UNIVERSIDADE NOVA DE LISBOA
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Electrotécnica

REALIZAÇÃO DE UMA ARQUITECTURA *DIFFSERV* COM CONFIGURAÇÃO TRANSPARENTE DE QOS

Por:
Sérgio Alexandre Braz Vieira

Dissertação apresentada na Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa para a obtenção do grau de Mestre em Engenharia Electrotécnica e de Computadores

Orientador: Doutor Luís Filipe Lourenço Bernardo
Co-Orientador: Doutor Paulo da Costa Luís da Fonseca Pinto

Lisboa
2009

Aos meus pais.

RESUMO

O foco desta dissertação é o estudo de métodos para a construção e configuração de redes com garantia de qualidade de serviço (QoS – *"Quality of Service"*) baseados na abordagem de serviços diferenciados (*DiffServ*). Pretende-se que a configuração da rede seja feita de forma transparente para o utilizador, aproveitando os pacotes de sinalização SIP (*"Session Initiation Protocol"*) utilizados em várias aplicações multimédia. A dissertação inclui também um enquadramento teórico, no qual se analisam o uso da Internet para serviços de voz (VoIP –*"Voice over Internet Protocol"*) e a estrutura do núcleo do sistema universal de telecomunicações móveis (UMTS – *"Universal Mobile Telecommunications System")* para serviços multimédia (IMS –*"Internet Protocol Multimedia Subsystem"*).

Ainda no âmbito da dissertação, foi elaborado um protótipo utilizando máquinas Linux que consiste numa rede *DiffServ* com dois tipos de nós: os encaminhadores de fronteira (ER – *"Edge Routers"*), que efectuam a negociação com os clientes (utilizando SIP); e os encaminhadores nucleares (CR – *"Core Routers"*), que distribuem os pacotes de acordo com a classe de serviço. A configuração dos CRs é feita dinamicamente em função da carga na rede, utilizando o protocolo COPS (*"Common Open Policy Service"*). A arquitectura realizada é compatível com os serviços VoIP e com o núcleo do UMTS para o serviço IMS.

PALAVRAS-CHAVE

Qualidade de Serviço, *DiffServ*, Configuração Transparente, Largura de Banda, COPS, SIP.

ABSTRACT

The focus of this dissertation is the study of methods for the construction and configuration of networks with guaranteed QoS, based upon the *DiffServ* approach. The configuration of the network is intended to be fully transparent to the user, taking advantage of the SIP signaling packets used in several multimedia applications. The dissertation also includes an overview of VoIP services and of the structure of the UMTS core for the IMS service.

Still within the breadth of this dissertation is the elaboration of a prototype, using Linux based machines, that consists on a *DiffServ* network with two types of nodes: the edge routers (ER), that negotiate with clients (using SIP); and the core routers (CR), that deal with the data packets according to their service class. The configuration of the CR is done dynamically according to the network load, using the COPS protocol. The architecture is compatible with VoIP services and with the UMTS core for the IMS service

KEYWORDS

Quality of Service, *DiffServ*, Transparent Configuration, Bandwidth, COPS, SIP.

AGRADECIMENTOS

Gostaria de agradecer a todos aqueles que de alguma forma contribuíram para a realização desta dissertação.

Agradeço ao Prof. Paulo Pinto e um agradecimento especial ao Prof. Luís Bernardo pela orientação, incentivo e apoio ao longo desta jornada e sobretudo a paciência e compreensão que demonstrou para comigo, durante este tempo. Nos momentos mais difíceis, foram estes factores que me ajudaram a manter o rumo e levar até ao fim esta demanda. Agradeço-lhe ainda o esforço e o tempo disponibilizado para rever o texto desta dissertação, mesmo em circunstâncias difíceis, e dos quais resultaram melhoramentos e correcções que elevaram substancialmente a qualidade da dissertação.

Aos meus colegas da FCT-UNL, particularmente aos colegas da secção de telecomunicações, gostaria de agradecer a amizade e a disponibilidade para me ajudarem, sempre que necessário. Gostaria de destacar Bernardo Oliveira, David Paulino, Diogo Rodrigues, Miguel Pereira, Miguel Silva, Pedro Faleiro e Sérgio Gaspar pela amizade e pelo convívio durante este tempo.

Aos meus amigos, gostaria de agradecer as palavras de ânimo e incentivo, que foram de grande importância nas horas mais difíceis.

Por último, mas de forma alguma menos importante, gostaria de agradecer aos meus pais todo o apoio e compreensão ao longo de todo o curso. Foi uma jornada atribulada e sem a sua ajuda, à custa de grande sacrifício da sua parte, dificilmente a teria terminado. Por isso, e para além de todo o carinho que me demonstraram ao longo da minha vida, estarei eternamente grato.

Lisboa, Fevereiro 2009

Sérgio Alexandre Braz Vieira

ACRÓNIMOS

3GPP	3rd Generation Partnership Project.
AF	Assured Forwarding.
ANA	Advanced Networking Applications.
API	Application Programming Interface.
ARM	Active Resource Management.
BAR	Bandwidth Allocation Request.
BB	Bandwidth Broker.
BBAS	Bandwidth Broker Administrative Server.
BBS	Bandwidth Broker Server.
BBTP	Bandwidth Broker Transfer Protocol.
BCIT	British Columbia Institute of Technology.
BE	Best Effort.
CANARIE	Canadian Advanced Network and Research for Industry and Education.
CATI	Charging and Accounting Technology for the Internet.
CITI	Center for Information Technology Integration.
COPS	Common Open Policy Service.
COPS-PR	COPS Usage for Policy Provisioning.
CR	Core Routers.
DID	Direct Inward Dialing.
DS	Differentiated Service.
DSCP	*DiffServ* code point.
EF	Expedite Forwarding.
ER	Edge Routers.
FTP	File Transfer Protocol.
GAIT	Group for Advanced Information Technology.
GARA	Globus Architecture for Reservation and Allocation.
GGSN	Gateway GPRS Support Node.
GPRS	General Packet Radio Service.
HTB	Hierarchic Token Bucket.
HTTP	Hypertext Transfer Protocol.
IETF	Internet Engineering Task Force.

IMS	Internet ProtocolMultimedia Subsystem.
IPsec	Internet Protocol Security Protocol.
IPv4	Internet Protocol version 4.
IPv6	Internet Protocol version 6.
ISP	Internet Service Provider.
IVR	Interactive Voice Response.
LDAP	Lightweight Directory Access Protocol.
LPDP	Local Policy Decision Point.
LSP	Layered Service Provider.
MIME	Multipurpose Internet Mail Extensions.
MPLS	Multi Protocol Label Switching.
MS	Mobile Station.
NIST	National Institute of Standards and Technology.
NWG	Network Working Group.
OSPF	Open Shortest Path First.
P2P	Peer to Peer.
PBN	Policy- Based Networking
PDA	Personal Digital Assistant.
PDP	Policy Decision Point.
PEP	Policy Enforcement Point.
PHB	Per Hop Behavior.
PIB	Policy Information Base.
PSTN	Public Switched Telephone Network.
QoS	Quality of Service.
RAA	Resource Allocation Answer.
RAR	Resource Allocation Request.
RATES	Routing and Traffic Engineering Server.
RFC	Request For Comment.
RH	Remote Host.
RSVP	Resource reSerVation Protocol.
RTP	Real-time Transport Protocol.
SDP	Session Description Protocol.
SGSN	Supporting GPRS Support Node.
SIBBS	Simple Interdomain Bandwidth Broker Signaling.

SIP	Session Initialization Protocol.
SLA	Service Level Agreement.
SLS	Service Level Specification.
SNMP	Simple Network Management Protocol.
TC	Traffic Class.
TCL	Tool Command Language.
TCP	Transmission Control Protocol.
TLS	Transport Layer Security.
TOS	Type Of Service.
UAC	User Agent Client.
UAS	User Agent Server.
UDP	User Datagram Protocol.
UMTS	Universal Mobile Telecommunications System.
URI	Universal Resource Identifiers.
VLL	Virtual Leased Line.
VoIP	Voice over Internet Protocol.
VPN	Virtual Private Network.
WAN	Wide Area Network.
WWW	World Wide Web.

ÍNDICE DE MATÉRIAS

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

CAPÍTULO 1.

INTRODUÇÃO

1.1 Enquadramento Científico

A popularidade da Internet cresce de dia para dia. O crescimento do número de utilizadores e da quantidade de dados que nela circulam tornam a sua infra-estrutura, bem como os seus mecanismos de encaminhamento de tráfego, inadequados para lidarem com as crescentes exigências do tráfego de dados multimédia.

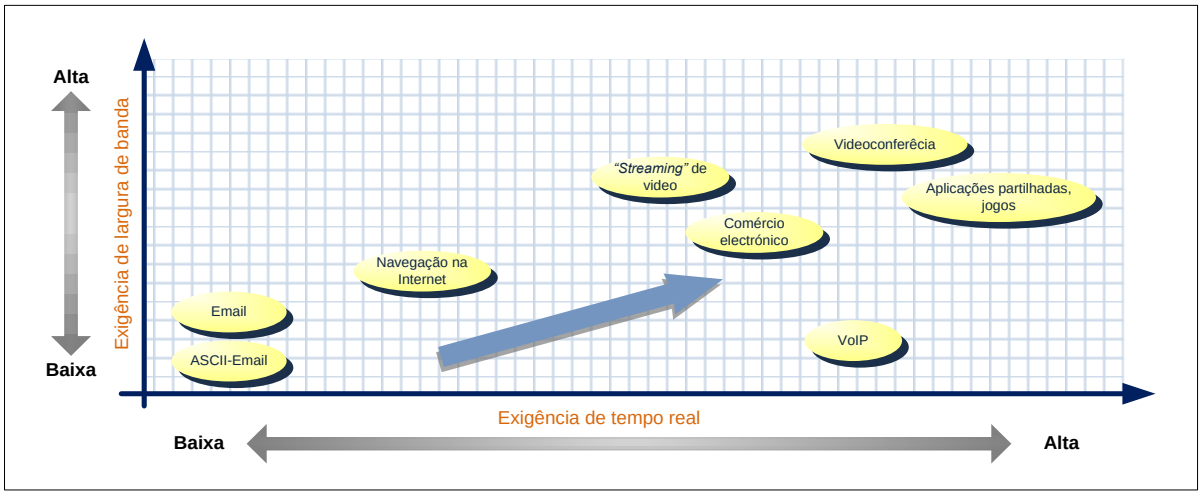

Figura 1.1– Exigências de QoS de diversos tipos de tráfego

Na Figura 1.1, são apresentados os principais tipos de tráfego de rede, ordenados de acordo com as suas exigências de QoS, tanto ao nível da largura de banda requerida (na vertical) como da sua susceptibilidade à latência e ao *"jitter"*. (na horizontal). É possível constatar que diferentes tipos de tráfego correspondem a diferentes exigências a nível de QoS. A titulo de exemplo, o correio electrónico (*"email"*) necessita de menos largura de banda que o *"streaming"* de vídeo e é também menos vulnerável à influência da latência e do *"jitter"*, pois esta só é perceptível ao utilizador quando estes factores atingem valores elevados. O mesmo já não se passa com o *"streaming"* de vídeo, que, para além de necessitar de uma maior largura de banda, requer também menores valores para a latência e para o *"jitter"*, para que a sua utilização seja feita em tempo real, com uma qualidade de serviço aceitável. No entanto, quando comparado com um serviço de videoconferência, o *"streaming"* de vídeo necessita de menos recursos para satisfazer as suas exigências de funcionamento em tempo real. A influência do *"jitter"* pode ser minimizada através da utilização de *"buffering"*, em que partes do vídeo são armazenados previamente ao seu visionamento, para que este seja feito sem variações na velocidade do som ou da imagem e a latência, se for constante, é notada apenas no inicio do visionamento. No caso da videoconferência, o conteúdo transmitido é gerado em tempo real, o que limita o uso de *"buffering"* e há uma interacção constante entre utilizadores, o que torna a influência da latência perceptível ao longo de toda a comunicação. Comparando dois tipos de tráfego com exigências de tempo real semelhantes, VoIP e videoconferência, verifica-se que no caso do VoIP é necessária menor largura de banda para se obter QoS, pois trata-se apenas da transmissão de áudio, enquanto no caso da videoconferência, é também necessário transmitir conteúdo de vídeo.

Actualmente, a Internet não faz distinções entre tipos de tráfego, o que leva a situações de congestão. Isto torna-se particularmente problemático quando se trata de tráfego que depende fortemente de parâmetros de QoS (Qualidade de Serviço), como transmissão de voz e imagem em tempo real, em que atrasos ou *"jitter"* na transmissão se tornam inaceitáveis. A solução mais simples passa pelo aprovisionamento em excesso de recursos. Mas esta é também uma solução dispendiosa e temporária, pois as necessidades de QoS de cada tipo de tráfego crescem de acordo com as exigências dos seus utilizadores (maiores resoluções de áudio e de vídeo, maior velocidade e fiabilidade na transferência de ficheiros, aplicações interactivas mais complexas). O que é considerado excessivo actualmente, pode rapidamente tornar-se insuficiente.

Uma alternativa reside num ambiente de rede que distinga vários tipos de tráfego com requisitos de QoS diferentes. No entanto, a obtenção de um serviço de QoS consistente numa

rede em larga escala torna-se problemática, pois é extremamente difícil (ou até impossível) a configuração manual dos parâmetros de filas de espera e dos mecanismos de processamento de tráfego correctos para cada elemento da rede, devido à sua heterogeneidade. É também necessário ter em conta que a configuração de QoS deve ser dinâmica, pois as necessidades de serviço em tempo real e a origem e destino dos fluxos de dados são variáveis. Para satisfazer estes requisitos, seria necessário desenvolver um protocolo uniforme, que forneça suporte à sinalização necessária para a configuração automática de QoS numa rede, envolvendo um processo de autenticação e autorização.

Nesta dissertação pretende-se apresentar um protocolo que satisfaça as necessidades apresentadas anteriormente, no intuito de criar um serviço global de QoS. Este protocolo assenta na utilização de SIP para sinalização, sendo possível obter as características do tipo de tráfego, assim como dos utilizadores. Essa informação é posteriormente utilizada para efectuar a gestão do tráfego, através do uso do protocolo COPS. A ligação entre os dois protocolos é obtida através de uma arquitectura de *"software"* que envolve a utilização de procuradores de largura de banda e de procuradores SIP. Embora o âmbito desta dissertação seja a utilização do protocolo desenvolvido num cenário de comunicação VoIP, a sua flexibilidade possibilita a sua adaptação para o suporte a outros tipos de aplicações.

1.2 Estrutura do Relatório

Esta dissertação encontra-se estruturada em seis capítulos.

Neste capítulo efectua-se o enquadramento científico da dissertação e a apresentação dos seus objectivos e da sua estrutura.

No capítulo 2 são apresentados conceitos básicos necessários para a elaboração e compreensão do trabalho realizado. São descritos os conceitos de Serviços Diferenciados, o protocolo COPS, o protocolo SIP e o serviço VoIP. Neste capítulo são também apresentadas referências bibliográficas de trabalhos relevantes sobre estes temas, para um conhecimento mais aprofundado.

O capítulo 3 apresenta trabalhos relevantes desenvolvidos sobre os temas em foco nesta dissertação (sistemas diferenciados, QoS, corretores de largura de banda, politicas de aprovisionamento).

No capítulo 4 apresenta-se uma possível solução de configuração transparente para o utilizador de uma rede com garantia de QoS, utilizando o sistema de mensagens do SIP e

compatível com os serviços VoIP e com o núcleo do UMTS para o serviço IMS. É apresentado o protótipo elaborado no âmbito desta dissertação e é feita uma descrição da sua implementação e do seu funcionamento, tanto a nível modular como a nível do sistema integrado.

Nos capítulos 5 são expostos os resultados experimentais obtidos com o protótipo, assim como uma descrição dos cenários de teste utilizados.

Para o capítulo 6 foram deixadas as considerações finais relacionadas com o tema da dissertação, uma discussão sobre os resultados experimentais apresentados no capítulo 5 e uma apresentação de perspectivas de trabalho futuro relacionadas com o tema apresentado.

CAPÍTULO 2.

CONCEITOS BÁSICOS

É possível definir o conceito de fluxo de diversas formas. Pode ser uma combinação dos endereços de origem e destino, do número do *"socket"* de origem e destino e da identificação de sessão. Pode também ser definido de uma forma mais geral como qualquer pacote de uma determinada aplicação ou interface de entrada.

O conceito de QoS refere-se à capacidade de uma rede prestar um serviço com a melhor qualidade possível a determinado tipo de tráfego, satisfazendo os seus requisitos técnicos em relação a factores como a latência (com grande influência para aplicações de tempo real e tráfego interactivo), *"jitter"* e perdas de pacotes. Para alcançar esse objectivo são utilizadas estratégias como gestão de congestão, policiamento e modelação de tráfego e ferramentas de eficiência de ligação. Quando se utilizam ferramentas de gestão de congestão, procura-se aumentar a prioridade de um fluxo através de um sistema de filas de espera, em que cada fila é atendida de forma diferente. De forma a evitar a congestão, o sistema de gestão descarta fluxos de menor prioridade em favor de outros de alta prioridade. Quando se utiliza uma estratégia de policiamento e modelação de tráfego, a prioridade de um feixe de dados é obtida limitando o débito dos restantes feixes. Ferramentas de eficiência de ligação limitam fluxos de grandes dimensões dando prioridade a fluxos menores.

Para estas estratégias, a melhoria de QoS para um determinado fluxo de tráfego é alcançada através do aumento da sua prioridade ou através da limitação da prioridade de fluxos concorrentes. A escolha da melhor estratégia a empregar (ou uma possível combinação de estratégias) é feita de acordo com as necessidades do tráfego com que a rede terá de lidar.

É também necessário garantir que, dando prioridade a determinados fluxos de tráfego, não se provoca a falha dos restantes fluxos de menor prioridade.

2.1 Serviços Diferenciados (*DiffServ*)

O grande crescimento do uso da Internet torna necessário o desenvolvimento de uma arquitectura escalável e que forneça garantias de QoS, como o *DiffServ*. A actual abordagem de "melhor esforço" utilizada pela Internet mostra-se insuficiente para a satisfação das necessidades de certas aplicações cujo uso tem vindo a crescer cada vez mais. Serviços como *e-mail* e transferência de ficheiros (FTP – *"File Transfer Protocol"*) são suportados pela actual estrutura da Internet. Com os melhoramentos a nível de largura de banda e das capacidades multimédia disponíveis para os utilizadores de computadores, torna-se evidente que cada vez mais serviços serão transmitidos pela Internet no futuro. O principal objectivo do *DiffServ* é acomodar um grande número de serviços e políticas de aprovisionamento em comunicações ponto-a-ponto ou de difusão dentro de um conjunto de redes. O *DiffServ* não define nenhum tipo de serviço em particular – apenas define o tratamento que cada pacote de dados recebe a cada salto (*"hop"*).

2.1.1 Arquitectura *DiffServ*

O diagrama da Figura 2.1 apresenta um domínio *DiffServ* típico com todos os seus principais elementos. É possível verificar a distinção entre os nós de fronteira e os nós interiores da rede. Numa arquitectura *DiffServ*, os nós de fronteira estabelecem a ligação entre um domínio *DiffServ* e outros domínios, que podem ter ou não a capacidade para *DiffServ*, existindo dois tipos: os nós de saída, que tratam o tráfego de saída do domínio e os nós de entrada, que tratam tráfego de entrada. Cada nó de fronteira pode desempenhar as funções de nó de saída e de entrada intercaladamente. Os nós interiores apenas se ligam a outros nós interiores ou de fronteira dentro do mesmo domínio. A maioria das operações *DiffServ* é efectuada nos encaminhadores situados nos nós de fronteira.

