Encaminhamento *Anycast* em Redes *IPv6*: uma proposta

Hugo Ferreira
Centro ALGORITMI
Universidade do Minho
E-mail: a50046@alunos.uminho.pt

Maria João Nicolau
Centro ALGORITMI
Universidade do Minho
E-mail: joao@dsi.uminho.pt

António Costa
Centro ALGORITMI
Universidade do Minho
E-mail: costa@di.uminho.pt

***Resumo*—O aparecimento do protocolo de comunicação *IPv6* introduziu um novo paradigma de comunicação, denominado *anycast* (um-para-um-de-muitos). Este novo paradigma, utiliza o conceito de grupo, à semelhança do que acontece com o *multicast*, mas em oposição a este, a informação é enviada apenas para um dos membros do grupo (tipicamente o mais próximo) e não para todos. Embora já se tenham passado alguns anos desde o seu aparecimento , o *anycast* tem sofrido uma lenta evolução, contribuindo para esta situação o facto de não existir ainda um protocolo normalizado, que permita às aplicações usar de forma generalizada este paradigma de comunicação.**

Tradicionalmente as soluções para o problema de encaminhamento *anycast* são simplesmente baseadas no encaminhamento *unicast* sem alterações. No entanto, e tratando-se de um paradigma que usa o conceito de grupo, é de esperar que os protocolos de encaminhamento *multicast*, ou alguma variante destes, possam vir a constituir uma boa solução para a implementação do *anycast* ao nível da rede. O presente artigo apresenta um levantamento de propostas relacionadas com o tema e propõe um novo protocolo de encaminhamento *anycast* baseado no protocolo *PIM-SM (Protocol Independent Multicast -Sparse Mode)*, denominado *Tree-based Anycast Protocol (TAP)*. As alterações propostas ao protocolo *PIM-SM* são apresentadas na especificação do sistema, tendo sido o seu correto funcionamento aferido recorrendo ao *Network Simulator 2 (ns-2.35)*.

Palavras-chave:*Anycast*, Encaminhamento, Redes, *IPv6*, *PIM-SM*, *TAP*.

I. Introdução

O protocolo de comunicação *IPv6* incluí três paradigmas de comunicação - o *unicast*, o *multicast* e o *anycast*. O *unicast* é o mais comum, estimando-se que noventa por cento do tráfego seja de um para um. O tráfego *multicast* (um-para-muitos) surgiu como uma evolução natural do antigo *broadcast* (um-para-todos), onde a comunicação deixa de ser ”para todos”e aparece o conceito de grupo. Passa a ser possível comunicar com um grupo definido de utilizadores. O *anycast* é originalmente proposto[1] a Novembro de 1993, sendo particularmente útil para implementar serviços que podem ser disponibilizados por múltiplos servidores. É atribuído um único endereço *anycast* a todos os sistemas terminais que disponibilize exatamente o mesmo serviço. Para os clientes desse serviço, basta-lhes dirigir o pedido ao endereço *anycast* e esperar que um dos sistemas (e apenas um) lhe responda. Do ponto de vista do serviço prestado é indiferente qual deles é, mas a escolha do sistema em melhores condições é um problema de encaminhamento específico da rede.

Os endereços *anycast* são sintaticamente indistinguíveis dos endereços *unicast*[2]. A lógica inerente a este tipo de endereçamento é de um endereço *unicast* em diferentes locais. A ideia visa permitir a um provedor de serviços aumentar a capacidade de balanceamento de carga, acrescentando um novo servidor numa outra rede. Quando um utilizador requer um determinado serviço, este é encaminhado para o servidor mais próximo da sua localização (com menor custo), utilizando o método de encaminhamento do *unicast*.

Na Figura 1 é possível observar um exemplo de uma comunicação *anycast*. Um endereço *anycast* único é atribuído aos três provedores de serviços - `Servidor Anycast 1`, `Servidor Anycast 2`, `Servidor Anycast 3`. Quando um `Cliente Anycast` deseja efetuar um pedido, cria um pacote *anycast* (pacote que contém um endereço *anycast* como destino) e envia esse pacote para a rede. Após o envio do pacote até à receção por parte do servidor mais próximo, é levado a cabo um processo de seleção (normalmente o caminho de menor custo). Aquando da receção do pacote *anycast* por parte do servidor mais próximo do cliente, é enviada a resposta ao pedido do cliente, idealmente utilizando o endereço *anycast* no campo de origem de modo a evitar problemas na altura da receção do pacote pelo cliente.

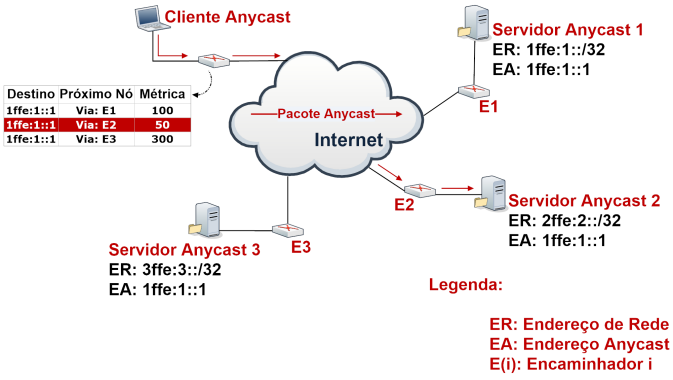

Figura 1: Esquema do encaminhamento *anycast*.

No encaminhamento global, o *anycast* prejudica a agregação de rotas pois permite que um mesmo endereço

apareça associado a várias redes distintas. Como podemos observar na Figura 1, é possível ao `Servidor Anycast 2` e ao `Servidor Anycast 3` estarem em redes diferentes da rede a que pertence o do endereço *anycast*. Um encaminhador que receba rotas relativas a estas redes, é facilmente iludido. Para conseguir distingui-las, é necessário adicionar entradas separadas a todos os encaminhadores ao longo da rede. Utilizando o mesmo exemplo, caso os servidores se encontrem em redes com endereços dispares pode ser muito complicado (ou mesmo impossível) efetuar a agregação de rotas. É necessário um novo sistema para encaminhamento, para retirar partido do potencial do *anycast*. Existem diversas propostas no sentido de atacar esta problemática, mas existem ainda muitos problemas por resolver[3][4][5][6][7][8].

Não obstante ao problema da agregação das rotas, o encaminhamento *anycast* possuí diversas características interessantes. Uma das características mais importantes do *anycast* quando realizado na camada de rede é não ser necessário ao cliente conhecer a topologia total da rede, o que lhe confere uma enorme escalabilidade. Quando o `Cliente Anycast` na Figura 1 faz um pedido, este é reencaminhado para o nó mais próximo dele (neste caso o `Servidor Anycast 1`). A grande vantagem acontece quando por qualquer motivo (por exemplo falha de *hardware*) o nó onde habita o `Servidor Anycast 1` deixa de estar acessível. Neste caso o seu pedido será na mesma respondido, mas por um servidor a uma maior distância.

