags=0<> mtu 33224
pfsync0: flags=0<> mtu 2020
```

Figura 4.13 – Visualização dos endereços IPv4 e IPv6 em OpenBSD.

Repare-se que a interface `"le1"` apenas tem configurado o endereço *link-local*, enquanto que a `"le2"` para além do endereço *link-local* tem também um endereço global associado. Este tipo de ligação tem algumas utilidades importantes tais como obter uma ligação que funcione, caso a normal rede IPv6 não esteja operacional, ou permitir uma configuração local a partir do Windows Server 2003.

- **Fedora Core 3**

Efectuou-se também com o comando "`ifconfig`", porém, a apresentação é um pouco diferente do OpenBSD, como demonstra a Figura 4.14.

```
[root@localhost ~]# ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:01:02:A4:34:64
          inet addr:192.168.0.2  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: b::201:2ff:fea4:3464/64 Scope:Global
          inet6 addr: fe80::201:2ff:fea4:3464/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:117 errors:0 dropped:0 overruns:0 frame:0
          TX packets:63 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:18996 (18.5 KiB)  TX bytes:5655 (5.5 KiB)
          Interrupt:11 Base address:0x1000
```

Figura 4.14 – Visualização dos endereços IPv4 e IPv6 no Linux.

A opção "`-a`" serve para mostrar toda a informação disponível.

- **Windows Server 2003**

A forma de verificar os endereços é igual à utilizada no Windows XP. A Figura 4.15 apresenta o resultado do comando "`ipconfig`" e a Figura 4.16 o resultado do comando "`netsh interface ipv6 show address`".

```
Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : 3Com 3C920 Integrated Fast
oller (3C905C-TX Compatible)
   Physical Address. . . . . . . . . : 00-06-5B-55-58-63
   DHCP Enabled. . . . . . . . . . . : No
   IP Address. . . . . . . . . . . . : 192.168.0.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   IP Address. . . . . . . . . . . . : b::206:5bff:fe55:5863
   IP Address. . . . . . . . . . . . : fe80::206:5bff:fe55:5863%6
   Default Gateway . . . . . . . . . : fe80::213:1aff:fe64:e8c1%6
```

Figura 4.15 – Visualização dos endereços IPv4 e IPv6, no Windows Server 2003.

```
netsh interface ipv6>show addr
Querying active state...

Interface 4: Local Area Connection

Addr Type  DAD State  Valid Life   Pref. Life   Address
---------  ---------  -----------  -----------  -------------------------
Public     Preferred   29d23h59m2s   6d23h59m2s b::206:5bff:fe55:5863
Link       Preferred      infinite     infinite fe80::206:5bff:fe55:5863
```

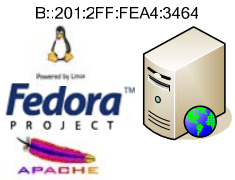
Figura 4.16 – Visualização dos endereços IPv6 no Windows Server 2003.

Destaca-se neste caso a falta do endereço privativo (ver Subsecção 3.3.4.1) existente no Windows XP. Esta situação acontece devido ao facto de o Windows Server 2003 ser destinado a servidores, o que não implica mobilidade.

Por fim utilizou-se o *ping* (Windows Server 2003) ou *ping6* (Windows XP, Linux, OpenBSD), de forma a verificar a conexão existente entre os diversos nós.

Convém não esquecer de desligar todas as *firewalls*, para que não exista tráfego IPv6 bloqueado. Existe uma interface gráfica em todos os sistemas testados, que permite desactivar este serviço. Apenas não foi instalada essa interface gráfica no OpenBSD, utilizando-se o comando "`pfctl -d`" para o desactivar.

4.2.1. Serviço HTTP



Um servidor de HTTP transmite a página a que acedemos e o seu suporte IPv6 é absolutamente necessário, para que se possa aceder a qualquer página por IPv6.

Foi então implementado um no Linux, o servidor Apache.

Apache

O Apache (ver [172]) suporta de forma nativa os endereços IPv6 (ver [173]) desde a sua versão 2.0.14. Porém, *patches* para suportar IPv6 nas versões 1.3.x também estão disponíveis. Neste caso, foi usada a versão 2.0.54, a mais recente e estável.

O Apache é *open-source*[20] e existem implementações Apache para as mais diversas plataformas. A escolha caiu sobre o Linux por o trazer de forma nativa, mas optou-se por fazer um *upgrade* à versão que vinha com a distribuição Fedora, mesmo não tendo a nova versão qualquer melhoria significativa, no que diz respeito ao IPv6.

1.º Passo: Activou-se o serviço httpd

Tal como acontecia no caso do IPv4 após activado o serviço, automaticamente são aceites pedidos e enviadas respostas para terminais IPv6.

2.º Passo: A partir do terminal acedeu-se ao servidor por HTTP

Conforme é demonstrado na Figura 4.17 foi efectuado um acesso HTTP ao servidor, com este a devolver uma mensagem. O *browser* é um cliente HTTP.

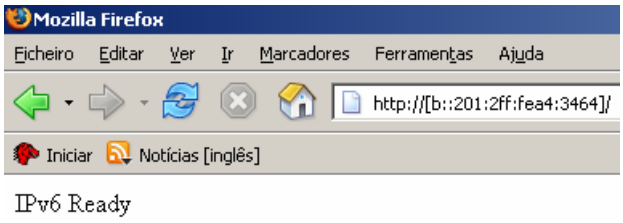


Figura 4.17 – Acesso ao servidor Apache com o Mozilla Firefox.

Atenção que para fazer um acesso deste tipo sem o serviço DNS, é necessário um *browser* que suporte endereços IPv6 literalmente (ver [84]). No caso do Internet Explorer, este tipo de endereços não é suportado. O Mozilla Firefox (exemplificado na Figura 4.17, ver [174]), o Opera ([175]), ou o Safari, são exemplos de *browsers* com suporte deste tipo de endereços. Outra restrição importante é que nenhum dos *browsers* suporta endereços *link-local*.

[20] Tipo de software em que são distribuídas as fontes de código. Geralmente é distribuído de forma gratuita, mas pode não ser.

3.º Passo: Configuração (opcional)

No caso da Figura 4.18, considerando o primeiro "*Listen*", o servidor ficou configurado para aceitar todos os pedidos no porto 80 de todos os endereços, tanto IPv4 como IPv6. Mas poderá ser configurado para aceitar só endereços IPv6, ou apenas alguns endereços IPv6.

Pode-se aceder ao ficheiro */etc/httpd/conf/httpd.conf* e substituir o "*Listen 80*" por "*Listen [::]:80*". Neste caso o Apache apenas aceitará pedidos a partir de endereços IPv6. Para aceitar apenas determinado endereço IPv6, basta colocar entre parênteses rectos o respectivo IP. Por exemplo, se o que se pretende é que apenas o cliente Windows XP aceda ao servidor, substitui-se o "*Listen 80*" por "*Listen [A::201:2FF:FEA4:3519]:80*", como é o caso do segundo "*Listen*" da Figura 4.18. Deve-se também proceder à remoção do primeiro "*Listen*", porque por ser mais abrangente ganha prioridade sobre o mais específico.

```
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, in addition to the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses (0.0.0.0)
#
#Listen 12.34.56.78:80
Listen 80
Listen [a::201:2ff:fea4:3519]:80
```

Figura 4.18 – Duas configurações do servidor Apache (ficheiro */etc/httpd/conf/httpd.conf*).

Todos os terminais conseguiram aceder à página alojada no servidor de HTTP. Em termos de formato dos pacotes, não existe qualquer alteração significativa para além da mudança nos campos de endereços.

Outros servidores de HTTP

Existem diversos servidores de HTTP para Linux (ver [119]) e com suporte IPv6, mas este é sem dúvida o mais utilizado. Um concorrente mais directo é o caso do IIS, o servidor de HTTP nativo do Windows. Esse servidor também já suporta IPv6, mas apenas na sua versão 6.1 (ver [134]), e essa versão só existe no Windows Server 2003 como se irá verificar mais à frente. Assim sendo, a opção mais aconselhada de um servidor de HTTP com suporte IPv6 para Windows XP, continua a ser o Apache.

4.2.2. Serviço DNS

Foi implementado um servidor de DNS no Windows Server 2003. Este tipo de servidor existe de forma nativa neste sistema operativo, não sendo necessária qualquer configuração adicional para o poder usar dele.

1.º Passo: Adicionaram-se os registos de associação

Os registos de associação utilizados foram de um endereço IPv6 a determinado nome. Esse processo é demonstrado na Figura 4.19.

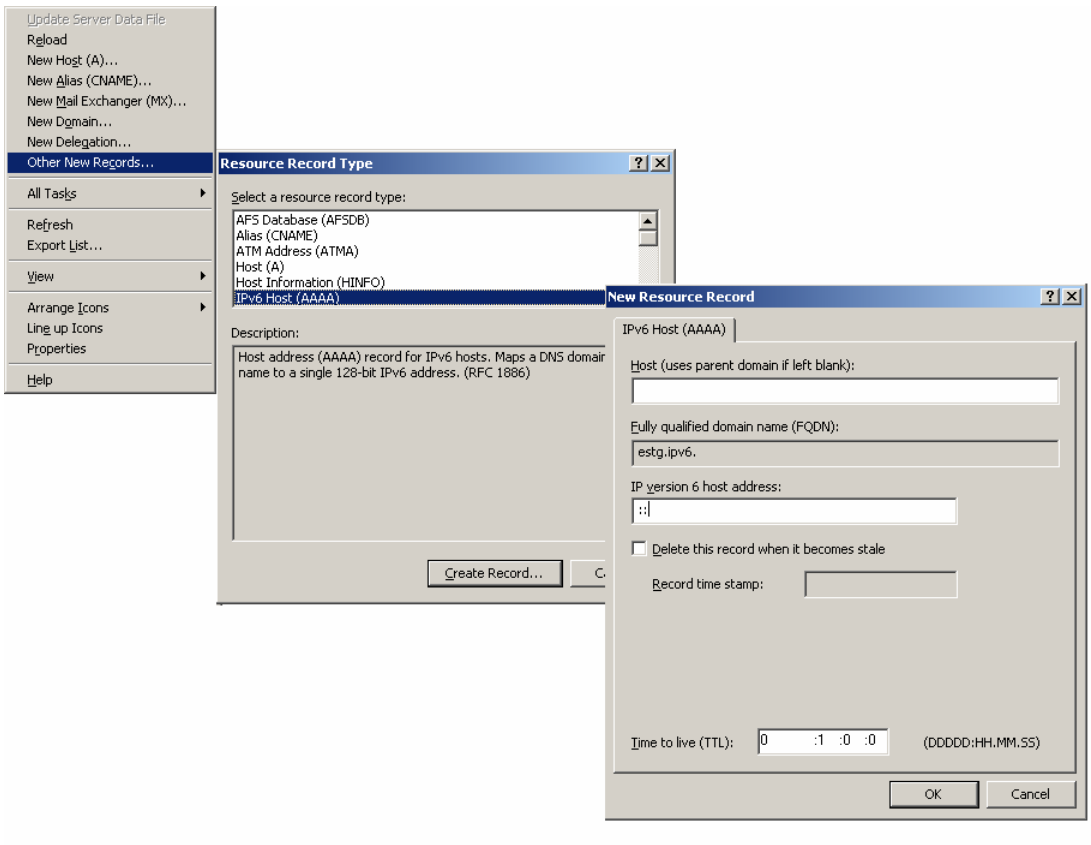


Figura 4.19 – Processo de inserção de um registo de associação.

- Acedeu-se a *Administrative Tools* > *DNS*.
- Com o botão direito do rato, acedeu-se a um menu onde se seleccionará "*Other New Records*".
- Seleccionou-se o tipo de *resource record*, para o IPv6 será o *AAAA*.
- Fez-se a associação entre nome e endereço IPv6.

Tanto o servidor de DNS do Windows Server 2003, como o cliente (*nslookup*) não suportam o *A6*, mas existem outros servidores que já o suportam, porém, o uso deste *resource record* é bastante reduzido.

Repetindo os passos atrás para a configuração de *resource records AAAA*, sendo um registo para cada máquina, a configuração ficou concluída. A configuração final do DNS é apresentada na Figura 4.20.

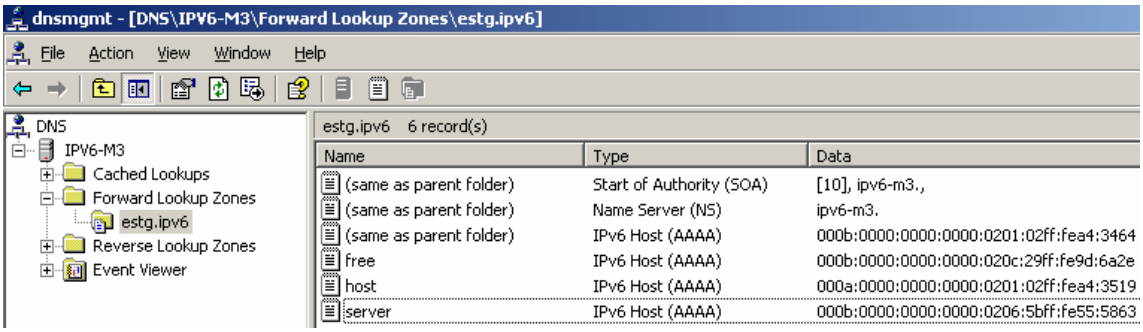


Figura 4.20 – Configuração final do DNS.

O domínio principal é *estg.ipv6*, o servidor de HTTP tem o mesmo nome do domínio (*estg.ipv6*), as outras máquinas têm o nome de *free.estg.ipv6*, *host.estg.ipv6* e *server.estg.ipv6*. Um cenário em mais detalhe é explicado na Figura 4.21.

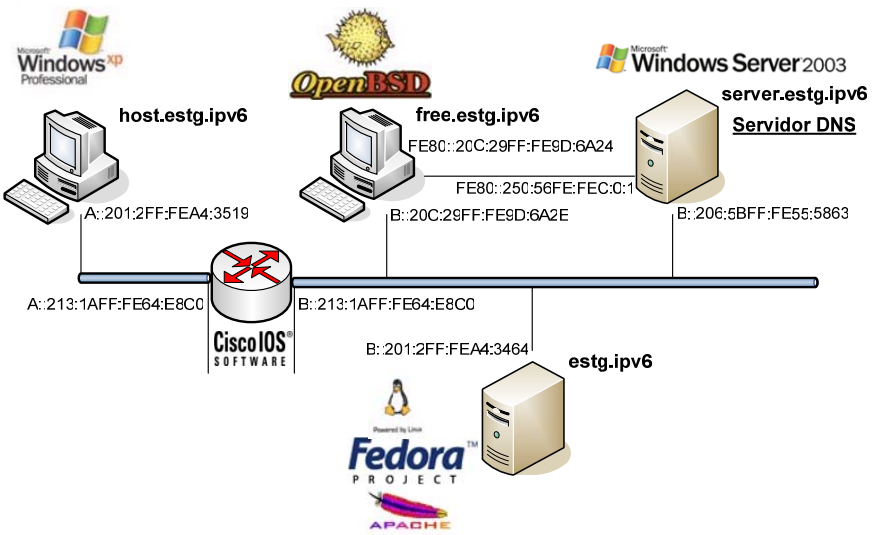


Figura 4.21 – Cenário 1 com servidor Apache e DNS.

2.º Passo: Configurou-se o servidor de DNS para aceitar pedidos IPv6

Para isso foi necessário recorrer à instalação do *package Windows Support Tools* existente no CD do Windows Server 2003, e proceder à execução do comando "`dnscmd /config /EnableIPv6`" (ver [132]). O processo de uma correcta execução é exemplificado na Figura 4.22.

```
F:\Program Files\Support Tools>dnscmd /config /EnableIPv6 1
Registry property EnableIPv6 successfully reset.
Command completed successfully.
```

Figura 4.22 – Comando para que o servidor de DNS suporte pedidos IPv6.

3.º Passo: Configurou-se em todos os clientes o endereço do servidor de DNS

- **Windows Server 2003**

O Windows Server 2003, funcionará como cliente do seu próprio servidor. Para adicionar o servidor de DNS foi necessário executar o comando "`netsh interface ipv6 add dns`". Neste caso o servidor será a própria máquina cliente, por isso será adicionado o endereço de *loopback* (ver Subsecção 2.4.3). O exemplo da sua configuração é mostrado na Figura 4.23.

```
netsh interface ipv6>add dns "Local Area Connection" ::1
Ok.
```

Figura 4.23 – Adicionar servidor de DNS no Windows Server 2003.

Após a inserção do servidor, verificou-se se depois, de efectuar o comando "`ipconfig /all`", surge a linha associada ao servidor configurado. A Figura 4.24 mostra o aspecto dessa linha.

```
DNS Servers . . . . . . . . . . . : ::1
```

Figura 4.24 – Verificação do servidor de DNS no Windows Server 2003.

- **Windows XP**

No Windows XP executou-se exactamente o mesmo comando, só que com o endereço IPv6 do servidor, como é exemplificado na Figura 4.25.

```
netsh interface ipv6>add dns "Local Area Connection" b::206:5bff:fe55:5863
Ok.
```

Figura 4.25 – Adicionar servidor de DNS no Windows XP.

A linha correspondente ao servidor de DNS, após o comando "`ipconfig /all`", é mostrada na Figura 4.26.

```
DNS Servers . . . . . . . . . . . : b::206:5bff:fe55:5863
```

Figura 4.26 – Verificação do servidor de DNS no Windows XP.

- **Linux**

O servidor pode ser adicionado numa interface gráfica, ou acedendo ao ficheiro */etc/resolv.conf*. No último caso, após o "*nameserver*" adiciona-se o endereço IPv6 do servidor de DNS. As duas formas de proceder à especificação do servidor de DNS estão exemplificadas na Figura 4.27.

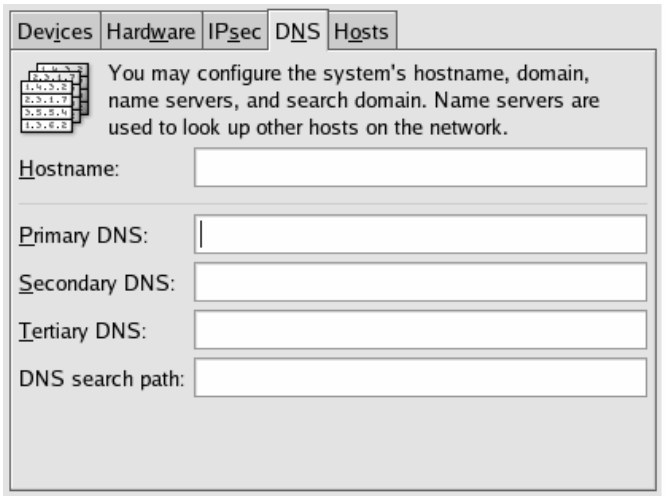


Figura 4.27 – Formas de adicionar o servidor de DNS no Linux (interface gráfica e ficheiro).

- **OpenBSD**

O adicionar de um servidor de DNS no sistema OpenBSD é exactamente igual ao do Linux. Porém, no nosso caso, o OpenBSD testado não possuía interface gráfica, pelo que foi configurado directamente no ficheiro. Com esta variante do BSD, o ficheiro */etc/resolv.conf* não existia, tendo sido necessário criá-lo e proceder à inserção do endereço do servidor. Como se pode ver mais à frente, apesar de ter sido configurado, o endereço IPv6 não foi aceite.

4.º Passo: Verificação de todo o correcto funcionamento do serviço DNS

Procedeu-se então a todos os testes através do utilitário *ping*, ou *ping6*, e de pedidos HTTP feitos pelos *browsers*.

- **Windows Server 2003 (cliente DNS)**

O primeiro a ser testado foi o próprio servidor, num processo de pedir resolução de endereços a si mesmo. O resultado está expresso na Figura 4.28, utilizando o Internet Explorer para aceder ao servidor Apache.

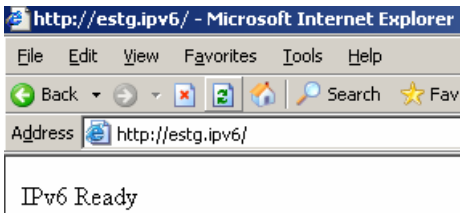


Figura 4.28 – Acesso *web* ao servidor Apache utilizando DNS, com o *browser* Internet Explorer.

Usando DNS, o Internet Explorer é completamente funcional para endereços IPv6, só existe problema no caso de o endereço ser posto na sua forma literal (utilizando a sua notação *column hexadecimal*).

Na Figura 4.29 apresenta-se o resultado de um *ping* feito ao terminal Windows XP, com o nome de domínio *host.estg.ipv6*.