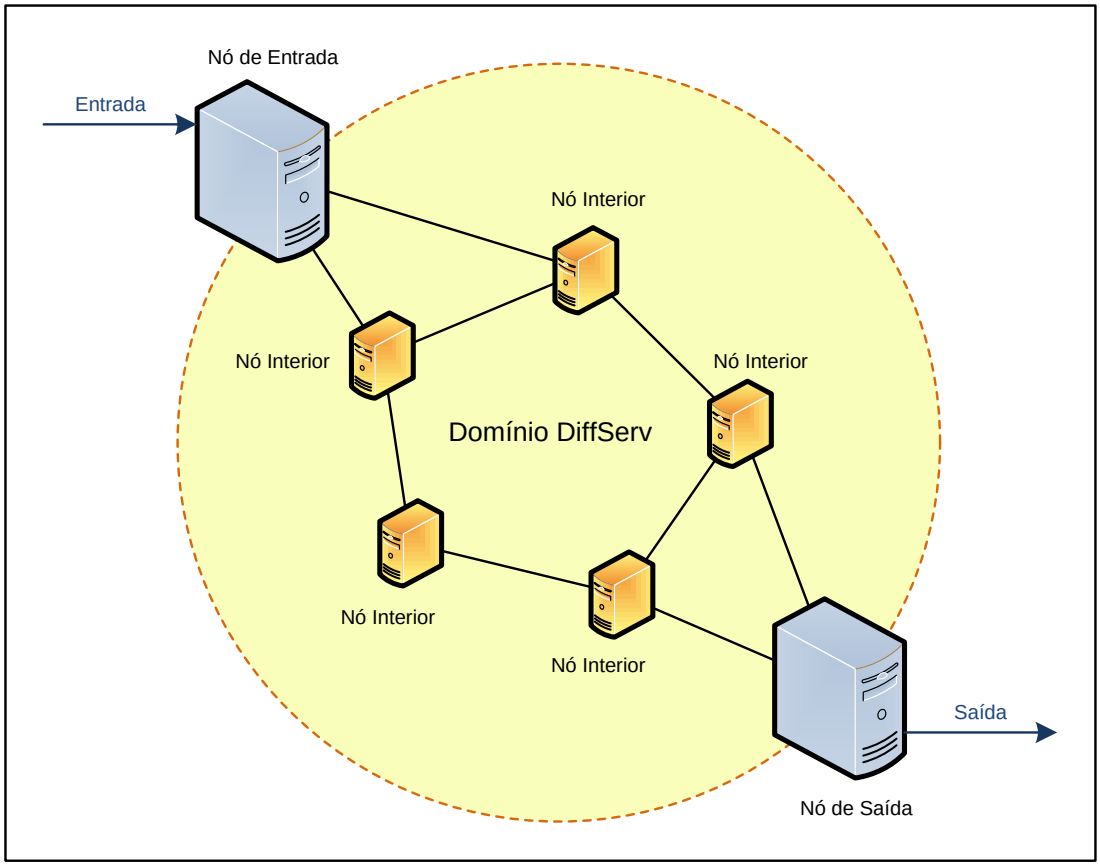

Figura 2.1– Arquitectura *DiffServ*

O RFC-2475 define o *DiffServ* como sendo uma arquitectura orientada para a implementação da diferenciação de serviços de forma escalável na Internet [RFC2475 1998].

Actualmente, os protocolos de transporte utilizados pela Internet são o IPv4 (*"Internet Protocol version 4"*) (de forma predominante) e o IPv6 (*"Internet Protocol version 6"*) (versão seguinte ao IPv4). Nos cabeçalhos dos pacotes definidos por ambos os protocolos, os campos utilizados pelo *DiffServ* são Tipo de Serviço (TOS – *"Type Of Service"*) para o IPv4 e o de Classe de Tráfego (TC – *"Traffic Class"*), para o IPv6. Estes dois campos foram renomeados como campo de Serviço Diferenciado (DS – *"Differentiated Service"*) devido ao seu uso na arquitectura *DiffServ* [RFC2474 1998].

Todos os pacotes que pertencem a uma determinada classe de tráfego são marcados com o mesmo valor no campo DS denominado *DiffServ "codepoint"* (DSCP – *"DiffServ Code Point"*). Cada DSCP define o tratamento que o pacote receberá em cada salto, ao passar por cada encaminhador *DiffServ*. Isto evita a necessidade de tratar cada fluxo de dados de forma separada, o que implicaria o armazenamento de informação acerca de cada fluxo ao nível dos encaminhadores. Assim, todos os pacotes marcados com o mesmo valor DSCP predefinido

são encaminhados da mesma forma. Esta capacidade de agregar fluxos de tráfego semelhantes aumenta a eficiência do funcionamento dos encaminhadores, especialmente em nós interiores.

O comprimento do campo DS é de oito bits, sendo apenas seis reservados para o DSCP para indicar o comportamento a cada salto (PHB – *"Per Hop Behaviour"*) de cada pacote ao longo do seu caminho (*"path"*), desde a fonte até ao seu destino. Os dois bits restantes não são utilizados, sendo ignorados pelos encaminhadores. A norma *DiffServ* especifica que todos os seis bits do DSCP devem ser utilizados para determinar o PHB. Qualquer especificação de PHB deve conter um valor de DSCP por omissão (*"default"*), possibilitando a remarcação de pacotes que cheguem a um nó com um valor desconhecido de DSCP, para evitar erros de funcionamento. Para a implementação *DiffServ*, o valor por omissão é '000000', que corresponde ao comportamento de Melhor Esforço (BE – *"Best Effort"*) de reenvio utilizado pela Internet.

O IETF (*"Internet Engineering Task Force"*) define dois tipos de comportamento de reenvio por salto: Reenvio Expedito (EF – "*Expedite Forwarding"*) e o Reenvio Assegurado (AF – *"Assured Forwarding"*).

O EF PHB pode ser utilizado para construir um serviço ponto-a-ponto com baixos valores para perdas, latência e *"jitter"* e largura de banda assegurada num domínio *DiffServ* [RFC2598 1999]. Este tipo de serviço também é designado de *"Premium Service"*, sendo destinado para aplicações que necessitem de uma largura de banda mínima assegurada. Todo o tráfego marcado com EF deve receber esta largura de banda mínima independentemente do tráfego em trânsito no nó. No entanto, é necessário garantir que o tráfego EF não bloqueie o restante tráfego de menor prioridade, o que pode ser obtido implementando mecanismos como a limitação por débito de repositório de testemunhos (*"token bucket rate"*), descartando o tráfego que excede o limite máximo. O valor de DSCP recomendado para usar com EF PHB é '101110'.

O AF PHB consiste num grupo de quatro classes de reencaminhamento. Dentro destas quatro classes, podem ser atribuídos aos pacotes IP três níveis diferentes de precedência de descarte [RFC2597 1999]. Isto permite aos fornecedores de serviços de Internet designar tráfego em diferentes classes dependendo dos serviços a que cada cliente tem direito. Desta forma, é possível estabelecer um valor monetário a cobrar diferente para cada classe. A cada uma das classes são aprovisionados determinados recursos e cada pacote é então colocado em fila de espera consoante a sua classe. Quando ocorre a saturação dos recursos aprovisionados a cada classe, entram em acção os diferentes níveis de precedência de descarte, que determinam a prioridade dentro de cada classe de AF. Quanto menor o nível de precedência

de descarte, maior será a prioridade do pacote e caso ocorra congestionamento no nó, os pacotes de menor prioridade (nível mais alto) serão descartados.

O AF PHB necessita de mais do que um DSCP. Quatro classes, cada uma com três níveis de precedência de descarte, equivalem a doze valores distintos para o DSCP. Na tabela seguinte encontram-se os diferentes valores de DSCP para AF PHB.

Precedência de Descarte	AF1	AF2	AF3	AF4
Baixa	001010	010010	011010	100010
Média	001100	010100	011100	100100
Alta	001110	010110	011110	100110

Tabela 1 – Valores de DSCP para AF PHB

A combinação dos dois tipos de PHB e da abordagem "melhor esforço" da Internet possibilita a transmissão de diferentes tipos de tráfego num domínio *DiffServ* tendo em conta os requisitos de QoS.

2.1.2 Encaminhador *DiffServ*

As duas operações fundamentais de um encaminhador *DiffServ* são a classificação e condicionamento de tráfego. Os pacotes que chegam a um encaminhador têm de ser classificados para que seja possível determinar qual o reencaminhamento a dar-lhes baseado no valor contido no DSCP. O condicionamento de tráfego consiste na medição, policiamento, modelação e/ou remarcação do tráfego, de forma a assegurar que este entra num domínio *DiffServ* conforme a politica de aprovisionamento de serviço predefinida [RFC2475 1998].

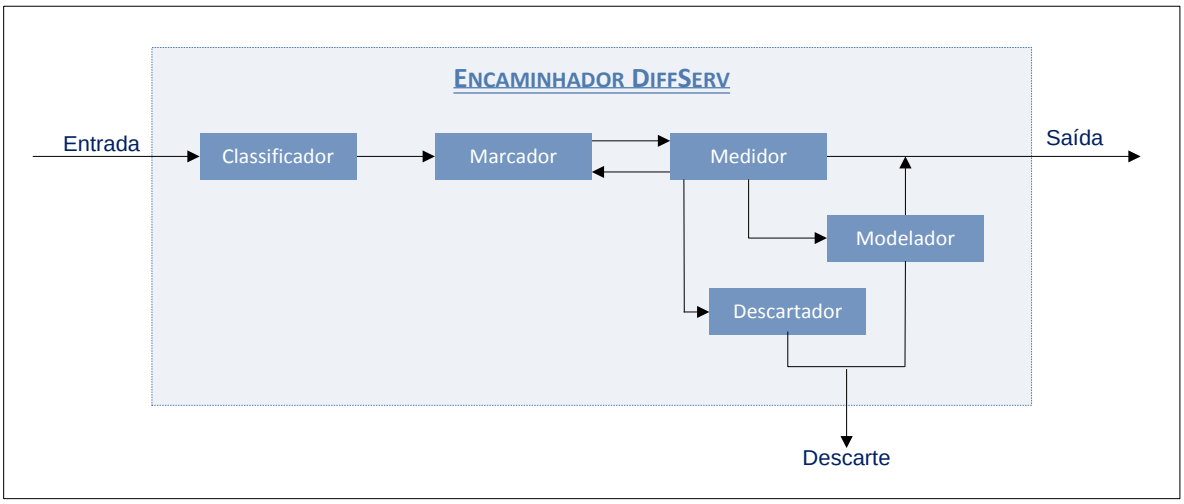

Figura 2.2– Componentes de um encaminhador *DiffServ*

Os componentes de um encaminhador *DiffServ* são os seguintes:

- Classificador – O classificador selecciona os pacotes de acordo com o conteúdo de determinados campos no seu cabeçalho. O encaminhador olha para o valor de DSCP no cabeçalho de um pacote para determinar qual o reencaminhamento a dar-lhe. Adicionalmente, pode ser necessário ao encaminhador determinar qual a origem, destino ou outros parâmetros contidos no cabeçalho do pacote de forma a proceder ao seu reencaminhamento. Na posse desta informação, o classificador pode colocar o pacote na fila de espera correcta a aguardar reencaminhamento;

- Medidor – O medidor é utilizado para verificar se um fluxo de entrada está de acordo com o perfil de tráfego negociado. Os pacotes que falham esta verificação são ou remarcados com um novo valor de DSCP, ou passados directamente para o modelador ou então são simplesmente descartados. O medidor pode ser utilizado para a gestão da contabilidade da rede [Jha 2002];

- Marcador – Os marcadores de pacotes colocam no campo DS do pacote o valor de DSCP, acrescentando o pacote a um agregado de pacotes classificado para receber o mesmo tratamento. Um marcador pode ser configurado para remarcar todos os pacotes que lhe são enviados com um determinado DSCP ou então remarcar com um valor de DSCP baseado no estado do medidor;

- Modelador/Descartador – Os modeladores de tráfego são necessários para atrasar alguns ou todos os pacotes de um fluxo de tráfego de forma a garantir que o fluxo se encontra de acordo com a especificação de tráfego negociada. O modelador tem

uma capacidade de armazenamento de pacotes limitada, o que resulta no descarte dos pacotes que excedem essa capacidade. Um descartador elimina os pacotes que se encontram fora da especificação de forma a tornar o fluxo conforme com essa mesma especificação e pode ser implementado como um modelador com uma capacidade de armazenamento nula.

2.1.3 Segurança

Um possível problema de segurança relacionado com o DSCP é o uso indevido de serviços por pacotes marcados com um código DSCP a que não têm direito. Isto torna necessária a autenticação e controlo de admissão de forma a garantir que cada pacote só é marcado com um DSCP a que tem direito, segundo um acordo de serviço previamente estabelecido. É possível aproveitar este uso indevido de serviços para provocar um ataque de negação de serviço, injectando pacotes marcados ilegalmente na rede, sobrecarregando o sistema e resultando na incapacidade de processar pacotes legítimos devido à falta de recursos disponíveis. A principal defesa contra este tipo de ataque consiste na combinação de condicionamento de tráfego na fronteira do domínio DS, com segurança e integridade na infra-estrutura da rede interna do mesmo [RFC2474 1998]. Isto implica de os nós de fronteira de um domínio *DiffServ* devem verificar se os pacotes se encontram devidamente marcados e se provêm de um domínio *DiffServ* autenticado. Podem também remarcar pacotes cuja origem não seja possível autenticar de forma a garantir a segurança no domínio. Ao aplicar este policiamento na fronteira do domínio, evita-se que os nós internos tenham de fazer qualquer processamento adicional dos pacotes, limitando-se a reencaminhá-los de acordo com o DSCP. Isto garante que os nós internos não sofrem perdas de velocidade de operação.

Outro problema a ter em conta é que a maioria dos encaminhadores actualmente em uso na Internet não processa o campo TOS dos pacotes IPv4. Assim sendo, para se implementar um ambiente *DiffServ* é necessário configurar os encaminhadores para efectuarem esse processamento.

2.1.4 Acordo de Nível de Serviço (SLA)

De forma a determinar qual o tratamento que os pacotes recebem a cada salto é necessário estabelecer uma forma de negociação prévia de serviços antes da transmissão. A arquitectura *DiffServ* define um Acordo de Nível de Serviço (SLA – *"Service Level*

Agreement") como um contrato entre um cliente e uma entidade provedora de serviço que especifica qual o serviço de reencaminhamento a que o cliente tem direito [RFC2475 1998]. Um SLA é um contrato formal no qual figuram factores como a disponibilidade da rede, acordos de pagamento, assim como outros factores legais e comerciais. Um SLA garante que o tráfego fornecido por um cliente que cumpra as condições estabelecidas previamente será transportado e entregue pelo provedor de serviços. Dependendo do contrato firmado, a falha no fornecimento do serviço pode levar a uma compensação monetária ao cliente ou outras consequências legais para a entidade provedora. Um SLA contém a especificação técnica do reencaminhamento que determinado fluxo de tráfego receberá e também parâmetros adicionais como privilégios de acesso.

Um SLA é um documento parcialmente técnico que é firmado entre os administradores de rede e os seus clientes, estabelecendo uma obrigação legal entre ambas as partes como qualquer outro contracto. A equipa de desenvolvimento de sinalização QBone defende que um SLA é estabelecido entre pessoas (utilizador e fornecedor de serviço) e não entre máquinas. Corretores de largura de banda não se envolvem em negociações de SLAs e não os comunicam aos seus pares [Chimento 2002].

2.1.5 Especificação de Nível de Serviço (SLS)

Uma vez que um SLA contém informação não técnica, não é passível de ser utilizado directamente por um corretor de largura de banda. Surge assim a especificação de nível de serviço (SLS – *"Service Level Specification"*), que contém exclusivamente os detalhes técnicos especificados por um SLA. Essencialmente, trata-se da tradução de um SLA em parâmetros que irão ser utilizados para configurar entidades na rede. Uma SLS dita como o tráfego será tratado dentro de um domínio *DiffServ*, desde as permissões de acesso de determinado fluxo até ao reencaminhamento que os pacotes desse fluxo receberão dentro do domínio. Quando um domínio aceita as condições descritas numa SLS, torna-se responsável por garantir que essa especificação é cumprida durante o tempo definido no acordo.

Uma SLS não é uma reserva de recursos mas sim um compromisso que permite fazer reservas baseadas nas condições descritas na SLS.

Quando o destino de um fluxo se encontra fora do domínio *DiffServ* de origem, torna-se necessária a existência de especificações de serviço entre domínios adjacentes na rota traçada pelo fluxo. Devido à complexidade da natureza dos parâmetros e dos recursos da rede, a equipa de desenvolvimento de sinalização QBone definiu apenas dois tipos de diferentes de

protocolos de negociação de SLS: Fase 0 e Fase 1. Nestas duas fases, os termos de SLSs bilaterais são propagados através de outro protocolo ou manualmente, para que dois corretores de largura de banda possam compreender a SLS em vigor entre estes [Chimento 2002].

2.1.6 Pedido de Aprovisionamento de Recursos (RAR)

Um Pedido de Aprovisionamento de Recursos (RAR – *"Resource Allocation Request"*) é um pedido de recursos seguindo a definição estabelecida na SLS. Se o RAR não cumpre ou excede as condições da SLS, então é rejeitado. Se o RAR é aceite, então os pontos de policiamento do domínio *DiffServ* têm de ser configurados de forma a lidarem como novo tráfego *DiffServ* de acordo com a SLS [Chimento 2002]. Uma Resposta de Aprovisionamento de Recursos (RAA – *"Resource Allocation Answer"*) é então retornada ao anfitrião (*"host"*) que aguarda a confirmação de que a rede foi reconfigurada com sucesso de acordo com a SLS em vigor. Um RAR só pode ser utilizado para requisitar recursos segundo uma SLS a que tenha direito. Assim sendo, é necessário incluir no RAR a identidade da SLS para a qual pretende reservar os recursos.

2.1.7 Gestão de Redes Baseada em Politicas (PBN)

A arquitectura de uma Rede Baseada em Políticas (PBN – *"Policy-Based Network"*) consiste em dois tipos de elementos:

- Pontos de decisão (PDP – *"Policy Decision Points"*);
- Pontos de aplicação de políticas (PEP – *"Policy Enforcement Points").*

Um PEP é um encaminhador (ou outro dispositivo de processamento de tráfego IP) que implementa um controlo de admissão para fluxos de dados baseado em políticas. Um PDP tem uma visão global da rede através dos seus PEPs e é o PDP que decide o tratamento a dar a cada fluxo de dados de acordo com as politicas em vigor, existindo pelo menos um PDP em cada domínio.

A interacção básica entre elementos começa no PEP quando este recebe uma notificação que requer uma decisão. Em resposta a este evento, o PEP formula um pedido de decisão e envia-o ao PDP. Este efectua a decisão baseando-se num repositório de políticas e retorna o resultado ao PEP, que aplica o resultado da decisão [RFC2753 2000]. Esta interacção é efectuada através da utilização do protocolo COPS, mais concretamente da sua extensão

COPS-PR, que estabelece o formalismo da comunicação entre ambos (este protocolo é descrito na secção 2.2). Esta interacção pode ser observada na Figura 2.3.