O *anycast* tem enumeras aplicações, das quais se destacam os serviços dependentes da localização, a descoberta de serviços e o balanceamento de carga.

Atualmente, uma das mais populares aplicações são os serviços dependentes da localização. A escolha do servidor mais próximo do requerente do serviço é uma questão cada vez mais critica. Normalmente passa por apresentar uma lista de servidores distribuídos por múltiplos locais e oferecer o poder de seleção ao requerente do serviço. Com a utilização do *anycast* neste tipo de aplicações, garante-se que é feita uma seleção mais precisa para o encaminhamento do pedido. Em vez de confiar no juízo do requerente, realiza-se um encaminhamento mais transparente a partir de qualquer parte do mundo. Com isto o *anycast* assume um papel essencial pois permite escolher o nó mais próximo, fazendo assim com que exista uma resposta mais célere e eficaz.

Para ser possível a descoberta de serviços, primeiramente é atribuído um endereço *anycast* a um serviço, e de seguida é atribuído aos nós que suportam os servidores desse mesmo serviço. Isto permite uma grande tolerância a falhas na rede, pois o pacote *anycast* será reencaminhado para um outro local apropriado. Este tipo de serviço assume uma grande importância em redes com grande escalabilidade, como redes móveis *ad hoc* ou de sensores, onde a sua topologia muda constantemente. Nestas redes, mais importante que descobrir os serviços, é saber que existe a disponibilidade para o obter.

Com o volume de tráfego a aumentar diariamente na *Internet*, a necessidade de providenciar a resposta por partes dos servidores aos pedidos torna-se cada vez mais difícil. Existem diversas opções de como melhorar a performance do sistema (desde aumento da capacidade de processamento ao aumento dos sistemas). Hoje em dia, a melhor opção é a distribuição de vários servidores por diferentes zonas (onde se pretende providenciar o serviço). Os diferentes pedidos vão sendo distribuídos pelos diferentes servidores fazendo com que os pedidos não sobrecarreguem somente uma rede. Diminui-se ainda o tempo de resposta, pois o processamento de pedidos é dividido pelos vários elementos do grupo.

A principal motivação do presente artigo é criar um protocolo de encaminhamento *anycast* inter-domínio, denominado de *Tree-based Anycast Protocol (TAP)*.

O artigo encontra-se divido em 5 secções. A próxima secção apresenta um estudo dos trabalhos relacionados, analisando cada uma das propostas, realçando os problemas que estas conseguem resolver e os que ainda deixam em aberto. Na secção 3 é apresentado o *TAP*, sendo especificadas as suas principais vertentes. A secção 4 descreve a implementação do *TAP* no *NS-2*, ilustrando o seu funcionamento com recurso à ferramenta *NAM*. É ainda realizada uma pequena comparação com o principal protocolo da literatura estudada. Por fim, a secção 5 apresenta as conclusões e o trabalho futuro.

II. Trabalho Relacionado

A proposta *GIA (Global IP Anycast)*[3], tem como característica distintiva o facto de introduzir um novo sistema de endereçamento, obrigando a uma mudança no *IPv6*. Através do aparecimento do endereço *anycast*, é possível aos encaminhadores reconhecer e tratar o tráfego de acordo com as especificações *anycast*. Utilizando o argumento de que um sistema autónomo (SA) só deverá possuir rotas para grupos *anycast* que lhe interessam, os autores do *GIA*, propõem uma divisão em três diferentes grupos - Grupo Interno, Grupo Externo Popular e Grupo Externo Impopular.

Quando um domínio possuí um grupo *anycast* no seu interior, é considerado um Grupo Interno. Como se trata de encaminhamento intra-domínio, o número de rotas acrescentadas às tabelas não é crítico, possuindo no máximo uma entrada para cada servidor. Se um domínio deteta um número significativo de utilizadores a aceder a um endereço *anycast* externo, catalogará esse serviço como Grupo Externo Popular. Quando existe um grande interesse por parte dos utilizadores do domínio num grupo *anycast* especifico, desencadeia-se uma série de procedimentos para tentar encontrar o melhor caminho para esse grupo. O *GIA* define um novo pacote *BGP*[9], que tem como objetivo obter o melhor caminho. O encaminhador fronteira (*border router*) do domínio, envia o pacote, sendo este reenviado através dos encaminhadores até a sua validade expirar ($TTL < 0$) ou um encaminhador responder a esse pedido. Um encaminhador responde a um pedido de procura se conhecer um caminho para o grupo melhor que o atual. Quando o encaminhador que originou a mensagem recebe a resposta, atualiza a sua tabela de encaminhamento *anycast* e cria um túnel entre si e o encaminhador que respondeu. Os pacotes *anycast* seguintes, passam a ser encaminhados através do melhor caminho descoberto. Todos os outros grupos,

que não interessam ao domínio, são considerados de Grupo Externo Impopular. Assim, o domínio adiciona uma rota padrão (*default route*), não sobrecarregando as suas tabelas de encaminhamento.

A proposta *PIAS (Proxy IP Anycast Service)*[4] , introduz uma nova abordagem ao encaminhamento *anycast*, a inclusão de redes sobrepostas. A ideia é de distribuir um grande número de *proxys* pela *Internet*, com o intuito de resolver o problema de escalabilidade do encaminhamento *anycast* ao nível da camada de rede. Os *proxys* funcionam como gestores da rede *anycast*, anunciando rotas para grupos *anycast* através do *BGP*[9]. A responsabilidade da entrega dos pacotes não recai sobre os encaminhadores, sendo estes entregues pelo *proxy* mais próximo, através de encaminhamento *unicast* (Figura 2). As soluções baseadas em redes sobrepostas nem sempre encaminham pelo melhor caminho, mas consegue melhorar a escalabilidade. Com este sistema de gestão de tráfego *anycast*, excluem-se as rotas de todo o grupo das tabelas de encaminhamento dos encaminhadores normais, introduzindo a melhor. Esta proposta alivia a carga colocada nos encaminhadores, mas acarreta a implementação de um grande número de *proxies*.