```
F:\Documents and Settings\Administrator>ping -6 host.estg.ipv6

Pinging host.estg.ipv6 [a::201:2ff:fea4:3519] from b::206:5bff:fe55:5863 with 32
 bytes of data:

Reply fron a::201:2ff:fea4:3519: time=1ms
Reply fron a::201:2ff:fea4:3519: time=1ms
Reply fron a::201:2ff:fea4:3519: time=1ms
Reply fron a::201:2ff:fea4:3519: time=1ms

Ping statistics for a::201:2ff:fea4:3519:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 1ms, Average = 1ms
```

Figura 4.29 – *Ping* utilizando DNS, ao *host.estg.ipv6* (Windows XP).

Foi especificada a opção `"-6"` que tem como objectivo, obrigar o pacote a usar tudo o que seja endereços IPv6 na sua rota. Se não se especificar, não existe servidor de DNS com endereço IPv4 definido e a resposta vai demorar uns certos segundos a aparecer. Este tempo de espera acontece porque ele tenta sempre optar por IPv4 e demora até se decidir pelo IPv6.

- **Windows XP**

Não é possível efectuar qualquer pedido DNS em IPv6 nesta versão do Windows. O Windows XP, apesar de permitir configurar um servidor de DNS em IPv6, não chega sequer a enviar o pedido IPv6 para a rede. Isto acontece, devido ao facto de o sistema operativo não estar sequer configurado para reconhecer endereços IPv6 neste tipo de pedidos.

Sendo assim, a única forma funcional de recorrer ao DNS para resolução de nomes de endereços IPv6, é pedir ao servidor em IPv4.

Se não for configurado qualquer servidor de DNS em IPv6, ao fazer `"ipconfig /all"` irá reparar-se que existem três servidores de DNS com endereços IPv6, como é visualizado na Figura 4.30.

```
DNS Servers . . . . . . . . . . . : fec0:0:0:ffff::1%1
                                    fec0:0:0:ffff::2%1
                                    fec0:0:0:ffff::3%1
```

Figura 4.30 – Os três servidores de DNS, com endereços IPv6 predefinidos pelo Windows.

Estes três servidores utilizam ainda os obsoletos endereços *site-local* (ver Subsecção 3.3.4.1). De qualquer forma, a ideia desta auto-configuração implementada pela Microsoft, poderá revelar-se bastante útil. A utilidade desta implementação é que basta atribuir um daqueles três endereços ao servidor para que os clientes estejam automaticamente configurados com o seu servidor de DNS.

- **OpenBSD**

A situação revelou-se idêntica ao Windows XP, mas neste caso nem sequer chegou a aceitar o endereço IPv6 no ficheiro */etc/resolv.conf*.

Na Figura 4.31 são demonstrados os dois possíveis funcionamentos do processo pedido/resposta para o Windows XP e OpenBSD.

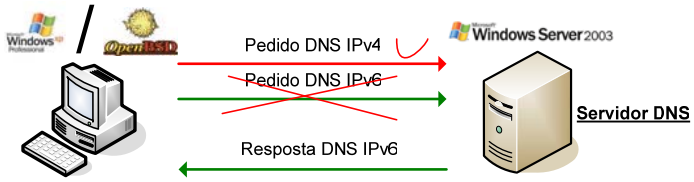


Figura 4.31 – Funcionamentos possíveis de pedidos DNS do Windows XP e OpenBSD.

Existem alguns problemas nestes pedidos e respostas que podem suceder (ver [86]). Há que ter algum cuidado, quando se juntam configurações IPv4 com IPv6 na configuração do DNS (ver [83]).

- **Linux**

Com Linux o processo pedido/resposta foi conseguido em IPv6 como exemplifica a Figura 4.32.

```
[root@localhost etc]# ping6 server.estg.ipv6
PING server.estg.ipv6(b::206:5bff:fe55:5863) 56 data bytes
64 bytes from b::206:5bff:fe55:5863: icmp_seq=0 ttl=64 time=0.230 ms
64 bytes from b::206:5bff:fe55:5863: icmp_seq=1 ttl=64 time=0.213 ms
64 bytes from b::206:5bff:fe55:5863: icmp_seq=2 ttl=64 time=0.219 ms
64 bytes from b::206:5bff:fe55:5863: icmp_seq=3 ttl=64 time=0.217 ms
```

Figura 4.32 – *Ping* utilizando DNS, efectuado ao *server.estg.ipv6* (Windows Server 2003)

Existem alguns outros cuidados a ter para verificar o correcto funcionamento IPv6, como o confirmar que não há cache DNS. Em Windows, o comando a usar será o `"ipconfig /flushdns"`, no Linux e OpenBSD basta garantir que o serviço "*nscd*" não é executado.

Outros servidores de DNS

O BIND é sem dúvida o mais popular de todos os servidores de DNS. O suporte IPv6 começou com a versão 8, mas de forma nativa só existe a partir da versão 9. Por ser tão popular, existem diversos manuais na Internet a explicar todo o procedimento a efectuar na configuração do servidor para IPv6 (ver [20] e [21]). Ao contrário do DNS no Windows Server 2003, existe suporte para o *resource record A6*.

A razão de ter sido escolhido o servidor de DNS do Windows Server 2003 foi exactamente por existirem poucas implementações IPv6 neste tipo de servidor.

4.2.3. Serviço DHCPv6

O serviço DHCP é um tipo de serviço bastante usado no IPv4, mas em relação ao IPv6 actualmente as suas utilizações são escassas. As razões deste fraco uso, podem ser devido ao processo de auto-

configuração *stateless* (ver Subsecção 3.8.1), ou também devido à falta de implementação de DHCPv6. Não existe qualquer serviço DHCPv6 disponibilizado pela Microsoft, nem existe nenhum servidor deste tipo incorporado de forma nativa em Linux ou OpenBSD. Sendo assim, foi utilizado uma implementação de um servidor e cliente feito numa faculdade polaca (ver [177]), o Dibbler (ver [165]).

1.º Passo: Instalou-se o Dibbler 0.4.0

- **Windows Server 2003**

A instalação do Dibbler no Windows Server 2003 é bastante simples, existindo uma interface gráfica para todo o procedimento de instalação.

- **Windows XP**

Mesma situação do Windows Server 2003.

- **Linux**

Pode-se instalar de três formas: a partir das fontes de código, a partir de um ficheiro *.tar*, ou a partir de um RPM.

Em todos os casos pode-se optar por instalar separadamente, servidor, cliente ou *relay agent*. Neste caso foram instalados servidor e cliente no Windows Server 2003 e clientes no Windows XP e Linux.

2.º Passo: Instalou-se servidor e respectivos clientes

Foram configurados os ficheiros *server.conf* e *client.conf*, que estão ambos na directoria onde se instalou o Dibbler. As configurações efectuadas são bastante simples (ver Anexos A.1.2, A.1.3 e A.1.4).

3.º Passo: Garantiu-se que o servidor e os clientes estão a ser executados

- **Windows Server 2003**

Acedeu-se no Menu Iniciar a *Dibbler* > *Server Install as service* como é exemplificado na Figura 4.33. A instalação do servidor como serviço está concluída.

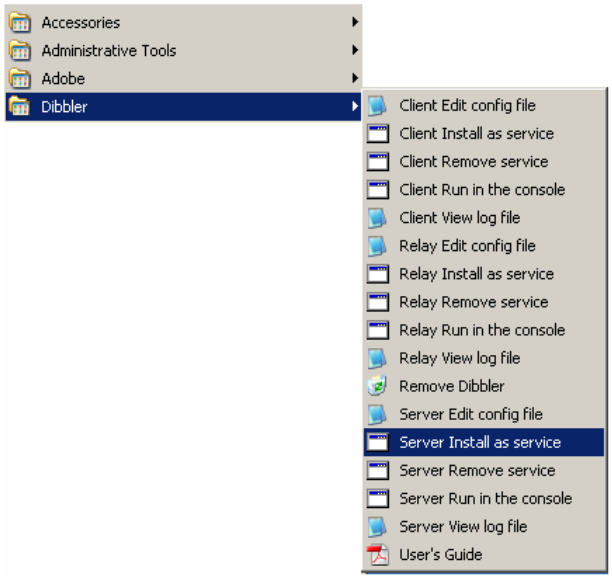


Figura 4.33 – Instalação do Dibbler como serviço.

De seguida fez-se o mesmo procedimento, mas agora para o cliente, *Dibbler > Client Install as service*.

Por fim verificou-se se o cliente e o servidor estão a ser correctamente executados. A Figura 4.34 exemplifica como devem estar as linhas associadas a ambos os serviços.

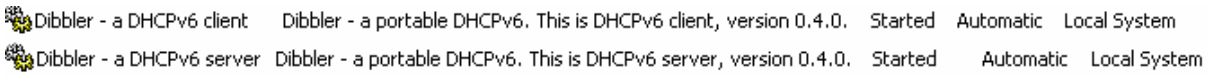


Figura 4.34 – Os serviços do cliente e servidor Dibbler no Windows Server 2003.

- **Windows XP**

O mesmo procedimento só que apenas para o cliente.

- **Linux**

O Dibbler no Linux não possui interface gráfica para a instalação e execução dos serviços (*daemons*). A forma utilizada foi usar o comando "`dibbler-client start`" na directoria onde foi instalado o Dibbler. O resultado desse comando está exposto na Figura 4.35.

```
[root@localhost dibbler]# ./dibbler-client start
| Dibbler - a portable DHCPv6, version 0.4.0(CLIENT)
| Authors : Tomasz Mrugalski<thomson(at)klub.com.pl>
.pl>
| Licence : GNU GPL v2 or later. Developed at Gdansk
| Homepage: http://klub.com.pl/dhcpv6/
Starting daemon...
```

Figura 4.35 – Execução do Dibbler no Linux.

Após executar o comando, procedeu-se à verificação do seu estado. Isso conseguiu-se através do comando "`dibbler-client-status`" e o resultado que se pretende está demonstrado na Figura 4.36.

```
[root@localhost dibbler]# ./dibbler-client status
| Dibbler - a portable DHCPv6, version 0.4.0(CLIENT)
| Authors : Tomasz Mrugalski<thomson(at)klub.com.pl>
.pl>
| Licence : GNU GPL v2 or later. Developed at Gdansk
| Homepage: http://klub.com.pl/dhcpv6/
Dibbler server: NOT RUNNING.
Dibbler client: RUNNING, pid=9099
Dibbler relay : NOT RUNNING.
```

Figura 4.36 – Estado dos diversos processos (cliente, servidor e *relay agent*) no Linux.

Convinha que o comando fosse executado automaticamente quando o sistema se inicia, pelo que foi inserido no ficheiro */etc/rc.local*. A Figura 4.37 mostra esse ficheiro.

```
[root@localhost etc]# cat rc.local
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local
/etc/dibbler/dibbler-client start
```

Figura 4.37 – Inserção do comando de execução do Dibbler no ficheiro de inicialização.

4.º Passo: Verificou-se os endereços dos diversos terminais clientes

O cenário que se pretende está exposto na Figura 4.38. A *pool* de endereços do servidor estende-se do endereço *CAFE::1* a *CAFE::FFFF*, e os clientes pedem os endereços de forma aleatória.

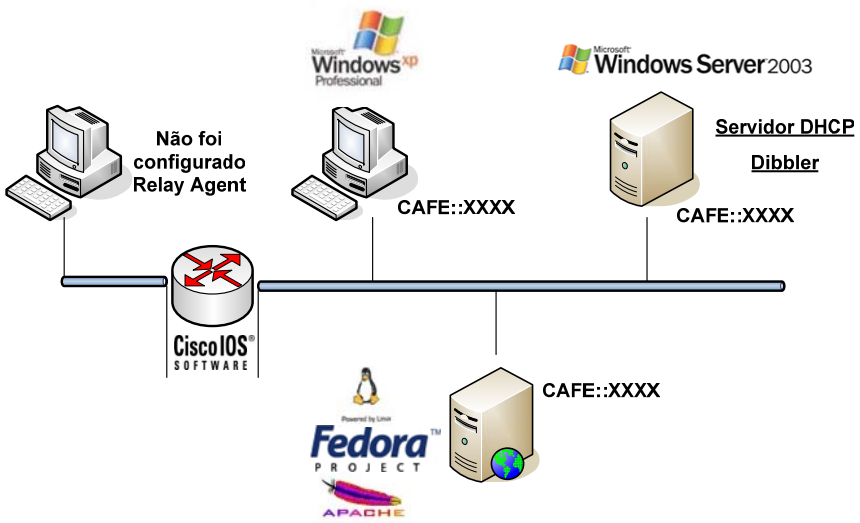


Figura 4.38 – Cenário 1 com servidor de DHCP.

- **Windows XP**

Foi configurado correctamente um dos endereços da *pool* especificada. Pode ser visualizado o sucesso na Figura 4.39.

```
Interface 4: Local Area Connection

Addr Type  DAD State  Valid Life   Pref. Life   Address
---------  ---------- ------------ ------------ ----------------------------
Manual     Preferred        48m37s       18m37s cafe::26d6
Link       Preferred      infinite     infinite fe80::201:2ff:fea4:3464
```

Figura 4.39 – Configuração dos endereços com DHCPv6 no Windows XP

O tipo deste endereço é considerado manual.

- **Windows Server 2003**

Também foi configurado correctamente um endereço IPv6 por DHCPv6 neste sistema operativo.

- **Linux**

Verificou-se também configurado correctamente, como explicita a Figura 4.40.

```
[root@localhost dibbler]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:01:02:A4
          inet addr:192.168.0.2  Bcast:192.168.0.
          inet6 addr: cafe::9148/128 Scope:Global
```

Figura 4.40 – Configuração dos endereços com DHCPv6 no Linux.

Na Figura 4.41 é mostrada a captura da mensagem DHCPv6 *Reply*. Esta mensagem é a responsável pela configuração final no processo de negociação de endereços. Pode-se verificar os campos da *Identify Association*, nomeadamente o *IA Address* > *IPv6 address*: *CAFE::9148* e o tempo de vida válido e preferido, predefinidos do endereço. Nota-se também através do *T1: 5*, que neste caso, de cinco em cinco segundos o cliente vai comunicar ao servidor para lhe enviar novas configurações através da mensagem DHCPv6 *Renew*. Por fim surge o *Status* a indicar o sucesso da negociação.

```
    7 63.270841 fe80::206:5bff:fe5 fe80::201:2ff:fea4 DHCPv6 Reply
▽ Identify Association
    option type: 3
    option length: 74
    IAID: 2
    T1: 5
    T2: 10
  ▽ IA Address
      option type: 5
      option length: 24
      IPv6 address: cafe::9148
      Preferred lifetime: 1800
      Valid lifetime: 3600
  ▽ Status code
      option type: 13
      option length: 30
      Status Code: Success (0)
      Status Message: All addresses were assigned.
```

Figura 4.41 – Captura do pacote, mais especificamente a IA da DHCPv6 Reply.

O Dibbler já tem diversos mecanismos operacionais praticamente a 100%, porém, ainda existem alguns *bugs* tanto no cliente, como no servidor e *relay agent* (ver [166], [167] e [168]). Apesar destes problemas, o projecto tenta já caminhar para outros sistemas operativos, nunca esquecendo claro a resolução de todos os *bugs* que infelizmente vão aparecendo.

Outros servidores de DHCPv6

A única implementação comercial deste tipo de servidor é uma implementação por parte da Cisco que está bastante completa (ver [137]), e é suportada a partir da versão 12.3(4)T do IOS. É também de referir o DHCPv6 alojado na SourceForge (ver [169]), que como todos os projectos da SourceForce é *open-source*. Este servidor é multi-plataforma só que a sua versão mais recente já data de 2003. Infelizmente devido ao facto da nossa versão do IOS não suportar DHCP, a escolha recaiu no Dibbler, por ser o mais actualizado entre os dois projectos *open-source*.

4.3. Cenário 2: Encaminhamento

Como já se viu, o encaminhamento é essencial numa rede, seja ela IPv4 ou IPv6 (ver Secção 3.9), pelo que era essencial neste projecto a implementação de cenários de teste que envolvessem protocolos de encaminhamento.

Na Figura 4.42 apresenta-se a rede de testes IPv6 utilizada para a configuração dos protocolos de encaminhamento RIP e OSPF. Foram testados o RIP e o OSPF, nas suas extensões para IPv6, por serem os dois protocolos de encaminhamento interiores mais utilizados em redes IP, e por serem suportados pela maioria dos *routers*. O primeiro é um protocolo de encaminhamento *distance vector* e o segundo *link state*.

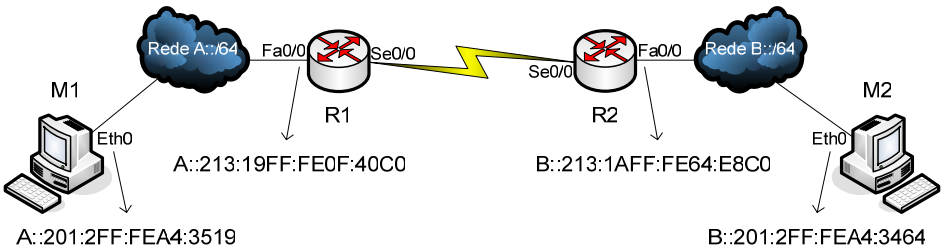


Figura 4.42 – Rede utilizada para a configuração dos protocolos de encaminhamento RIP e OSPF.

Na rede apresentada são utilizados dois *routers* Cisco IOS (R1 e R2), e duas máquinas, uma Microsoft Windows XP SP2 (M1) e uma Linux com a distribuição Fedora Core 3 (M2), com suporte IPv6 instalado e activo.

Começou por se configurar os dois *routers* para que estes anunciassem prefixos para as redes e as máquinas se auto-configurassem. Desta forma assegura-se a comunicação entre as máquinas e os *routers* de cada rede. Na Figura 4.43 e na Figura 4.44 apresentam-se, respectivamente, as configurações efectuadas no *router* R1 e no *router* R2.

```
R1(config)#ipv6 unicast-routing
R1(config)#
R1(config)#interface FastEthernet0/0
R1(config-if)#ipv6 enable
R1(config-if)#ipv6 address A::/64 eui-64
R1(config-if)#
R1(config-if)#exit
R1(config)#
R1(config)#interface Serial0/0
R1(config-if)#ipv6 enable
R1(config-if)#clockrate 115200
R1(config-if)#
R1(config-if)#exit
R1(config)#
```

Figura 4.43 – Configuração das interfaces do *router* R1.

De notar nas duas configurações o comando "`ipv6 unicast-routing`" necessário para activar o encaminhamento de pacotes IPv6 nos *routers*. Importante também é a configuração dos prefixos nas interfaces *FastEthernet*, através do comando "`ipv6 address`", e da indicação que os endereços IPv6 devem ser criados a partir da norma EUI-64.

```
R2(config)#ipv6 unicast-routing
R2(config)#
R2(config)#interface FastEthernet0/0
R2(config-if)#ipv6 enable
R2(config-if)#ipv6 address B::/64 eui-64
R2(config-if)#
R2(config-if)#exit
R2(config)#
R2(config)#interface Serial0/0
R2(config-if)#ipv6 enable
R2(config-if)#clockrate 115200
R2(config-if)#
R2(config-if)#exit
R2(config)#
```

Figura 4.44 – Configuração das interfaces do *router* R2.

Não é necessária qualquer configuração nas máquinas, pois estas auto-configuram os seus endereços com base nos prefixos divulgados pelos *routers*. Por esta altura ambos os *routers* e ambas as máquinas do cenário possuem configurados os endereços IPv6 apresentados na Figura 4.42. Para comprovar a conectividade entre a máquina M1 e o *router* R1, e a máquina M2 e o *router* R2 utilizou-se o *ping6* em ambas as máquinas. Na Figura 4.45 pode-se ver o resultado do *ping6* entre a máquina M1 (*A::201:2FF:FEA4:3519*) e o *router* R1 (*A::213:19FF:FE0F:40C0*), prova da conectividade entre os dois nós.

```
C:\>ping6 a::213:19ff:fe0f:40c0

Pinging a::213:19ff:fe0f:40c0
from a::201:2ff:fea4:3519  with 32 bytes of data:

Reply from a::213:19ff:fe0f:40c0: bytes=32 time<1ms
Reply from a::213:19ff:fe0f:40c0: bytes=32 time<1ms
Reply from a::213:19ff:fe0f:40c0: bytes=32 time<1ms
Reply from a::213:19ff:fe0f:40c0: bytes=32 time<1ms

Ping statistics for a::213:19ff:fe0f:40c0:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figura 4.45 – Resultado do *ping6* entre a máquina M1 e o *router* R1.

Na Figura 4.46 pode-se ver o resultado do *ping6* entre a máquina M2 (*B::201:2FF:FEA4:3464*) e o *router* R2 (*B::213:1AFF:FE64:E8C0*).

```
[root@localhost /]# ping6 -c 4 b::213:1aff:fe64:e8c0
PING b::213:1aff:fe64:e8c0(b::213:1aff:fe64:e8c0) 56 data bytes
64 bytes from b::213:1aff:fe64:e8c0: icmp_seq=0 ttl=64 time=1.59 ms
64 bytes from b::213:1aff:fe64:e8c0: icmp_seq=1 ttl=64 time=0.702 ms
64 bytes from b::213:1aff:fe64:e8c0: icmp_seq=2 ttl=64 time=0.670 ms
64 bytes from b::213:1aff:fe64:e8c0: icmp_seq=3 ttl=64 time=0.737 ms

--- b::213:1aff:fe64:e8c0 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.670/0.926/1.597/0.389 ms, pipe 2
```

Figura 4.46 – Resultado do *ping6* entre a máquina M2 e o *router* R2.