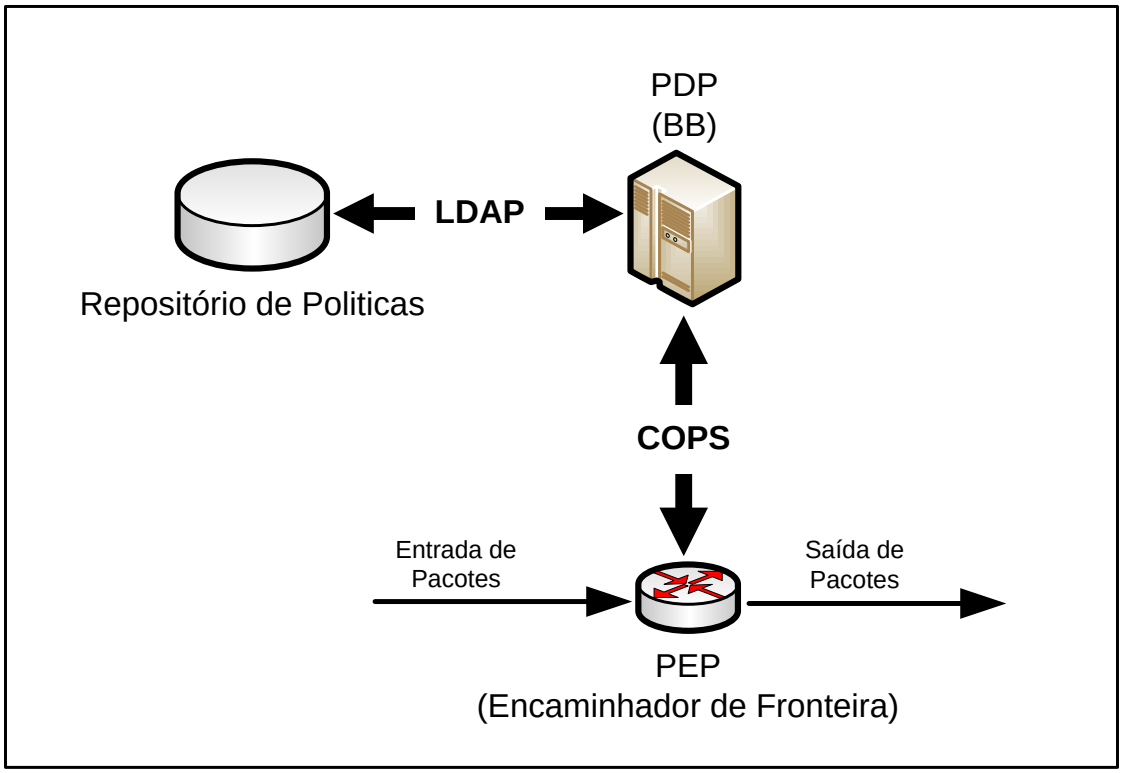

Figura 2.3 – Arquitectura de PBN

As políticas são guardadas num repositório. Um repositório centralizado pode ser utilizado para a distribuir e partilhar informação acerca de políticas entre sistemas múltiplos. Este repositório pode ser implementado como um serviço de directórios utilizando LDAP (*"Lightweight Directory Access Protocol"*) para aceder às políticas nele armazenadas, sendo este o padrão para o armazenamento de políticas em sistemas inter-operáveis.

As políticas num directório têm normalmente um elevado grau de abstracção e cabe ao sistema de políticas (PDP e PEP) traduzi-las em especificações para a configuração da rede. Tipicamente, as políticas são expressas como condições tais como:

```
SE: UTILIZADOR = Premium;
E: NOME = xpto;
ENTÃO: largura_de_banda > 10 Mbits/s.
```

2.1.8 Corretor de Largura de Banda (BB – *"Bandwidth Broker"*)

Um corretor de largura de banda é uma forma de PDP. O BB está envolvido no aprovisionamento de recursos de rede, especialmente de largura de banda. Assim sendo, é necessário que o BB tenha conhecimento do estado dos recursos da rede (capacidade total de largura de banda e largura de banda disponível instantânea).

Geralmente, um BB pode receber um RAR de um anfitrião no mesmo domínio ou de um BB de outro domínio. O BB responde então a esse pedido de acordo com o estado de largura de banda e informação de políticas correntes da rede, podendo retornar uma confirmação ou negação de serviço. Os efeitos de um RAR podem ser a configuração de encaminhadores de fronteira ou a geração de RARs adicionais para a reserva de recursos em domínios fronteiros.

É possível a uma entidade provedora de serviços de Internet (ISP – "*Internet Service Provider"*) negociar dinamicamente diferentes SLAs com um determinado cliente. Em alternativa, um ISP pode cobrar taxas diferentes de acordo com a largura de banda requisitada. Para que isto seja possível é necessário que o BB seja capaz de comunicar com BBs remotos de forma a negociar a SLS a utilizar e com PEPs locais para determinar o estado da rede e alterar a sua configuração. O BB tem em consideração a capacidade total da rede em satisfazer os pedidos recebidos. Por exemplo, se um utilizador requisita que a sua largura de banda mínima seja aumentada de 1 Mbps para 4 Mbps, o BB terá de verificar o estado da rede durante esse período, assim como os compromissos assumidos com outros utilizadores antes de tomar alguma decisão.

Um BB é uma entidade complexa que requer a integração de diversas tecnologias tais como uma interface para a comunicação dentro de cada domínio e entre domínios, protocolo de comunicação e base de dados. As políticas organizacionais podem ser configuradas utilizado o BB. A nível das interacções entre domínios, as responsabilidades de um BB incluem a configuração dos encaminhadores de fronteira para implementar o aprovisionamento de recursos e o controlo de admissão de tráfego. Um BB pode ainda ser utilizado numa arquitectura *DiffServ* para providenciar QoS a aplicações em tempo real tais como VoIP. O mecanismo de controlo de admissão de tráfego do BB permite optimizar a utilização dos recursos da rede, o que resulta num aumento da margem de lucro dos ISPs.

2.2 Protocolo COPS

2.2.1 Introdução

O protocolo COPS [RFC2748 2000] é uma das formas de aprovisionamento de políticas de gestão de tráfego em redes, existindo uma extensão ao protocolo específica para esse fim: o COPS-PR (*"COPS Usage For Policy Provisioning"*) [RFC3084 2001].

Este protocolo permite a provisão de políticas para múltiplos clientes referentes a QoS ou segurança. Adicionalmente, através das extensões COPS-PR, fornece também as funcionalidades de suporte à gestão de redes baseada em políticas (PBN – *"Policy Based Networking"*).

A nível técnico, o protocolo COPS assenta num modelo cliente/servidor que opera sobre TCP (*"Transmission Control Protocol"*), em que o servidor atende pedidos de políticas por parte do cliente.

As extensões COPS-PR definem os mecanismos e convenções que regulam a comunicação entre servidor de aprovisionamento (PDP) e o cliente (PEP), sendo definidas independentemente do tipo de política a ser aprovisionada [RFC2753 2000]. Outras definições a ter em conta são:

- Elemento ou Nó de Rede: encaminhadores, *"switches"*, *"hubs"*.
- Política: combinação de regras e serviços em que as regras definem os critérios para acesso e utilização dos recursos.
- Objecto de Política: contém informação de política em resposta a uma decisão de atribuição de recursos
- Elemento de Política: unidades de informação do Objecto de Política.
- Modelo *"Soft State"*: mecanismo automático de limpeza do estado instalado num PEP ou PDP. É accionado em caso de falha de comunicação ou falha do elemento de rede.
- Estado Instalado: solicitação nova e única feita de um PEP para um PDP, que deve ser explicitamente removida.
- LPDP (*"Local Policy Decision Point"*): um PDP existente no mesmo elemento de rede que o cliente PEP.

2.2.2 Principais Características

As principais características do COPS e do COPS-PR são especificadas da seguinte forma:

- Do lado do cliente, o PEP envia mensagens de pedido, actualização ou eliminação ao servidor e recebe as decisões retornadas.

- Do lado do servidor, o PDP recebe as mensagens enviadas pelo cliente e retorna as decisões tomadas.

- A definição do protocolo é orientada para a administração, configuração e aplicação de políticas, sendo extensível e podendo suportar objectos auto-identificados e informações específicas relativas aos clientes.

- A ligação TCP é iniciada pelo PEP através do porto 3288, sendo mantida enquanto ambas as partes (PEP e PDP) se encontrarem activas.

- A autenticação e a segurança do canal de comunicação entre o PDP e o PEP são feitas através do uso de IPsec (*"Internet Protocol Security Protocol"*) ou de TLS (*"Transport Layer Security"*).

- A informação sobre o estado de pedido/resposta da comunicação entre clientes e o servidor é armazenada, sendo partilhada entre ambos até ser explicitamente apagada. Os estados de diferentes eventos podem estar associados, podendo levar a que novos pedidos obtenham respostas diferentes.

- A informação de configuração pode ser enviada ao cliente pelo servidor em modo não solicitado, podendo também o servidor remover esta configuração quando se torna obsoleta.

Suporta dois modelos para controlo baseado em políticas:

- O *"outsourcing"*, em que o PEP delega a responsabilidade de tomar decisões de politicas a um PDP externo, sempre que for necessário;

- O aprovisionamento (*"provisioning"*), em que o PDP envia decisões de politicas para o PEP de forma não solicitada como resposta a eventos externos ou eventos do próprio PEP, podendo ser feito com a configuração completa ou parcial.

2.2.3 Funcionamento

O PEP e os seus parâmetros de configuração são descritos em solicitações de políticas que são enviadas pelos encaminhadores de fronteira para o servidor de aprovisionamento, sempre que se registam alterações nestes parâmetros.

As decisões efectuadas pelo COPS no servidor de aprovisionamento são enviadas para os encaminhadores de fronteira, não só em resposta às solicitações mas também devido a eventos externos ou do PDP, tais como actualizações de políticas e SLA.

No PDP, os atributos de suporte enviados pelo PEP têm prioridade sobre as políticas no envio de decisões. Exemplificando, se um atributo é especificado como não suportado pelo PEP mas é especificado por uma política no PDP, então essa política não é enviada para o PEP em causa. Existe, no entanto, uma excepção no tratamento de tabelas de atributos de suporte *'Notify'*, que são enviadas apenas por convenção e não afectam as políticas enviadas pelo servidor (as decisões deste são baseadas no suporte às tabelas *'install'* e *'install-notify'*).

2.3 Protocolo de Início de Sessão (SIP)

O SIP é um protocolo de controlo de nível de aplicação desenvolvido pelo Grupo de Trabalho de Engenharia da *Internet* (IETF – *"Internet Engineering Task Force"*) [RFC3261 2002].

Este protocolo é utilizado para criar, modificar e terminar sessões com um ou mais utilizadores. O conceito de sessão refere-se à comunicação estabelecida entre dois ou mais utilizadores, funcionando como emissores e receptores e ao estado dos utilizadores durante essa comunicação. Alguns exemplos de sessões SIP são chamadas telefónicas VoIP, conferências multimédia, jogos de comutador em rede, distribuição (*"streaming"*) de conteúdos multimédia (vídeo e áudio), trocas de ficheiros, etc.

O SIP não é um protocolo de uso geral. A sua função é tornar a ligação possível mas é necessário utilizar outro meio (possivelmente outro protocolo) para a estabelecer. Por exemplo, dois protocolos que são normalmente utilizados em conjunto com o SIP são o RTP (*"Real-time Transport Protocol"*) [RFC35502003] e o SDP (*"Session Description Protocol"*) [RFC45662006]. O RTP é utilizado para o transporte de dados multimédia em tempo real (áudio, vídeo e texto), permitindo a sua codificação e divisão em pacotes. O SDP é utilizado para descrever e codificar as características dos participantes de uma sessão. Esta descrição é

posteriormente utilizada na negociação das características da sessão (negociação dos *"codecs"* a utilizar para codificar e descodificar os dados em trânsito durante a sessão, negociação do protocolo de transporte a utilizar) para que todos os intervenientes estejam de acordo.

A implementação do SIP foi feita em conformidade com o modelo de *"Internet"*. Trata-se de um protocolo de sinalização entre terminais (*"end-to-end"*), o que significa que a lógica interveniente na comunicação encontra-se nos terminais da ligação (exceptuando o encaminhamento das mensagens SIP). A informação de estado também é armazenada nos terminais da ligação. Este tipo de arquitectura apresenta vantagens de escalabilidade e distributividade, à custa de um excesso (*"overhead"*) de mensagens mais elevado, causada pela troca de mensagens entre terminais. Este conceito de ligação entre terminais diverge do modelo utilizado pela Rede Publica de Comutação Telefónica (PSTN– *"Public Switched Telephone Network"*), em que a lógica e informação de estado referente à ligação se encontra na rede e os terminais (telefones) são elementos simples. Um dos objectivos do SIP é fornecer o mesmo tipo de funcionalidade que a PSTN tradicional mas a sua arquitectura entre terminais permite-lhe ainda a implementação de serviços adicionais que dificilmente poderiam ser implementados em redes telefónicas tradicionais.

A robustez do SIP deriva das seguintes características:

- Reutilização de componentes – o SIP utiliza extensões MIME (*"Multipurpose Internet Mail Extensions"*) para o transporte de informações adicionais e utiliza identificadores universais de recursos (URI – *"Uniform Resource Identifier"*) para endereçamento, tal como o HTTP (*"Hypertext Transfer Protocol"*);

- Escalabilidade – os utilizadores podem estar situados em qualquer ponto da Internet e serem convidados para várias sessões simultaneamente;

- Interoperabilidade – uma vez que o SIP é uma norma aberta, é possível à comunidade de desenvolvimento construir uma grande variedade de implementações que podem comunicar com outros produtos baseados em SIP.

2.3.1 Identificador Uniforme de Recursos (URI)

Uma entidade SIP é identificada utilizando SIP URI. Um exemplo de SIP URI tem a seguinte forma:

```
sip:utilizador@dominio.com
```

Um SIP URI é composto por uma parte referente ao utilizador e outra referente ao domínio separadas pelo símbolo '@'. É possível constatar que os SIP URIs têm uma estrutura semelhante aos endereços de correio electrónico (*"e-mail"*), o que os torna fáceis de memorizar.

2.3.2 Elementos de Rede

Numa configuração de rede simples é possível utilizar apenas dois agentes de utilizador SIP para estabelecer uma troca de mensagens SIP entre dois utilizadores. No entanto, uma rede SIP típica possui mais tipos de elementos SIP como procuradores, *"registrars"* e servidores de redireccionamento. Estes elementos são normalmente entidades lógicas. Para minimizar a utilização de recursos (como tempo de processamento), estas entidades podem ser agregadas, dependendo da implementação e configuração da arquitectura de rede utilizada.

2.3.2.1 Agentes de Utilizador

Dois terminais de ligação que utilizam SIP para se encontrarem e negociarem uma sessão entre si designam-se por agentes de utilizador (*"user agents"*). Este tipo de entidade é normalmente uma aplicação de *"software"* que opera no computador de cada utilizador mas também pode ser um telemóvel, um PDA (*"Personal Digital Assistant"*), um portal PSTN, um sistema automatizado IVR (*"Interactive Voice Response"*), etc.

Cada agente é composto por duas unidades lógicas: o Agente Servidor de Utilizador (UAS – *"User Agent Server"*) e o Agente Cliente de Utilizador (UAC – *"User Agent Client"*). O UAS é a parte responsável pela recepção de pedidos e pelo envio de respostas, enquanto o UAC é responsável pela recepção de respostas e pelo envio de pedidos. Por vezes o agente de utilizador é designado como UAC ou UAS dependendo da função que desempenha.

Numa interacção SIP básica, o agente de um utilizador inicia uma secção SIP, estabelecendo uma ligação ao agente de outro utilizador. Envia então um pedido 'INVITE',

aguardando em seguida a respectiva resposta. Caso a sessão seja aceite pelo agente receptor, este envia uma resposta '200 OK', que é confirmada com uma resposta do tipo 'ACK', enviada pelo agente do utilizador que iniciou a sessão. Ambos os agentes podem terminar a sessão enviado um pedido 'BYE' (para uma descrição mais aprofundada das mensagens SIP, ver secção 2.3.3.).

Quando um agente inicia uma ligação, envia um pedido 'INVITE' e aguarda a respectiva resposta, a sua função é de UAC. Quando o agente funciona como destinatário de uma ligação, a sua função é de UAS, recebendo pedidos 'INVITE' e enviando as respectivas respostas. A situação inverte-se se o agente destinatário decidir terminar a ligação enviado uma mensagem 'BYE', pois actua agora como UAC enquanto o agente do utilizador que iniciou a chamada actuará agora como UAS, recebendo o 'BYE' e enviando a subsequente resposta.

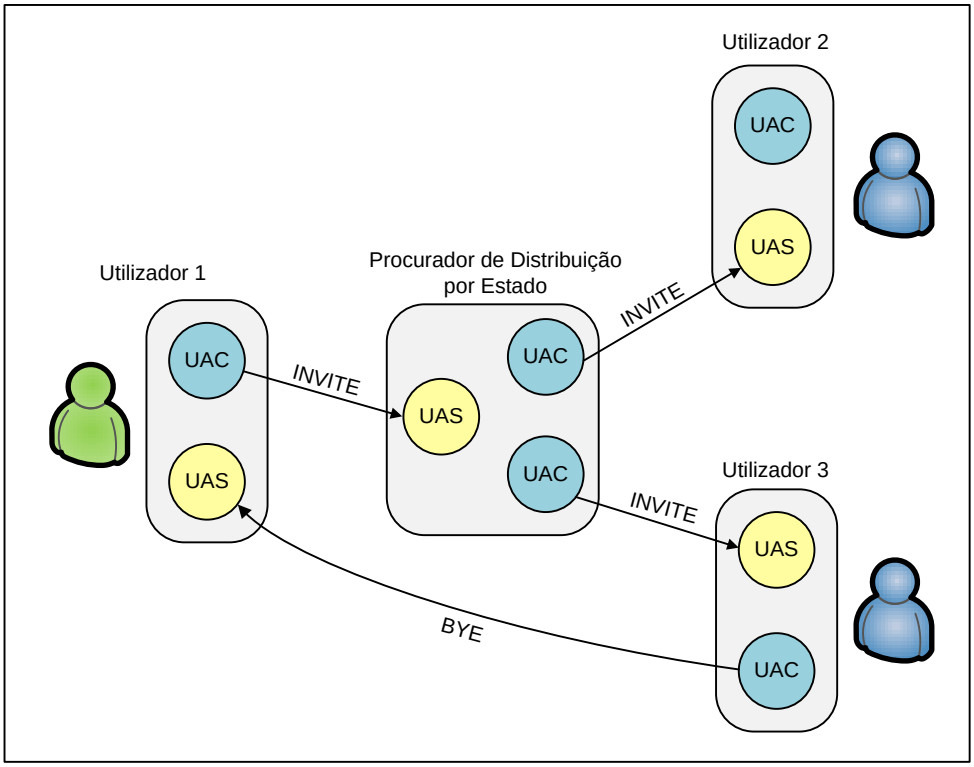

Figura 2.4 – UAC e UAS

Na Figura 2.4 é possível observar três agentes de utilizador e um procurador de distribuição por estado (*"stateful forking proxy"*). A parte do procurador que recebe o 'INVITE' proveniente do 'Utilizador 1' funciona como UAS, criando depois duas instâncias

de UAC para o reenviar por estado (*"statefully"*) para os destinatários ('Utilizador 2' e 'Utilizador 3').

2.3.2.2 Procuradores SIP

O SIP permite a criação de infra-estruturas de rede designadas procuradores (*"proxy servers"*). Este tipo de entidade permite o registo de utilizadores, o encaminhamento de pacotes de início de sessão de acordo com a localização do destinatário, autenticação e outras funções.

Existem dois tipos básicos de procurador – por estado e sem estado (*"stateless"*).

- **Procuradores sem estado**

Este tipo de procurador apenas reencaminha mensagens, fazendo-o independentemente para cada uma. Embora as mensagens sejam normalmente agregadas em transacções, procuradores sem estado não desempenham o controlo de transacções. Podem ser usados para balancear a carga de tráfego, tradução de mensagens ou simples encaminhamento.