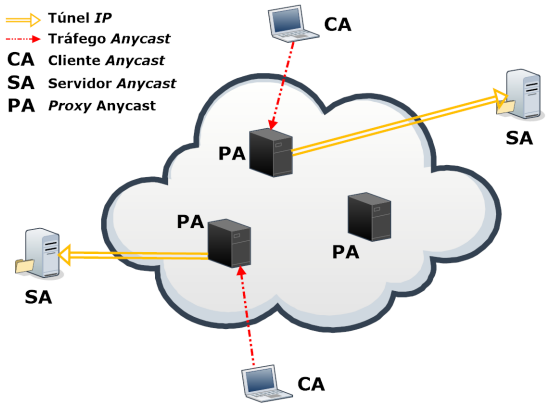

Figura 2: Arquitetura do *PIAS*.

O encaminhamento *multicast* e *anycast* possuem muitas propriedades semelhantes, o que levou a que vários investigadores estudassem a adaptação dos principais algoritmos *multicast* para a realidade *anycast*. Um do trabalhos mais notórios nesta vertente[6], escolheu três protocolos (*DVMRP*[10], *MOSPF*[11] e *PIM-SM*[12]) e procedeu ao estudo de uma possível implementação. Os dois primeiros protocolos consomem muitos recursos, sendo propostas para serem aplicadas a pequenas redes. O último protocolo (*PIM-SM*) é um dos protocolos de encaminhamento mais utilizados ao nível global, com muitas vantagens sobre os principais concorrentes. O facto de consumir poucos recursos e ser desenhado para redes alargadas, constitui-se como o maior candidato à adaptação.

Num segundo artigo[7], os investigadores focaram-se na adaptação do *PIM-SM*, para criar um protocolo de encaminhamento *anycast* denominado *PIA-SM*. O *PIA-SM*, mantém a lógica do seu protocolo base e constrói árvores unidirecionais, centradas num ponto de encontro (*RP*). O *RP* é a raiz da árvore que une os membros do grupo. Ao contrário do que acontece no *multicast*, o *RP* e todos os nós da árvore partilhada somente encaminham para uma das interfaces, a que possui melhor métrica (Figura 3). Quando o servidor recebe o pacote, estabelece a ligação com o cliente, por *unicast*.

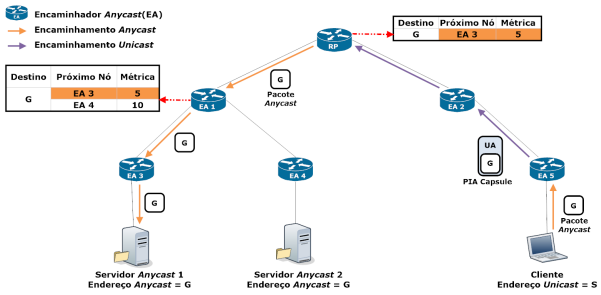

Figura 3: Encaminhamento dos pacotes *anycast* pelo *PIA-SM*

A implementação do *PIA-SM*, não contempla o caso de existir mais de que um cliente. Caso existam vários clientes simultaneamente, o servidor escolhido é sempre o mesmo, mesmo que esteja sobrecarregado. Uma outra variante do *PIA-SM*[8] aborda este problema, introduzindo um outro campo à tabela de encaminhamento para além da métrica, campo esse que tem em consideração o facto de o servidor estar ocupado.

O *PIA-SM* possui ainda uma última consideração a ser feita, relativamente à forma como é calculada a sua métrica. Como a métrica é calculada quando os servidores se juntam à árvore partilhada, a métrica obtida é relativa ao *RP*, o que pode originar que o servidor escolhido não é realmente o mais próximo do cliente, mas o mais próximo do *RP*, adulterando o princípio do encaminhamento *anycast*[2].

III. Especificação do *TAP*

O *PIM-SM* está otimizado para redes com recetores dispersamente distribuídos ao longo da rede. É designado por independente do protocolo, pois ao contrário de outros protocolos de encaminhamento *multicast* não depende de nenhum protocolo de encaminhamento *unicast* específico. Constrói árvores unidirecionais, centradas num ponto de encontro, normalmente designado por *Rendezvous Point (RP)*. O papel do *RP* é fundamental no protocolo, servindo de ponto de encontro entre as fontes e os recetores. O facto de utilizar um ponto de encontro para fazer esta ligação, permite aos encaminhadores reduzirem o número de entradas, pois simplesmente precisam de possuir a entrada (*,G). Devido a esta característica, a adaptação deste protocolo *multicast* à realidade *anycast*, parece ser muito vantajosa, tendo já sido realizadas algumas adaptações[7][8].

Um último aspeto a diferenciar entre a metodologia *multicast* e *anycast*, é a forma como decorre a comunicação. Enquanto no tráfego *multicast*, o grupo de clientes estabelece ligação a uma fonte (ou a um conjunto de fontes), no caso de tráfego *anycast* a ligação é entre o cliente e o servidor mais

próximo[1]. Comparando diretamente as duas metodologias, um recetor (ou recetores) no *multicast* equipara-se a um grupo de servidores no *anycast*, e a fonte no *multicast* equipara-se a um cliente.

A. *Atribuição do Endereço* Anycast *a um Grupo*

A primeira grande diferença entre o paradigma de comunicação *multicast* e o paradigma de comunicação *anycast* reside no facto de a primeira possuir um endereçamento próprio e distinguível dos endereços *unicast*. Os endereços *anycast* são absolutamente indistinguíveis dos endereços *unicast*[2], o que resulta num desafio extra para o seu encaminhamento na rede. O desafio acontece porque um encaminhador padrão[2] não consegue distinguir o tráfego e deste modo encaminhá-lo de forma distinta.

A abordagem seguida pelos autores de outra proposta que se focou igualmente no *PIM-SM*[7], considera que o endereço *anycast* atribuído a um grupo é o endereço *unicast* do nó inicial[3]. Quando um cliente efetuar um pedido, mesmo que nunca passe por um encaminhador que implemente o protocolo proposto (adiante designado por `Encaminhador Anycast (EA)`), viajará sempre até ao nó inicial.

Um encaminhador *anycast* (`EA`) é um encaminhador especial, que com um protocolo específico *anycast*, consegue identificar e encaminhar o tráfego, como acontece no *multicast*.

A criação de um esquema de endereçamento próprio é uma alternativa à abordagem anterior. O protocolo *GIA*[3] defende que, para ser possível um correto funcionamento do paradigma de comunicação *anycast*, é necessário conseguir distinguir o endereço *anycast* do *unicast*. A introdução deste novo tipo de endereço, dificulta a aceitação de um novo protocolo, pois o endereçamento utilizado é diferente do previsto na norma (indistinguível dos endereços *unicast*). Apesar desta desvantagem, as comunicações tornam-se mais seguras, pois o endereço *unicast* dos sistemas terminais nunca é conhecido. Os clientes comunicam sempre para o endereço do grupo, e os servidores respondem com o endereço de grupo no campo de origem. Outra vantagem é o facto da redução da carga sobre um `EA`, pois reduz o número de pesquisas (*lookup*) nas tabelas de encaminhamento. Um `EA` na primeira proposta (endereço indistinguível do *unicast*), é obrigado a verificar a tabela de encaminhamento *anycast* e, caso não encontre correspondência, também tem de verificar a tabela *unicast* para reencaminhar um pedido.