Com as configurações efectuadas, cada *router* apenas conhece as suas redes, pelo que não possui informação de como chegar às outras redes. Não é assim possível a comunicação entre máquinas de diferentes redes. Como se pode ver na tabela de encaminhamento do *router* R1 (Figura 4.47), este apenas conhece a sua rede *A::/64*.

```
R1#sh ipv6 route
IPv6 Routing Table - 4 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
C   A::/64 [0/0]
     via ::, FastEthernet0/0
L   A::213:19FF:FE0F:40C0/128 [0/0]
     via ::, FastEthernet0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

Figura 4.47 – Tabela de encaminhamento do *router* R1.

O mesmo acontece com o *router* R2 que só conhece a sua rede *B::/64*. Na Figura 4.48 pode ver-se a tabela de encaminhamento deste *router*.

```
R2#sh ipv6 route
IPv6 Routing Table - 4 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
C   B::/64 [0/0]
     via ::, FastEthernet0/0
L   B::213:1AFF:FE64:E8C0/128 [0/0]
     via ::, FastEthernet0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

Figura 4.48 – Tabela de encaminhamento do *router* R2.

Para provar que realmente as duas máquinas não conseguem comunicar efectuou-se um *ping6* entre elas. Na Figura 4.49 está o resultado do *ping6* sem sucesso entre a máquina M1 (*A::201:2FF:FEA4:3519*) e a máquina M2 (*B::201:2FF:FEA4:3464*). As respostas recebidas são do *router* R1 (*A::213:19FF:FE0F:40C0*) indicando que não sabe como chegar ao destino.

```
C:\>ping6 b::201:2ff:fea4:3464

Pinging b::201:2ff:fea4:3464
from a::201:2ff:fea4:3519  with 32 bytes of data:

Reply from a::213:19ff:fe0f:40c0: No route to destination.
Reply from a::213:19ff:fe0f:40c0: No route to destination.
Reply from a::213:19ff:fe0f:40c0: No route to destination.
Reply from a::213:19ff:fe0f:40c0: No route to destination.

Ping statistics for b::201:2ff:fea4:3464:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figura 4.49 – Resultado do *ping6* sem sucesso entre a máquina M1 e a máquina M2.

Na Figura 4.50 apresenta-se o *ping6* em sentido contrário, da máquina M2 (*B::201:2FF:FEA4:3464*) para a máquina M1 (*A::201:2FF:FEA4:3519*). As respostas recebidas são do *router* R2 (*B::213:1AFF:FE64:E8C0*) informando que não tem informação de como chegar àquele endereço destino.

```
[root@localhost /]# ping6 -c 4 a::201:2ff:fea4:3519
PING a::201:2ff:fea4:3519(a::201:2ff:fea4:3519) 56 data bytes
From b::213:1aff:fe64:e8c0 icmp_seq=0 Destination unreachable: No route
From b::213:1aff:fe64:e8c0 icmp_seq=1 Destination unreachable: No route
From b::213:1aff:fe64:e8c0 icmp_seq=2 Destination unreachable: No route
From b::213:1aff:fe64:e8c0 icmp_seq=3 Destination unreachable: No route

--- a::201:2ff:fea4:3519 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3000ms
```

Figura 4.50 – Resultado do *ping6* sem sucesso entre a máquina M2 e a máquina M1.

4.3.1. OSPF

Para que as duas máquinas das duas redes diferentes possam trocar pacotes entre si, é necessário que ambos os *routers* corram um protocolo de encaminhamento. Na Figura 4.51 apresentam-se as configurações necessárias para activar o OSPF no *router* R1. Na Figura 4.52 apresentam-se as mesmas configurações mas para activar o OSPF no *router* R2.

```
R1(config)#ipv6 router ospf 1
R1(config-rtr)#router-id 0.0.0.1
R1(config-rtr)#
R1(config-rtr)#exit
R1(config)#
R1(config)#interface FastEthernet0/0
R1(config-if)#ipv6 ospf 1 area 0
R1(config-if)#
R1(config-if)#exit
R1(config)#
R1(config)#interface Serial0/0
R1(config-if)#ipv6 ospf 1 area 0
R1(config-if)#
R1(config-if)#exit
R1(config)#
```

Figura 4.51 – Configurações para activar o OSPF no *router* R1.

No comando `"ipv6 router ospf"`, o `"1"` serve para identificar este processo OSPF. O comando `"router-id"` serve para especificar um número de 32 *bits*, no formato de um endereço IPv4, que é a identificação do *router* para efeitos de convergência com outros *routers*. Com IPv4 não é necessário especificar o *router-id* pois o *router* obtém-o a partir de um endereço IP configurado. De notar que é necessário activar o OSPF em todas as interfaces das quais e pelas quais se querem anunciar redes.

```
R2(config)#ipv6 router ospf 1
R2(config-rtr)#router-id 0.0.0.2
R2(config-rtr)#
R2(config-rtr)#exit
R2(config)#
R2(config)#interface FastEthernet0/0
R2(config-if)#ipv6 ospf 1 area 0
R2(config-if)#
R2(config-if)#exit
R2(config)#
R2(config)#interface Serial0/0
R2(config-if)#ipv6 ospf 1 area 0
R2(config-if)#
R2(config-if)#exit
R2(config)#
```

Figura 4.52 – Configurações para activar o OSPF no *router* R2.

Depois de configurar e activar o OSPF nos dois *routers*, já se podem ver alterações nas tabelas de encaminhamento. Na Figura 4.53 e na Figura 4.54 podem ver-se as tabelas de encaminhamento dos *routers* R1 e R2 após configuração do OSPF.

```
R1#sh ipv6 route
IPv6 Routing Table - 5 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
C   A::/64 [0/0]
     via ::, FastEthernet0/0
L   A::213:19FF:FE0F:40C0/128 [0/0]
     via ::, FastEthernet0/0
O   B::/64 [110/782]
     via FE80::213:1AFF:FE64:E8C0, Serial0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

Figura 4.53 – Tabela de encaminhamento do *router* R1 após configuração do OSPF.

Nas tabelas de encaminhamento dos *routers* já se podem ver as rotas adquiridas através de OSPF, aquelas marcadas com "O" antes do prefixo da rede a atingir.

```
R2#sh ipv6 route
IPv6 Routing Table - 5 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
O   A::/64 [110/782]
     via FE80::213:19FF:FE0F:40C0, Serial0/0
C   B::/64 [0/0]
     via ::, FastEthernet0/0
L   B::213:1AFF:FE64:E8C0/128 [0/0]
     via ::, FastEthernet0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

Figura 4.54 – Tabela de encaminhamento do *router* R2 após configuração do OSPF.

Neste momento a máquina M1 já pode comunicar com a máquina M2 e vice-versa, pois o *router* R1 já possui informação da rede *B::/64* do *router* R2, e o *router* R2 já possui informação da rede *A::/64* do *router* R1. Na Figura 4.55 é mostrado o resultado de um *ping6* com sucesso da máquina M1 (*A::201:2FF:FEA4:3519*) para a máquina M2 (*B::201:2FF:FEA4:3464*).

```
C:\>ping6 b::201:2ff:fea4:3464

Pinging b::201:2ff:fea4:3464
from a::201:2ff:fea4:3519  with 32 bytes of data:

Reply from b::201:2ff:fea4:3464: bytes=32 time=14ms
Reply from b::201:2ff:fea4:3464: bytes=32 time=14ms
Reply from b::201:2ff:fea4:3464: bytes=32 time=14ms
Reply from b::201:2ff:fea4:3464: bytes=32 time=14ms

Ping statistics for b::201:2ff:fea4:3464:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 14ms, Maximum = 14ms, Average = 14ms
```

Figura 4.55 – Resultado de um *ping6* com sucesso da máquina M1 para a máquina M2.

Na Figura 4.56 apresenta-se o resultado de um *ping6* com sucesso da máquina M2 (*B::201:2FF:FEA4:3464*) para a máquina M1 (*A::201:2FF:FEA4:3519*), prova de que ambas as máquinas conseguem realmente comunicar.

```
[root@localhost /]# ping6 -c 4 a::201:2ff:fea4:3519
PING a::201:2ff:fea4:3519(a::201:2ff:fea4:3519) 56 data bytes
64 bytes from a::201:2ff:fea4:3519: icmp_seq=0 ttl=62 time=18.8 ms
64 bytes from a::201:2ff:fea4:3519: icmp_seq=1 ttl=62 time=17.8 ms
64 bytes from a::201:2ff:fea4:3519: icmp_seq=2 ttl=62 time=22.5 ms
64 bytes from a::201:2ff:fea4:3519: icmp_seq=3 ttl=62 time=17.7 ms

--- a::201:2ff:fea4:3519 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 17.767/19.253/22.584/1.975 ms, pipe 2
```

Figura 4.56 – Resultado de um *ping6* com sucesso da máquina M2 para a máquina M1.

Em anexo sequem as configurações dos dois *routers* do cenário com o protocolo de encaminhamento OSPF configurado (ver Anexos A.2.1 e A.2.2).

4.3.2. RIP

Em relação ao protocolo de encaminhamento RIP apenas são necessários dois comandos para o activar: o comando `"ipv6 router rip "nome""` no contexto de configuração global, em que o argumento `"nome"` serve para identificar o processo do RIP no *router*; e o comando `"ipv6 rip "nome" enable"` no contexto de configuração de cada interface onde se quer activar o encaminhamento com RIP. Na Figura 4.57 é apresentada esta configuração efectuada no *router* R1, e na Figura 4.58 a mesma configuração mas no *router* R2.

```
R1(config)#ipv6 router rip teste-ripng
R1(config-rtr)#
R1(config-rtr)#exit
R1(config)#
R1(config)#interface FastEthernet0/0
R1(config-if)#ipv6 rip teste-ripng enable
R1(config-if)#
R1(config-if)#exit
R1(config)#
R1(config)#interface Serial0/0
R1(config-if)#ipv6 rip teste-ripng enable
R1(config-if)#
R1(config-if)#exit
R1(config)#
```

Figura 4.57 – Configurações para activar o RIP no *router* R1.

Neste exemplo usou-se o nome `"teste-ripng"` para identificar o processo do protocolo de encaminhamento RIP nos dois *routers*, mas este nome não precisa ser igual nos dois *routers*.

```
R2(config)#ipv6 router rip teste-ripng
R2(config-rtr)#
R2(config-rtr)#exit
R2(config)#
R2(config)#interface FastEthernet0/0
R2(config-if)#ipv6 rip teste-ripng enable
R2(config-if)#
R2(config-if)#exit
R2(config)#
R2(config)#interface Serial0/0
R2(config-if)#ipv6 rip teste-ripng enable
R2(config-if)#
R2(config-if)#exit
R2(config)#
```

Figura 4.58 – Configurações para activar o RIP no *router* R2.

Na Figura 4.59 pode ver-se a tabela de encaminhamento do *router* R1 com o RIP já configurado. Nesta tabela já consta a informação de como chegar à rede *B::/64*.

```
R1#sh ipv6 route
IPv6 Routing Table - 5 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
C   A::/64 [0/0]
     via ::, FastEthernet0/0
L   A::213:19FF:FE0F:40C0/128 [0/0]
     via ::, FastEthernet0/0
R   B::/64 [120/2]
     via FE80::213:1AFF:FE64:E8C0, Serial0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

Figura 4.59 – Tabela de encaminhamento do *router* R1 após configuração do RIP.

A tabela de encaminhamento do *router* R2 é apresentada na Figura 4.60. Pode ver-se que este *router* já sabe como chegar à rede *A::/64* pela presença de uma rota na tabela com este prefixo. Antes do prefixo pode ver-se a letra "R", indicadora de que esta rota foi aprendida através do RIP.

```
R2#sh ipv6 route
IPv6 Routing Table - 5 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   A::/64 [120/2]
     via FE80::213:19FF:FE0F:40C0, Serial0/0
C   B::/64 [0/0]
     via ::, FastEthernet0/0
L   B::213:1AFF:FE64:E8C0/128 [0/0]
     via ::, FastEthernet0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

Figura 4.60 – Tabela de encaminhamento do *router* R2 após configuração do RIP.

Em anexo podem ver-se todas as configurações dos dois *routers* com o protocolo de encaminhamento RIP configurado (ver Anexos A.2.3 e A.2.4).

4.4. Cenário 3: Acesso ao exterior por túnel

Actualmente, um dos elementos essenciais de uma rede de computadores é o acesso ao exterior. Considerando uma rede IPv6 e tendo em conta a infra-estrutura de *backbone* existente, existem duas formas principais de proporcionar este acesso: utilizando um túnel IPv6 sobre IPv4 (ver Subsecção 3.10.2) e de forma IPv6 nativa. Como já foi visto anteriormente, os túneis IPv6 sobre IPv4 são uma solução muito útil na interligação de redes IPv6 através da infra-estrutura IPv4. Nesta parte do trabalho apresenta-se o caminho percorrido até ao estabelecimento do túnel com a FCCN para acesso ao *backbone* IPv6.

4.4.1. Túnel com terminal

Foi estabelecido inicialmente, para teste do acesso exterior, um túnel *Host-to-Router* a partir de um terminal Microsoft Windows XP SP2 Professional. No diagrama de rede da Figura 4.61 pode ver-se como, nesta fase inicial, foi estabelecida a ligação ao exterior. Podem ver-se no diagrama os principais dispositivos e endereços IPv4 e IPv6 intervenientes no estabelecimento do túnel.

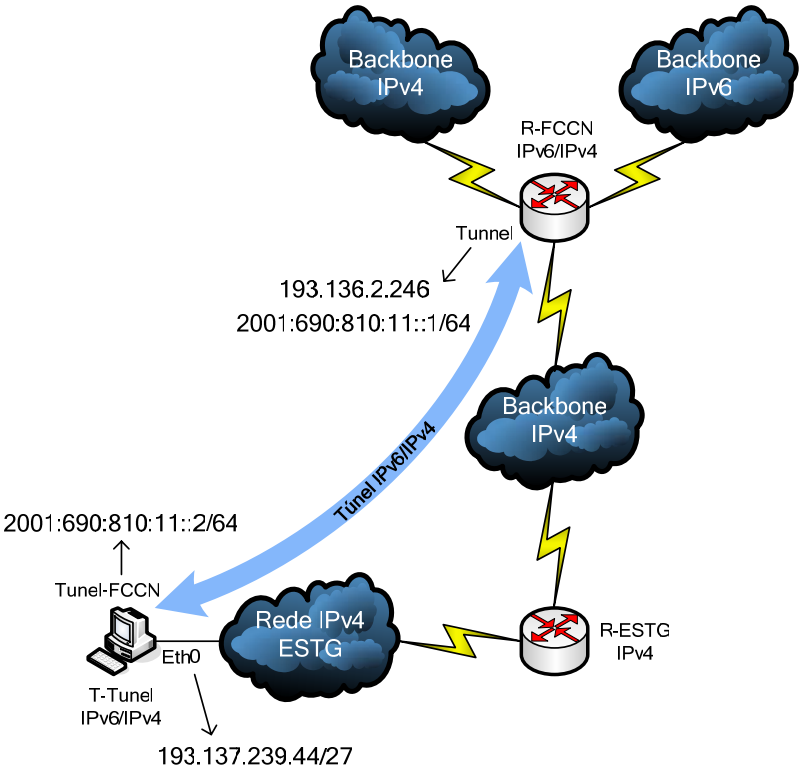


Figura 4.61 – Cenário de acesso ao exterior por túnel IPv6 sobre IPv4.

O túnel foi estabelecido entre o endereço IPv4 *193.137.239.44* do lado da ESTG-Leiria (terminal T-Tunel), e o endereço IPv4 *193.136.2.246* do lado da FCCN (*router* R-FCCN). Para além do endereço IP *193.137.239.44/27*, o terminal onde foi estabelecido o túnel possuía configurado o *default gateway* com o IP *193.137.239.62* e o servidor de DNS *193.137.239.226* da ESTG-Leiria. O IP utilizado público na ligação IPv4 é de uma ligação directa a um *router* da ESTG, com ligação directa à Internet.

Foi necessário, para a comunicação IPv6 para o exterior, que fosse permitido o trânsito de pacotes IPv6 encapsulados em IPv4 no *router* de acesso ao exterior da ESTG.

Os endereços IPv6 inicialmente configurados nas extremidades do túnel eram o *3FFE:31FF:0:FFFF::36/127* do lado da FCCN e o *3FFE:31FF:0:FFFF::37/127* do lado da ESTG, e o prefixo usado nas redes era o *3FFE:3102::/48*. Como a rede GÉANT, através da qual a FCCN tem trânsito IPv6 nativo para o mundo, não reencaminha tráfego tendo como origem endereços do espaço

de endereçamento do 6Bone (endereços com o prefixo *3FFE::/16*), o tráfego proveniente dos endereços IPv6 da ESTG não poderia sair da rede da FCCN para a GÉANT. Deixaram, então, de ser utilizados os endereços *3FFE::/16* e passaram a usar-se, por indicação da FCCN, endereços do espaço de produção (com o prefixo *2001::/16*); os endereços utilizados nas extremidades do túnel passaram a ser os apresentados na imagem: o *2001:690:810::1/64* do lado da FCCN e o *2001:690:810::2/64* do lado da ESTG. Nas redes passou-se a usar, o prefixo *2001:690:2067::/48* que fica no "terreno de expansão" do espaço da ESTG-Leiria, mas não está registado. Utilizou-se este prefixo em vez daquele que está atribuído à ESTG-Leiria pela FCCN, o *2001:690:2060::/48*, devido ao facto de este último estar reservado para ser administrado pelos administradores do acesso ao exterior da ESTG-Leiria.

A Tabela 4.2 resume os endereços configurados no terminal T-Tunel para acesso IPv4 ao exterior, configuração do túnel e acesso IPv6.

	IPv4	**IPv6**
Endereço IP	*193.137.239.44/27*	*2001:690:810:11::2/64*
Default Gateway	*193.137.239.62*	*2001:690:810:11::1/64*
Servidor de DNS	*193.137.239.226*	

Tabela 4.2 – Endereços configurados no terminal T-Tunel.

Para estabelecer a ligação por túnel à Internet, começou por se configurar a ligação IPv4 no terminal T-Tunel. Configurou-se o endereço IP, a máscara, o *default gateway* e o servidor de DNS. Na Figura 4.62 apresenta-se a configuração dos endereços através da interface gráfica do Windows XP.

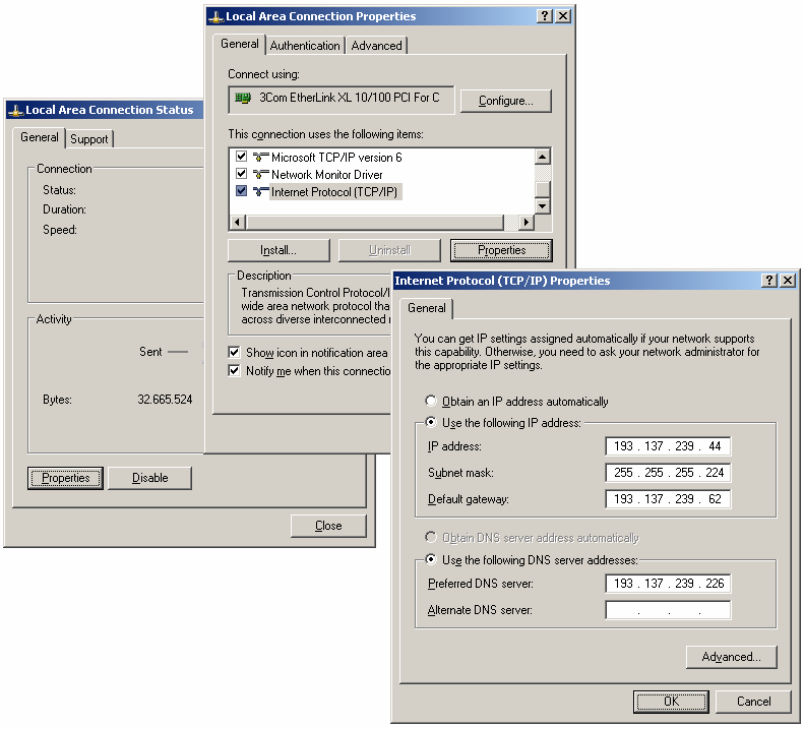


Figura 4.62 – Forma de configurar os endereços através da interface gráfica do Windows XP.

Na Figura 4.63 apresentam-se os comandos de NetShell executados para configurar os endereços especificados.

```
C:\>netsh interface ip set address name="Local Area Connection" source=static ad
dr=193.137.239.44 mask=255.255.255.224
Ok.

C:\>netsh interface ip set address name="Local Area Connection" gateway=193.137.
239.62 gwmetric=0
Ok.

C:\>netsh interface ip set dns name="Local Area Connection" source=static addr=1
93.137.239.226 register=primary
Ok.
```

Figura 4.63 – Comandos executados para configurar os endereços no terminal T-Tunel.

Para confirmar a conectividade IPv4 à Internet efectuou-se um *ping* ao endereço IPv4 da extremidade do túnel da FCCN. Na Figura 4.64 é apresentado o resultado do *ping* efectuado do terminal T-Tunel ao endereço *193.136.2.246* (*router* R-FCCN), prova da conectividade entre as duas máquinas.