Os procuradores sem estado são mais simples e mais rápidos que os procuradores por estado. No entanto são mais limitados, não sendo capazes de efectuar a reabsorção de mensagens transmitidas ou encaminhamento mais avançado como bifurcação de mensagens ou travessia recursiva.

- **Procuradores por estado**

Este tipo de procurador é mais complexo. Ao receber um pedido, um procurador por estado cria um estado e faz a sua manutenção até que a transacção termine. Algumas transacções, como as iniciadas com um 'INVITE', podem ser longas (neste caso, duram até que o receptor atenda ou recuse a chamada). Como um procurador por estado necessita de manter um estado para cada ligação durante a sua duração, isto apresenta um limite para o seu desempenho.

Um procurador por estado é capaz de reabsorver mensagens transmitidas pois mantém o estado da transmissão, sabendo assim se já recebeu a mesma mensagem.

A sua capacidade para associar mensagens SIP em transacções permite-lhe bifurcar mensagens, isto é, enviar duas ou mais mensagens após receber uma.

É também possível efectuar formas mais complexas de encaminhamento com este tipo de procurador. Por exemplo, o reenvio de uma chamada destinada a um determinado utilizador para o seu telemóvel quando este não a atende no seu destino original (falha na

transacção com o destino original). Um procurador sem estado não é capaz de executar esta tarefa pois não tem como saber o resultado da transacção com o destino original.

- **Utilização de procuradores**

Um exemplo típico da utilização de procuradores pode ser uma entidade centralizada (por exemplo uma companhia ou uma universidade) que possui o seu próprio procurador SIP, que é utilizado por todos os utilizadores englobados nessa entidade.

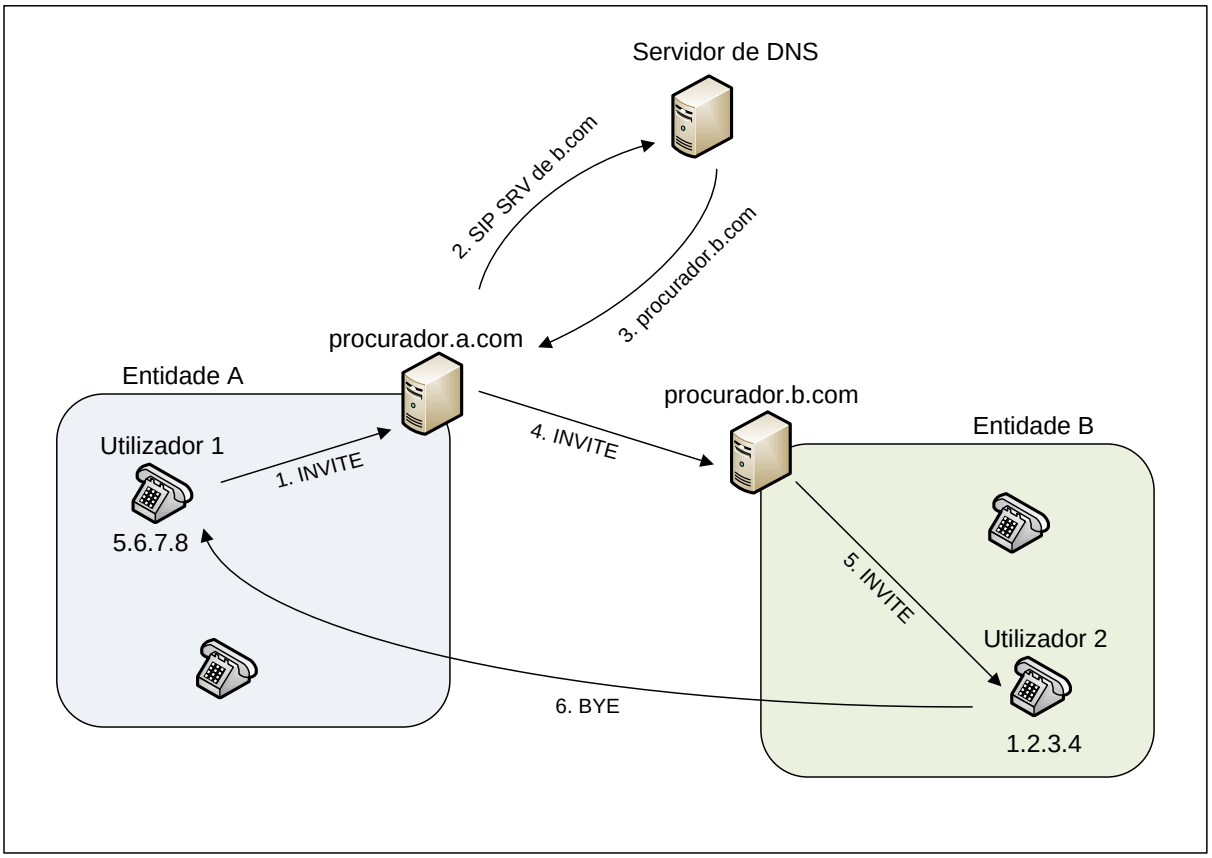

Figura 2.5 - Estabelecimento de uma sessão utilizando um procurador SIP

Na Figura 2.5 é apresentada uma situação em que o 'Utilizador 1' inserido na 'Entidade A' estabelece uma sessão com o 'Utilizador 2' da 'Entidade B'. As entidades 'A' e 'B' têm o seu próprio procurador SIP.O 'Utilizador 1' utiliza o endereço `sip:utilizador2@b.com` para comunicar com o 'Utilizador 2'. O agente de utilizador do 'Utilizador 1' não tem conhecimento de como enviar o 'INVITE' até ao seu destino final mas está configurado para enviar o seu tráfego para o procurador SIP da sua entidade no endereço `procurador.a.com`. O procurador então determina que o destinatário se encontra noutra entidade e reenvia o pedido para o seu procurador. O seu endereço pode estar pré-configurado no procurador A ou

alternativamente pode utilizar o registo DNS SRV para o encontrar. O 'INVITE' ao chegar a `procurador.b.com` é então reenviado para o 'Utilizador 2'. O 'Procurador B' tem conhecimento que o 'Utilizador 2' se encontra associado ao endereço IP `1.2.3.4` e utiliza esse endereço.

2.3.2.3 Registrar

Para que um procurador SIP tenha conhecimento da localização de um determinado utilizador é necessário que este se registe no procurador. Isto é feito utilizando um *"registrar"*. Esta é uma entidade SIP responsável por receber registos de utilizadores, extrair a informação relacionada com a sua localização (endereço IP, porto, nome de utilizador) e armazenar essa informação numa base de dados. Utilizado o exemplo da Figura 2.5, o *"registrar"* estabelece uma relação entre o endereço `sip:utilizador2@b.com` com o endereço `sip:utilizador2@1.2.3.4.co:5060` e armazena-a na base de dados de localizações. Este repositório de informação é posteriormente utilizado pelo procurador SIP da entidade (neste caso, o 'Procurador B') para encontrar os utilizadores da sua entidade.

O "*registrar*" é normalmente apenas uma entidade lógica. Devido ao seu estreito relacionamento com o procurador, estas duas entidades costumam ser implementadas na mesma localização.

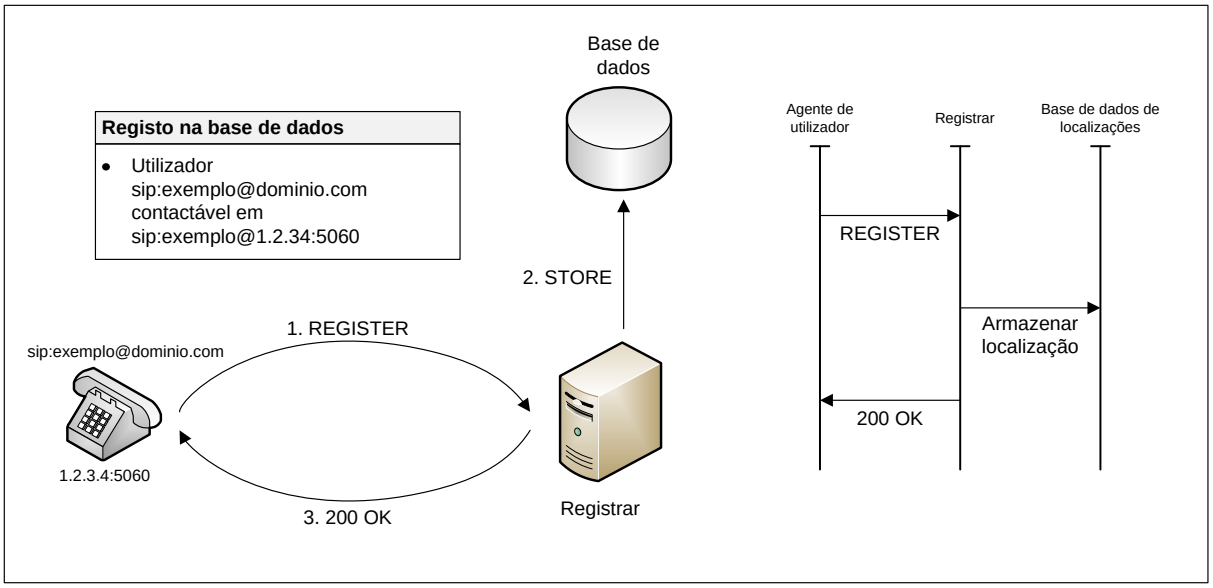

Figura 2.6 - Operação de registo de um utilizador

A Figura 2.6 mostra uma operação típica de registo SIP. Uma mensagem 'REGISTER' contendo o endereço de registo `sip:exemplo@dominio.com` e o endereço de contacto `sip:exemplo@1.2.3.4:5060` (em que 1.2.3.4 é o endereço IP do utilizador e 5060 o porto utilizado pelo seu agente para a comunicação SIP) é enviada ao "*registrar*". Este então extrai esta informação e armazena-a na base de dados de localizações. Caso a operação seja concluída com sucesso, o "*registrar*" envia uma resposta '200 OK' ao utilizador, concluindo o processo de registo.

Cada registo tem um prazo de validade limitado. Este prazo pode ser indicado no cabeçalho da mensagem 'REGISTER' no campo 'Expires' ou no parâmetro 'expires' do campo 'Contact'. O agente do utilizador necessita de renovar o registo para que este não expire.

2.3.2.4 Servidor de Reencaminhamento

Esta entidade recebe pedidos e envia respostas contendo a lista de actuais localizações associadas a determinado utilizador. Ao receber um pedido, o servidor de reencaminhamento procura na base de dados de localizações criada por um "*registrar*" as entradas associadas a determinado utilizador. Cria então uma lista contendo os resultados dessa procura e envia-a para a aplicação que originou o pedido com uma resposta de classe '3XX'. A lista é então extraída pelo agente do utilizador originário da chamada, permitindo-lhe enviar um 'INVITE' directamente para o utilizador destinatário.

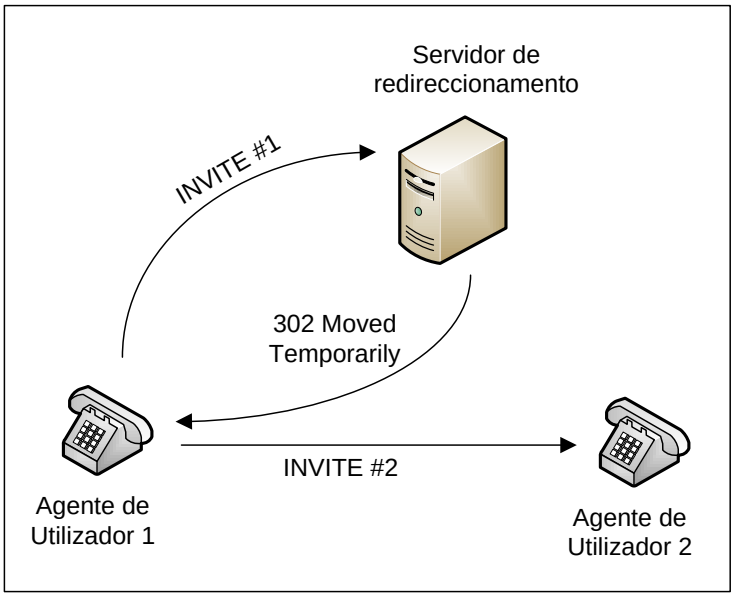

Figura 2.7 - Exemplo de reencaminhamento SIP

A Figura 2.7 mostra um exemplo de funcionamento de um servidor de reencaminhamento. O 'Utilizador 1' pretende estabelecer uma ligação com o 'Utilizador 2'. O seu agente envia um 'INVITE' para o servidor de reencaminhamento. Este então procura na base de dados de localizações as entradas referentes ao utilizador de destino e envia os resultados da procura ao agente do 'Utilizador 1' numa mensagem '302' indicativa de que o utilizador de destino alterou a sua localização temporariamente. O agente do 'Utilizador 1' recebe a mensagem e extrai a lista com a localização ou localizações alternativas para o 'Utilizador 2' e utiliza-a para enviar um segundo 'INVITE' directamente para o agente do 'Utilizador 2'.

2.3.3 Mensagens SIP

A comunicação utilizando SIP é normalmente designada por sinalização SIP. Esta é efectuada através de séries de mensagens que podem ser transportadas independentemente através da rede, sendo normalmente transportadas em datagramas UDP (*"User Datagram Protocol"*).

A primeira linha de cada mensagem constitui o seu cabeçalho (*"header"*), sendo o resto designado por corpo (*"body"*) da mensagem. No cabeçalho é indicado o tipo de mensagem que pode ser um pedido (*"request"*) ou uma resposta (*"response"*). Um pedido é utilizado para iniciar uma determinada acção ou para informar o recipiente de algo. Uma resposta é utilizada para confirmar a recepção e processamento de um pedido e contém informação respectiva ao estado desse processamento.

2.3.3.1 Pedidos SIP

Um exemplo de um pedido SIP tem a seguinte forma:

```
INVITE sip:utilizador@dominio1.com SIP/2.0
Via: SIP/2.0/UDP 195.37.77.100:5060;rport
Max-Forwards: 10
From: "utilizador1" <sip:utilizador1@dominio1.com>;tag=76ff7a07-c091-4192-84a0-d56e91fe104f
To: <sip:utilizador2@dominio2.com>
Call-ID: d18015e0-bf17-4afa-8141-d913a793d96@213.20.128.35
CSeq: 2 INVITE
Contact: <sip:213.20.128.35:9315>
User-Agent: Windows RTC/1.0
Proxy-Authorization: Digest username="utilizador1", realm="dominio1.com",algorithm="MD5",
     uri="utilizador2@dominio2.com", nonce=3cef753800000001771328f5ae1b8b7f0d742da1feb5753c",
     response="53fe98db10e1074b03b3e06438bda70f"
```

```
Content-Type=application/sdp
Content-Length: 451

v=0
o=jku2 0 0 IN IP4 213.20.128.35
s=session
c=IN IP4 213.20.128.35
b=CT:1000
t=0 0
m=audio 57442 RTP/AVP 97 111 112 6 0 8 4 5 3 101
a=rtpmap:97 red/8000
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap: 112 G7221/16000
a=fmtp: 112 bitrate=24000

a=rtpmap: 6 DVI4/16000
a=rtpmap: 0 PCMU/8000
a=rtpmap: 4 G723/8000
a=rtpmap: 3 GSM/8000
a=rtpmap: 101 telephone-event/8000
a=fmtp: 101 0-16
```

Na primeira linha do cabeçalho é indicado o tipo de mensagem (‘INVITE’), que é utilizado para iniciar uma sessão. O URI `sip:utilizador@dominio1.com` presente também nesta linha é designado *“Request URI”*e indica qual o próximo salto (*“hop”*) da mensagem (neste caso será o anfitrião ‘dominio1.com’).

Um pedido SIP pode conter um ou mais campos ‘Via’ no cabeçalho. Estes campos são utilizados para registarem o caminho percorrido pela mensagem e este registo é posteriormente utilizado para encaminhar as respostas SIP da mesma forma. O ‘INVITE’ do exemplo contém apenas um campo ‘Via’ criado pelo agente do utilizador que o enviou. Através da informação nele contida é possível determinar que o agente do utilizador está a correr no anfitrião ‘195.37.77.100’ e no porto ‘5060’.

Nos campos ‘To’ e ‘From’ está presente a identificação do receptor e do emissor, respectivamente, da mensagem. O campo ‘From’ contém o parâmetro ‘tag’, que irá servir como identificador de diálogo (ver secção 2.3.4).

O campo ‘Call-ID’ é também um identificador de diálogo e a sua função é identificar mensagens pertencentes à mesma chamada, cujo valor deste campo será idêntico. O campo ‘CSec’ é utilizado para ordenar pedidos. Como os pedidos podem ser enviados através de meios de transporte que não assegurem a manutenção da sua ordem de envio, é necessária

uma forma de numerar as mensagens para que o receptor consiga identificar retransmissões e pedidos desordenados.

No campo 'Contact' é inserido o endereço IP e o porto através do qual o agente do emissor aguarda mensagens enviadas pelo receptor.

O cabeçalho da mensagem é separado do corpo por uma linha em branco. O corpo do 'INVITE' contém a descrição dos tipos de média aceites pelo emissor e é codificado em SDP.

Para além do 'INVITE', outros importantes tipos de pedidos SIP são:

- 'ACK' – Esta mensagem declara a recepção da resposta final a um 'INVITE'. O estabelecimento de uma sessão envolve uma troca de três mensagens, devido à natureza assimétrica da comunicação.

- 'BYE' – As mensagens 'BYE' são utilizadas para terminar sessões. Quando uma das partes deseja encerrar a sessão, envia um 'BYE' às restantes partes.

- 'CANCEL' – É utilizado para cancelar uma sessão ainda não estabelecida, isto é, quando o receptor ainda não respondeu com uma resposta final mas o emissor deseja abortar a sessão.

- 'REGISTER' – O pedido 'REGISTER' tem como função dar a conhecer ao *"registrar"* a localização do utilizador em termos do seu endereço IP e porto.

Os pedidos listados geralmente não têm corpo pois este é desnecessário na maioria dos casos mas é possível adicionar-lhes um.

Apesar de existirem outros tipos de pedidos SIP, a sua descrição e discussão não se encontra dentro do âmbito desta dissertação.

2.3.3.2 Respostas SIP

Quando um agente de utilizador ou um procurador recebe um pedido, este envia uma resposta. Cada pedido deve ser respondido, excepto os pedidos do tipo 'ACK'.