Na implementação do sistema, foi escolhida a introdução de um novo tipo de endereçamento *anycast*.

B. *Escolha do Melhor Servidor*

O paradigma de comunicação *anycast* necessita da introdução de um fator de seleção, que permita ao `EA` encaminhar o tráfego para o servidor mais capaz. Normalmente esse fator utilizado é o número de saltos, entre o cliente e o servidor.

Como a fixação de uma métrica pode ser algo limitador, é possível definir o cálculo da métrica de acordo com o desejo do criador do grupo *anycast*. Esta métrica pode passar por um sem número de parâmetros como o número de saltos, largura de banda, perda de pacotes, latência, carga atual do servidor, fiabilidade do trajeto, entre outros. Alguns destes parâmetros obrigam ao cálculo da métrica pelo percurso fim-a-fim, onde todos os nós a atualizam. A equação 1 é um exemplo de uma métrica combinada, sendo calculada fim-a-fim, onde todos os nós (ao longo da árvore) atualizariam a métrica.

$$\textbf{Métrica} = \textbf{f}(N^{o}deSaltos) \oplus \textbf{f}(\text{Carga do Servidor}) \oplus \oplus \textbf{f}(\text{Largura de Banda Disponível}) \quad (1)$$

Ao longo do capítulo, é adotado como métrica, um parâmetro fixo calculado previamente pelo servidor, de modo a simplificar a sua explicação e implementação. O parâmetro escolhido foi a carga total do servidor.

C. *Descoberta de Encaminhadores Vizinhos*

O funcionamento do protocolo baseia-se no facto que todos os `EA` têm conhecimento dos caminhos possíveis na sua topologia. No *PIM-SM* são trocadas mensagens *PIM Hello*, que funcionam como mensagens de reconhecimento de vizinhos. A Figura 4 ilustra como se procede a localização dos vizinhos. Todos os encaminhadores `EA` trocam mensagens de reconhecimento entre si (a laranja na Figura 4), sendo estas enviadas para o endereço *multicast* do segmento de rede. Com todas as ligações identificadas, designa-se um encaminhador responsável (*DR*) pelo envio de mensagens periódicas de aviso para o *RP*. Tal como no *PIM-SM*, o *DR* é eleito baseado no endereço lógico de cada interface, sendo escolhido o de maior valor. Cada segmento da rede tem de possuir um *DR*, como vemos na Figura 4.

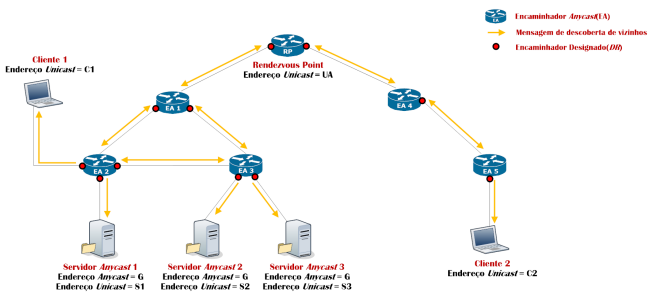

Figura 4: Descoberta de vizinhos.

Às mensagens de reconhecimento de vizinhos pode ainda ser adicionado uma opção que permita autenticar os *EA*, utilizando segurança ao nível da rede (`IPSec`).

[1] Servidor com melhor métrica.

[2] Encaminhador que não possui nenhum protocolo de encaminhamento *anycast*.

[3] Primeiro servidor a ativar-se.

D. Criação da Árvore Partilhada

A escolha do *RP* é uma das decisões mais importantes deste tipo de protocolo, existindo diversos estudos para otimizar o seu posicionamento[13][14]. Este é utilizado como ponto de registo dos servidores, para proporcionar o encaminhamento de pacotes. A construção da árvore partilhada pode ser dividida em duas parcelas, o registo dos servidores junto do `EA` mais próximo da sua origem e deste com o *RP*.

O registo de um servidor *anycast* é semelhante ao que acontece com os recetores no *multicast*. É gerada uma mensagem de ligação ao grupo desejado, enviando esta mensagem para o `EA` mais próximo. Como um administrador pode pretender implementar mais do que um servidor (para o mesmo grupo *anycast*) no mesmo segmento de rede, o `EA` deve comunicar sempre com o endereço de origem na mensagem de registo. O endereço de origem identifica o endereço local do servidor, sendo a componente inicial da mensagem de registo, como podemos verificar na Figura 5.

Figura 5: Pacote da mensagem de registo.

O registo de um servidor *anycast* junto do *RP* é efetuado pelo `EA` que é o *DR* no segmento de rede, aquando da receção da mensagem de ligação na rede local (a laranja na Figura 6). O `EA` reage ao pedido de admissão do novo servidor, criando na tabela de encaminhamento *anycast*, uma entrada (*,G). Através do envio sucessivo de mensagens de ligação (equivalente ao *join* no *PIM-SM*, a azul na Figura 6) em direção ao *RP*, constrói-se um ramo da árvore partilhada. O reenvio da mensagem de ligação serve-se sempre do nó do caminho mais curto até ao *RP*, ao longo de cada segmento.

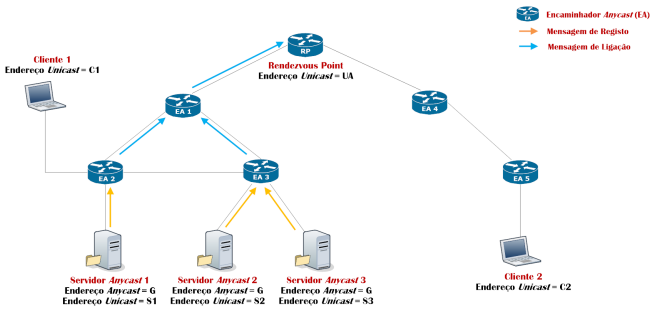

Figura 6: Criação da Árvore Partilhada.

O papel desempenhado pelo encaminhador designado, não se esgota apenas na receção de pedidos e estabelecimento das respetivas ligações, devendo acrescentar o endereço do *RP* para efetuar periodicamente notificações, de modo a conservar o(s) servidor(es) ativo(s). Um `EA` apaga a entrada (*,G) e retira o endereço do *RP* da lista de notificações, quando esse grupo não lhe interessar. O interesse deste pode ser por ter um membro diretamente ligado ou então servir de elo de ligação para outros membros.