```
C:\>ping 193.136.2.246

Pinging 193.136.2.246 with 32 bytes of data:

Reply from 193.136.2.246: bytes=32 time=14ms TTL=251
Reply from 193.136.2.246: bytes=32 time=13ms TTL=251
Reply from 193.136.2.246: bytes=32 time=14ms TTL=251
Reply from 193.136.2.246: bytes=32 time=15ms TTL=251

Ping statistics for 193.136.2.246:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 13ms, Maximum = 15ms, Average = 14ms
```

Figura 4.64 – Resultado do *ping* efectuado do terminal T-Tunel ao endereço *193.136.2.246*.

O próximo passo é configurar o túnel IPv6 sobre IPv4 para acesso ao *backbone* IPv6. Na Figura 4.65 apresentam-se os comandos executados para efectuar essa configuração.

```
C:\>netsh interface ipv6 add v6v4tunnel interface="Tunel-FCCN" localaddress=193.
137.239.44 remoteaddress=193.136.2.246
Ok.

C:\>netsh interface ipv6 add address interface="Tunel-FCCN" address=2001:690:810
:11::2
Ok.

C:\>netsh interface ipv6 add route prefix=::/0 interface="Tunel-FCCN" metric=0 n
exthop=2001:690:810:11::1
Ok.
```

Figura 4.65 – Comandos executados para configurar o túnel no terminal T-Tunel.

Esta configuração baseou-se em três acções:

- Criar a interface virtual que servirá de túnel, neste caso um túnel IPv6 sobre IPv4, indicando o nome da interface, e os endereços IPv4 origem e destino do túnel, ou seja, as extremidades do túnel;
- Configurar o endereço IPv6 da nossa extremidade do túnel da ESTG;
- Configurar uma rota por defeito indicando que todo o tráfego IPv6 é encaminhado pelo túnel.

Neste caso, o endereço origem do túnel é o *193.137.239.44*, endereço do terminal T-Tunel e o endereço destino é o *193.136.2.246*, endereço do *router* R-FCCN. O endereço IPv6 da extremidade do túnel da ESTG é o *2001:690:810:11::2*. A rota por defeito aponta todo o tráfego (prefixo *::/0*) para o túnel configurado, sendo o endereço de próximo salto o endereço do *router* R-FCCN, *2001:690:810:11::1*.

Do lado da FCCN o túnel já estava configurado (*router* R-FCCN), tendo como endereço origem o endereço *193.136.2.246* e endereço destino o endereço *193.137.239.44* (o inverso do túnel configurado na nosso terminal). O endereço da extremidade do túnel da FCCN é o

2001:690:810:11::1. No *router* R-FCCN o tráfego com destino ao nosso endereço IPv6 é encaminhado pelo túnel configurado.

À partida, a partir deste momento, já estaria o túnel configurado e pronto a transportar tráfego IPv6 sobre IPv4. Efectuou-se, então, um *ping6* do terminal T-Tunel (*2001:690:810:11::2*) ao endereço *2001:690:810:11::1* do *router* R-FCCN para testar a conectividade IPv6 e o funcionamento do túnel. O resultado desse *ping6* com sucesso é apresentado na Figura 4.66, e prova a conectividade entre os dois nós.

```
C:\>ping6 2001:690:810:11::1

Pinging 2001:690:810:11::1
from 2001:690:810:11::2 with 32 bytes of data:

Reply from 2001:690:810:11::1: bytes=32 time=19ms
Reply from 2001:690:810:11::1: bytes=32 time=17ms
Reply from 2001:690:810:11::1: bytes=32 time=19ms
Reply from 2001:690:810:11::1: bytes=32 time=22ms

Ping statistics for 2001:690:810:11::1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 17ms, Maximum = 22ms, Average = 19ms
```

Figura 4.66 – Resultado do *ping6* efectuado do terminal T-Tunel ao *router* R-FCCN.

```
⊞ Frame 19025 (114 bytes on wire, 114 bytes captured)
⊞ Ethernet II, Src: 00:01:02:a4:35:19, Dst: 00:d0:04:72:03:fc
⊟ Internet Protocol
    Version: 4
    Header length: 20 bytes
  ⊞ Differentiated Services Field: 0x00
    Total Length: 100
    Identification: 0x6c3f (27711)
  ⊞ Flags: 0x00
    Fragment offset: 0
    Time to live: 128
    Protocol: IPv6 (0x29)
    Header checksum: 0x58fd (correct)
    Source: 193.137.239.44 (193.137.239.44)
    Destination: 193.136.2.246 (193.136.2.246)
⊟ Internet Protocol Version 6
    Version: 6
    Traffic class: 0x00
    Flowlabel: 0x00000
    Payload length: 40
    Next header: ICMPv6 (0x3a)
    Hop limit: 128
    Source address: 2001:690:810:11::2
    Destination address: 2001:690:810:11::1
⊟ Internet Control Message Protocol v6
    Type: 128 (Echo request)
    Code: 0
    Checksum: 0x75a9 (correct)
    ID: 0x0000
    Sequence: 0x01e9
    Data (32 bytes)
```

Figura 4.67 – Captura do *Echo Request* do terminal T-Tunel para o *router* R-FCCN.

Na Figura 4.67 apresenta-se o *Echo Request* resultante do *ping6* do terminal T-Tunel para o *router* R-FCCN. Na Figura 4.68 é apresentado o *Echo Reply*, resposta à mensagem anterior, do *router* R-FCCN para o terminal T-Tunel. Nestas capturas o que é importante é reparar no facto de o pacote IPv6 ser encasulado dentro de um pacote IPv4, e nos endereços origem e destino de cada pacote. Compreende-se assim melhor o funcionamento dos túneis.

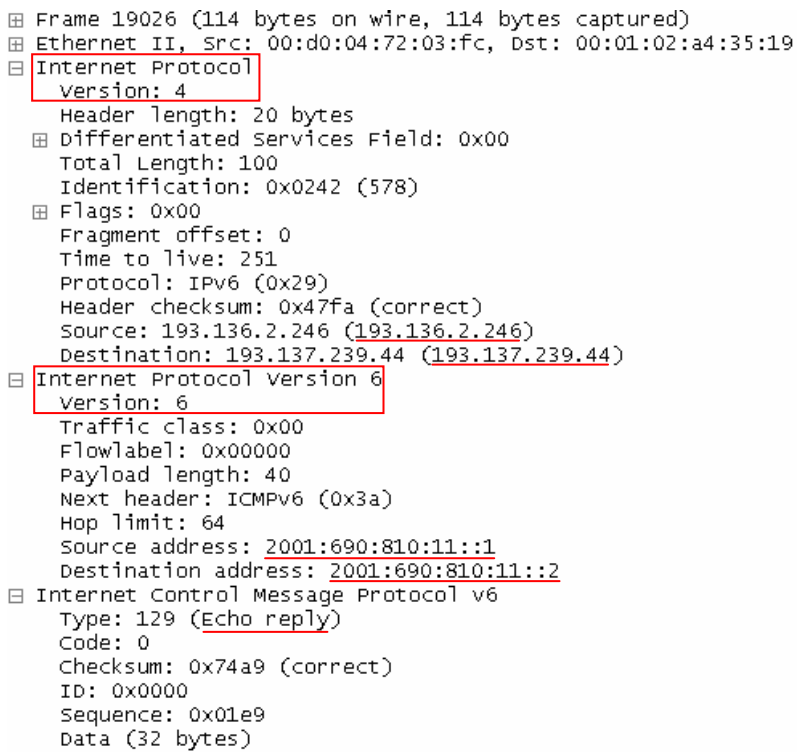


Figura 4.68 – Captura do *Echo Reply* do *router* R-FCCN para o terminal T-Tunel.

Para confirmar realmente o sucesso da ligação à Internet utilizou-se uma ferramenta *online*, disponibilizada em [118] pelo projecto JOIN. Efectuou-se, então, um *ping6 online* do servidor do projecto JOIN (endereço IPv6 *2001:638:500:131:2E0:81FF:FE24:37C6*) para o endereço *2001:690:810:11::2* do terminal T-Tunel. Na Figura 4.69 mostra-se o resultado obtido do *ping6* feito *online*.

```
Result of 'ping6 -s 56 -W 10 -c 5 2001:690:810:11::2 ':

PING 2001:690:810:11::2(2001:690:810:11::2) 56 data bytes
64 bytes from 2001:690:810:11::2: icmp_seq=1 ttl=118 time=76.1 ms
64 bytes from 2001:690:810:11::2: icmp_seq=2 ttl=118 time=75.1 ms
64 bytes from 2001:690:810:11::2: icmp_seq=3 ttl=118 time=80.3 ms
64 bytes from 2001:690:810:11::2: icmp_seq=4 ttl=118 time=75.2 ms
64 bytes from 2001:690:810:11::2: icmp_seq=5 ttl=118 time=72.6 ms

--- 2001:690:810:11::2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4003ms
rtt min/avg/max/mdev = 72.607/75.918/80.363/2.521 ms
```

Figura 4.69 – Resultado do *ping6 online* para o terminal T-Tunel.

Para não existirem dúvidas em relação à conectividade IPv6 ponto-a-ponto, efectuou-se um *ping6* do terminal T-Tunel para o endereço IPv6 do servidor do grupo JOIN com o endereço IPv6 *2001:638:500:131:2C0:81FF:FE24:37C6*. O resultado desse *ping6* com sucesso é apresentado na Figura 4.70.

```
C:\>ping6 2001:638:500:131:2e0:81ff:fe24:37c6

Pinging 2001:638:500:131:2e0:81ff:fe24:37c6
from 2001:690:810:11::2 with 32 bytes of data:

Reply from 2001:638:500:131:2e0:81ff:fe24:37c6: bytes=32 time=78ms
Reply from 2001:638:500:131:2e0:81ff:fe24:37c6: bytes=32 time=74ms
Reply from 2001:638:500:131:2e0:81ff:fe24:37c6: bytes=32 time=76ms
Reply from 2001:638:500:131:2e0:81ff:fe24:37c6: bytes=32 time=71ms

Ping statistics for 2001:638:500:131:2e0:81ff:fe24:37c6:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 71ms, Maximum = 78ms, Average = 74ms
```

Figura 4.70 – Resultado do *ping6* do terminal T-Tunel para o servidor do projecto JOIN.

Testou-se, então, o acesso *web* abrindo a página da FCCN (*http://www.fccn.pt/*). Como se tem configurado o endereço de um servidor de DNS IPv4 que resolve nomes em endereços IPv6, através do registo *AAAA*, o nome *www.fccn.pt* é resolvido no respectivo endereço IPv6 *2001:690:A00:40AA:2C0:9FFF:FE20:E261*. Na Figura 4.71 pode ver-se a resposta do servidor de DNS responsável pela resolução de endereços.

```
⊞ Frame 14639 (137 bytes on wire, 137 bytes captured)
⊞ Ethernet II, Src: 00:d0:04:72:03:fc, Dst: 00:01:02:a4:35:19
⊞ Internet Protocol
⊞ User Datagram Protocol, Src Port: 53 (53), Dst Port: 1693 (1693)
⊟ Domain Name System (response)
    Transaction ID: 0x4202
  ⊞ Flags: 0x8180 (Standard query response, No error)
    Questions: 1
    Answer RRs: 1
    Authority RRs: 2
    Additional RRs: 0
  ⊞ Queries
  ⊟ Answers
    ⊟ www.fccn.pt: type AAAA, class IN, addr 2001:690:a00:40aa:2c0:9fff:fe20:e261
        Name: www.fccn.pt
        Type: AAAA (IPv6 address)
        Class: IN (0x0001)
        Time to live: 4 hours
        Data length: 16
        Addr: 2001:690:a00:40aa:2c0:9fff:fe20:e261
  ⊞ Authoritative nameservers
```

Figura 4.71 – Resposta do servidor de DNS responsável pela resolução de endereços.

Como o acesso IPv6 é preferido ao IPv4, o pedido HTTP é feito por IPv6. Na Figura 4.72 pode ver-se um *screenshot* da página da FCCN, na qual é dito qual o endereço IPv6 utilizado no acesso ao *site*, prova de que a ligação foi feita por IPv6.

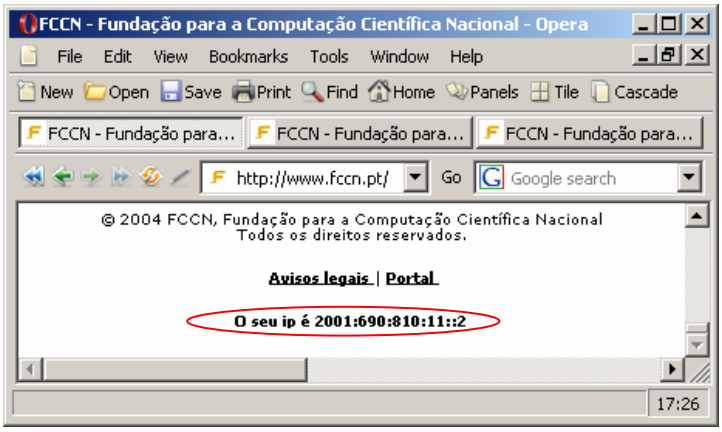


Figura 4.72 – Página da FCCN com indicação do endereço IPv6 utilizado no acesso ao *site*.

Sem servidor de DNS não existe resolução de nomes pelo que, se não tivesse sido configurado o endereço de um, o acesso à página da FCCN em vez de ser feito da forma *http://www.fccn.pt/* teria de

ser feito através da forma http://[2001:690:A00:40AA:2C0:9FFF:FE20:E261]/ com o endereço IPv6 do *site* entre parênteses rectos ("[" e "]"). Na Figura 4.73 exemplifica-se esta forma de acesso.

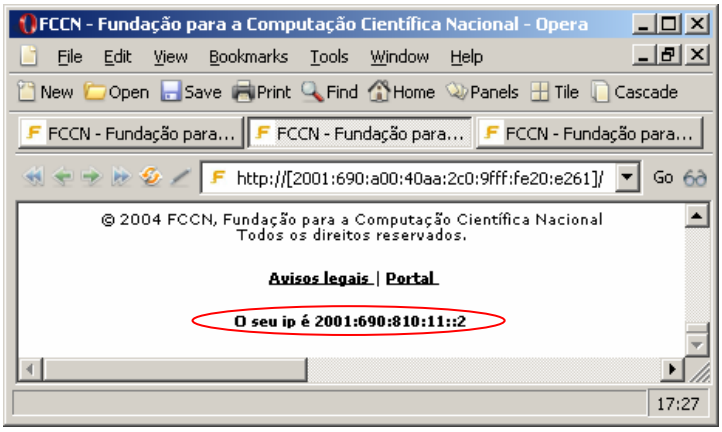


Figura 4.73 – Página da FCCN acedida através do endereço IPv6 entre parênteses rectos.

A título de comparação, na Figura 4.74 apresenta-se a página da FCCN acedida através de IPv4, com a indicação do endereço IPv4 através do qual foi feito o acesso ao *site*.

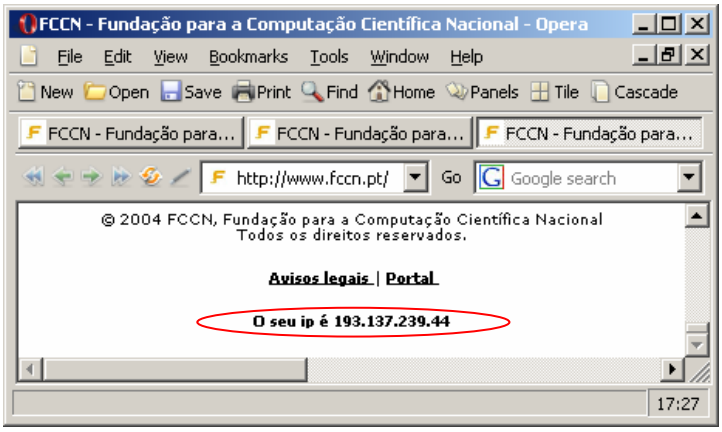


Figura 4.74 – Página da FCCN acedida através de IPv4.

Em anexo seguem as configurações do terminal T-Tunel. As configurações referem-se ao contexto de interface do NetShell (ver Anexo A.3.1).

4.4.2. Túnel com router e rede interior

Depois de se ter configurado o túnel IPv6 sobre IPv4 da ESTG para a FCCN, ficou-se a perceber melhor o funcionamento deste mecanismo, mas era necessário ir mais além. O objectivo era criar uma rede heterogénea completamente IPv6, em que todas as máquinas tivessem acesso IPv6 ao exterior. Esse objectivo foi conseguido através do cenário representado no diagrama de rede da Figura 4.75. No cenário ambos os *routers* (R-Tunel e R-Rede) são *Cisco IOS*; os terminais e servidores utilizados possuem os sistemas operativos Microsoft Windows XP SP2 e Server 2003, Fedora Core 3 e Mac OS X.

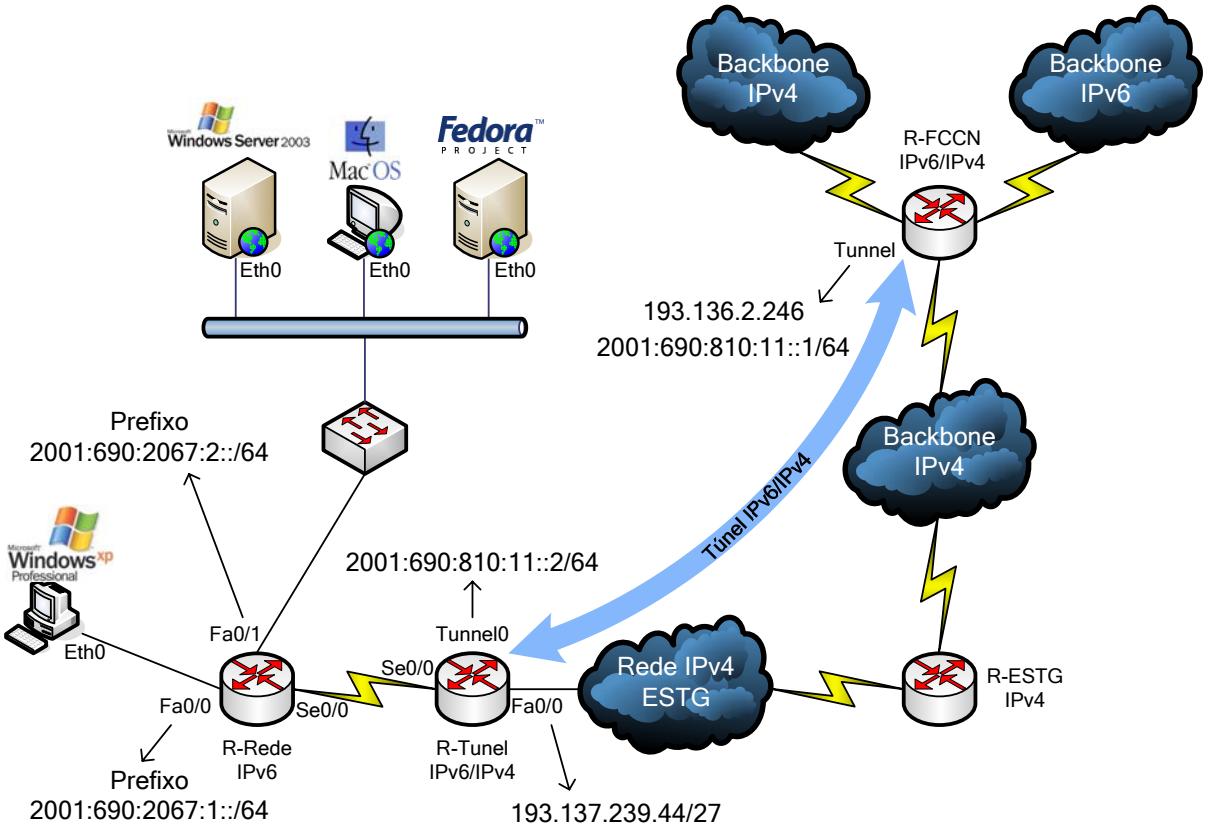


Figura 4.75 – Diagrama de rede heterogenia com acesso ao exterior por túnel.

Para se implementar este cenário seguiram-se os passos apresentados a seguir.

1.º Passo: Configuração do *router* R-Tunel

Entre as principais configurações necessárias no *router* R-Tunel incluem-se: a configuração da ligação IPv4 do *router* ao exterior; a configuração do túnel com o *router* R-FCCN; e a configuração das rotas de encaminhamento IPv4 e IPv6. Estas configurações são muito semelhantes às efectuadas no terminal T-Tunel, mas desta vez no *router*. Na Figura 4.76 são apresentadas as configurações efectuadas no *router* R-Tunel para garantir conectividade IPv4 e IPv6 deste *router* ao exterior.

```
R-Tunel(config)#ipv6 unicast-routing
R-Tunel(config)#
R-Tunel(config)#interface FastEthernet0/0
R-Tunel(config-if)#ipv6 enable
R-Tunel(config-if)#ip address 193.137.239.44 255.255.255.224
R-Tunel(config-if)#
R-Tunel(config-if)#exit
R-Tunel(config)#
R-Tunel(config)#interface Tunnel0
R-Tunel(config-if)#ipv6 enable
R-Tunel(config-if)#ipv6 address 2001:690:810:11::2/64
R-Tunel(config-if)#tunnel source 193.137.239.44
R-Tunel(config-if)#tunnel destination 193.136.2.246
R-Tunel(config-if)#tunnel mode ipv6ip
R-Tunel(config-if)#
R-Tunel(config-if)#exit
R-Tunel(config)#
R-Tunel(config)#ip route 0.0.0.0 0.0.0.0 193.137.239.62
R-Tunel(config)#
R-Tunel(config)#ipv6 route ::/0 2001:690:810:11::1
R-Tunel(config)#
```

Figura 4.76 – Configurações efectuadas no *router* R-Tunel.

Depois de garantir a conectividade deste *router* à Internet, via IPv4 e IPv6, configurou-se a interface *Serial0/0* do *router* para se poder estabelecer a ligação com o *router* R-Rede, que suportará a ligação das redes internas. Configurou-se também, através do comando "`ipv6 route`", uma rota estática de forma a que o tráfego com destino a endereços IPv6 com o prefixo *2001:680:2067::/48*, prefixo usado pelas máquinas das redes internas, seja encaminhado através da interface *Serial0/0* para o *router* R-Rede. Na Figura 4.77 podem ver-se estas configurações.