Por exemplo, uma resposta SIP tem a seguinte forma:

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 192.168.1.30:5060;received=66.87.48.68
From: sip:utilizador@dominio.com;tag=794fe65c16edfdf45da4fc39a5d2867c.b713
Call-ID: 2443936363@192.168.1.30
CSeq: 63629 INVITE
Contact: sip:utilizador@66.87.48.68:5060;transport=udp;q=0.00;expires=120
Server: SIP Express router (0.8.11pre21xrc (i386/linux))
Content-Length: 0
```

```
Warning: 392  195.37.77.101.:5060 "Noisy feedback tells:
     pid=5110 req_src_ip=66.87.48..68 req_src_port=5060 in_uri=sip:dominio.com
     out_uri=sip:dominio.com via_cnt==1"
```

É possível verificar que as respostas SIP têm uma estrutura semelhante à dos pedidos, exceptuando a primeira linha. Nas respostas, esta linha contém a versão do protocolo utilizado (SIP/2.0), o código de resposta (200) e texto adicional explicativo.

Os códigos de resposta são formatados como números inteiros de 100 a 699 e indicam o tipo da resposta. Estes tipos podem ser agrupados em 6 classes:

- 1XX – São respostas provisórias que indicam ao seu receptor que o pedido efectuado foi recebido mas o estado do seu processamento ainda é desconhecido. Estas respostas são enviadas quando o pedido não é processado de imediato. O emissor deixa de retransmitir o pedido quando recebe uma mensagem provisória. Alguns exemplos do uso deste tipo de mensagens são o envio de uma resposta com o código '100' por parte de um procurador SIP, quando este recebe um 'INVITE' para reenviar ou quando um agente de utilizador envia uma resposta com o código '180', indicando que o telefone do receptor está a tocar.

- 2XX – São respostas finais positivas. Uma resposta final é a última resposta que um emissor de um pedido recebe em relação ao mesmo. Assim sendo, estas respostas contêm o resultado final do processamento associado ao pedido efectuado. Respostas cujo código se situa no intervalo 200-299 são respostas positivas indicativas de que o pedido foi processado com sucesso. Por exemplo, o código '200 OK' é utilizado como resposta quando o utilizador aceita o convite para uma sessão (como resposta final a um 'INVITE'). Uma UAC pode receber várias respostas '200' relacionadas com o mesmo 'INVITE' pois um procurador de distribuição (*"forking proxy"*) pode distribuir o pedido por vários UAS, que podem aceitá-lo. Neste caso, cada resposta é identificada pelo parâmetro 'tag' no campo 'To' do seu cabeçalho.

- 3XX – Estas respostas são utilizadas para redireccionar o seu receptor. Contêm informação acerca da nova localização do utilizador que se pretende contactar ou acerca de um serviço alternativo para o contactar. Respostas de redireccionamento são normalmente enviadas por procuradores quando recebem um pedido que não podem satisfazer por algum motivo. Podem então fornecer a localização alternativa para o utilizador contactar o destinatário da chamada. Esta localização pode ser de um

procurador alternativo ou a localização actual do destinatário. O utilizador então reenvia o pedido directamente para a nova localização fornecida. As respostas do tipo 3XX são finais.

- 4XX – São respostas finais negativas. Uma resposta desta classe significa que o pedido não pôde ser processado devido a um problema no lado do utilizador da comunicação. A razão pode ser devido a erros de sintaxe na sua formulação ou não pode ser satisfeito pelo servidor que o recebe.

- 5XX – É outra classe de respostas finais negativas. São enviadas quando o pedido não pode ser processado mas a razão está relacionada com o lado do servidor da comunicação. O pedido é aparentemente válido mas o servidor falhou ao tentar processá-lo. Quando recebe uma resposta destas, o utilizador normalmente volta a tentar efectuar o pedido mais tarde.

- 6XX – Esta classe de resposta indica que o pedido não pode ser processado por nenhum servidor. É normalmente utilizada quando um servidor tem algum tipo de informação específica acerca de determinado utilizador. Por exemplo, quando um utilizador não deseja participar na sessão, o seu UAS envia uma resposta '603 Decline'.

Para além da classe da resposta, a primeira linha contém ainda uma frase descritiva. O seu propósito é indicar ao utilizador, através de uma frase de texto, o resultado do processamento do seu pedido. O código numérico da classe, apesar de facilmente interpretado por máquinas, torna-se complicado para o utilizador humano memorizar o significado das 599 opções possíveis.

A associação entre um pedido e a respectiva resposta é feita através do campo 'CSeq' do cabeçalho da resposta. Para além do número de sequência, este campo contém ainda o tipo de pedido.

2.3.4 Transacções e Diálogos SIP

Embora as mensagens SIP sejam enviadas independentemente através da rede, estas são normalmente agrupadas por agentes de utilizador e procuradores em transacções. Por esta razão, por vezes o protocolo SIP é descrito como um protocolo transaccional.

Uma transacção é uma sequência de mensagens SIP trocadas entre elementos SIP numa rede e consiste num pedido e nas respectivas respostas. Inclui as respostas provisórias assim como as respostas finais. No entanto, as mensagens 'ACK' constituem uma excepção. Estas

só são consideradas de uma transacção iniciada com um 'INVITE' quando a resposta final não é positiva (não é da classe 2XX). Isto deve-se à importância da entrega das respostas '200 OK' pois estas determinam o estabelecimento da sessão, podendo ser enviadas por múltiplas entidades quando um pedido é distribuído a vários receptores. Sendo imperativo que o utilizador que enviou o 'INVITE' receba todas as respostas '200 OK', estes são retransmitidos pelos agentes receptores do 'INVITE' até receberem a mensagem 'ACK'.

Uma entidade SIP que tem noção das transacções efectuadas é denominada de estadual. Geralmente, estas entidades criam um estado associado a uma transacção e armazenam-no em memória ao longo da duração da transacção. Quando recebe uma resposta ou um pedido, uma entidade estadual procura relacionar a mensagem com uma transacção existente. Para tal, utiliza o identificador de transacção contido no cabeçalho da mensagem e compara-o com os identificadores de transacções existentes. Caso seja estabelecida uma correspondência, a entidade actualiza o estado utilizando o conteúdo da mensagem.

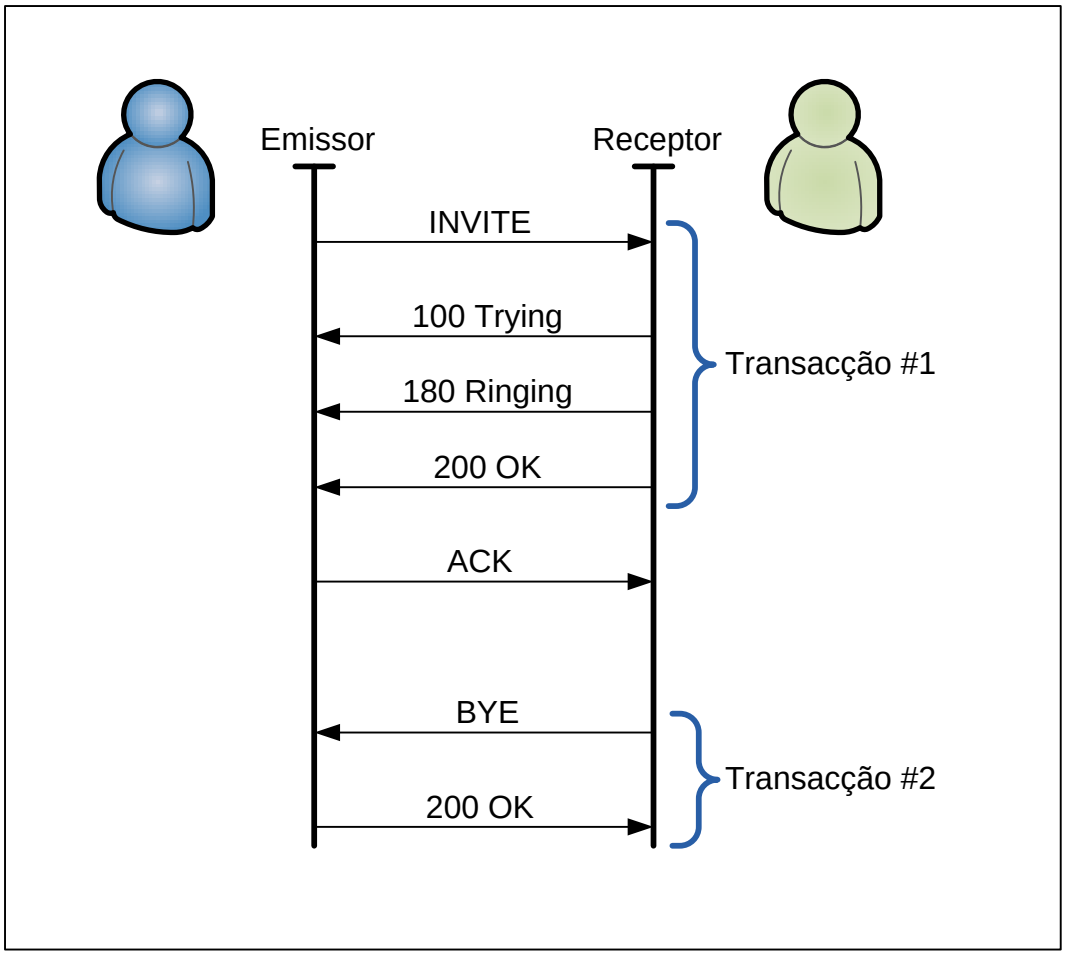

Figura 2.8 - Transacções SIP

As transacções podem ser agrupadas em diálogos. Um diálogo representa uma relação par-a-par SIP entre dois agentes de utilizador. Por exemplo, uma transacção que inclui um 'INVITE' inicia uma sessão, enquanto outra, contendo um 'BYE', encerra essa mesma sessão. Juntas, estas duas transacções formam um diálogo. Um diálogo SIP é persistente e facilita a sequenciação e encaminhamento das mensagens entre agentes.

Os diálogos são identificados através dos campos 'Call-ID', 'To' e 'From' no cabeçalho das mensagens. Aquelas que têm o conteúdo destes campos em comum pertencem ao mesmo diálogo. Através do campo 'CSeq' é estabelecida a ordem de sequência das mensagens dentro de cada diálogo, sendo o seu valor utilizado também para identificar uma transacção dentro de um diálogo. Apenas uma transacção pode estar activa em cada direcção da comunicação. Pode-se então afirmar que um diálogo é uma sequência de transacções.

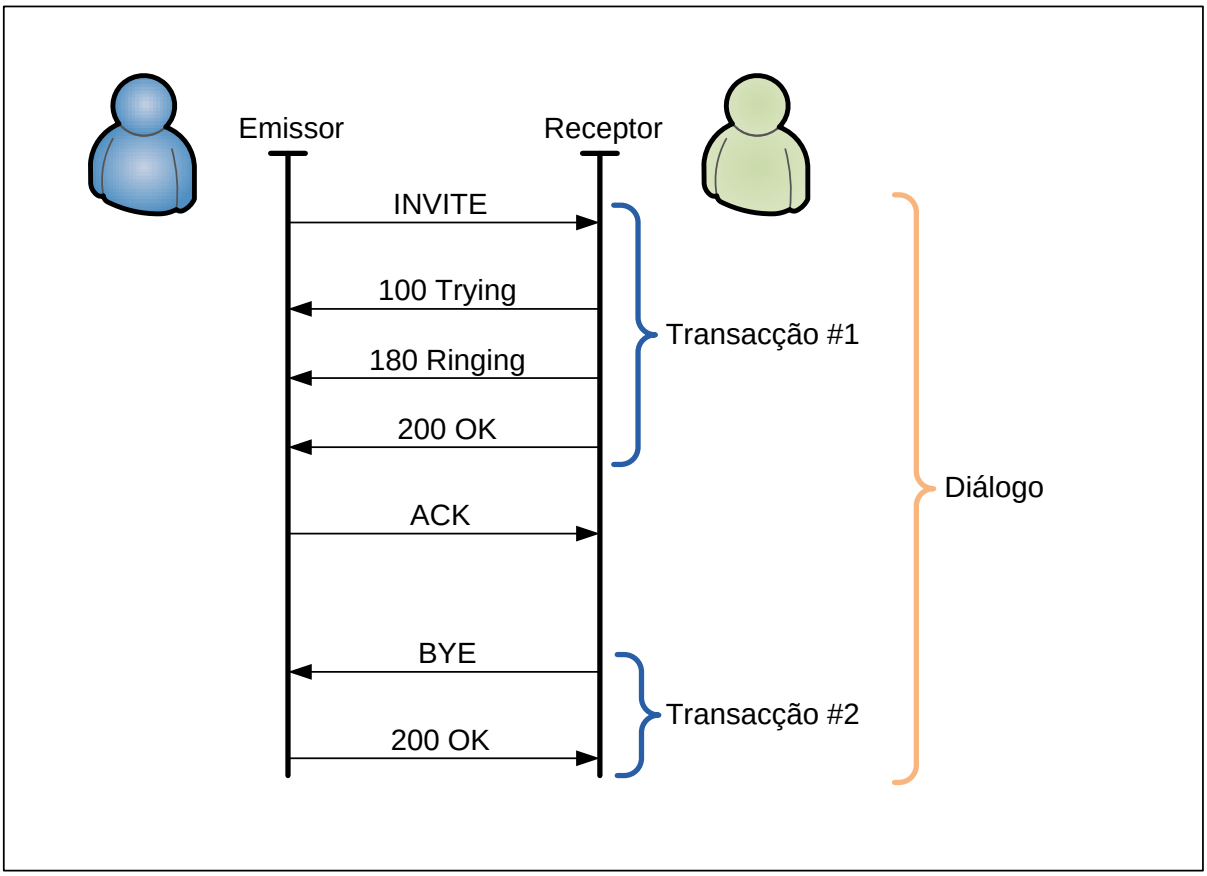

Figura 2.9 - Diálogo SIP

Na Figura 2.9 pode ver-se uma extensão da Figura 2.8. É apresentado um diálogo SIP entre dois utilizadores, composto por duas transacções. A 'Transacção #1' inicia a sessão (assim como o diálogo) e a 'Transacção #2' encerra tanto a sessão como o diálogo.

Nem todas as mensagens podem ser agrupadas em diálogos. Isto permite estabelecer relações explícitas entre mensagens e enviar mensagens sem relação fora de um diálogo. Por exemplo, a mensagem 'INVITE' estabelece um diálogo que será encerrado com um 'BYE'. Ambas formam um diálogo. Mas se um dos utilizadores decidir enviar uma mensagem do tipo 'MESSAGE', este pedido não irá estabelecer nenhum diálogo.

2.3.5 Cenários SIP

Nesta secção apresentam-se alguns cenários típicos de tráfego SIP.

2.3.5.1 Registo

É necessário que os utilizadores se registem junto de um *"registrar"* para se poderem ligar a outros utilizadores. A operação de registo é composta por uma mensagem 'REGISTER' seguida por uma resposta '200 OK' enviada pelo *"registrar"*, caso o registo tenha sido efectuado com sucesso. Normalmente é necessário algum de tipo de autorização para estabelecer um registo, logo uma resposta '407' pode ser enviada se o utilizador não apresentar credenciais válidas.

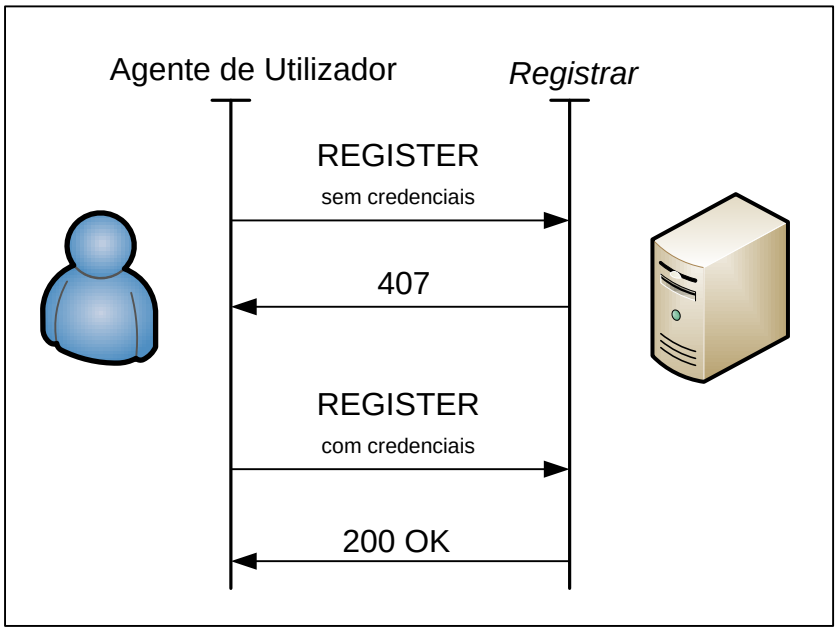

Figura 2.10 - Registo SIP

2.3.5.2 Início de Sessão

O início de uma sessão consiste no envio de um pedido 'INVITE' da parte de um agente de utilizador. Este pedido é normalmente enviado para um procurador, que responde com um '100 Trying', indicando que está a tentar estabelecer uma ligação para impedir retransmissões do 'INVITE' e reenvia o pedido para o destinatário.

Todas as respostas provisórias por parte do receptor do 'INVITE' são enviadas ao seu emissor. Por exemplo, quando o agente de utilizador do receptor recebe o 'INVITE' e o telefone começa a tocar, envia uma resposta provisória '180 Ringing'. Quando o utilizador atende a chamada, o seu agente envia uma resposta final '200 OK'. Esta é retransmitida até que o emissor envie um 'ACK' e este seja recebido pelo receptor da chamada, estabelecendo uma sessão SIP.

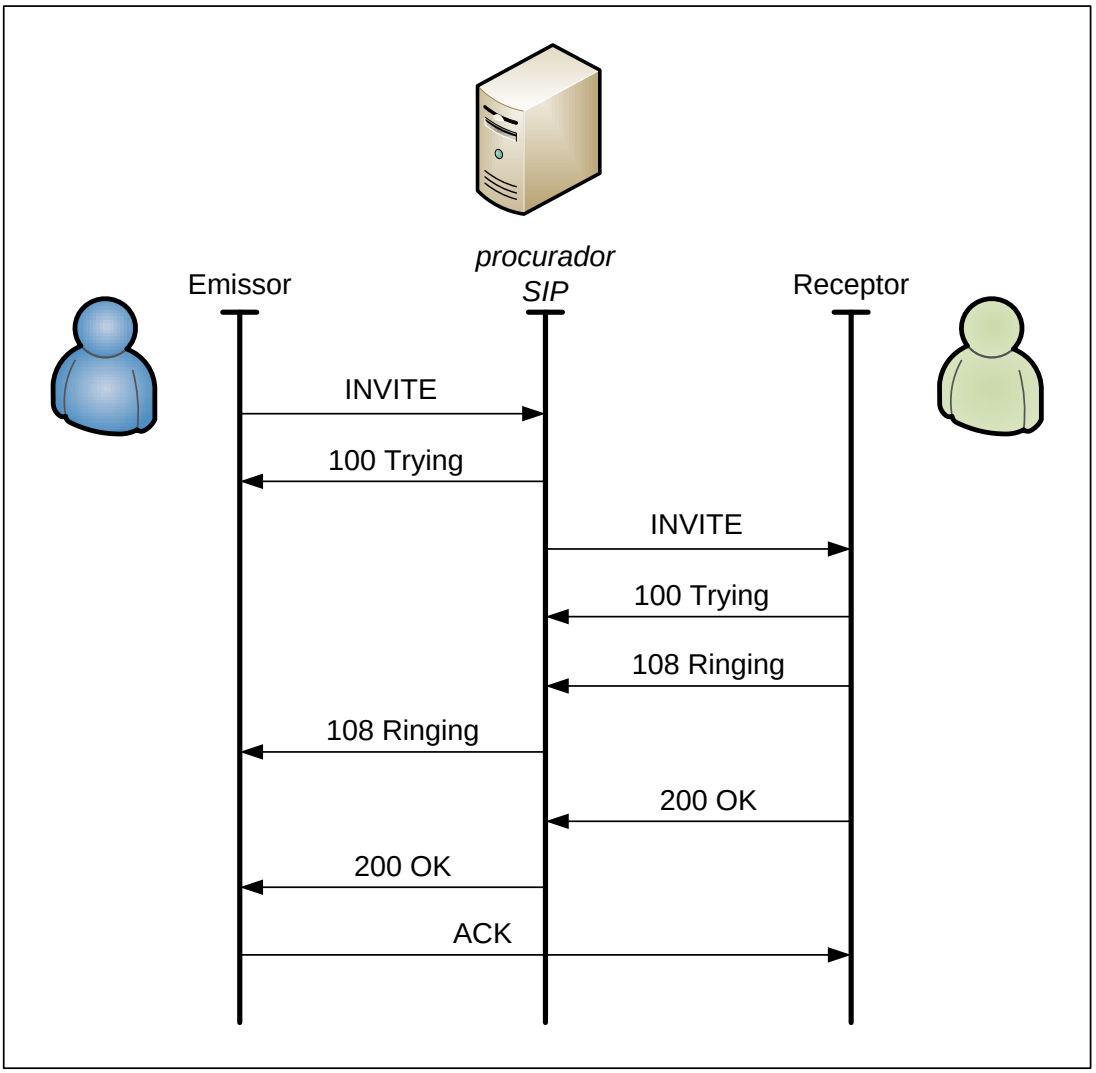

Figura 2.11 - Fluxo de mensagens de um início de sessão

2.3.5.3 Encerramento de sessão

O encerramento de uma sessão é efectuado através do envio de um ’BYE’ dentro de um diálogo iniciado por um ‘INVITE’. As mensagens ‘BYE’ são enviadas directamente de um agente de utilizador para outro, excepto se um procurador no caminho da ligação entre os utilizadores indique que permanece no caminho, utilizando encaminhamento com registo.

O ‘BYE’ é enviado quando uma das partes da sessão deseja encerrá-la. A outra parte responde com um ‘200 OK’ e a sessão é encerrada.

2.4 VoIP

O VoIP (*Voice over*IP) é o protocolo utilizado para a transmissão de voz através da Internet (conhecida por telefonia através da Internet) ou através de qualquer outra rede baseada em IP. Utiliza protocolos de controlo de sessão, como o SIP para o estabelecimento e terminação de chamadas e *“codecs”* (normas de codificação) de áudio para a codificação de voz, permitindo a sua transmissão em redes IP sob a forma de áudio digital através um feixe de dados áudio. A utilização de *“codecs”* varia conforme as implementações de VoIP, sendo disponibilizadas várias opções. A escolha do *“codec”* a utilizar é feita de acordo com os recursos de rede disponíveis (largura de banda) e reflecte-se na qualidade do áudio transmitido, que pode ir desde o áudio comprimido até ao áudio de alta-fidelidade em estéreo. Na tabela seguinte, apresentam-se alguns tipos de *“codecs”* normalmente utilizados.

Norma de codificação (*“codec”*)	**Ritmo binário médio (kbit/s)**	**Atraso de codificação (ms)**
G.711	64	
G.726	16-40	2
G.729 A / G.729 AB	8	10
G.723.1	5.3-6.4	30
G.729E	11.8	10

Tabela 2 – Normas de codificação de voz (*"codecs"*)

A agregação de transmissão de dados com chamadas telefónicas permite uma considerável redução de custos por parte do utilizador, especialmente se tiver acesso a uma rede cuja capacidade esteja subutilizada e que suporte VoIP sem custos adicionais. As chamadas entre utilizadores VoIP são normalmente grátis enquanto chamadas de VoIP para uma rede telefónica de comutação de circuitos são tarifadas, ficando o custo a cargo do

utilizador de VoIP. Quanto às chamadas de uma rede telefónica para VoIP, existem duas formas possíveis de as realizar: através de marcação interna directa (DID – *"Direct Inward Dialing"*) e através de números de acesso. Ao utilizar DID, em que o emissor é ligado directamente ao receptor através de números de acesso, é necessária a introdução do número de extensão do receptor VoIP.

A utilização de VoIP concede uma grande mobilidade pois permite encaminhamento das chamadas destinadas a um utilizador para o seu telefone VoIP, independentemente do local onde se encontra ligado à rede (possibilitando a recepção de chamadas onde quer que se ligue à Internet). Actualmente, é possível para telefones VoIP integrarem serviços disponíveis para a Internet tais como trocas de mensagens e ficheiros de dados em paralelo com a conversação e conferências áudio e vídeo.

Uma das desvantagens do serviço VoIP é a sua dependência de uma ligação à Internet. A qualidade e fiabilidade de uma ligação telefónica VoIP dependem da qualidade, fiabilidade e velocidade da ligação à Internet utilizada.

CAPÍTULO 3.

TRABALHO RELACIONADO

3.1 Pesquisa e Desenvolvimento de BBs

Neste capítulo pretende-se apresentar uma visão geral do estado actual de alguns dos principais esforços de diversos grupos de pesquisa no desenvolvimento de BBs (*"Bandwidth Brokers"*).

3.1.1 Grupo de Trabalho QBone

Das várias propostas de desenvolvimento de um BB, uma das mais significativas foi a levada a cabo pelo *"Internet2 QBone Bandwidth Broker Advisory Coucil"*. Foi um dos primeiros grupos a definir detalhadamente os requisitos para um BB e foi também dos primeiros a propor um modelo simples de protocolo de sinalização intra-domínio para o seu funcionamento, o SIBBS (*"Simple Interdomain Bandwidth Broker Signaling"*) [QBONE].

O Grupo de Trabalho QBone (*"QBone Bandwidth Broker Work Group"*) nasceu em 1999, resultante de uma evolução do *"Internet2 QBone Bandwidth Broker Advisory Coucil"*. O seu principal objectivo foi preencher as lacunas existentes no protocolo SIBBS, de forma a criar um protocolo abrangente de comunicação entre BBs num cenário inter-domínio.

3.1.2 CANARIE ANA