E. Criação da Árvore Centrada no Cliente

Finalizado o processo da ativação dos servidores *anycast* junto do *RP*, é agora possível aos clientes efetuarem pedidos. A Figura 7 ilustra o processo de descobrimento dos servidores por parte dos clientes. Quando `Cliente 1 (C1)` deseja iniciar a comunicação com os servidores, "encapsula" o pacote *anycast* de descoberta numa mensagem *unicast*, enviando até ao *RP* do grupo (a azul na Figura 7). O facto de ser utilizado tráfego *unicast* permite que outros encaminhadores padrão sejam capazes de encaminhar o tráfego, mesmo não possuindo perceção do tráfego *anycast*. Ao receber o pacote, o *RP* "desencapsula" a mensagem e envia o pacote *anycast* com auxílio da árvore partilhada do grupo (a laranja na Figura 7)

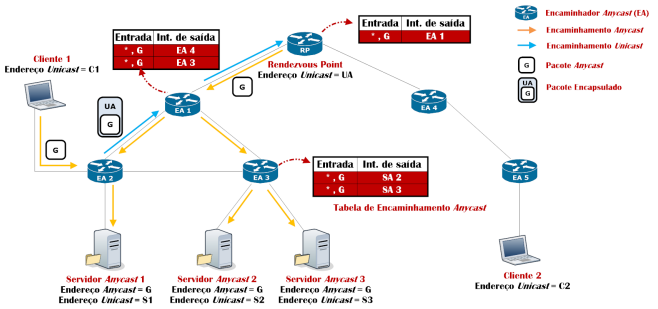

Figura 7: Pedido de localização por parte do Cliente 1.

Quando um cliente começa o processo de localização dos servidores, um temporizador é iniciado, aguardando que os servidores construam uma nova árvore centrada no cliente, de modo a poder realizar pedidos. No caso do tempo se esgotar sem nenhum servidor se conectar a si, é enviado um novo pedido e reiniciado o temporizador.

Logo que o pacote de localização alcança um servidor, este calcula a sua carga total. Como acontece na mensagem de registo na árvore partilhada, é enviado um pacote para o `EA` mais próximo. Sendo este processo replicado por todos os servidores do grupo, garante-se que um cliente poderá realmente escolher o servidor com menor carga. A Figura 8 ilustra a estrutura da mensagem para o registo na árvore centrada no cliente, onde é possível verificar uma alteração no pacote, a inclusão da métrica do servidor.

Figura 8: Pacote da mensagem de registo na árvore centrada no cliente.

Calculada a carga total para cada servidor no momento atual, é altura de comutar da árvore partilhada para a árvore

centrada na fonte. Como é possível verificar na Figura 9, todos os servidores efetuam a ligação ao C1, anunciando a sua métrica. Inicialmente todos os servidores apresentam a mesma carga, sendo escolhido o servidor que enviou a mensagem primeiro. Na Figura, o C1 receberia primeiro a mensagem do SA 1, pois este está mais próximo da sua localização.

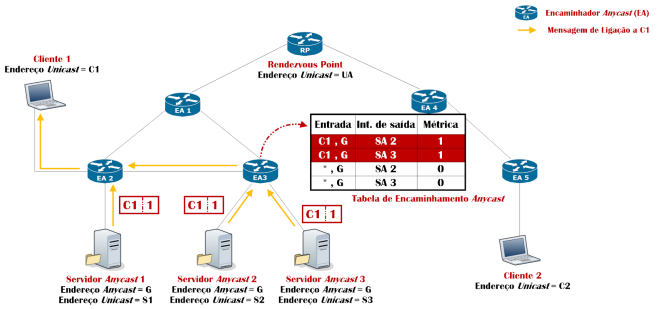

Figura 9: Criação da árvore centrada no cliente.

Quando um servidor decide centrar-se no cliente, o encaminhador designado (*DR*) do seu segmento de rede, começa por adicionar uma entrada (C,G) à sua tabela de encaminhamento *anycast*. A constituição da nova entrada possui uma grande diferença em relação ao comportamento do *PIM-SM*. Enquanto o *PIM-SM* inicialmente coloca a 0 a *flag* SPT-BIT, aqui é colocada a 1, pois a comunicação deverá ser automaticamente comutada para a árvore centrada no cliente.É enviado um pedido de exclusão para o grupo em direção ao *RP*, de modo a alerta-lo (e aos outros *EA*), que a partir daquele momento a comunicação decorrerá a através da árvore de centrada no cliente.

A exemplo do que acontece na construção de uma árvore partilhada, a árvore centrada no cliente deve adicionar o endereço do cliente à lista de notificações periódicas. Os *EA* entre o cliente e os servidores, devem atualizar/criar as suas entradas (C,G) em conformidade com o seu funcionamento.

F. Abandono do Grupo por parte dos Servidores

O abandono por parte de um servidor do grupo pode acontecer por dois motivos, por pedido do servidor ou por não efetuar a renovação da ligação.

Se o servidor decide abandonar o grupo, pode abandonar por completo (árvore partilhada e todas as centradas no cliente de que faça parte) ou então simplesmente uma árvore centrada no cliente. O primeiro caso está normalmente relacionado com o desejo de terminar todas as comunicações no sistema terminal, começando o EA por percorrer todas a entradas e remover a ligação de saída para o servidor. Caso o EA verifique que a lista de interfaces de saída ficou vazia, deve notificar o próximo nó do abandono, nó do caminho mais curto em relação ao destino final (*RP* para árvores partilhadas e cliente para árvores centradas). O segundo caso pode acontecer quando um servidor estiver com demasiadas comunicações e decide abandonar alguns clientes para prestar um melhor serviço. Esta consideração caberá sempre ao gestor dos servidores. A aplicação, utilizada pelo cliente, deverá enviar para o *RP*, periodicamente, pedidos de localização. O envio deste pedido tem como objetivo não deixar expirar o ramo de ligação ao servidor e permitir a novos servidores efetuarem as suas ligações.

A Figura 10 demonstra o pedido de abandono do SA 1 do grupo. Normalmente, quando um servidor abandona a rede, é necessário reiniciar o processo de descoberta de servidores. Supondo que se trata de uma comunicação importante, o tempo de resposta por parte dos servidores é minimizado.