```
R-Tunel(config)#interface Serial0/0
R-Tunel(config-if)#ipv6 enable
R-Tunel(config-if)#
R-Tunel(config-if)#exit
R-Tunel(config)#
R-Tunel(config)#ipv6 route 2001:690:2067::/48 Serial0/0
R-Tunel(config)#
```

Figura 4.77 – Configurações efectuadas no *router* R-Tunel para a ligação ao *router* R-Rede.

No Anexo A.3.2, consta toda a configuração do *router* R-Tunel.

Estando o *router* R-Tunel todo configurado e pronto a encaminhar o tráfego proveniente das redes internas, pode-se configurar o *router* R-Rede e as redes internas a ele ligadas.

2.º Passo: Configuração do *router* R-Rede

O *router* R-Rede é o *router* ao qual se ligam as redes internas. Este *router* possui duas interfaces *FastEthernet*, pelo que se ligaram duas redes: uma apenas com um terminal Windows XP, e outra com três servidores *web*, um no Windows Server 2003, outro no Fedora Core 3 e outro no Mac OS X.

Na configuração deste *router* há a destacar que nas interfaces *FastEthernet* se configuraram os prefixos que se queriam divulgados para as redes. Para a rede ligada à interface *FastEthernet0/0* atribuiu-se o prefixo *2001:690:2067:1::/64* e para a rede ligada à interface *FastEthernet0/1* o prefixo *2001:690:2067:2::/64*. Ambos os prefixos derivam do prefixo *2001:690:2067::/48* que nos foi atribuído pela FCCN. Todos os endereços configurados com base nos prefixos divulgados pelo *router* R-Rede serão formados a partir da norma EUI-64.

Na Figura 4.78 podem ver-se as configurações efectuadas no *router* R-Rede. Para além das interfaces *FastEthernet*, configuraram-se ainda o encaminhamento de tráfego IPv6 *unicast* ("`ipv6 unicast-routing`"), a interface *Serial0/0* e uma rota estática.

```
R-Rede(config)#ipv6 unicast-routing
R-Rede(config)#
R-Rede(config)#interface FastEthernet0/0
R-Rede(config-if)#ipv6 enable
R-Rede(config-if)#ipv6 address 2001:690:2067:1::/64 eui-64
R-Rede(config-if)#
R-Rede(config-if)#exit
R-Rede(config)#
R-Rede(config)#interface FastEthernet0/1
R-Rede(config-if)#ipv6 enable
R-Rede(config-if)#ipv6 address 2001:690:2067:2::/64 eui-64
R-Rede(config-if)#
R-Rede(config-if)#exit
R-Rede(config)#
R-Rede(config)#interface Serial0/0
R-Rede(config-if)#ipv6 enable
R-Rede(config-if)#clockrate 115200
R-Rede(config-if)#
R-Rede(config-if)#exit
R-Rede(config)#
R-Rede(config)#ipv6 route ::/0 Serial0/0 2001:690:810:11::2
R-Rede(config)#
```

Figura 4.78 – Configurações efectuadas no *router* R-Rede.

A rota estática configurada serve para o *router* saber que todo o tráfego de qualquer prefixo ou endereço que não seja local, deve ser encaminhado pela interface *Serial0/0*, sendo o endereço de próximo salto o endereço IPv6 do *router* R-Tunel *2001:690:810:11::2*.

Em anexo pode consultar-se a configuração completa do R-Rede (ver Anexo A.3.3).

3.º Passo: Configuração da segunda rede interna

Desta vez, com excepção de um terminal, todos os outros fazem o papel de cliente e servidor. Existiu também a preocupação de tentar usar exclusivamente IPv6 em toda a rede. Os sistemas utilizados foram:

- Windows Server 2003 Enterprise – Servidor de HTTP;
- Linux Fedora Core 3 – Servidor de HTTP;
- Mac OS X 10.3.7, Darwin 7.7 – Servidor de HTTP;
- Windows XP – Cliente.

- **Windows Server 2003**

A opção foi instalar o IIS 6.1, a partir do CD do Windows Server 2003. Suporta IPv6 de forma nativa, sem ser necessário qualquer tipo de configuração. O procedimento a efectuar está descrito na Figura 4.79.

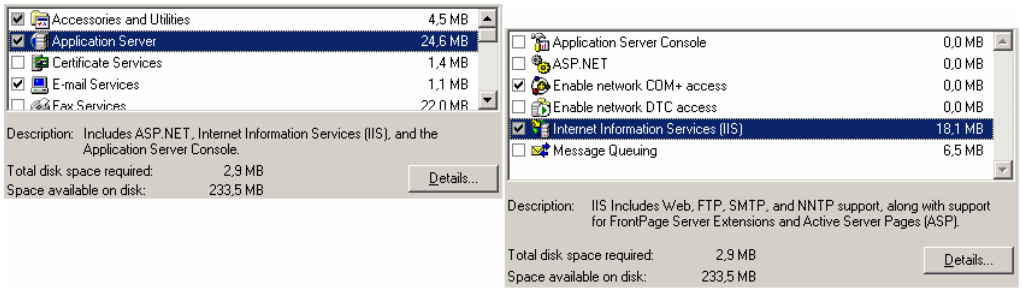


Figura 4.79 – Instalação do IIS 6.1 no Windows Server 2003.

O suporte em IPv6 do IIS é muito básico pois a interface gráfica que permite por exemplo, restringir endereços IPv4, não existe para IPv6, não sendo assim possível restringir endereços IPv6. Toda a interface gráfica de configuração funciona como se não existissem os endereços IPv6. Para além disso SMTP, NNTP e FTP não têm suporte IPv6 com este servidor.

Foi desactivada também a *stack* IPv4, como se exemplifica na Figura 4.80.

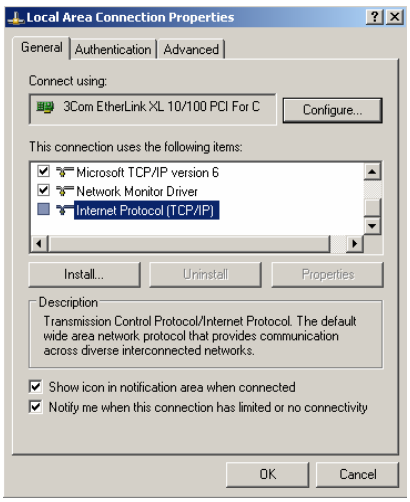


Figura 4.80 – Desactivar a stack IPv4 no Windows.

Para o cliente Windows XP efectuou-se o mesmo processo.

- **Mac OS X e Darwin**

Esta é uma nova plataforma que foi usada também como servidor de http, pelo que foi verificado primeiro se todo o suporte IPv6 funcionava correctamente. O processo será semelhante aos já explicados em outros sistemas operativos.

1. **Verificou-se os endereços IPv6, tanto *link-local*, como globais**

   Na Figura 4.81 pode-se reparar que ao seleccionar a opção "Automatically" automaticamente ficou associado um prefixo a um endereço global. Podia-se também a través da interface gráfica adicionar um endereço IPv6 manualmente.

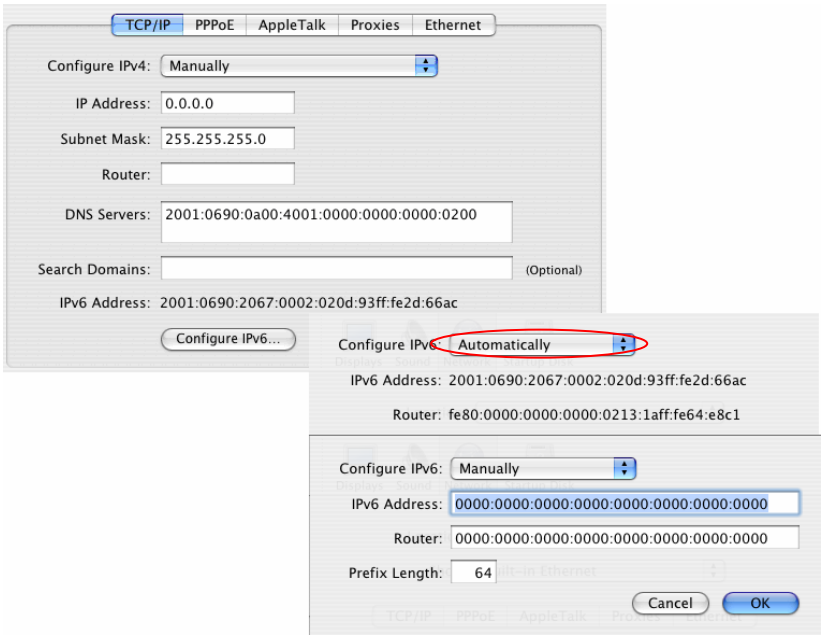


Figura 4.81 – Configuração de endereços IPv6 em Mac OS X.

Pode-se também verificar os endereços, recorrendo a uma consola, e utilizando o comando "`ifconfig`", nesse caso pode-se dizer que se está a utilizar o sistema Darwin, baseado em FreeBSD. O aspecto será então o representado na Figura 4.82.

```
profs-Computer:/ ipv6$ ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        inet6 ::1 prefixlen 128
        inet6 fe80::1 prefixlen 64 scopeid 0x1
        inet 127.0.0.1 netmask 0xff000000
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
en0: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        inet6 fe80::20d:93ff:fe2d:66ac prefixlen 64 scopeid 0x4
        inet6 2001:690:2067:2:20d:93ff:fe2d:66ac prefixlen 64 autoconf
        ether 00:0d:93:2d:66:ac
        media: autoselect (100baseTX <half-duplex>) status: active
        supported media: none autoselect 10baseT/UTP <half-duplex> 10baseT/UTP <full-duplex> 10baseT/UTP
<full-duplex,hw-loopback> 100baseTX <half-duplex> 100baseTX <full-duplex> 100baseTX <full-duplex,hw-loopb
ack> 1000baseTX <full-duplex> 1000baseTX <full-duplex,hw-loopback> 1000baseTX <full-duplex,flow-control>
1000baseTX <full-duplex,flow-control,hw-loopback>
fw0: flags=8822<BROADCAST,SMART,SIMPLEX,MULTICAST> mtu 4078
        lladdr 00:0d:93:ff:fe:2d:66:ac
        media: autoselect <full-duplex> status: inactive
        supported media: autoselect <full-duplex>
```

Figura 4.82 – Configuração de endereços IPv6 em Darwin (Mac).

Optou-se por não especificar qualquer endereço IPv4, ficando o terminal com o endereço *unspecified* (ver Subsecção 2.4.3).

2. Configurou-se o servidor Apache

O procedimento normal seria apenas aceder às *Network Preferences* >.*Sharing* > *Personal Web Sharing* e de seguida clicar no botão "*Star*t". A Figura 4.83 exemplifica.

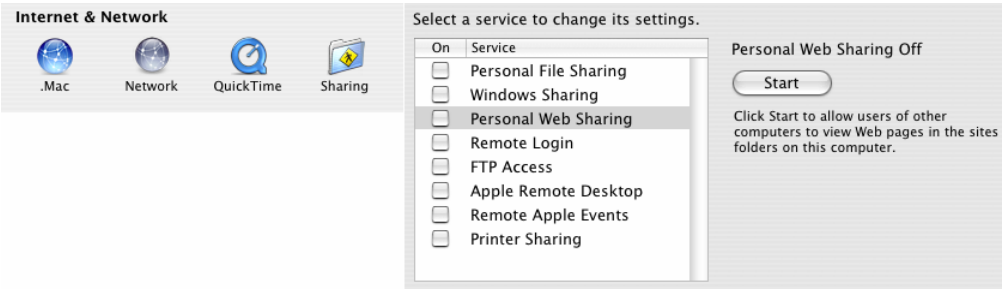


Figura 4.83 – Procedimento normal para activar o *Personal Web Server* (Apache).

3. Configurar o servidor, recorrendo a um *update* ao Apache.

O Mac apesar de usufruir de uma versão relativamente actualizada do OS X, a versão do Apache não era a 2, mas sim a 1.3. Devido ao facto de a versão 1.3 do Apache não suportar de forma nativa o IPv6, foi necessário optar: ou se utilizavam as *patches* pouco estáveis do KAME, ou então fazia-se um *upgrade* para a versão mais recente e estável que já suporta o IPv6. Foi escolhida a segunda opção, muito pela razão da estabilidade usufruída.

Existem alguns manuais na Internet, com o objectivo de facilitar todo este processo de instalação a partir das fontes de código do apache (ver [161]).

Aconselha-se a ida à página *web* do serviço em questão, e de lá retirar as respectivas fontes (ver [162]). Por vezes os *packages* de instalação que tornam todo processo mais simples de instalar, contêm alguns *bugs*. Isto pode fazer com que o tempo que se poupa a instalar, não compense o que se passa a configurar. No caso em questão verifica-se que não existe uma implementação em Mac OS X do Apache, mas existe em Darwin. Por outras palavras significa que existe apache para o Mac, mas sem qualquer tipo de interface gráfica. A versão usada foi a 2.0.50.

- **Linux**

A implementação é exactamente igual à do Cenário 1 (ver Subsecção 4.2.1), apenas muda o prefixo.

4.º Passo: Testes

Foram feitos diversos testes para implementação tanto na rede interna, como fora dela.

1. Foi dado suporte IPv6 ao Safari (*browser* do Mac)

Curiosamente o Safari no início suportava apenas endereços IPv6 na sua forma literal, exactamente o contrário do Internet Explorer. O Safari evoluiu em termos de IPv6, suportando agora respostas DNS e existe uma forma de o fazer efectuar pedidos em IPv6.

Em algumas versões do Safari, basta digitar no terminal o seguinte comando `"defaults write com.apple.safari IncludeDebugMenu 1"`, para aceder a um menu no *browser* que permite então endereços IPv6.

No caso do Safari testado, este comando não funcionou, tendo sido necessário instalar um utilitário intitulado *Safari enhancer* que nos permite activar esse menu escondido. Este utilitário tem variadas opções mas a única que nos interessa é a opção *Debugging Menu*. A aplicação *Safari enhancer*, deve ter o aspecto da Figura 4.84.

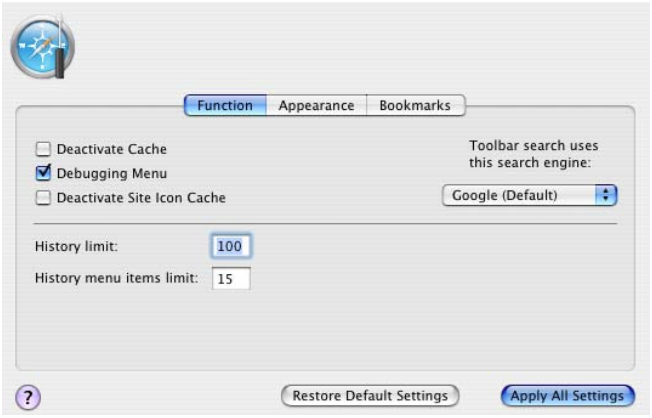


Figura 4.84 – Aplicação *Safari enhancer* com as opções necessárias para o IPv6 prioritário.

O passo seguinte é abrir o Safari, aceder ao menu *Debug* > *Supported Protocols* e desactivar a opção *http: (Simple Loader*), como está demonstrado na Figura 4.85. A partir desse momento o Safari pode utilizar endereços IPv6.

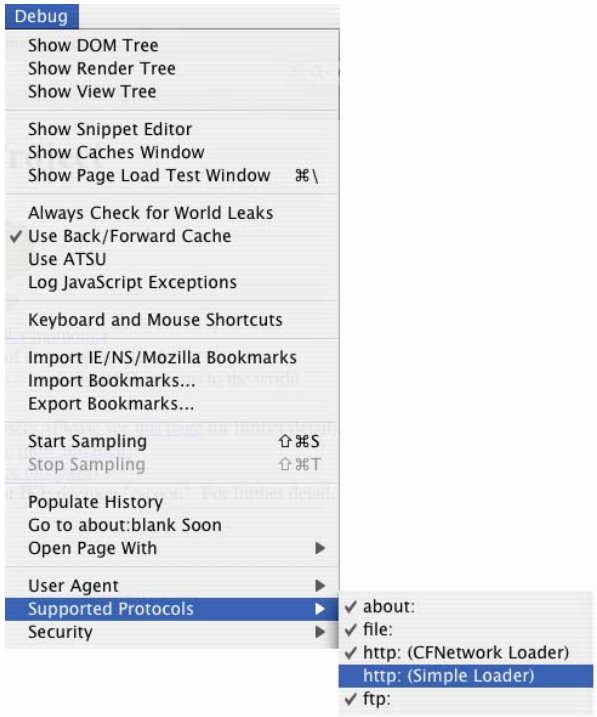


Figura 4.85 – Menu *Debug* para activar IPv6 no *browser* Safari.

2. Alguns testes entre a rede interna (os diversos servidores ou clientes)

Estes testes abordam apenas a rede interna demonstrada na Figura 4.86.

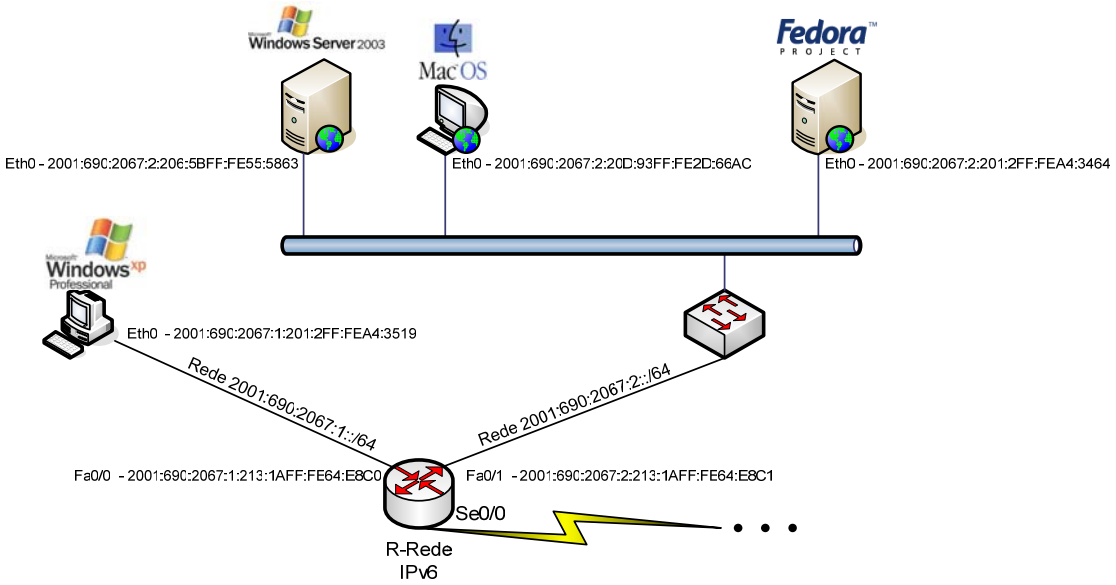


Figura 4.86 – Parte interna da rede com túnel.

Tentou-se explorar ao máximo a compatibilidade dos diversos sistemas operativos, utilizando o serviço HTTP. Os resultados dos testes estão expostos na Tabela 4.3. A tabela mostra se os pedidos HTTP de determinado cliente a determinado servidor tiveram sucesso ou não.

Servidor / Cliente	Windows xp Professional	Windows Server 2003	Fedora	Mac OS
Windows xp Professional	✓	✓	✓	✓
Windows Server 2003	✓	✓	✓	✓
Fedora	✓	✓	✓	✓
Mac OS	✓	✗	✓	✓

Tabela 4.3 – Tabela de compatibilidades de servidores e clientes HTTP.

A única falha na rede heterogénea IPv6, foi verificada num pedido HTTP do cliente Mac OS X ao servidor Windows Server 2003. A Figura 4.87 demonstra a captura desse erro.

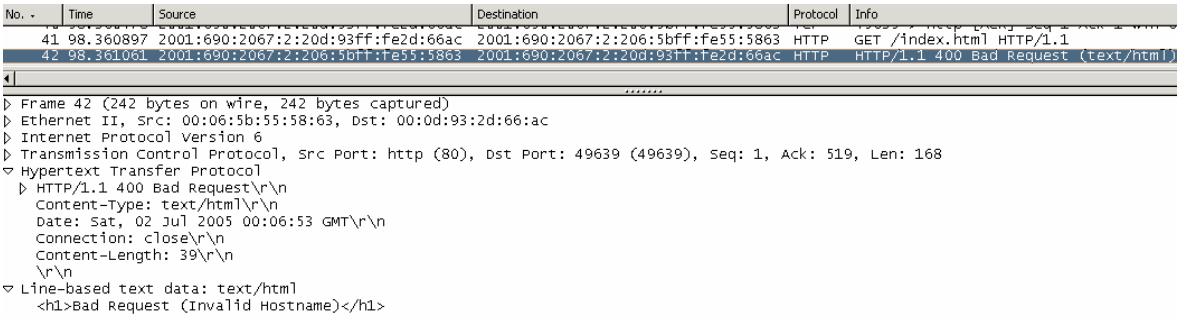


Figura 4.87 – Captura do erro aquando do pedido do Mac OS para Windows Server 2003.

Com IPv4, toda a negociação HTTP é efectuada correctamente entre estes dois terminais. Isso significa que o problema tem origem na própria implementação IPv6 de algum dos sistemas

operativos. Demonstra-se, assim, ainda algum carácter pouco estável do IPv6, quando se recorre a diversos sistemas operativos.

3. Teste utilizando a rede externa.

Foi especificado um servidor de DNS IPv6 para que se pudessem efectuar pedidos ao exterior com resolução de nomes. O servidor escolhido pertence a um projecto da FCCN (ver [20]). A Figura 4.88 demonstra a configuração do endereço do servidor de DNS no sistema Mac OS X.

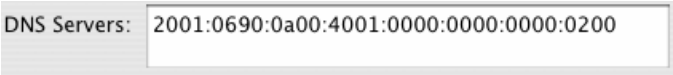


Figura 4.88 – Inserção do Servidor de DNS no Mac OS X.

Efectuou-se então o acesso à Internet utilizando o *browser* Safari, que recorre ao servidor de DNS (Figura 4.89).

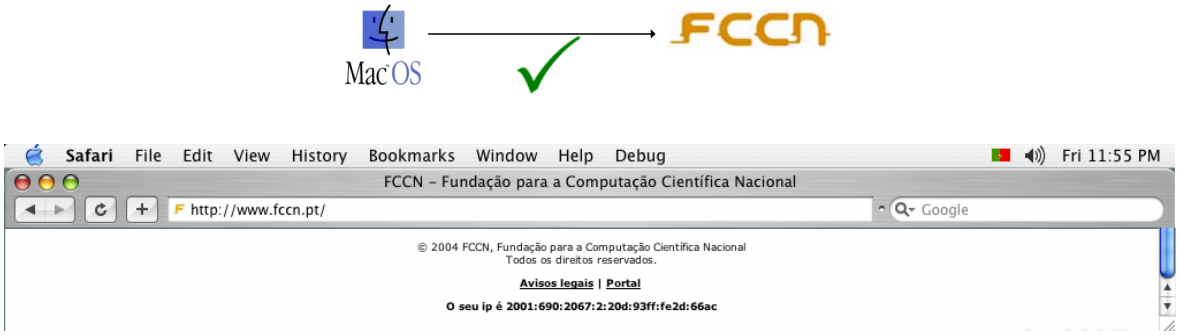


Figura 4.89 – Pedido do Mac OS X ao sítio da FCCN.

4. Acesso a partir de uma rede doméstica.

Por fim acedeu-se à rede interna implementada através de um terminal algures na Internet. Na Figura 4.90 está traçado o percurso dos pacotes do terminal de casa , até chegarem à rede criada no outro extremo.

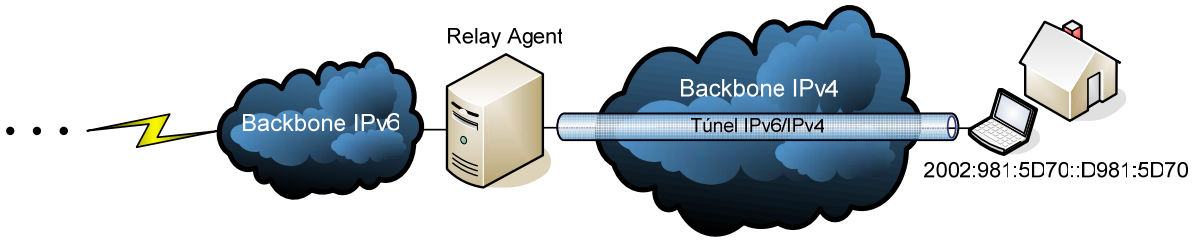


Figura 4.90 – Percurso da rede doméstica.

Infelizmente, os ISPs em Portugal ainda não fornecem prefixos IPv6, pelo que é necessário usar também os túneis. Neste caso foi utilizada a interface 6to4 automaticamente configurada no Microsoft Windows (ver [133]). A configuração desta interface pode ser visualizada na Figura 4.91.