A CANARIE (*"Canadian Advanced Network and Research for Industry and Education"*) é uma corporação com fins não-lucrativos subsidiada pelo governo canadiano, composta por universidades e outras instituições ligadas à educação e também por companhias e agências governamentais [CANARIE 2008]. A CANARIE é responsável pela manutenção de uma rede de larga escala (WAN – *"Wide Area Network"*), a CA*net (ou CAnet) cujos *"links"* são licenciados a organizações educacionais e de pesquisa canadianas, assim como pelo desenvolvimento e implementação de tecnologias de rede para fins educativos e de pesquisa. A Ca*net é também utilizada para testar protótipos de aplicações num ambiente de rede de larga escala.

Em 1997, a rede da CANARIE encontrava-se na versão CA*net II (actualmente CA*net 4), com uma capacidade no seu núcleo de 155 Mbits/s. Nessa altura, a CANARIE pretendeu implementar um serviço de videoconferência sobre essa rede. O seu desenvolvimento foi atribuído à CANARIE ANA (*"Advanced Networking Applications"*). Durante este processo, surgiu o conceito da provisão de serviços diferenciados na Ca*net, o que levou ao projecto de um BB que fosse capaz de levar a cabo essa tarefa. O desenvolvimento do BB da CANARIE ANA foi feito com a colaboração do Grupo de Tecnologia Informática Avançada (GAIT – *"Group for Advanced Information Technology"*) do Instituto de Tecnologia da Columbia Britânica (BCIT – *"British Columbia Institute of Technology"*) [BCIT 2008].

O funcionamento modular básico do BB da CANARIE ANA é descrito abaixo, podendo ser visualizado na Figura 3.1.

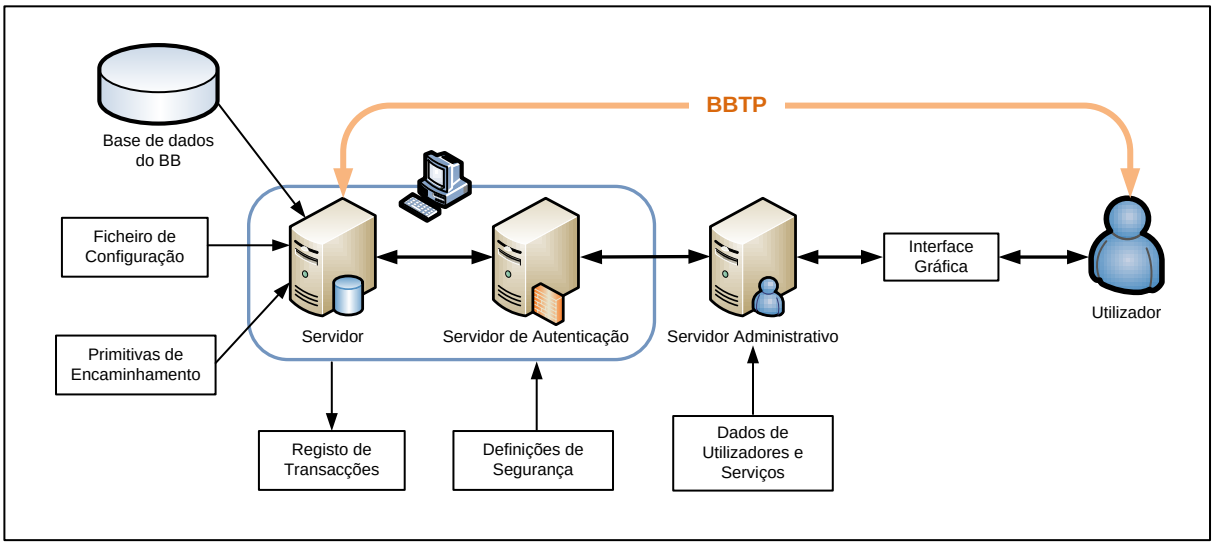

Figura 3.1– Esquema funcional do BB da CANARIE ANA

Toda a informação referente aos SLAs e ao aprovisionamento de largura de banda é armazenada na base de dados do BB. O servidor do BB é um programa de processos múltiplos implementado para operar dentro da sua região de confiança. As suas funcionalidades principais são a validação de dados, a aplicação de políticas, a actualização do registo de transacções e a limpeza de dados obsoletos. O servidor do BB e o servidor de autenticação do BB têm de funcionar na mesma máquina, sendo o último responsável por garantir a segurança necessária para as operações do anterior;

Existe uma entidade, o servidor administrativo (BBAS – *"Bandwidth Broker Administrative Server"*), que guarda informações acerca do cliente e que é responsável pela gestão dos contractos entre os clientes e a entidade que fornece o serviço, assim como do estabelecimento de tarifas aplicáveis a esse serviço.

O servidor do BB (BBS – *"Bandwidth Broker Server"*) é configurado utilizando um ficheiro de configuração contendo informação acerca dos portos a utilizar, onde guardar os ficheiros de registos, quais o serviços oferecidos, as definições das primitivas do encaminhador e outras informações acerca de outros encaminhadores na mesma rede. O ficheiro que contém as definições das primitivas do encaminhador pode ser utilizado para configurar qualquer encaminhador específico. O BBS guarda também o registo de todas as transacções efectuadas num ficheiro de texto.

A interacção entre o cliente e o BB é feita utilizando o protocolo de transferência do BB (BBTP – *"Bandwidth Broker Transfer Protocol"*). Este protocolo é baseado num modelo cliente/servidor em que a interacção é iniciada pelo cliente. Após estabelecer uma ligação com o servidor, o cliente envia um pedido, ao qual o servidor responde adequadamente e encerra a ligação. A comunicação BBTP é feita através de ligações TCP/IP, sendo possível utilizar qualquer outro protocolo de fiabilidade semelhante.

O BBTP define dois tipos de mensagens: mensagens de pedido e mensagens de resposta. Os cabeçalhos das mensagens BBTP podem ser de três tipos: geral, de pedido, de resposta e de entidade. Existem também definidas três classes de códigos de estado: sucesso, erro do cliente e erro do servidor. No entanto, a nível de segurança, a autenticação de clientes não se encontra definida no BBTP.

As funcionalidades implementadas por este projecto foram:

- Definição fixa de classes de serviço;
- Aprovisionamento distribuído estático;
- Mecanismo de definição de encaminhadores;

- Configuração de encaminhadores pelo utilizador;
- Geração de relatórios estáticos;
- Interacção segura com o BB;
- Registo de transacções;
- Armazenamento de dados relativos a SLAs;
- Recepção de RARs;
- Configuração de encaminhadores num único domínio;
- Interface entre o utilizador e o BB através de uma linha de comandos ou utilizado um *"web browser"* com uma ligação segura;
- Comunicação entre BB e os encaminhadores no seu domínio através de uma interface telnet.

3.1.3 Universidade do Kansas

O conceito de BB do grupo de pesquisa da Universidade do Kansas é responsável por todas as operações a nível interno e externo. A nível interno, o BB recebe os pedidos de QoS por parte dos clientes e aprovisiona recursos segundo as políticas do seu domínio. A nível externo, o BB providencia QoS ao tráfego que atravessa a fronteira do seu domínio através da manutenção de SLAs bilaterais. A Figura 3.2 apresenta um modelo básico da sua arquitectura.

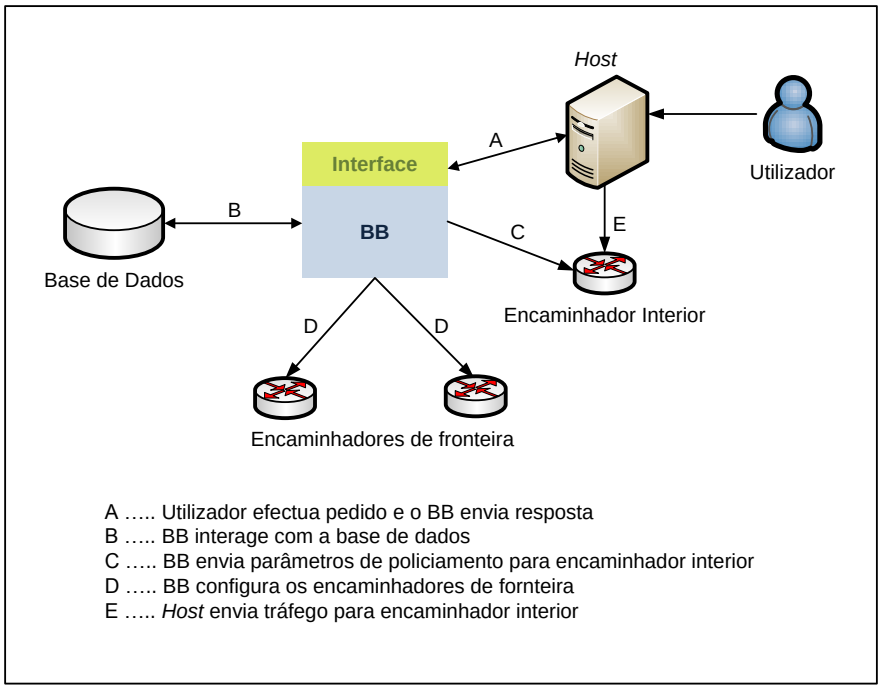

Figura 3.2– Arquitectura do modelo de BB da Universidade do Kansas

A base de dados do BB contém informação acerca dos SLAs, pedidos de aprovisionamento de largura de banda (BAR – *"Bandwidth Allocation Request"*) e do mapeamento entre largura de banda e o respectivo *"DiffServ code point"* (DSCP). Cada BAR contém informação acerca do SLA utilizado, *"leaf router"*, origem, destino, ritmo e tempo. O BB armazena informação acerca das capacidades de largura de banda e de *DiffServ* de cada encaminhador bem como do seu papel no domínio.

O cliente interage directamente com o BB através de um BBTP (ligação A na Figura 3.2), sendo a interface entre os utilizadores autorizados e o BB fornecida por um servidor de rede Apache.

O BB então verifica os dados do pedido de largura de banda na sua base de dados. Caso a sua validade seja estabelecida, o BB envia os parâmetros de políticas e de marcação de tráfego ao *"leaf router"* mencionado no BAR. O BB configura também o encaminhador de saída enviando-lhe os parâmetros necessários. O anfitrião passa então a enviar o tráfego através do *"leaf router"*.

A implementação do BB é feita segundo um modelo cliente/servidor, sendo o BB o servidor que, no arranque, estabelece uma ligação com os encaminhadores do domínio e efectua a sua configuração básica. Tanto os anfitriões do domínio como o administrador da rede podem efectuar pedidos como cliente. O servidor do BB mantém uma base de dados *MySQL*. Os SLAs são adicionados estaticamente pelo administrador da rede.

Os BARs são gerados pelo anfitrião *DiffServ*. Ao receber um pedido, o BB verifica o RAR junto dos SLAs armazenados na sua base de dados. Se o pedido é aceite, o BB envia uma mensagem de sucesso ao cliente após reconfigurar o encaminhador de saída e do DSCP ter sido enviado ao anfitrião para este o inserir nos pacotes do fluxo. Para além do controlo de admissão, o BB também efectua o controlo da validade. Quando um fluxo expira, o seu aprovisionamento de recursos é descartado. Quando um SLA expira, todos os BARs que lhe estão associados são apagados. Existe um *"daemon"* DS a funcionar em cada anfitrião que faz uma estimativa da largura de banda necessária utilizando a operação 'setsocket' no núcleo. O mesmo *"daemon"* é também responsável por gerar pedidos ao BB por parte do anfitrião. O BB mantém uma base de dados de todos os encaminhadores no seu domínio. O BB necessita de estar ligado aos encaminhadores da periferia para efectuar a sua reconfiguração, sendo os encaminhadores centrais configurados estaticamente.

3.1.4 Merit

O grupo de pesquisa da Merit propôs um modelo de BB multi-domínio, no qual introduz a implementação do suporte para linhas virtuais de aluguer (VLL – *"Virtual Leased Line"*), concentrando os seus esforços de investigação no papel desempenhado pelo BB na autorização e estabelecimento das mesmas. A forma como o BB desempenha esta tarefa designa-se por EF PHB num cenário multi-domínio. Neste modelo existe o suporte a dois tipos de serviços VLL:

- VLL de logo prazo, em que a ligação é estabelecida durante largos períodos de tempo e não necessita de nenhum tipo de sinalização, sendo a única responsabilidade do BB gerir o encaminhamento e o fluxo de dados;
- VLL de curto prazo, em que a ligação é estabelecida durante períodos curtos e que necessita de sinalização explicita para o seu estabelecimento e término.

Se uma VLL se estende a outros domínios, cabe ao BB contactar esses domínios e negociar os parâmetros de QoS para a VLL. Podem ser estabelecidos dois tipos de SLS:

- SLS comprometidos, em que existe um compromisso explícito em suportar determinada largura de banda;
- SLS aberto, em que não existe um destino definido e como tal não é possível que os requisitos de largura de banda ou fiabilidade sejam cumpridos.

O BB mantém um registo de todas as políticas e da topologia da rede. Dentro do domínio deve ser capaz de estabelecer encaminhamento, assim como proceder a decisões de controlo de admissão. Ao nível da interacção com outros domínios, o BB responde de acordo com RARs recebidos de BBs de outros domínios. Neste caso, não existe requisito de administração de encaminhadores por parte do BB.

Os esforços deste grupo de pesquisa foram posteriormente incluídos no projecto do grupo QBone, apresentado na secção 3.1.1.

3.1.5 Arquitectura Globus para Reserva e Aprovisionamento (GARA)

O modelo GARA (*"Globus Architecture for Reservation and Allocation"*) é uma arquitectura global para a reserva e aprovisionamento de recursos. Os principais problemas na garantia de QoS entre o destino e a origem de um fluxo residem na descoberta dinâmica de recursos disponíveis e na reserva imediata ou antecipada desses mesmos recursos quando estes são administrados separadamente. O protótipo implementado pelo projecto GARA procura encontrar a solução para estes problemas.

O modelo GARA é uma extensão do conceito da arquitectura de gestão de recursos do Globus (um *"middleware"* para *"grids"*) e providencia a gestão para recursos administrados separadamente. A arquitectura do modelo GARA consiste em três componentes:

- Serviço de informação – define a hierarquia do espaço de nomes a utilizar e os métodos de acesso aos recursos através do uso de LDAP;
- Agentes de co-aprovisionamento – recebem os pedidos de aplicações que solicitam a reserva de recursos, calculam os requisitos necessários e direccionam o pedido para o gestor de recursos local adequado ou gestor de aprovisionamento de recursos Globus (GRAM – *"Globus Resource Allocation Manager"*);
- Gestores de recursos locais – recebem e autenticam os pedidos e, caso o pedido seja autenticado com sucesso, indicam ao gestor de tarefas local para aprovisionar os recursos e enviam o apontador para a tarefa à aplicação que fez o pedido.

O GARA possui ainda uma funcionalidade que permite a reserva de recursos heterogéneos. Isto é conseguido através da introdução do conceito de recurso como um objecto genérico, englobando fluxos de rede, blocos de memória e de disco e até processos. O aprovisionamento imediato de recursos é feito em simultâneo com a reserva.

O suporte à reserva antecipada de recursos é obtido fazendo a reserva e o aprovisionamento separadamente. É retornada uma resposta indicando que a reserva foi feita

mas o aprovisionamento só é feito na altura indicada no pedido. A entidade responsável pelas reservas de recursos é o agente de co-reserva. O seu funcionamento é semelhante ao do agente de co-aprovisionamento só que após calcular os requerimentos de recursos, efectua a sua reserva e não o seu aprovisionamento.

Embora não exista nenhuma entidade equivalente a um BB no protótipo GARA, o seu funcionamento como um todo fornece algumas das funcionalidades básicas de um BB. O modelo funcional do GARA é apresentado na Figura 3.3.

O GARA tem uma API (*"Application Programing Interface"*) que permite o suporte de reservas entre a origem e o destino. Os gestores de recursos funcionam como BBs dentro de um domínio. São responsáveis pelo controlo de admissão e pela modelação e marcação de tráfego dentro do seu domínio. A GARA API é utilizada para contactar os gestores de recursos para efectuar as reservas. Esta interface permite também ao utilizador contactar o gestor de recursos para especificar os requerimentos de reserva. O controlo do acesso aos gestores de recursos é feito através de uma autenticação de chave pública. No caso de reservas inter-domínio ponto-a-ponto, a aplicação contacta a API ponto-a-ponto e comunica-lhe o seu pedido. Quando o pedido é aceite, todos os gestores de recursos reconfiguram os encaminhadores a que estão ligados para suportar o fluxo. O controlo dos encaminhadores por parte do gestor de recursos é feito através de um *"script"* TCL (*"Tool Command Language"*), contendo o *"login"*, a palavra-chave e a interface para configurar os encaminhadores.

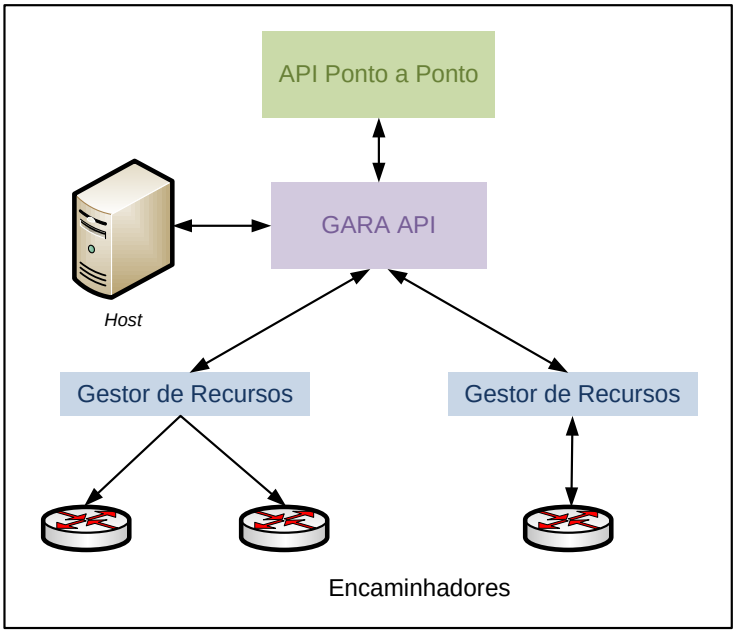

Figura 3.3 – Arquitectura do modelo GARA

3.1.6 Tecnologia de Contas e Tarifários Para a Internet (CATI)

O principal objectivo do CATI (*"Charging and Accounting Technology for the Internet"*) foi desenvolver um mecanismo de contas de utilizador e de tarifários de serviço baseado nos protocolos IP actualmente disponíveis.

A interligação segura entre redes privadas foi um dos factores que levou à emergência das redes privadas virtuais (VPN – *"Virtual Private Network"*). O suporte de QoS em VPNs pode ser obtido através do uso de corretores de serviços. Este tipo de mecanismos disponibiliza os seus serviços de acordo com termos específicos, tendo a capacidade de negociar com o cliente o custo dos serviços e de proceder à sua execução caso exista um acordo entre ambas as partes. Os corretores são desenhados numa forma hierárquica escalável, como pode ser visto na Figura 3.4.

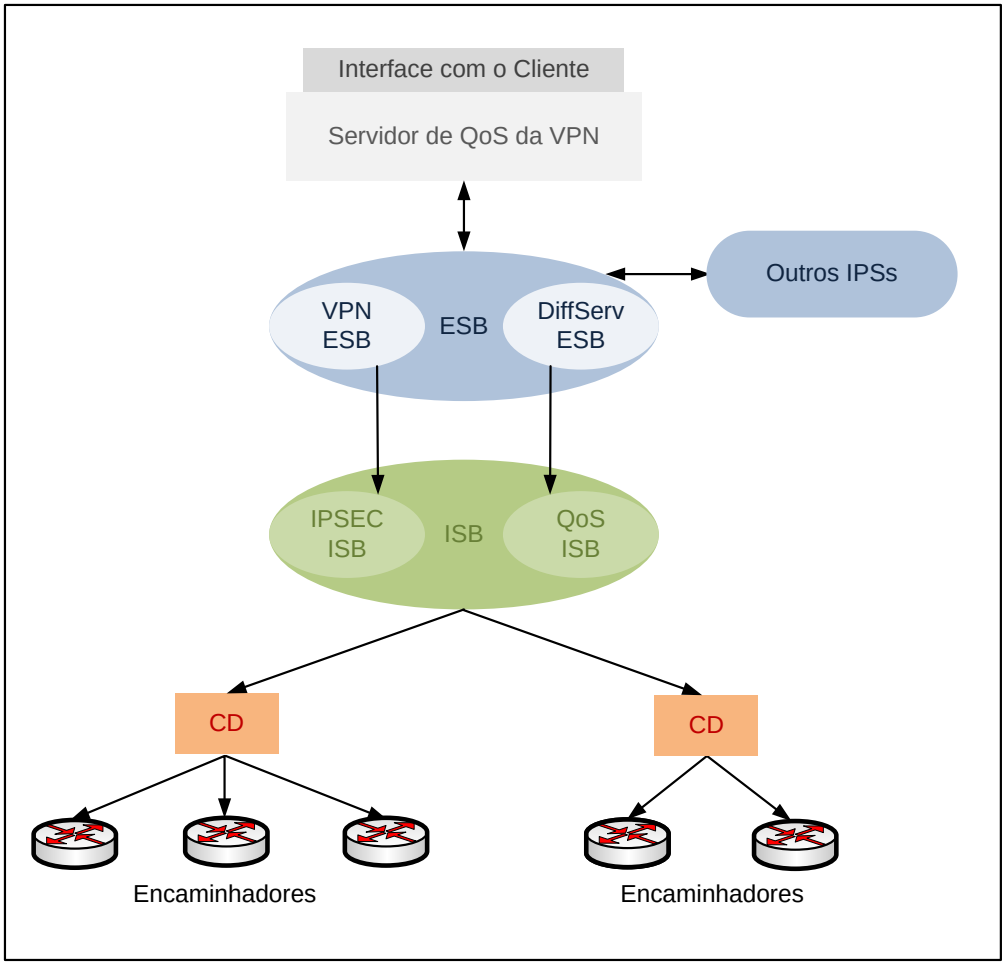

Figura 3.4 – Interacção entre corretores de serviços

3.1.7 Comparação

Na tabela seguinte é apresentado um sumário de todas as implementações discutidas nesta secção. É feita uma comparação entre as implementações baseada nos componentes básicos de uma arquitectura de um BB apresentados na secção 2.