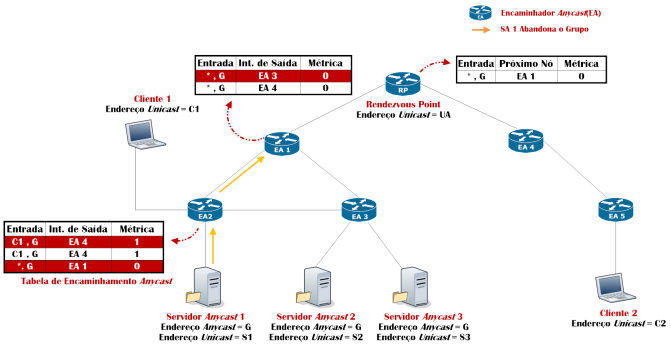

Figura 10: Abandono por parte do SA 1

A arquitetura proposta, proporciona uma funcionalidade que permite comutar automaticamente para outro servidor *anycast* sem voltar a refazer o serviço de localização. Tendo o servidor SA 1 abandonado o grupo, passa a ser o servidor com a segunda melhor métrica até então (SA 2), a responder aos pedidos do C1, como podemos verificar pela Figura 11.

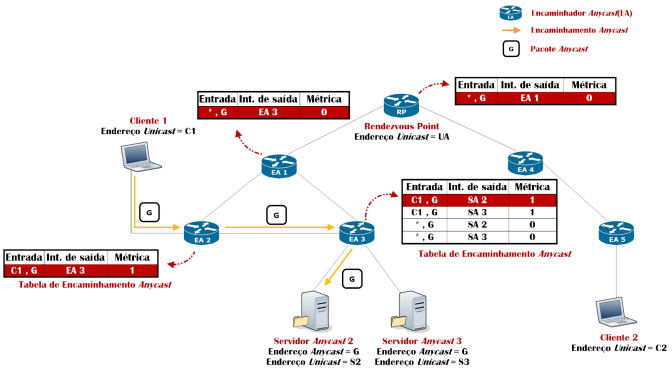

Figura 11: Resposta a pedido do C2 após o abandono do SA 1

G. Atualização da Métrica de um Cliente por parte dos Servidores

O balanceamento de carga é uma técnica consensual para quem quer implementar um serviço e manter uniforme a carga de trabalho dos seus sistemas terminais. Como o valor do atraso na resposta dos sistemas terminais é proporcional à carga que eles suportam no momento, é necessário repartir a carga or vários sistemas terminais, para evitar possíveis congestionamentos. A solução proposta engloba a possibilidade

de balanceamento de carga, através da atualização da métrica do servidor.

A Figura 12 demonstra o cenário de quando a métrica do SA 2 se degrada. Esta degradação pode passar por vários fatores, como por exemplo estar a receber demasiados pedidos. O SA 2 recalcula a sua métrica e comunica-o ao EA 3. Por sua vez, este verifica a sua tabela de encaminhamento e atualiza-a. Sendo que a menor métrica não sofreu nenhuma alteração (SA 3 tinha a mesma métrica), o EA 3 não necessita de informar o *EA 2*, o próximo encaminhador em direção ao cliente. Se houvesse uma alteração na menor métrica, o *EA 3* deveria reenviar a mensagem de atualização pelo caminho mais curto em direção do C1, sendo o processo repetido em todos os encaminhadores.

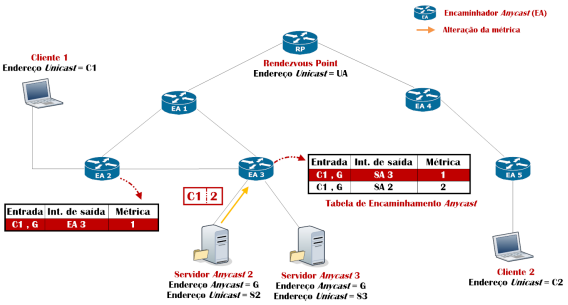

Figura 12: Alteração da métrica do SA 3

Perante a atualização da tabela de encaminhamento de EA 3, este agora passa a direcionar os pacotes provenientes do C1 para o SA 3. A Figura 11, demonstra o processo de encaminhamento após atualização da métrica, por parte do SA 2.

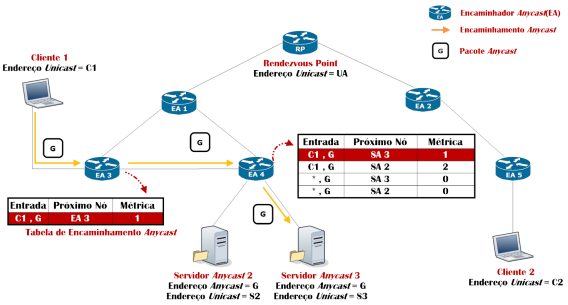

Figura 13: Comutação dos pedidos de C2 para SA 1

IV. Implementação e Teste do Sistema

A. Validação do Sistema

O sistema especificado na secção anterior, foi implementado com auxílio do simulador de rede *NS-2.35*[15]. O código implementado para este simulador é escrito em duas linguagens orientadas a objetos, o C++ para as tarefas amiudamente executadas (mais rápido) e o OTcl para o controlo de ações (mais flexível).

O código base utilizado na implementação do sistema é baseado numa implementação do *PIM-SM* existente[16], o que permitiu construir uma solução mais completa. Apesar de esta solução possuir um classificador e um agente para o protocolo *PIM-SM*, foi necessário adaptá-la ao sistema especificado. As classes classifier-pim.cc e classifier-pim.h são responsáveis pelo reenvio das mensagens entre nós, sendo escritas em C++. A implementação do agente do *PIM.tcl* ocorre na classe PIM.tcl, escrita em OTcl, sendo esta responsável pela troca de mensagens entre os nós, pela implementação do algoritmo e a construção das tabelas de encaminhamento.

O classificador implementado na abordagem base[16], reenvia os pacotes de acordo com os endereços de origem e o destino (grupo). Ao receber um pacote, o classificador verifica a sua tabela e encaminha os pacotes para todas as interfaces(nós) nele registados. Como no tráfego *anycast* é necessário escolher de acordo com os endereços de origem e destino mais a métrica, foi necessário acrescentar essa possibilidade. O método de encaminhamento *multicast* (para todos) foi conservado, sendo utilizado para as mensagens de descoberta de servidores. Foi necessário criar uma nova estrutura, com um par de campos interligados, métrica e respetivo nó. Assim, quando cada servidor se centrar no cliente, cria uma nova entrada, com a sua respetiva métrica. O encaminhamento passa a ser efetuado de acordo com o nó com melhor métrica, passando o cliente a comunicar com um único servidor. A Figura 14 mostra os campos constituintes do novo classificador *anycast*.