```
Adaptador Tunnel 6to4 Tunneling Pseudo-Interface:

        Sufixo DNS específico da ligação. : netvisao.pt
        Endereço IP . . . . . . . . . . . : 2002:d981:5d70::d981:5d70
        Gateway predefinido . . . . . . . : 2002:c058:6301::1
                                            2002:836b:213c:1:e0:8f08:f020:8
                                            2002:c058:6301::c058:6301
```

Figura 4.91 – Configuração da interface 6to4 do terminal localizado em casa.

Verificam-se três *gateways* predefinidos, todos eles com endereços de *relay agents*[21], o primeiro é a tradução para hexadecimal do endereço *anycast* 192.88.99.1 (ver [70]), exclusivamente especificado para *relay agents* 6to4. Para o caso de não ser possível encontrar o *relay agent* através do endereço *anycast*, tenta-se utilizar os endereços próprios da Microsoft para os seus *relay agents*. Neste caso o *relay agent* da Microsoft com o nome *6to4.ipv6.microsoft.com* tem o endereço IPv6 *2002:836B:213C:1:E0:8F08:F020:8*.

Foi efectuado então um trace, que pode ser visualizado na Figura 4.92, desde casa até à rede interna criada anteriormente.

```
C:\Documents and Settings\Miguel>tracert 2001:690:2067:2:201:2ff:fea4:3464

Rastreio da rota para 2001:690:2067:2:201:2ff:fea4:3464 até o máximo de 30 salto
s

  1    27 ms    20 ms    21 ms  2002:c058:6301::1
  2    35 ms    35 ms    32 ms  2001:690:810:11::2
  3    57 ms    67 ms    68 ms  2001:690:2067:1:213:1aff:fe64:e8c0
  4    99 ms    54 ms    53 ms  2001:690:2067:2:201:2ff:fea4:3464