Arquitectura	QBone	CANARIE ANA	Kansas	CATI	GARA
Tabela de Encaminhamento	Sim	Não	Não	Não	Não
Repositório de Dados	Armazena SLAs, dados de utilizadores, parâmetros de configuração de encaminhadores, politicas	Armazena SLAs, informação de reserva de largura de banda	Armazena SLAs; BARs, relações entre DSCPs e largura de banda a reservar, topologia da rede, capacidades *DiffServ* dos encaminhadores	Armazena SLAs, recursos de ligações, tarifas de serviços	Parâmetros de configuração de encaminhadores
Interface com o utilizador	Disponível mas não é especificada	Linha de comandos Unix	Ligação TCP	GUI	GARA API
Interface com o administrador de rede	Disponível mas não é especificada	Linha de comandos Unix	GUI	GUI	GARA API
Configuração de encaminhadores	Ligação TCP	Interface Telnet	Interface Telnet para encaminhadores Cisco, ligação TCP para encaminhadores Linux	*"Daemon"* de configuração, utilizando *"scripts"*	Gestor de recursos
Controlo de Admissão	Não é especificado	Não é especificado	Não é especificado	Corretor de serviço interior	Gestor de recursos
Segurança	Não é especificado	Não é especificado	Não é especificado	MD5 e SHA-1	Autenticação por chave pública no acesso ao gestor de recursos
Módulo Funcional Principal	Não é especificado	Não é especificado	Não é especificado	Não é especificado	API entre terminais

Tabela 3 – Comparação de implementações diferentes de BBs

As implementações discutidas são protótipos experimentais. Os modelos discutidos anteriormente apresentam especificações de funcionamento intra-domínio mas quanto ao funcionamento inter-domínio, a sua implementação não chegou a ser finalizada. Nesse

aspecto, o projecto da QBone apresenta-se como o mais completo, graças à sua especificação do protocolo SIBBS para a comunicação de BBs inter-domínio. Mas tal como o seu nome indica (*"Simple Inter-Domain Bandwidth Broker Specification"*), este é um protocolo simples, do tipo pedido/resposta entre BBs, com a troca de informação essencial para a requisição de determinado serviço.

Em termos de fiabilidade, todos os projectos excepto o GARA apresentam uma falha comum: se a entidade que faz a gestão dos recursos do domínio (gestor de recursos, no caso do GARA, BB para os restantes) falha, perdem-se todos os serviços e rotas do seu domínio. Seria necessário implementar uma solução de recurso em caso de falha ou uma solução distribuída, que minimizasse os efeitos dessa falha. No caso do GARA, como a gestão de cada recurso é feita de forma independente por diferentes gestores, isto torna-o mais robusto.

Quanto à escalabilidade, esta é limitada pelo tipo e quantidade de informação armazenada na sua base de dados. Nesse aspecto, o GARA é aquele que guarda menos informação pois a descoberta de recursos e a sua reserva é feita de forma dinâmica.

Em termos de segurança, os projectos GARA e CATI são os mais sólidos, sendo utilizada encriptação na comunicação com o utilizador e entre as entidades que compõem a sua estrutura. Nos restantes projectos, esta necessidade é discutida mas não implementada. No entanto, uma a vulnerabilidade a ataques do tipo Negação de Serviço é comum a todos.

Resumindo, apesar da sua importância académica como modelos experimentais, nenhuma destas implementações apresenta a funcionalidade completa de um procurador de banda. A principal barreira para alcançar esse objectivo tem sido o desenvolvimento e implementação de um protocolo de comunicação inter-domínio entre BBs que abranja todas as especificações necessárias.

3.2 Escalabilidade de Corretores de Largura de Banda

Actualmente regista-se um crescimento rápido do uso de aplicações como VoIP e de aplicações de entrega de conteúdos em tempo real, que necessitam de dinamismo na gestão e controlo de QoS. Ao lidar com um grande volume de fluxos de dados, a capacidade de um BB pode ser insuficiente. Um BB mal dimensionado pode tornar-se num ponto de estrangulamento no que diz respeito à reserva eficiente de recursos de uma rede, mesmo em cenários em que a capacidade desta não se encontra totalmente utilizada. Uma solução

proposta para resolver este problema de escalabilidade consiste na utilização de uma arquitectura de múltiplos BBs [Zhang 2001]. Nesta solução, existe um BB central (cBB – *"central BB"*) e um determinado número de BBs de fronteira (eBB – *"edge BB"*) em cada domínio. Os estados de QoS são representados em dois níveis: ligação (*"link"*) e caminho (*"path"*). As ligações são estabelecidas entre as entidades que compõem a rede, enquanto os caminhos são compostos pelas ligações que é necessário percorrer para ir de um ponto a outro na rede. A informação referente ao estado de QoS de cada ligação do domínio é armazenada numa base de dados. Através desta informação é obtido o estado de QoS de cada caminho possível na rede, que é armazenado noutra base de dados. O modelo desta arquitectura utiliza uma abordagem baseada em quotas por caminho. Nesta abordagem, a largura de banda é reservada em quotas pelos diferentes caminhos requisitados, de forma a limitar os acessos de pedidos de fluxos à base de dados de ligações [Zhang 2001]. O cBB é a entidade responsável pela manutenção da base de dados de informação de estado de QoS das ligações e pela atribuição de quotas de tráfego aos eBBs. Cada eBB é responsável por um subconjunto exclusivo da base de dados de estado de QoS dos caminhos, realizando o controlo de admissão desses mesmos caminhos. Quando um fluxo de dados chega a um encaminhador de fronteira, o RAR correspondente é enviado ao eBB responsável por esse encaminhador. Se o eBB tem recursos suficientes disponíveis, o fluxo é admitido. Caso isto não se verifique, então o RAR é enviado para o cBB, que procede ao controlo de admissão através da verificação da largura de banda disponível em todas as ligações ao longo do caminho na base de dados de QoS de ligação. Esta arquitectura, embora apresente uma solução para o problema de escalabilidade da rede, não é isenta de problemas. Para reduzir o processamento excessivo por parte do BB, a capacidade deste efectuar uma gestão de recursos optimizada e eficiente é degradada com a introdução um mecanismo de reserva de largura de banda baseado em quotas por caminho. O aumento do tamanho da quota pode reduzir de forma significativa a eficiência da reserva de largura de banda a cada fluxo. Cada eBB tem um número de caminhos para os quais efectua controlo de admissão. No entanto, a distribuição de largura de banda por ligações comuns a diferentes caminhos é feita de forma ambígua. Este facto introduz também um desperdício de largura de banda ao longo destes caminhos. Se a base de dados de ligação for acedida frequentemente, então o processamento excessivo resultante não justifica a selecção desta abordagem baseada em cotas.

3.3 Gestão Activa de Recursos

A utilização de um BB na arquitectura *DiffServ* permite ao utilizador reservar largura de banda de forma a garantir as suas necessidades de QoS. Existe, no entanto, a possibilidade do utilizador sobrestimar as suas necessidades, resultando numa gestão ineficaz de recursos. Uma solução proposta para este problema é a utilização de uma gestão activa de recursos (ARM – *Active Resource Management*) [Anan 2001]. Com este tipo de gestão, cada fluxo é monitorizado pelo BB e a largura de banda utilizada é avaliada em relação à que foi reservada. A largura de banda excedente (que está reservada ao fluxo mas não está a ser utilizada) é reaproveitada para outros fluxos. O tráfego é marcado utilizando o DSCP correspondente ao SLS para esse fluxo. Numa arquitectura que utilize ARM, o BB monitoriza a taxa de tráfego de cada fluxo e quando a largura de banda ocupada é menor que a reservada, o BB disponibiliza esse excedente para outros fluxos, de forma a evitar o desperdício de recursos na rede. Se a taxa de tráfego aumentar e o utilizador necessitar de mais largura de banda, esta é-lhe de novo reservada com vista a satisfazer o SLA estabelecido. Não é feita uma reserva de largura de banda para serviços BE. Estes obtêm a largura de banda necessária através do excedente de outros fluxos.

3.4 Uso de BBs em Outras Arquitecturas de Rede

Em [RFC2638 1999] é introduzido o conceito da implementação de BB num domínio *DiffServ* para a partilha controlada de largura de banda na Internet. Com os avanços na pesquisa deste tema, tornou-se aparente que um BB consegue preencher outros requisitos quando utilizado em outras arquitecturas de rede. Em seguida, apresentam-se algumas perspectivas acerca da incorporação de BBs em diferentes arquitecturas de rede com vista a optimizar a gestão de recursos.

3.4.1 MPLS

Existem determinadas funcionalidades do protocolo de encaminhamento por rótulos (MPLS – *"MultiProtocol Label Switching"*) destinadas à realização de QoS que podem ser complementadas com o uso de um BB.

3.4.1.1 RATES

Numa arquitectura de rede que utilize MPLS existe um servidor de encaminhamento e gestão de tráfego (RATES – *"Routing and Traffic Engineering Server"*) proposto em [Aukia 2000]. Esta entidade funciona como um BB de intra-domínio.

Quando recebe um pedido, o RATES tenta encontrar um novo LSP (*"Layered Service Provider"*) sem alterar os já existentes. Embora tenha a capacidade de criar novos LSPs com uma largura de banda garantida, os requisitos de QoS de um SLA podem ter outros parâmetros a satisfazer, tais como *"jitter"*, latência e perdas de pacotes. De forma a obter informações sobre o estado das ligações de rede, o RATES é implementado como um "*link state peer"* na rede. Um pedido desencadeia a computação de uma rota para um novo pedido, podendo este pedido ser formulado por um encaminhador de entrada ou através de uma GUI por um utilizador. O RATES possui informação completa acerca da topologia da rede, sendo-lhe possível recuperar de possíveis falhas de ligação através do reencaminhamento do tráfego. Quando o RATES calcula o melhor caminho para satisfazer as exigências de um LSP, comunica ao encaminhador de entrada utilizando COPS de forma a proceder à sua implementação. Adicionalmente, envia decisões baseadas em políticas e reconfigura os encaminhadores de acordo com os parâmetros de classificação dos pacotes.

3.4.1.2 Abordagem de QoS em *Unified Layer 3*

O modelo básico da abordagem de QoS nível unificado 3 (*unified level 3*) deriva do *DiffServ*. Este modelo foi apresentado em [Clayton 1998]. As principais funcionalidades providenciadas pelos componentes da arquitectura são as seguintes:

- Existência de um gestor de QoS/BB que efectua controlo de admissão e gere os recursos de rede;
- Possibilidade de cada anfitrião do domínio efectuar pedidos de QoS ao gestor de Qos/BB;
- Os encaminhadores de fronteira efectuam policiamento e modelação de tráfego para cada fluxo de dados;
- Os encaminhadores de fronteira do domínio são responsáveis pelo policiamento e modelação do tráfego entre domínios.

O modelo fornece suporte a futuras reservas de recursos através de negociações de QoS flexíveis. Se um pedido não pode ser satisfeito devido a escassez de recursos, então são

apresentadas ao utilizador alternativas com requisitos de QoS inferiores que possam ser satisfeitos imediatamente. Em caso de indisponibilidade, um sinal de 'ocupado' é enviado ao utilizador.

Quando um pedido é aceite, então o BB configura cada encaminhador no caminho predeterminado do fluxo policiado. Os encaminhadores de admissão inicialmente procedem à etiquetagem dos pacotes que pode ser traduzida nos encaminhadores de fronteira, de forma a proceder ao policiamento de fluxos agregados. Esta etiquetagem é feita segundo os parâmetros de QoS com os quais o BB é configurado. Nos encaminhadores de saída, as etiquetas são removidas e os pacotes são redireccionados para o seu destino. O conceito básico em causa é a aplicação de um redireccionamento rápido de pacotes baseado em MPLS numa rede *DiffServ*, onde o encaminhamento é feito baseado na tradução das etiquetas.

3.4.2 BB em *IntServ* sobre modelo *DiffServ*

A arquitectura *IntServ* apresenta problemas de escalabilidade enquanto a arquitectura *DiffServ* não providencia fortes garantias de serviço. Procurou-se então obter um modelo que permitisse utilizar os pontos fortes de cada uma das arquitecturas de forma a colmatar as fraquezas da outra. Este objectivo foi alcançado através da integração de *IntServ* com *DiffServ.*

3.4.2.1 Modelo básico

Ao implementar *IntServ* em redes isoladas (*"stub networks"*) são satisfeitos os requisitos de garantia de serviço do utilizador especificados nos SLAs. Neste tipo de rede, os pedidos de reserva de recursos são feitos utilizando RSVP. A ligação entre várias redes isoladas é feita através de domínios de trânsito, com capacidade de *DiffServ* e a agregação dos fluxos que nele transitam é baseada em DSCP. A utilização de *DiffServ* no domínio de trânsito resolve o problema de escalabilidade e os encaminhadores do domínio de trânsito não necessitam de armazenar informação para cada fluxo individualmente.

O modelo básico desta arquitectura híbrida consiste em redes *IntServ* isoladas que utilizam um domínio de trânsito *DiffServ* para estabelecer ligações entre si e é apresentado em [Braun 1999]. O protocolo de sinalização utilizado em *IntServ* é o RSVP. Assume-se que cada anfitrião utiliza este tipo de sinalização para efectuar pedidos de QoS e que alguns terão também a capacidade de marcar o seu tráfego com DSCP. Os encaminhadores de fronteira suportam RSVP e comunicam com um BB inserido no domínio *IntServ*. Estes

encaminhadores suportam também *DiffServ*. Os encaminhadores de fronteira do domínio *DiffServ* providenciam funcionalidades de controlo de tráfego de acordo com os SLAs acordados entre os domínios, comunicando também com o BB do domínio *IntServ* de forma a condicionar o tráfego e gerir os recursos disponíveis. Cada rede *"stub"* possui anfitriões com capacidade para *IntServ* mas esta característica não é necessária para todos os encaminhadores nessa rede. Redes *"stub"* podem usar um BB para providenciar QoS adequada às suas necessidades. As mensagens RSVP atravessam o domínio de trânsito *DiffServ* de forma transparente.