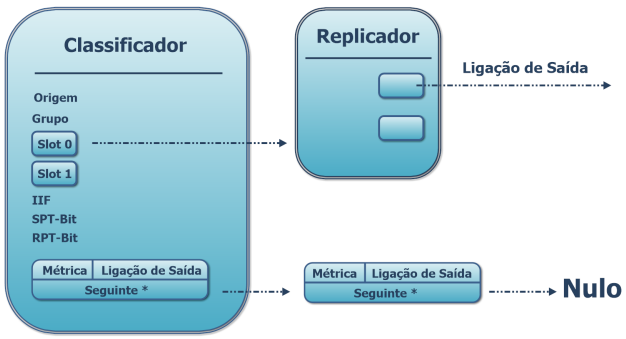

Figura 14: Composição do classificador *anycast*.

Os métodos tradicionais do *PIM-SM* para a manutenção das árvores (partilhada e centrada) tiveram que sofrer alterações, para que passassem a contemplar a métrica nas suas mensagens. A classe PIM.tcl sofreu várias alterações nas suas principais funções (*join-group*, *leave-group*, *recv-join*, entre outras), para passar a tratar o tráfego segundo o encaminhamento *anycast*. Foi ainda necessário incluir um novo tipo de mensagem de atualização, não disponível no protocolo base. Quando um nó recebe uma mensagem de atualização, atualiza a métrica do nó que originou a mensagem e verifica se o valor mínimo se alterou. Tendo este valor sido alterado, é enviado então uma mensagem para o nó seguinte, em direção ao destino, *RP* e/ou cliente(s). O processo descrito é repetido sucessivamente até que a métrica mínima não seja alterada ou chegue ao destino. De modo a proceder

a criação da nova mensagem de atualização, foi necessário modificar a classe `packet.h` e `mcast_ctrl.cc`. Para finalizar, foi necessário implementar duas aplicações, para os clientes e servidores, de modo a simular o comportamento especificado no capítulo anterior. A nova aplicação no cliente, efetua pedidos de descoberta de localização até receber a primeira mensagem de resposta de um servidor, passando de seguida a enviar pedidos para o servidor com a melhor métrica. A aplicação que corre nos servidores responde aos pedidos vindos dos clientes automaticamente, simulando o comportamento real de servidor, divulgando somente o seu endereço *anycast*.

O cenário utilizado para explicar o sistema na secção anterior, foi implementado para testar a solução, com o auxílio de uma aplicação de suporte do *NS*, o *NAM*. O *NAM* é uma ferramenta de animação, que permite criar topologias e visualizar a animação dos pacotes durante o seu encaminhamento. O primeiro passo foi criar a topologia e de seguida os nós, que seriam clientes, servidores ou simples encaminhadores. Finalizada a implementação, é altura de dar indicações aos nós para efetuarem a sua ligação, abandono ou atualização no instante de tempo pretendido. A Figura 15 demonstra o encaminhamento dos pacotes do `C1` para o `SA 1`, o servidor com melhor métrica.

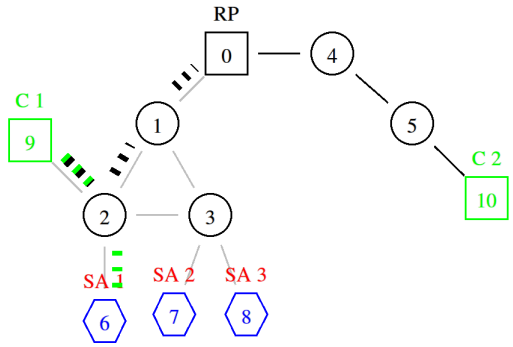

Figura 15: Resposta do pedido do `C1` pelo `SA 1`.

A falha (ou abandono) durante a comunicação de um servidor é sempre um problema para a rede, sendo muitas vezes necessário voltar a iniciar a comunicação. A implementação efetuada permite a fácil comutação de um servidor para outro, de modo a minimizar o tempo perdido. A Figura 16 capta o momento em que o `SA 1` deixa de poder atender o `C1`, podendo-se verificar a imediata comutação para o `SA 2`.

As ligações ao longo da *Internet*, estão constantemente a alterar a sua qualidade. O protocolo implementado prevê essa situação, podendo, a qualquer momento, um servidor proceder à atualização da sua métrica em relação a qualquer nó. Quando o `SA 2`, no cenário anterior, vê que a sua ligação deteriorou-se, procede a atualização da sua métrica. Na Figura 17 é possível verificar o encaminhamento após a atualização por parte do `SA 2`, passando o tráfego a ser encaminhado o `SA 3`.

O cenário elaborado permite testar o correto comportamento do *TAP* implementado e exequibilidade da proposta.

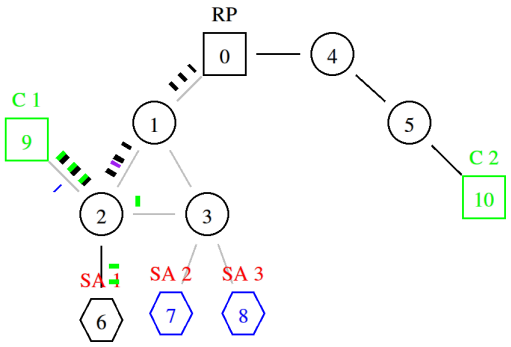

Figura 16: Resposta do pedido do `C1` comuta do `SA 1` para o `SA 2`.

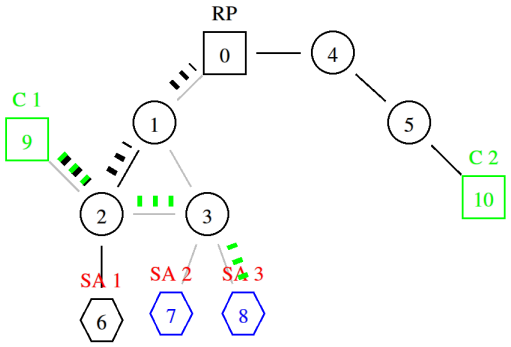

Figura 17: Resposta do pedido do `C1` comuta do `SA 2` para o `SA 3`.

B. Análise dos resultados em relação ao PIA

O *PIA*[7] foi o protocolo escolhido para comparar com o *TAP*, principalmente por este também utilizar uma abordagem baseada no *PIM-SM*. Embora o protocolo tenha sido implementado com sucesso, não houve tempo para desenvolver uma aplicação que tivesse um comportamento como especificado[7]. Esta limitação não permitiu efetuar grandes testes entre os dois protocolos, ficando a faltar a realização de um melhor conjunto de testes no futuro.

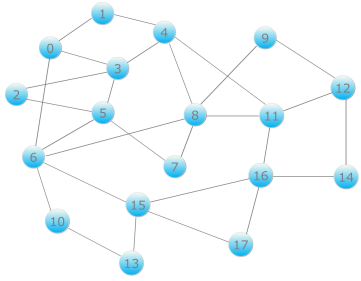

Figura 18: Topologia implementada com 18 nós.