Rastreio concluído.
```

Figura 4.92 – Trace de casa à rede interna criada.

Pode verificar-se, então, que o rastreio começou pelo endereço *anycast*. Seguidamente percorreu o nosso *router* de fronteira, o nosso *router* da rede interna, tendo terminado num dos nossos servidores de HTTP, o Windows Server 2003 com IIS. O acesso a este servidor está exposto na Figura 4.93.

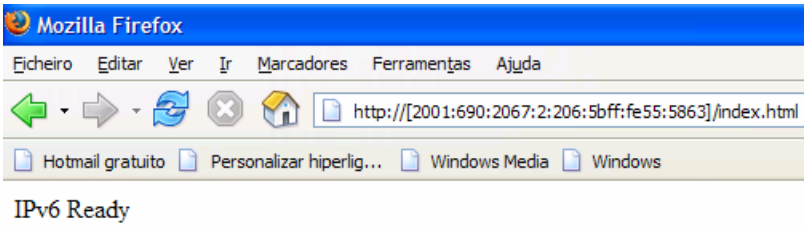


Figura 4.93 – Acesso ao servidor HTTP do Windows Server 2003.

Provou-se assim a atingibilidade do servidor *web* da nossa rede interna através da Internet.

[21] Em IPv6 um *relay agent* é o responsável pela trasnposição de um endereço IPv6 encapsulado para uma normal rede IPv6.

4.5. Cenário 4: Acesso nativo ao exterior

Outra forma de estabelecer um acesso à Internet é através de uma ligação nativa IPv6. Era objectivo deste projecto a configuração e teste desta solução. Apesar de ter sido iniciada a sua configuração, não foi possível em tempo útil ter este cenário pronto a testar, mas não obstante apresenta-se o que teve (ou teria) de ser feito.

Antes de se partir para a configuração da ligação IPv6 nativa, a ligação à Internet da rede da ESTG estava como é apresentado na Figura 4.94. O *router* de acesso, um Cisco 2600, tinha apenas uma interface *Ethernet* que servia toda a rede. A ligação à FCCN era feita através de uma interface ATM com um PVC com encapsulamento `"aal5mux ip"`.

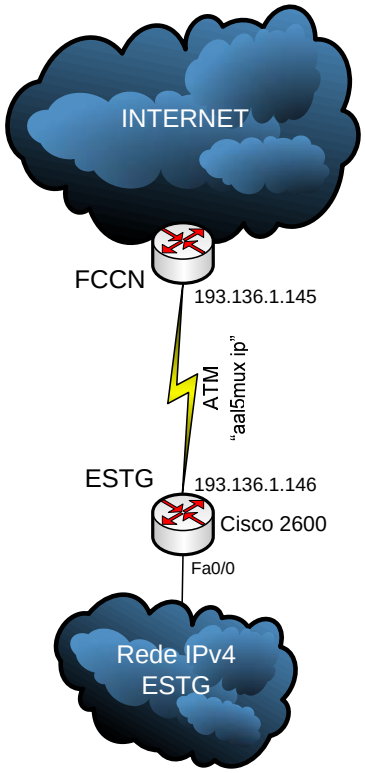


Figura 4.94 – Ligação à Internet da rede da ESTG.

Para se poder usufruir de uma ligação IPv6 nativa no *campus* da ESTG, foi instalado um *router* Cisco 2621-XM com suporte IPv6 e duas interfaces *FastEthernet*, de forma a se poder separar internamente o tráfego IPv4 e IPv6; uma das interfaces *FastEthernet* ficaria para a rede IPv4 e outra para a rede IPv6 pura. Para se poder ter tráfego IPv6 puro na ligação ATM existente foi necessário alterar o encapsulamento do PVC de `"aal5mux ip"` para `"all5snap"` (comando `"encapsulation all5snap"`) nos dois lados da ligação. Na interface ATM do *router* da ESTG foi necessário configurar, para além do endereço IPv4 *193.136.1.146*, o endereço IPv6 *2001:690:810:1B::2/64* (comando `"ipv6 address 2001:690:810:1B::2/64"`); na interface ATM do lado da FCCN foi configurado o endereço *2001:690:810:1B::1/64*. Na Figura 4.95 é apresentado o cenário da ligação da ESTG à Internet, com IPv4 e IPv6.

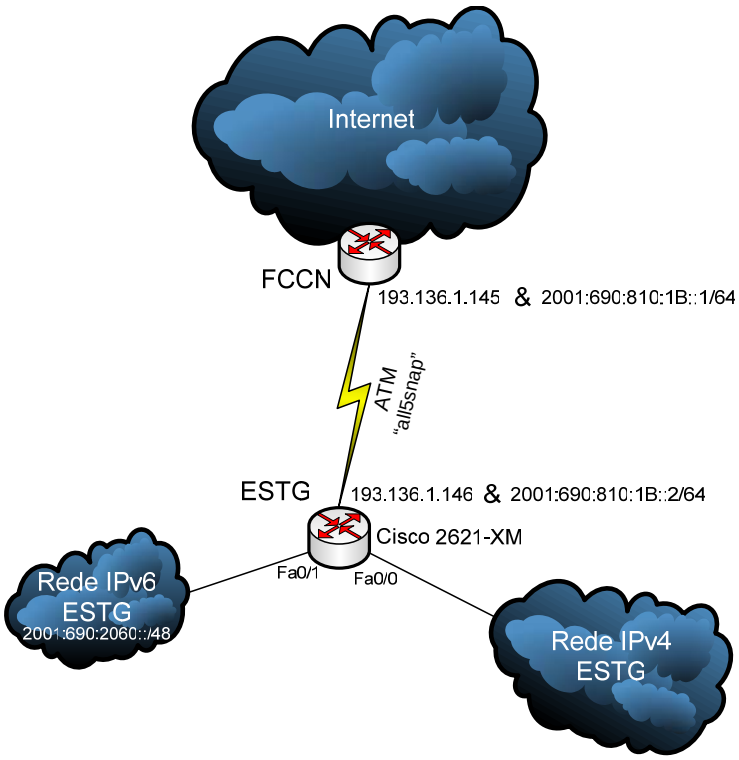


Figura 4.95 – Cenário da ligação da ESTG à Internet, com IPv4 e IPv6.

Para além de alterar o encapsulamento do PVC da linha ATM, e de configurar o endereço IPv6, foi necessário configurar através do comando "`ipv6 route ::/0 2001:690:810:1B::1`", no *router* de acesso da ESTG ao exterior, uma rota por defeito para todo o tráfego IPv6 ser encaminhado para o IP *2001:690:810:1B::1*, endereço do *router* da FCCN, extremo IPv6 da ligação. Do lado da FCCN, todo o tráfego com destino a endereços com prefixo *2001:690:2060::/48* (prefixo atribuído à ESTG-Leiria) é encaminhado para o endereço *2001:690:810:1B::2* do *router* da ESTG.

A rede IPv6 da ESTG utilizará o prefixo *2001:690:2060::/48*, pelo que é necessário adicionar mais uma rota de encaminhamento ao *router* de acesso ao exterior da Escola; é necessária uma rota que encaminhe todo o tráfego com destino a endereços formados com aquele prefixo pela interface *FastEthernet0/1*, isso é conseguido através do comando "`ipv6 route 2001:690:2060::/48 FastEthernet0/1`").

Com as alterações e configurações efectuadas é possível ter uma rede IPv6 pura na ESTG-Leiria, e assim aceder ao *backbone* IPv6.

4.6. Análise e Conclusões

O processo de implementação começou pela configuração de uma rede interna, passou pelos exemplos de alguns serviços que podiam ser implementados, ligou-se depois ao exterior e foram então testadas as diversas negociações IPv6 tanto na própria rede interna, como da rede interna para o exterior.

O serviço HTTP é um dos serviços que mais suporte IPv6 tem. Por essa mesma razão foram testados diversos servidores e clientes HTTP de diversos sistemas operativos, com o objectivo de verificar até que ponto consegue chegar a compatibilidade das implementações IPv6. O cenário é bastante positivo, verificando-se apenas uma falha na interacção de dois sistemas operativos: o Mac OS X e o Windows Server 2003.

No que diz respeito ao DNS, foi usado o Windows Server 2003 como servidor. O suporte é muito limitado comparando por exemplo com o BIND (o mais popular dos servidores de DNS). Existem problemas no cliente Windows XP e OpenBSD no que diz respeito a efectuar pedidos IPv6 ao servidor. Estranhamente, o Linux é mais compatível com o servidor do Windows Server 2003 do que

o Windows XP. O Linux efectua pedidos em IPv6 ao servidor sem problemas, enquanto que o Windows XP não.

Também foram implementados clientes e servidores de DHCP. Destes três serviços (HTTP, DNS e DHCP), o DHCP é de longe o que está mais atrasado no seu desenvolvimento, talvez também por ser, destes serviços, o que a mais modificações obriga. Porém existem já implementações bastante funcionais, como é o caso da implementação testada (Dibbler).

Foram testados os principais serviços de Internet e de redes internas. Na maioria dos restantes serviços, as mudanças necessárias passam apenas praticamente pela aceitação do novo tipo de endereços. Na Figura 4.96 está exposto um fluxograma de um possível processo para instalação de um serviço.

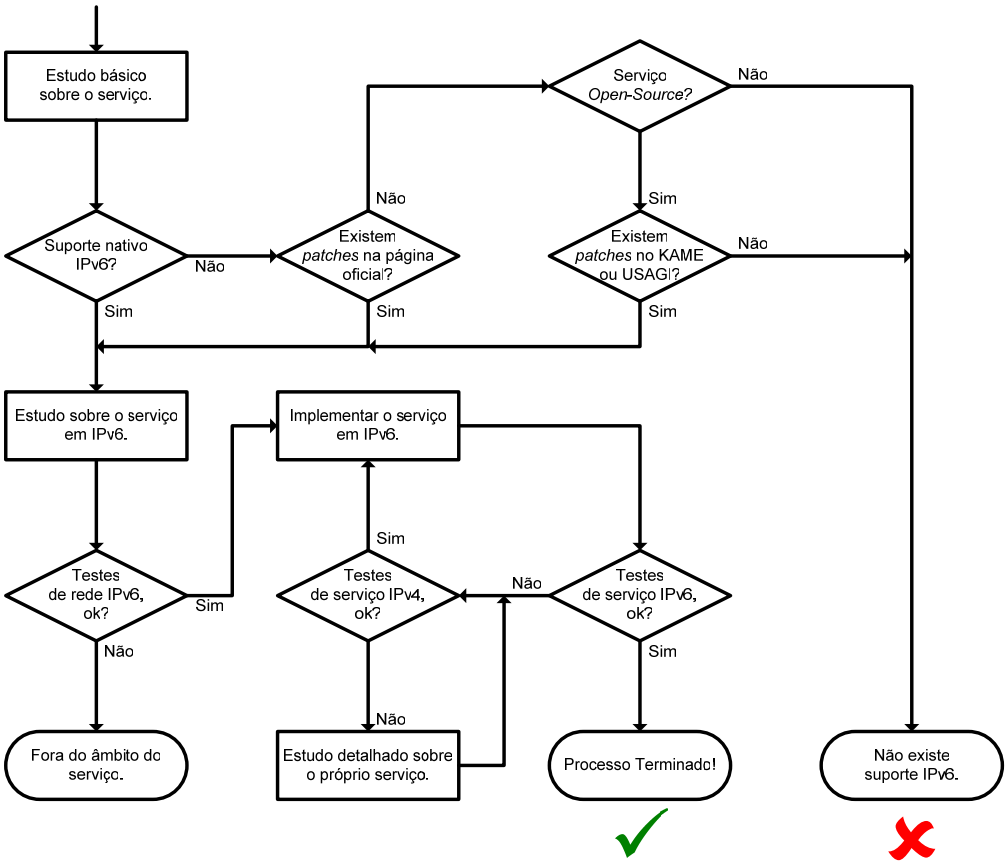


Figura 4.96 – Fluxograma para instalação de serviço IPv6.

No Linux já existe uma lista considerável de serviços com suporte IPv6 porém, em termos de interfaces gráficas que aceitem o novo tipo de endereços, a lista é muito reduzida, tanto em Windows como Linux ou BSD. O Mac OS X ainda é o sistema operativo que tem o maior suporte IPv6 em termos de interface amigável para o utilizador. Este tipo de sistema aproveita todas as grandes funcionalidades que o BSD proporciona (devido ao Darwin) em IPv6 e junta-lhe uma interface gráfica bastante interessante em certos serviços.

Uma ligação nativa IPv6 e uma rede pura IPv6 é sempre bom, mas nem tudo são pontos positivos. Ainda existem muitos serviços que não funcionam em IPv6, e tendo uma rede só com IPv6 poderá não ser muito proveitoso. Por exemplo, no Microsoft Windows XP, apesar de ser possível especificar o endereço de um servidor de DNS IPv6, não é possível efectuar transporte de mensagens DNS sobre IPv6 (ver [126]), logo, não existindo IPv4, não poderá existir resolução de nomes e endereços IPv6 naquele sistema operativo. Há também o problema do acesso a conteúdos que apenas estão disponíveis em IPv4. Com túnel beneficiamos de ligação IPv4 e IPv6 simultânea, mas isso implica *dual stack*, mais configurações, mais trabalho, *overhead* de tráfego, mas tendo em conta o desenvolvimento actual do IPv6, o uso simultâneo das duas versões do protocolo ainda é a melhor escolha.

Ao longo de toda a implementação, chegou-se a diversas conclusões que permitem agora fornecer algumas sugestões, umas mais ligadas directamente ao IPv6, outras menos, mas que também podem ajudar à implementação IPv6. Algumas delas são:

- Ao utilizar Linux, aconselha-se o uso de uma distribuição que disponibilize um serviço que permita actualizar automaticamente (precisando apenas de executar um comando) tanto o *kernel*, como os mais variados serviços e aplicações. Alguns exemplos dessas aplicações são: o Portage no Gentoo, (ver [170]), Yum no Fedora (ver [171]), e o APT no Debian (ver [22]). Estas aplicações são bastantes úteis pois, no que diz respeito ao IPv6, por vezes, revelam-se cruciais. Isso acontece, devido ao facto de estar numa fase de grande desenvolvimento.
- Ao fazer configurações IPv6 em Linux, tentar usar sempre a linha de comandos para as efectuar. As interfaces gráficas por vezes têm alguns *bugs*, que por vezes podem afectar a correcta configuração do IPv6.
- Por vezes é preferível testar o serviço primeiro em IPv4 e só depois IPv6. Esta opção depende muito do à vontade que o utilizador tem com o serviço. Se o utilizador tiver pouco à vontade com o serviço, pode ocorrer o problema ser da configuração do próprio serviço e não ter nada a ver com a implementação IPv6.
- Ter bastante atenção nos endereços quando se utiliza a sua forma literal. Ao configurar endereços IPv6 a sua extensão é muito propícia a erros de troca de caracteres.
- O conceito de começar no mais simples e a pouco e pouco ir avançando, é extremamente aplicável no IPv6. Convém então, primeiro verificar os endereços *link-local*, só depois configurar o *router* e daí visualizar endereços globais, etc.

5. Conclusão

O projecto começou por abordar diversas noções teóricas e foi progredindo ao longo de todos os capítulos para uma vertente cada vez mais prática, terminando na prática pura. A informação em termos de IPv6 é imensa, e talvez por isso tenha sido complicado encontrar informação bastante recente. Apesar de terem existido diversas mudanças, principalmente a nível da atribuição do endereçamento IPv6, a base teórica do IPv6 continua a mesma de quando foi especificado. O mesmo não se verifica com a parte prática: em termos de implementação, o suporte IPv6 dos sistemas operativos, serviços e aplicações tem variado ao longo do tempo, com a tendência de existir cada vez mais suporte. Sendo assim, a parte prática de implementação de todo este projecto funcionará apenas como um ponto de situação, correndo felizmente o risco de ver o seu suporte aumentado numa questão de meses. O que se verifica actualmente é que todo o núcleo principal IPv6 está criado, faltando apenas algum acompanhamento desta tecnologia por parte de alguns protocolos e principalmente aplicações. A quantidade de RFCs associada ao IPv6 ultrapassa já a centena, o que é bastante representativo de tudo o que já está especificado para a sua implementação, e grande parte delas tem a ver com determinada relação que o IPv6 tem com outras tecnologias. A culpa é do choque em cadeia proporcionado pelo IPv6 em muitos outros assuntos relacionados com as redes, tendo o IPv6 de esperar sempre pelas modificações relativas a eles.

O que nos chamou mais a atenção em toda a parte prática foi, sem dúvida, a facilidade e rapidez em todas as configurações. O mecanismo de auto-configuração é também, indubitavelmente, uma das grandes vantagens, que por si só pode justificar o uso de IPv6. Depois de se usar este novo protocolo, voltar ao IPv4 para administrar uma rede pode revelar-se até aborrecido. Infelizmente, parece-nos ainda prematuro implementar uma rede de grandes dimensões em IPv6. Isso acontece principalmente devido à quantidade de serviços que pode ser disponibilizada por uma rede dessas dimensões, pois existiriam sempre alguns serviços com um suporte IPv6 muito rudimentar ou até serviços sem suporte deste protocolo. Pelo contrário, para uma rede de pequenas dimensões, a sua implementação é já bastante fiável, pois os principais serviços já têm um bom suporte IPv6.

No que diz respeito às questões teóricas, verifica-se o resultado de uma aprendizagem de cerca de 15 anos com o IPv4. Tudo o que era bom no IPv4 ficou ainda melhor, e tudo o que era menos bom foi substituído por diferentes mecanismos que dão qualidades ao IPv6, desde a eficácia à facilidade.

No âmbito deste projecto elaborou-se um guia de instalação de rápida do IPv6 como forma de divulgar e promover o uso do novo protocolo no *campus* da ESTG. Criou-se também uma página *web* que pretende ser a face do projecto IPv6@ESTG-Leiria.

O projecto pretende ser motivador e ser um ponto de partida para diversos projectos relacionados com esta tecnologia, mas bem mais específicos, aspirando ser a base de um conjunto de projectos IPv6.

Referências

Livros:

[1] HAGEN, S. – *IPv6 Essentials*, O'Reilly, ISBN 0596001258, 2002.

[2] DAVIES, J. – *Understandig IPv6*, Microsoft Press, ISBN 0735612455, 2002.

[3] MALONE, D.; MURPHY, N. R. – *IPv6 Network Administration*, O'Reilly, ISBN 0596009348, 2005.

Documentos:

[4] ROSÁRIO, C.; DIAS, H. – *Projecto Piloto IPv6*, ESTG-Leiria, 2001.

[5] SANTOS, F.; ROMANO, L. – *DHCP Relay Agent IPv6*, ESTG-Leiria, 2003.

[6] GOMES, N.; SILVA, R. – *Trabalho sobre IPv6 da disciplina de Tecnologias de Redes sem Fios*, ESTG-Leiria, 2004.

[7] *Cisco IOS IPv6 Configuration Library*, Cisco Systems, Inc., 2005, http://www.cisco.com/en/US/products/sw/iosswrel/ps5187/products_configuration_guide_book09186a00801d65f9.html.

[8] *The ABCs of IP Version 6*, Cisco Systems, Inc., 2002, http://www.cisco.com/application/pdf/en/us/guest/products/iosswrel/c1127/cdccont_0900aecd8018e369.pdf.

[9] *IPv6 Routing at a Glance*, Cisco Systems, Inc., 2005, http://www.cisco.com/application/pdf/en/us/guest/tech/tk872/c1482/cdccont_0900aecd80260051.pdf.

[10] HAGEN, S. – *Sunny Paper: The IPv6 Case - Questions and Answers*, Sunny Connection, 2004, http://www.sunny.ch/downloadfiles/SunnyPaper_IPv6case.pdf.

[11] LEE, D.; STEWART, E. – *Internet Protocol version 6 (IPv6) Conformance and Performance Testing*, Ixia, 2004, http://www.ixiacom.com/white_papers/pdfs/ipv6.pdf.

[12] BIERINGER, P. – *Linux IPv6 HOWTO (en) v0.48*, 11 de Janeiro de 2005, http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/pdf/Linux+IPv6-HOWTO.pdf.

[13] PALET, J.; et al. – *Deliverable D3.4 IPv6 Overall Status v1.3*, IPv6 TF-SC Consortium, 29 de Fevereiro de 2004, http://www.ipv6tf-sc.org/html/public/ipv6tf-sc_pu_d3_4v1_3.pdf.

[14] PALET, J.; et al. – *Deliverable D4 Final Project Report v1.9*, IPv6 TF-SC Consortium, 29 de Novembro de 2004, http://www.ipv6tf-sc.org/html/public/ipv6tf-sc_pu_d4v1_9.pdf.

[15] *IPv6 Address Allocation and Assignment Policy*, APNIC, ARIN, RIPE NCC, 22 de Janeiro de 2003, http://www.ripe.net/ripe/docs/ipv6policy.html.

[16] *IPv6 Resource Allocations*, APNIC, 9 de Maio de 2005, http://geoff.telstra.net/iso3166/v6.html.

[17] *TCP/IP Fundamentals for Microsoft Windows*, Miscrosoft, 2004, http://www.microsoft.com/technet/itsolutions/network/evaluate/technol/tcpipfund/tcpipfund_ch04.mspx.

[18] TORTONESI, M. – *Obtaining the best IPv6 support with Linux*, Deep Space 6, 2004, http://www.deepspace6.net/docs/best_ipv6_support.html.

[19] BIERINGER, P.; et al. – *Current Status of IPv6 Support for Networking Applications*, Deep Space 6, 2004, http://www.deepspace6.net/docs/ipv6_status_page_apps.html.

[20] BAPTISTA, M.; DOMINGUES, M. – *DNS em IPv6 v1.12*, FCCN, 2004, http://www.fccn.pt/files/documents/D2.06.1.PDF.

[21] GORDON, D.; HADDAD, I. – *Building a Linux IPv6 DNS Server*, Ericsson, 2003, http://www.linux.ericsson.ca/ipv6/dns_v6.pdf.

[22] SILVA, G. N. – *APT HOWTO v1.8.10.4*, Março de 2005, Debian, http://www.debian.org/doc/manuals/apt-howto/index.en.html.

Artigos:

[23] MORTON, D. – *Understandig IPv6*, PC Network Advisor, Vol. 83, 1997, http://www.pcsupportadvisor.com/nasample/c0655.pdf.

[24] KRAPF, E. – *Whatever Happened To IPv6*, Abril de 2001, Business Communications Review, http://www.bcr.com/bcrmag/2001/04/p14.php.

[25] HAGEN, S. – *IPv6: Revitalizing the Internet Revolution*, O'Reilly, 25 de Setembro de 2002, http://www.oreillynet.com/pub/a/network/2002/09/25/IPv6Essentials.html.

[26] MONTFORT, N. – *Breaking Protocol*, Wired Magazine, Dezembro de 1999, http://www.wired.com/wired/archive/7.12/ipv6.html.

[27] *Together we're stronger*, 6LINK, 13 de Março de 2002, http://www.6link.org/press.html.

[28] HOUSTON G.; LADID, L.; BOUND J. – *Response by IPv6 Forum to ISP Column article entitled "Waiting for IP version 6"*, Internet Society, 2004, http://www.isoc.org/pubs/isp/ipv6response.shtml.

[29] FORD, M. – *Security and IPv6*, BT Exact, 2005, http://www.ipv6.bt.com/tutorials/security.html.

[30] *IPv6 FAQs*, IPv6style, 2004, http://www.ipv6style.jp/en/faq/latest.shtml.

Apresentações:

[31] CHOWN, T. – *IPv6 and Qos*, IPv6 Forum, 2003, http://www.6init.com/public/iir_qos04.pdf.

[32] *O projecto 6NET*, FCCN, 2004, http://www.minhocampusparty.org/stuff/interno/ipv63.ppt.

[33] LOPÉZ TOLEDO, A. – *QoS in IPv6*, Madrid 2002 Global IPv6 Summit, Março de 2002, http://www.ipv6-es.com/02/in/i-agenda.htm, http://www.ipv6-es.com/02/docs/alberto_lopez_1.pdf.

[34] O'DONOVAN, S. – *Making IPv6 a Reality*, Microsoft Corporation, 2004, http://usipv6.unixprogram.com/usipv6_reston_2004/wed/ODonovan.pdf.

[35] *IPv6 Concepts, Cisco Systems*, Inc., 2004, http://www.cisco.com/warp/public/732/Tech/ipv6/docs/ipv6concepts.pdf.

[36] *IPv6 Deployment, Cisco Systems*, Inc., 2004, http://www.cisco.com/warp/public/732/Tech/ipv6/docs/ipv6deployment.pdf.

[37] TREMBLAY, J. – *Introduction to ICMPv6, Neighbor Discovery, Autoconfiguration and IPv6 DNS*, Hexago, 2004, http://www.hexago.com/docs/doc_20041207-16.pdf.

[38] BLANCHET, N.; PARENT, F. – *IPv6 transition mechanisms*, Viagénie, Maio de 2000, http://www.viagenie.qc.ca/en/ipv6/presentations/IPv6-transition-mechanisms_v1.pdf.

[39] LOURDELET, B. – *IPv6 Routing Protocols*, Cisco Systems, Inc., 2001, http://aristote1.aristote.asso.fr/Presentations/IPv6-2002/P/Lourdelet-Routing/IPv6-routing.pdf.

[40] SANTOS, L.; ANTUNES, M. – *Diapositivos das aulas de IPv6 de Laboratório de Redes*, ESTG-Leiria, 2005.

RFCs:

[41] POSTEL, J. – *Internet Protocol (RFC0791) (STD0005)*, 1981, ftp://ftp.rfc-editor.org/in-notes/rfc791.txt, ftp://ftp.rfc-editor.org/in-notes/std/std5.txt.

[42] POSTEL, J. – *Internet Control Message Protocol (RFC0792) (STD0005)*, 1981, ftp://ftp.rfc-editor.org/in-notes/rfc792.txt, ftp://ftp.rfc-editor.org/in-notes/std/std5.txt.

[43] PLUMMER, D. C. – *Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware (RFC0826) (STD0037)*, 1982, ftp://ftp.rfc-editor.org/in-notes/rfc826.txt, ftp://ftp.rfc-editor.org/in-notes/std/std37.txt.

[44] DEERING, S., Ed. – *ICMP Router Discovery Messages (RFC1256), 1991*, ftp://ftp.rfc-editor.org/in-notes/rfc1256.txt.

[45] PARTRIDGE, C.; MENDEZ, T.; MILLIKEN W. – *Host Anycasting Service (RFC1546)*, 1993, ftp://ftp.rfc-editor.org/in-notes/rfc1546.txt.

[46] THOMSON, S.; HUITEMA, C. – *DNS Extensions to support IP version 6 (RFC1886)*, 1995, ftp://ftp.rfc-editor.org/in-notes/rfc1886.txt.

[47] REKHTER, Y.; MOSKOWITZ, B.; KARRENBERG, D.; GROOT, G. J. de; LEAR, E. – *Address Allocation for Private Internets (RFC1918) (BCP0005)*, 1996, ftp://ftp.rfc-editor.org/in-notes/rfc1918.txt, ftp://ftp.rfc-editor.org/in-notes/bcp/bcp5.txt.

[48] ELZ, R. – *A Compact Representation of IPv6 Addresses (RFC1924)*, 1996, ftp://ftp.rfc-editor.org/in-notes/rfc1924.txt.

[49] MCCANN, J.; DEERING, S.; MOGUL, J. – *Path MTU Discovery for IP version 6 (RFC1981)*, 1996, ftp://ftp.rfc-editor.org/in-notes/rfc1981.txt.

[50] MALKIN, G.; MINNEAR, R. – *RIPng for IPv6 (RFC2080)*, 1997, ftp://ftp.rfc-editor.org/in-notes/rfc2080.txt.

[51] KENT, S.; ATKINSON, R. – *IP Authentication Header (RFC2402)*,1998, ftp://ftp.rfc-editor.org/in-notes/rfc2402.txt.

[52] KENT, S.; ATKINSON, R. – *IP Encapsulating Security Payload (ESP) (RFC2406)*, 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2406.txt.

[53] DEERING, S.; HINDEN, R. – *Internet Protocol, Version 6 (IPv6) Specification (RFC2460)*, 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2460.txt.

[54] NARTEN, T.; NORDMARK, E.; SIMPSON, W. – *Neighbor Discovery for IP Version 6 (IPv6) (RFC2461)*, 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2461.txt.

[55] THOMSON, S.; NARTEN, T. – *IPv6 Stateless Address Autoconfiguration (RFC2462)*, 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2462.txt.

[56] CONTA, A.; DEERING, S. – *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification (RFC2463)*, 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2463.txt.

[57] NICHOLS, K.; BLAKE, S.; F., BLACK, D. – *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers (RFC2474)*, 1998, ftp://ftp.rfc-editor.org/in-notes/rfc2474.txt.

[58] MARQUES, P.; DUPONT, F. – *Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing (RFC2545)*, 1999, ftp://ftp.rfc-editor.org/in-notes/rfc2545.txt

[59] DEERING, S.; FENNER, W.; HABERMAN, B. – *Multicast Listener Discovery (MLD) for IPv6 (RFC2710)*, 1999, ftp://ftp.rfc-editor.org/in-notes/rfc2710.txt.

[60] PARTRIDGE, C.; JACKSON, A. – *IPv6 Router Alert Option (RFC2711)*, 1999, ftp://ftp.rfc-editor.org/in-notes/rfc2711.txt.

[61] COLTUN, R.; FERGUSON, D.; MOY, J. – *OSPF for IPv6 (RFC2740)*, 1999, ftp://ftp.rfc-editor.org/in-notes/rfc2740.txt.

[62] NORDMARK, E. – *Stateless IP/ICMP Translation Algorithm (SIIT) (RFC2765)*, 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2765.txt.

[63] TSIRTSIS, G.; SRISURESH, P. – *Network Address Translation - Protocol Translation (NAT-PT) (RFC2766)*, 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2766.txt.

[64] FERGUSON, P.; SENIE, D. – *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing (RFC2827) (BCP0038)*, 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2827.txt, ftp://ftp.rfc-editor.org/in-notes/bcp/bcp38.txt.

[65] BATES, T.; REKHTER, Y.; CHANDRA, R.; KATZ, D. – *Multiprotocol Extensions for BGP-4 (RFC2858)*, 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2858.txt.

[66] CRAWFORD, M.; HUITEMA, C. – *DNS Extensions to Support IPv6 Address Aggregation and Renumbering (RFC2874)*, 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2874.txt.

[67] GILLIGAN, R.; NORDMARK, E. – *Transition Mechanisms for IPv6 Hosts and Routers (RFC2893)*, 2000, ftp://ftp.rfc-editor.org/in-notes/rfc2893.txt.

[68] NARTEN, T.; DRAVES, R. – *Privacy Extensions for Stateless Address Autoconfiguration in IPv6 (RFC3041)*, 2001, ftp://ftp.rfc-editor.org/in-notes/rfc3041.txt.

[69] CARPENTER, B.; MOORE, K. – *Connection of IPv6 Domains via IPv4 Clouds (RFC3056)*, 2001, ftp://ftp.rfc-editor.org/in-notes/rfc3056.txt.

[70] HUITEMA, C. – *An Anycast Prefix for 6to4 Relay Routers (RFC3068)*, 2001, ftp://ftp.rfc-editor.org/in-notes/rfc3068.txt.

[71] HAGINO, J.; YAMAMOTO, K. – *An IPv6-to-IPv4 Transport Relay Translator (RFC3142)*, 2001, ftp://ftp.rfc-editor.org/in-notes/rfc3142.txt.

[72] IAB; IESG – *IAB/IESG Recommendations on IPv6 Address Allocations to Sites (RFC3177)*, 2001, ftp://ftp.rfc-editor.org/in-notes/rfc3177.txt.

[73] DROMS, R., Ed.; BOUND, J.; VOLZ, B.; LEMON, T.; PERKINS, C.; CARNEY, M. – *Dynamic Host Configuration Protocol for IPv6 (DHCPv6) (RFC3315)*, 2003, ftp://ftp.rfc-editor.org/in-notes/rfc3315.txt.

[74] BUSH, R.; DURAND, A.; FINK, B.; GUDMUNDSSON, O.; HAIN, T. – *Representing Internet Protocol version 6 (IPv6) Addresses in the Domain Name System (DNS) (RFC3363)*, 2002, ftp://ftp.rfc-editor.org/in-notes/rfc3363.txt.

[75] AUSTEIN, R. – *Tradeoffs in Domain Name System (DNS) Support for Internet Protocol version 6 (IPv6) (RFC3364)*, 2002, ftp://ftp.rfc-editor.org/in-notes/rfc3364.txt.

[76] CAIN, B.; DEERING, S.; KOUVELAS, I.; FENNER, B.; THYAGARAJAN, A. – *Internet Group Management Protocol, Version 3 (RFC3376)*, 2002, ftp://ftp.rfc-editor.org/in-notes/rfc3376.txt.

[77] HINDEN, R.; DEERING, S. – *Internet Protocol Version 6 (IPv6) Addressing Architecture (RFC3513)*, 2003, ftp://ftp.rfc-editor.org/in-notes/rfc3513.txt.

[78] HINDEN, R.; DEERING, S.; NORDMARK, E. – *IPv6 Global Unicast Address Format (RFC3587)*, 2003, ftp://ftp.rfc-editor.org/in-notes/rfc3587.txt.

[79] RAJAHALME, J.; CONTA, A.; CARPENTER, B.; DEERING, S. – *IPv6 Flow Label Specification (RFC3697)*, 2004, ftp://ftp.rfc-editor.org/in-notes/rfc3697.txt.

[80] FINK, R.; HINDEN, R. – *6bone (IPv6 Testing Address Allocation) Phaseout (RFC3701)*, 2005, ftp://ftp.rfc-editor.org/in-notes/rfc3701.txt.

[81] VIDA, R., Ed.; COSTA, L., Ed. – *Multicast Listener Discovery Version 2 (MLDv2) for IPv6 (RFC3810)*, 2004, ftp://ftp.rfc-editor.org/in-notes/rfc3810.txt.

[82] HUITEMA, C.; CARPENTER, B. – *Deprecating Site Local Addresses (RFC3879)*, 2004, ftp://ftp.rfc-editor.org/in-notes/rfc3879.txt.

[83] DURAND, A.; IHREN, J. – *DNS IPv6 Transport Operational Guidelines (RFC3901) (BCP0091)*, 2004, ftp://ftp.rfc-editor.org/in-notes/rfc3901.txt, ftp://ftp.rfc-editor.org/in-notes/bcp/bcp91.txt.

[84] BERNERS-LEE, T.; FIELDING, R.; MASINTER, L. – *Uniform Resource Identifier (URI): Generic Syntax (RFC3986) (STD0066)*, 2005, ftp://ftp.rfc-editor.org/in-notes/rfc3986.txt, ftp://ftp.rfc-editor.org/in-notes/std/std66.txt.

[85] DEERING, S.; HABERMAN, B.; JINMEI, T.; NORDMARK, E.; ZILL, B. – IPv6 *Scoped Address Architecture (RFC4007)*, 2005, ftp://ftp.rfc-editor.org/in-notes/rfc4007.txt.

[86] PETERSON, J. – *A Presence Architecture for the Distribution of GEOPRIV Location Objects (RFC4079)*, 2005, ftp://ftp.rfc-editor.org/in-notes/rfc4079.txt.

Internet Drafts:

[87] KING, S.; FAX, R.; HASKIN, D.; LING, W.; MEEHAN, T.; FINK, R.;PERKINS, C. E. – *The Case for IPv6 (draft-ietf-iab-case-for-ipv6-06)*, 25 de Dezembro de 1999, http://www.6bone.net/misc/case-for-ipv6.html.

[88] HINDEN, R.; HABERMAN, B. – *Unique Local IPv6 Unicast Addresses (draft-ietf-ipv6-unique-local-addr-09)*, 24 de Janeiro de 2005, ftp://ftp.rfc-editor.org/in-notes/internet-drafts/draft-ietf-ipv6-unique-local-addr-09.txt.

[89] HINDEN, R.; DEERING, S. – *IP Version 6 Addressing Architecture (draft-ietf-ipv6-addr-arch-v4-04)*, 23 de Maio de 2005, ftp://ftp.rfc-editor.org/in-notes/internet-drafts/draft-ietf-ipv6-addr-arch-v4-04.txt.

[90] HOPPS, C. – *Routing IPv6 with IS-IS (draft-ietf-isis-ipv6-05)*, 2 de Junho de 2005, ftp://ftp.rfc-editor.org/in-notes/internet-drafts/draft-ietf-isis-ipv6-05.txt.

Normas:

[91] *Intermediate System to Intermediate System Intra-Domain Routeing Information Exchange Protocol for use in Conjunction with the Protocol for Providing the Connectionless-Mode Network Service (ISO 8473) (ISO/IEC 10589:2002)*, 2.ª ed., 2002, http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=30932&ICS1=35.

[92] *3GPP TS 23.221 V6.3.0*, 3rd Generation Partnership Project, Junho de 2004, http://www.3gpp.org/ftp/Specs/archive/23_series/23.221/.

[93] *Guidelines For 64-Bit Global Identifier (EUI-64) Registration Authority*, IEEE, 1998, http://standards.ieee.org/regauth/oui/tutorials/EUI64.html.

Websites:

[94] The IPv6 Portal, http://www.ipv6tf.org/.

[95] IPv6 Forum, http://www.ipv6forum.com/.

[96] IPv6 Forum: Global Forum Events, http://www.ipv6forum.org.uk/navbar/events/global.htm.

[97] Presentations from the IPv6 Forum World Summit, http://www.ipv6forum.org.uk/navbar/globalsummit/slides/paris99.htm.

[98] IPv6 Ready Logo Program, http://www.ipv6ready.org/.

[99] IPv6 Ready Logo Phase-1, http://www.ipv6ready.org/logo_db/approved_list.php.

[100] IPv6 Ready Logo Phase-2, http://www.ipv6ready.org/logo_db/approved_list_p2.php.

[101] IPv6 Information Page, http://www.ipv6.org/.

[102] The Internet Engineering Task Force, http://www.ietf.org/.

[103] RFC Editor Webpage, http://www.rfc-editor.org/.

[104] IP Version 6 Working Group (ipv6), http://www.ietf.org/html.charters/ipv6-charter.html.

[105] Next Generation Transition (ngtrans) working group, http://www.ietf.org/html.charters/OLD/ngtrans-charter.html.

[106] IPv6 Operations (v6ops) working group, http://www.ietf.org/html.charters/v6ops-charter.html.

[107] 6Bone Home Page, http://www.6Bone.net/.

[108] IST IPv6 Cluster, http://www.ist-ipv6.org/.

[109] 6NET, http://www.6net.org/.

[110] Euro6IX, http://www.euro6ix.org/.

[111] GÉANT2 Home, http://www.geant2.net/.

[112] ALICE Home, http://alice.dante.net/.

[113] FCCN, Fundação para a Computação Científica Nacional, http://www.fccn.pt/.

[114] Task Force Portuguesa de IPv6, http://www.ipv6-tf.com.pt/.

[115] IANA, http://www.iana.org/.

[116] IPv6style, http://www.ipv6style.jp/en/.

[117] Tunisia Next Generation Network, http://www.ipv6net.tn/.

[118] JOIN – Test for IPv6 connectivity, http://www.join.uni-muenster.de/TestTools/IPv6_Verbindungstests.php?lang=en.

[119] JOIN – IPv6 in different Software, http://www.join.uni-muenster.de/Implementationen/Software.php?lang=en#Webserver.

[120] Wikipedia – IPv6, http://en.wikipedia.org/wiki/IPv6.

[121] Wikipedia – IPv4, http://en.wikipedia.org/wiki/IPv4.

[122] Wikipedia – IANA, http://en.wikipedia.org/wiki/IANA.

[123] TCP/IP Guide – Internet Protocol Version 6 (IPv6) / IP Next Generation (IPng), http://www.tcpipguide.com/free/t_InternetProtocolVersion6IPv6IPNextGenerationIPng.htm.

[124] Windows Server 2003: Microsoft Internet Protocol Version 6 (IPv6), http://www.microsoft.com/ipv6/, http://www.microsoft.com/windowsserver2003/technologies/ipv6/default.mspx.

[125] Microsoft IPv6 Technology Preview for Windows 2000, http://msdn.microsoft.com/downloads/sdks/platform/tpipv6.asp.

[126] Microsoft TechNet – Frequently Asked Questions About the IPv6 Protocol for Windows XP, http://www.microsoft.com/technet/prodtechnol/winxppro/plan/faqipv6.mspx.

[127] Microsoft Windows Server 2003 – Updating IPv6.exe Commands to Netsh Commands, http://www.microsoft.com/windowsserver2003/technologies/ipv6/ipv62netshtable.mspx.

[128] Microsoft Windows Server 2003 – Frequently Asked Questions About the IPv6 Protocol for the Windows Server 2003 Family, http://www.microsoft.com/windowsserver2003/techinfo/overview/ipv6faq.mspx.

[129] Trumpet Software International Products: IPv6, http://www.trumpet.com.au/ipv6.htm.

[130] MSDN – IPsec6.exe, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winsock/winsock/ipsec6_exe_2.asp.

[131] Microsoft Windows XP – Using IPSec between two local link hosts, http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/sag_ip_v6_imp_conf5.mspx.

[132] Microsoft TechNet – IPv6 Configuration Items, http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/ServerHelp/af7d4a80-cfe8-4d7d-a830-8eb6e6ebd6b7.mspx.

[133] Microsoft Windows XP – IPv6 traffic between nodes in different sites across the Internet (6to4), http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/sag_ip_v6_imp_conf2.mspx.

[134] Microsoft Technet – IPv6 and IIS 6.0 (IIS 6.0), http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/4c7c6bce-213a-4125-bc36-2202e3b4c8c8.mspx.

[135] Cisco IOS Software Technologies – IPv6, http://www.cisco.com/warp/public/732/Tech/ipv6/, http://www.cisco.com/ipv6/.

[136] Start Here: Cisco IOS Software Release Specifics for IPv6 Features, http://www.cisco.com/en/US/products/sw/iosswrel/ps5187/products_configuration_guide_chapter09186a00801d65ed.html.

[137] Cisco IOS IPv6 Command Reference - IPv6 Commands: debug ipv6 dhcp through debug ipv6 rip, http://www.cisco.com/en/US/products/sw/iosswrel/ps5187/products_command_reference_chapter09186a00801d6639.html.

[138] Cisco IOS IPv6 Provider Edge Router (6PE) over MPLS, http://www.cisco.com/en/US/products/sw/iosswrel/ps1835/products_data_sheet09186a0080115501.html.

[139] USAGI Project – Linux IPv6 Development Project, http://www.linux-ipv6.org/.

[140] KAME Project, http://www.kame.net/.

[141] KAME Project Releases, http://www.kame.net/project-overview.html#release.

[142] KAME Patches Directory, ftp://ftp.kame.net/pub/kame/misc/.

[143] TAHI Project, http://www.tahi.org/.

[144] The Linux Kernel Archives, http://www.kernel.org/.

[145] Debian IPv6 Project, http://people.debian.org/%7Ecsmall/ipv6/index.html.

[146] Gentoo IPv6 Router Guide, http://www.gentoo.org/doc/en/ipv6.xml.

[147] IPv6 on Fedora Core mini-HOWTO, http://linux.yyz.us/ipv6-fc2-howto.html.

[148] IPv6 & Linux - Current Status - Applications, http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-apps.html.

[149] IPv6 & Linux - Current Status - Distributions, http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-distributions.html.

[150] IPv6 & Linux - Current Status - Kernel and mandatory features, http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-kernel.html.

[151] FreeBSD Handbook – Chapter 25.10 IPv6, http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/network-ipv6.html.

[152] NetBSD Documentation: NetBSD IPv6 Networking, http://www.netbsd.org/Documentation/network/ipv6/.

[153] Solaris TCP/IP Administration, http://docs.sun.com/app/docs/doc/816-4554/6maoq01lc?q=ipv6&a=view.

[154] IPv6 enablement at IBM, http://www-306.ibm.com/software/os/zseries/ipv6/.

[155] HP-UX 11i Internet Protocols, http://www.hp.com/products1/unix/operating/internet/ipv.html.

[156] IPv6 – Compaq Tru64 Unix, http://h18000.www1.hp.com/IPv6/Tru64UNIX.html.

[157] IPv6 – Compaq TCP/IP services for OpenVMS, http://h18000.www1.hp.com/IPv6/OpenVMS.html.

[158] GNU Zebra, http://www.zebra.org/.

[159] IP Infusion, http://www.ipinfusion.com/.

[160] FreeS/WAN Project, http://www.freeswan.org/.

[161] PHPmac – Installing Apache 2 on Mac OS X, http://www.phpmac.com/articles.php?view=23.

[162] Darwin Apache Archive, http://archive.apache.org/dist/httpd/binaries/darwin/.

[163] IPSEC: secure IP over the Internet, http://lartc.org/howto/lartc.ipsec.html.

[164] IPSec and IPv6: the KAME snapshot for FreeBSD, http://www.eurescom.de/~public-web-deliverables/P1100-series/P1113/D1/41_ipsec_bsd.html.

[165] Dibbler – a portable DHCPv6, http://klub.com.pl/dhcpv6/.

[166] Dibbler – bugs do cliente, http://klub.com.pl/cgi-bin/bugzilla/buglist.cgi?product=Dibbler+client&bug_status=NEW&bug_status=ASSIGNED&bug_status=REOPENED&order=Importance.

[167] Dibbler – bugs do servidor, http://klub.com.pl/cgi-bin/bugzilla/buglist.cgi?product=Dibbler+server&bug_status=NEW&bug_status=ASSIGNED&bug_status=REOPENED&order=Importance.

[168] Dibbler – bugs do relay agent, http://klub.com.pl/cgi-bin/bugzilla/buglist.cgi?product=Dibbler+relay&bug_status=NEW&bug_status=ASSIGNED&bug_status=REOPENED&order=Importance.

[169] dhcpv6 @ sourceforge, http://dhcpv6.sourceforge.net/.

[170] Gentoo-Portage, http://www.gentoo-portage.com/.

[171] Linux@DUKE:Yum, http://linux.duke.edu/projects/yum/.

[172] The Apache Software Foundation, http://www.apache.org/.

[173] Apache HTTP Server – Binding, http://httpd.apache.org/docs-2.0/bind.html.

[174] Firefox, http://www.mozilla.org/products/firefox/.

[175] Opera Web Browser, http://www.opera.com/.

[176] UMTS Forum, http://www.umts-forum.org/.

[177] Faculty of Electronics, Telecommunications and Informatics, Gdansk University, Poland, http://www.eti.pg.gda.pl.

Outros:

[178] *Internet Protocol Version 6 Address Space*, IANA, 23 de Junho de 2005, http://www.iana.org/assignments/ipv6-address-space.

[179] *IPv6 Global Unicast Address Assignments*, IANA, 8 de Julho de2005, http://www.iana.org/assignments/ipv6-unicast-address-assignments.

[180] *Internet Protocol Version 6 Multicast Addresses*, IANA, 24 de Junho de 2005, http://www.iana.org/assignments/ipv6-multicast-addresses.

[181] *ICMPv6 Type Numbers*, IANA, 25 de Junho de 2005, http://www.iana.org/assignments/icmpv6-parameters.

[182] *Protocol Numbers*, IANA, 18 de Outubro de 2004, http://www.iana.org/assignments/protocol-numbers.

[183] Notícias Lusa, 2005.

Anexos

A.1. Configurações relativas ao cenário dos serviços

Nesta secção apresentam-se as configurações relativas ao cenário de serviços, a configuração do *router* e do servidor e clientes DHCPv6.

A.1.1. Configurações do router