No entanto, foi possível realizar uma comparação entre os dois protocolos, medindo a distância do servidor escolhido até ao cliente. A topologia presente na Figura 18 foi utilizada como cenário para os testes, sendo posicionados aleatoria-

mente nesses nós um *RP*, um cliente e quatro servidores. O cenário foi simulado 100 vezes para os dois protocolos, *TAP* e *PIA*, sendo o gráfico presente na Figura 19 o seu resultado.

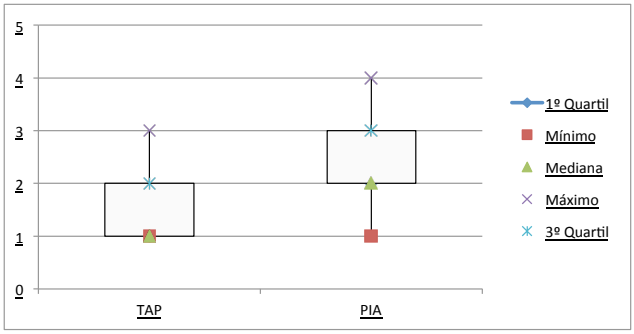

Figura 19: Distância entre o cliente e o servidor escolhido para os dois protocolos (*TAP* e o *PIA*).

Ao observar o gráfico na Figura 19, é possível afirmar que o protocolo *PIA* nem sempre escolhe o servidor mais próximo da localização. Uma importante conclusão a retirar das simulações é que o valor das localizações centrais das amostras é menor no *TAP*, o que atesta a eficiência do novo protocolo, em relação ao *PIA*. Apesar de o *TAP* ser melhor quando as métricas levam em conta a posição dos servidores, é necessário, para trabalho futuro, realizar um conjunto de testes exaustivos ao *TAP*, obtendo novos resultados em relação ao *PIA*.

V. Conclusão e trabalho futuro

O encaminhamento de tráfego *anycast* em redes *IPv6* ainda não é uma realidade, principalmente devido ao facto de não existir um protocolo normalizado que retire partido das suas potencialidades. Este artigo aborda uma proposta para um novo protocolo, tendo o *PIM-SM* como base, que pretende solucionar esse problema.

As principal vantagem deste novo protocolo é o facto de o cliente comunicar realmente com o servidor mais próximo (com melhor métrica) do seu local. A segurança é outra das suas vantagens, pois os clientes nunca conhecem o endereço do servidor, comunicando sempre para o grupo. A última grande vantagem é a grande disponibilidade da rede. No caso da falha de um servidor, os pacotes são automaticamente comutados para outro servidor. O grande entrave à aceitação desta proposta prende-se com o facto de ser necessário um endereçamento distinguível do *unicast*, obrigando a uma alteração da norma[2].

O trabalho ainda se encontra a decorrer, estando o protótipo a ser avaliado. O *TAP* está a ser testado em diferentes tipos de topologias de rede, para obter dados mais consistentes em relação ao seu desempenho. O trabalho futuro passa, claramente, por realizar mais testes ao *TAP* e comparar com a abordagem referida no trabalhado relacionado[7].

Agradecimentos

Este trabalho é financiado por Fundos FEDER através do Programa Operacional Fatores de Competitividade – COMPETE e por Fundos Nacionais através da FCT – Fundação para a Ciência e Tecnologia no âmbito do Projeto: FCOMP-01-0124-FEDER-022674

Referências

[1] C. Partridge, T. Mendez, and W. Milliken, "Host Anycasting Service." RFC 1546 (Informational), Nov. 1993.

[2] R. Hinden and S. Deering, "IP Version 6 Addressing Architecture." RFC 1884 (Historic), Dec. 1995. Obsoleted by RFC 2373.

[3] D. Katabi and J. Wroclawski, "A framework for scalable global ip-anycast (gia)," *SIGCOMM Comput. Commun. Rev.*, vol. 30, pp. 3–15, Aug. 2000.

[4] H. Ballani and P. Francis, "Towards a global ip anycast service," *SIGCOMM Comput. Commun. Rev.*, vol. 35, pp. 301–312, Aug. 2005.

[5] T. Stevens, M. De Leenheer, C. Develder, F. De Turck, B. Dhoedt, and P. Demeester, "Astas: Architecture for scalable and transparent anycast services," *J. Commun. Netw.*, vol. 9, no. 4, pp. 1229–2370, 2007.

[6] S. Doi, S. Ata, H. Kitamura, and M. Murata, "Ipv6 anycast for simple and effective service-oriented communications," *IEEE Communications Magazine*, vol. 42, pp. 163–171, 2004.

[7] S. Matsunaga, S. Ata, H. Kitamura, and M. Murata, "Design and Implementation of IPv6 Anycast Routing Protocol: PIA-SM," in *Advanced Information Networking and Applications*, vol. 2, pp. 839–844, 2005.

[8] A. Sulaiman, B. Ali, S. Khatun, and G. Kurup, "An enhanced ipv6 anycast routing protocol using protocol independent multicast-sparse mode (pim-sm)," in *Telecommunications and Malaysia International Conference on Communications, 2007. ICT-MICC 2007. IEEE International Conference on*, pp. 588 –593, may 2007.

[9] k. claffy, "Border Gateway Protocol (BGP) and Traceroute Data Workshop Report," tech. rep., Cooperative Association for Internet Data Analysis (CAIDA), Oct 2011.

[10] D. Waitzman, C. Partridge, and S. Deering, "Distance Vector Multicast Routing Protocol." RFC 1075 (Experimental), Nov. 1988.

[11] J. Moy, "MOSPF: Analysis and Experience." RFC 1585 (Informational), Mar. 1994.

[12] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)." RFC 4601 (Proposed Standard), Aug. 2006. Updated by RFCs 5059, 5796, 6226.

[13] F. Font and D. Mlynek, "Choosing the set of rendezvous points in shared trees minimizing traffic concentration," in *Communications, 2003. ICC '03. IEEE International Conference on*, vol. 3, pp. 1526 – 1530 vol.3, may 2003.

[14] D. Katabi, "The use of ip-anycast for building efficient multicast trees," in *Global Telecommunications Conference, 1999. GLOBECOM '99*, vol. 3, pp. 1679 –1688 vol.3, 1999.

[15] S. Mccanne, S. Floyd, and K. Fall, "ns2 (network simulator 2)." http://www-nrg.ee.lbl.gov/ns/.

[16] A. S. e. V. F. António Costa, Maria João Nicolau, "Implementação e Teste do PIM-SM no Network Simulator," tech. rep., Conferência sobre Redes de Computadores (CRC2002), Faro, Portugal, Sep 26-27 2002.