```
Current configuration : 977 bytes
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R1
!
boot-start-marker
boot-end-marker
!
no network-clock-participate slot 1
no network-clock-participate wic 0
no aaa new-model
ip subnet-zero
!
ip cef
ip audit po max-events 100
ipv6 unicast-routing
no ftp-server write-enable
!
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
 ipv6 address A::/64 eui-64
 ipv6 enable
!
interface Serial0/0
 no ip address
 shutdown
!
interface FastEthernet0/1
 no ip address
 duplex auto
 speed auto
 ipv6 address B::/64 eui-64
 ipv6 enable
!
interface Serial0/1
 no ip address
 shutdown
!
ip classless
!
ip http server
no ip http secure-server
!
```

```
line con 0
line aux 0
line vty 0 4
 login
!
end
```

A.1.2. Configuração do servidor de DHCPv6 no Windows Server 2003

```
#---------------------------------
#server.conf (Windows Server 2003)
#---------------------------------

log-level 8
log-mode short

#interface escolhida
iface "Local Area Connection"
{
    #abrangência
    class
    {
        pool CAFE::1-CAFE::FFFF #pool de endereços [MIN]-[MAX]
    }
}
```

A.1.3. Configuração do cliente de DHCPv6 no Windows XP

```
#------------------------
#client.conf (Windows XP)
#------------------------
#
# This is a configuration file for Dibbler client.
#

# modo de logging
log-mode short

# Interface escolhida
iface eth0
{
    # abrangência (scope)
    IA
    {
        address #endereço aleatório da pool de endereços do servidor
    }
}
```

A.1.4. Configuração do cliente de DHCPv6 no Linux

```
#-------------------
#client.conf (Linux)
#-------------------
#
# This is a configuration file for Dibbler client.
#

# modo de logging
log-mode short

# Interface escolhida
iface eth0
{
    # abrangência (scope)
```

```
    IA
    {
        address #endereço aleatório da pool de endereços do servidor
    }
}
```

A.2. Configurações dos routers do cenário de encaminhamento

Nesta secção apresentam-se as configurações dos dois *routers* utilizados no cenário de encaminhamento.

A.2.1. Configuração do router R1 com OSPF

```
Current configuration : 853 bytes
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R1
!
boot-start-marker
boot-end-marker
!
no network-clock-participate slot 1
no network-clock-participate wic 0
no aaa new-model
ip subnet-zero
!
ip cef
ip audit po max-events 100
ipv6 unicast-routing
no ftp-server write-enable
!
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
 ipv6 address A::/64 eui-64
 ipv6 enable
 ipv6 ospf 1 area 0
!
interface Serial0/0
 no ip address
 ipv6 enable
 ipv6 ospf 1 area 0
 no fair-queue
!
interface Serial0/1
 no ip address
 shutdown
!
ip classless
!
ip http server
no ip http secure-server
!
ipv6 router ospf 1
 router-id 0.0.0.1
 log-adjacency-changes
!
line con 0
line aux 0
line vty 0 4
!
```

```
end
```

A.2.2. Configuração do router R2 com OSPF

```
Current configuration : 949 bytes
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R2
!
boot-start-marker
boot-end-marker
!
no network-clock-participate slot 1
no network-clock-participate wic 0
no aaa new-model
ip subnet-zero
!
ip cef
ip audit po max-events 100
ipv6 unicast-routing
no ftp-server write-enable
!
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
 ipv6 address B::/64 eui-64
 ipv6 enable
 ipv6 ospf 1 area 0
!
interface Serial0/0
 no ip address
 ipv6 enable
 ipv6 ospf 1 area 0
 no fair-queue
 clockrate 115200
!
interface FastEthernet0/1
 no ip address
 shutdown
 duplex auto
 speed auto
!
interface Serial0/1
 no ip address
 shutdown
!
ip classless
!
ip http server
no ip http secure-server
!
ipv6 router ospf 1
 router-id 0.0.0.2
 log-adjacency-changes
!
line con 0
line aux 0
line vty 0 4
!
end
```

A.2.3. Configuração do router R1 com RIP

```
Current configuration : 853 bytes
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R1
!
boot-start-marker
boot-end-marker
!
no network-clock-participate slot 1
no network-clock-participate wic 0
no aaa new-model
ip subnet-zero
!
ip cef
ip audit po max-events 100
ipv6 unicast-routing
no ftp-server write-enable
!
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
 ipv6 address A::/64 eui-64
 ipv6 enable
 ipv6 rip teste-ripng enable
!
interface Serial0/0
 no ip address
 ipv6 enable
 ipv6 rip teste-ripng enable
 no fair-queue
!
interface Serial0/1
 no ip address
 shutdown
!
ip classless
!
ip http server
no ip http secure-server
!
ipv6 router rip teste-ripng
!
line con 0
line aux 0
line vty 0 4
!
end
```

A.2.4. Configuração do router R2 com RIP

```
Current configuration : 949 bytes
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R2
!
boot-start-marker
```

```
boot-end-marker
!
no network-clock-participate slot 1
no network-clock-participate wic 0
no aaa new-model
ip subnet-zero
!
ip cef
ip audit po max-events 100
ipv6 unicast-routing
no ftp-server write-enable
!
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
 ipv6 address B::/64 eui-64
 ipv6 enable
 ipv6 rip teste-ripng enable
!
interface Serial0/0
 no ip address
 ipv6 enable
 ipv6 rip teste-ripng enable
 no fair-queue
 clockrate 115200
!
interface FastEthernet0/1
 no ip address
 shutdown
 duplex auto
 speed auto
!
interface Serial0/1
 no ip address
 shutdown
!
ip classless
!
ip http server
no ip http secure-server
!
ipv6 router rip teste-ripng
!
line con 0
line aux 0
line vty 0 4
!
end
```

A.3. Configurações relativas ao cenário de acesso ao exterior

Nesta secção apresentam-se as configurações relativas ao cenário de acesso ao exterior.

A.3.1. Configuração do terminal T-Tunel

```
#========================
# Interface configuration
#========================
pushd interface

reset all

popd
# End of interface configuration
```

```
#========================
# Interface configuration
#========================
pushd interface ipv6

install
reset

set state v4compat=enabled
add route prefix=::/0 interface="Tunel-FCCN" metric=0 nexthop=2001:690:810:11::1

add route prefix=::/96 interface="Automatic Tunneling Pseudo-Interface" metric=0

add v6v4tunnel interface="Tunel-FCCN" localaddress=193.137.239.44 remoteaddress=
193.136.2.246
add address interface="Tunel-FCCN" address=2001:690:810:11::2

popd
# End of interface configuration

# ---------------------------------
# ISATAP Configuration
# ---------------------------------
pushd interface ipv6 isatap

popd
# End of ISATAP configuration

# ---------------------------------
# 6to4 Configuration
# ---------------------------------
pushd interface ipv6 6to4

reset

popd
# End of 6to4 configuration

#========================
# Port Proxy configuration
#========================
pushd interface portproxy

reset

popd
# End of Port Proxy configuration

# ---------------------------------
# Interface IP Configuration
# ---------------------------------
pushd interface ip

# Interface IP Configuration for "Local Area Connection"

set address name="Local Area Connection" source=static addr=193.137.239.43 mask=
255.255.255.224
set address name="Local Area Connection" gateway=193.137.239.62 gwmetric=0
set dns name="Local Area Connection" source=static addr=193.137.239.226 register
=PRIMARY
set wins name="Local Area Connection" source=static addr=none

popd
# End of interface IP configuration
```

A.3.2. Configuração do router R-Tunel

```
Current configuration : 1035 bytes
```

```
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R-Tunel
!
boot-start-marker
boot-end-marker
!
no network-clock-participate slot 1
no network-clock-participate wic 0
no aaa new-model
ip subnet-zero
!
ip cef
ip audit po max-events 100
ipv6 unicast-routing
no ftp-server write-enable
!
interface Tunnel0
 no ip address
 ipv6 address 2001:690:810:11::2/64
 ipv6 enable
 tunnel source 193.137.239.44
 tunnel destination 193.136.2.246
 tunnel mode ipv6ip
!
interface FastEthernet0/0
 ip address 193.137.239.44 255.255.255.224
 duplex auto
 speed auto
 ipv6 enable
!
interface Serial0/0
 no ip address
 ipv6 enable
!
interface Serial0/1
 no ip address
 shutdown
!
ip classless
ip route 0.0.0.0 0.0.0.0 193.137.239.62
!
ip http server
no ip http secure-server
!
ipv6 route 2001:690:2067::/48 Serial0/0
ipv6 route ::/0 2001:690:810:11::1
!
line con 0
line aux 0
line vty 0 4
 login
!
end
```

A.3.3. Configuração do router R-Rede

```
Current configuration : 977 bytes
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
```

```
hostname R-Rede
!
boot-start-marker
boot-end-marker
!
no network-clock-participate slot 1
no network-clock-participate wic 0
no aaa new-model
ip subnet-zero
!
ip cef
ip audit po max-events 100
ipv6 unicast-routing
no ftp-server write-enable
!
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
 ipv6 address 2001:690:2067:1::/64 eui-64
 ipv6 enable
!
interface Serial0/0
 no ip address
 ipv6 enable
 clockrate 115200
!
interface FastEthernet0/1
 no ip address
 duplex auto
 speed auto
 ipv6 address 2001:690:2067:2::/64 eui-64
 ipv6 enable
!
interface Serial0/1
 no ip address
 shutdown
!
ip classless
!
ip http server
no ip http secure-server
!
ipv6 route ::/0 Serial0/0 2001:690:810:11::2
!
line con 0
line aux 0
line vty 0 4
 login
!
end
```