As redes *IntServ* classificam o tráfego com base nas suas especificações enquanto redes *DiffServ* classificam-no com base no DSCP. Existe, por isso, uma necessidade de estabelecer uma relação entre estes dois tipos de classificação. A forma mais simples é associar o serviço de carga controlado e o serviço garantido do *IntServ* com AF e EF do *DiffServ*, respectivamente. É também possível estabelecer uma associação específica para cada utilizador. O encaminhador de fronteira da rede *"stub"* processa a mensagem RSVP e estabelece contacto com o BB do seu domínio. Este BB agrega os requisitos do fluxo e envia o pedido correspondente ao BB do domínio de trânsito. O pedido é então verificado pelo BB do domínio de trânsito utilizando o seu repositório de políticas. Se o percurso do fluxo passar por BBs adicionais no domínio de trânsito, então o primeiro BB contacta todos os outros BBs a jusante. Caso o pedido seja aceite por todos os BBs do domínio de trânsito e pelo BB do domínio *"stub"* de destino, então todos os BBs do domínio de trânsito configuram os seus encaminhadores de fronteira com o perfil de tráfego adequado. O último BB de trânsito a jusante necessita de enviar mensagens ao BB local do domínio *"stub"* de destino para estabelecer a reserva de recursos para o fluxo agregado no seu domínio.

3.4.2.2 Controlo de Admissão em *IntServ* sobre o modelo *DiffServ*

Em [Hwang 2000] é proposto um mecanismo para controlo de admissão em *IntServ* sobre o modelo *DiffServ*. É atribuído a cada classe *DiffServ* um valor de prioridade para um encaminhador de saída/entrada baseado na reserva de largura de banda para essa classe associado às ligações de saída/entrada nesse nó da rede. Quando um fluxo RSVP requisita um serviço ao domínio *DiffServ* fica associado a um PHB. É então atribuído ao fluxo um testemunho (*"token"*) cujo valor é determinado pelo seu PHB e pelos seus requisitos de QoS. O BB procede à gestão dos testemunhos para as ligações entrada/saída em cada nó da rede, sendo responsável pela provisão de QoS. Cada encaminhador armazena os valores de

prioridade de cada classe e dos testemunhos de cada fluxo que o atravessa. Estes valores são então utilizados pelo BB para tomar decisões relativamente ao controlo de admissão.

3.4.3 BB em *DiffServ* para a rede móvel

A implementação de uma arquitectura *DiffServ* numa rede móvel possibilita-lhe o suporte a diversas aplicações e serviços multimédia [Liljebladh 1999]. No entanto, a integração destes dois tipos de rede não é trivial, levantando alguns problemas cuja solução potencializa diversas áreas de investigação. Alguns destes problemas podem ser solucionados através da inserção de um BB na arquitectura. O principal problema reside na necessidade da existência de SLAs dinâmicos devido à constante mudança da localização dos anfitriões móveis e a presença de um BB na arquitectura permite o suporte a essa funcionalidade. Uma abordagem de optimização de encaminhamento sugere a ligação directa entre os anfitriões móveis e o BB da sua rede, levando à necessidade de se estabelecer um protocolo de sinalização entre estas entidades. Quando um anfitrião se move de um domínio para outro é necessário que os BBs dos domínios atravessados comuniquem entre si de forma a trocarem informação acerca dos requisitos de QoS para esse anfitrião. Esta comunicação deve ser feita de forma directa e é fundamental para evitar áreas de incerteza (*"black holes"*) entre domínios.

3.4.3.1 *DiffServ* na rede GPRS

A compatibilidade entre *DiffServ* e GPRS é discutida em [Karagiannis 2000]. A área das comunicações móveis sofreu uma grande expansão nos últimos anos e o GPRS foi um dos primeiros serviços de dados baseados em pacotes oferecidos pelas redes GSM (segunda geração). Para além de serviços de transmissão de voz, este tipo de comunicações é também utilizado para transferir dados. O serviço geral de rádio por pacotes (GPRS – *"General Packet Radio Service"*) foi o primeiro serviço de dados baseado em pacotes oferecido pelas redes GSM (segunda geração).

Uma estação móvel (MS – *"Mobile Station"*) corresponde ao equipamento utilizado para aceder ao serviço de telecomunicações. A entidade física responsável pela comunicação entre a rede GPRS e utilizadores contidos na sua área de serviço é denominada nó de suporte GPRS (SGSN – *"Supporting GPRS Support Node"*). A ligação entre a rede móvel e outras redes é feita através de um portal (*"gateway"*) de suporte GPRS (GGSN – *"Gateway GPRS*

Support Node"). É necessário expandir as funcionalidades do GGSN de forma a torná-lo compatível com *DiffServ*.

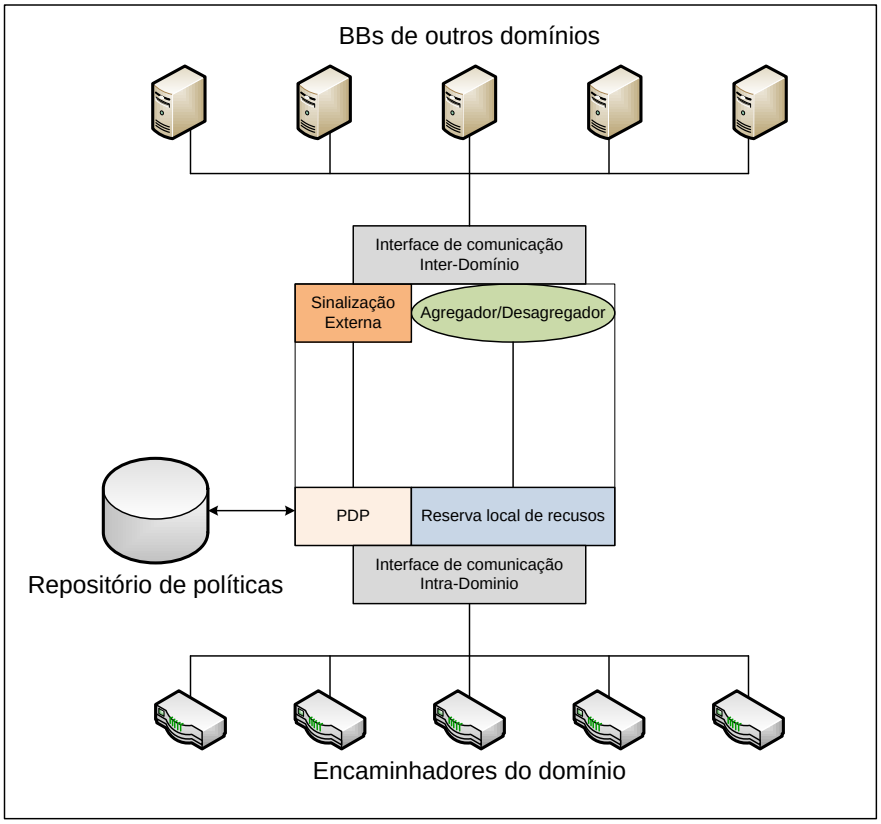

Figura 3.5 - BB numa rede de acesso móvel

A Figura 3.5 apresenta a arquitectura de um BB numa rede de acesso móvel. O seu papel na rede pode ser sumarizado pelos seguintes pontos:

- O BB deve ser capaz de agregar mensagens de reserva RSVP no sentido "*uplink*"e desagregá-las no sentido oposto (*"downlink"*);
- Deve ser capaz de comunicar com BBs de domínios *DiffServ* vizinhos;
- Deve ser capaz de manter um repositório de políticas de forma a poder desempenhar as funções de PDP e de configurar PEPs.

Uma rede GPRS não é uma rede *DiffServ*. Como tal, o GGSN é utilizado para estabelecer a ligação com o *DiffServ* através da implementação de um mecanismo de marcação de tráfego antes de este entrar no domínio *DiffServ*. Em relação ao BB do domínio *DiffServ*, o GGSN é visto como um BB do domínio GPRS, pois apresenta algumas funcionalidades de BB.

Existem dois cenários diferentes a considerar neste tipo de arquitectura. Num destes cenários, um anfitrião remoto (RH – *"Remote Host"*) encontra-se no domínio de acesso *DiffServ* e o domínio GPRS comunica com esse anfitrião através de uma rede *DiffServ*. Neste caso, o GGSN funciona como um agregador e o BB de acesso como desagregador. O MS envia o seu pedido ao GGSN, que o reencaminha para o BB de acesso. Através da utilização de COPS, o BB de acesso comunica a sua decisão para o RH e também quais os recursos disponíveis para aceder à MS.O BB de acesso também notifica os BBs vizinhos acerca das suas decisões. Desta forma, os BBs vizinhos podem configurar os encaminhadores do seu domínio e enviar ao GGSN as suas próprias decisões. O GGSN então actualiza o seu estado de encaminhamento (*"path state"*) e envia uma mensagem de confirmação a BB de acesso através do BB vizinho. Quando o RH envia uma mensagem de aprovação ao BB de acesso, esta é reencaminhada para o SGSN através do GGSN. O SGSN notifica então o MS e estabelece-se a comunicação entre a MS e o RH.

No segundo cenário, o RH procura estabelecer uma comunicação com a MS, enviando uma mensagem de pedido COPS ao BB de acesso. O BB traduz a mensagem para uma mensagem ponto-a-ponto RSVP e envia-a para o GGSN com o respectivo perfil de QoS. Ao receber a mensagem, o GGSN procede à sua conversão para uma mensagem do tipo "criar pedido em contexto PDP" e envia-a ao SGSN, que a reencaminha para a MS. Uma resposta positiva por parte da MS traduz-se no envio de uma mensagem do tipo "aceitação de activação em contexto PDP" para o GGSN. Uma mensagem ponto-a-ponto RSVP de aceitação de reserva de recursos é enviada ao BB de acesso, que comunica a decisão ao RH através de COPS.

3.4.4 QoS em UMTS através do serviço IMS

O serviço IMS foi concebido pelo 3GPP (*"3rd Generation Partnership Project"*) como forma de fornecer serviços IP multimédia em redes de comunicação móveis, num esforço de evolução para além do GSM. Tornou-se assim parte integrante dos padrões de funcionamento de sistemas de telefonia móvel 3G (terceira geração) em redes UMTS. Para facilitar a sua integração com a *Internet*, o IMS utiliza protocolos da IETF (como o SIP) sempre que possível.

Relativamente à QoS do UMTS, são definidas quatro classes de serviço:

- Conversação;
- *"Streaming"*;
- Interactiva;
- *"Background"*.

A principal distinção entre estas classes de serviço é a sensibilidade do seu tráfego em relação à latência. A classe de conversação é destinada a tráfego com uma grande sensibilidade à latência enquanto o tráfego da classe *'Background*' é o menos sensível a esse problema.

As classes de 'Conversação' e de '*Streaming*' são principalmente utilizadas para fluxos de tráfego em tempo real. É também a sua sensibilidade à latência que as distingue. Os serviços de conversação como a telefonia com vídeo são aplicações de tempo real extremamente sensíveis à latência, sendo o seu tráfego transportado na classe de conversação.

As classes 'Interactiva' e '*Background*' têm requisitos menos exigentes em termos de QoS, sendo utilizadas para o transporte de aplicações tradicionais da *Internet* como correio electrónico, *Telnet*, FTP, WWW e *News*, cujo desempenho é menos vulnerável ao *"jitter"* e à latência. A principal diferença entre as classes 'Interactiva' e '*Background*' é que a primeira, tal como o nome indica, é destinada a aplicações que requerem uma interacção com o utilizador em tempo real, enquanto a classe '*Background*' destina-se normalmente à transferência de ficheiros. O tráfego da classe 'Interactiva' tem maior prioridade relativamente ao tráfego da classe '*Background*'. Assim, as aplicações de '*Background*' apenas utilizam recursos de transmissão quando estes não estão em uso por parte de aplicações interactivas. Esta característica é muito importante em ambientes de redes em fios, em que a largura de banda disponível é pequena, quando comprada a redes fixas.

No entanto, o uso das classes de tráfego não é rígido no sentido de que determinado tipo de tráfego só pode ser transportado por determinada classe. Por exemplo, um serviço interactivo pode utilizar a classe de 'Conversação' se a aplicação ou o utilizador tiverem requisitos elevados de QoS em relação à latência.

O conceito e a arquitectura de QoS em redes UMTS encontra-se descrito em pormenor em [3GPP 2007].

3.5 Resumo

Um BB pode desempenhar controlo de admissão através da gestão de políticas. Possui ainda a capacidade de aplicar essas políticas através da configuração dos encaminhadores de fronteira do seu domínio *DiffServ*. Dentro do domínio *DiffServ*, o BB aplica e gere SLAs dinamicamente. Ao nível da rede, o BB desempenha um importante papel no fornecimento de QoS ponto-a-ponto aos utilizadores do seu domínio. Devido ao grande número de responsabilidades a cargo desta entidade, desenvolver um BB padrão que as desempenhe eficazmente não é trivial, o que originou vários trabalhos de pesquisa e desenvolvimento acerca deste tema.

O protocolo SIBBS é uma possibilidade para a comunicação entre BB de domínios distintos mas o seu desenvolvimento é ainda parcial, existindo ainda a necessidade de encontrar um protocolo de comunicação entre domínios *DiffServ* para BBs que seja eficiente e seguro. A segurança é um dos principais problemas no desenvolvimento de um protocolo deste tipo. Para além da preservação da integridade das mensagens, também a segurança ponto-a-ponto na comunicação é de importância crítica.

Não se pode assumir que o BB está ligado de forma segura e confiável a todos os encaminhadores do seu domínio. Isto aumenta a necessidade de monitorizar o estado das ligações da rede para se obter uma comunicação segura intra-domínio. Esta funcionalidade deve ser integrada nas funcionalidades do BB, ultrapassando assim os problemas criados por alterações na topologia da rede.

Os SLAs são negociados dinamicamente entre BBs devido às constantes alterações na natureza do tráfico na rede, existindo a necessidade de desenvolver um mecanismo através do qual um BB possa optimizar o suporte a SLAs dinâmicos. Esta funcionalidade é um factor crítico na optimização da utilização de recursos da rede.

CAPÍTULO 4.

CONFIGURAÇÃO TRANSPARENTE DE QOS

4.1 Introdução

Nesta dissertação foi desenvolvido um sistema que realiza a configuração dos parâmetros de QoS de forma transparente para o utilizador, numa rede multi-domínio com serviços diferenciados.

O protótipo desenvolvido consiste numa versão modificada da implementação de um protótipo de BB (*"Bandwidth Broker"*) desenvolvido em Java, levada a cabo por Khoi Ba Pham e Richmond Nguyen da *"University of New South Wales"*, sob a orientação de Sanjay Jha e Shaleeza Sohail, no âmbito do seu projecto final de Bacharelato em Engenharia de Telecomunicações [Pham 2003].

A principal modificação feita à arquitectura original consiste na integração da interface com o utilizador num procurador SIP (SIP *"proxy"*), de forma a conseguir que todo o processo de pedido e reserva de recursos se processe de forma transparente para o utilizador. O procurador SIP utilizado foi o *'JAIN-SIP-presence-proxy'*, desenvolvido pelo NIST (*"National Institute of Standards and Technology"*) [NIST 2008].

Na arquitectura original, era necessária a troca de mensagens entre o BB e o utilizador, através de uma aplicação BBClient. Na arquitectura implementada nesta dissertação, a troca é efectuada entre o BB e o procurador SIP modificado, utilizando o protocolo originalmente

destinado para a comunicação entre BB e a interface de utilizador BBClient, com algumas modificações. Esta interacção pode ser observada na Figura 4.1, onde é apresentado um cenário envolvendo uma ligação VoIP entre dois utilizadores e a correspondente configuração de QoS por parte do BB. Os elementos dessa figura, apresentados a tracejado, são transparentes do ponto de vista dos utilizadores.

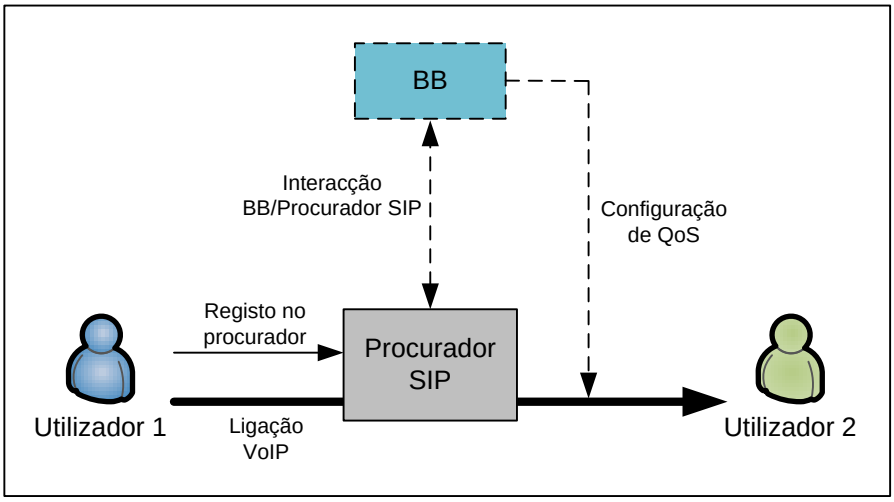

Figura 4.1 – Configuração transparente de QoS, num cenário de ligação VoIP

Foi eliminada a necessidade do utilizador realizar um *"login"* através do BBClient, o que requeria a introdução por parte do utilizador do porto e endereço do BBServ (servidor do BB) do seu domínio. Era ainda necessário ao utilizador introduzir a sua identificação de SLA e a sua senha de validação. Posteriormente, o utilizador teria de introduzir os parâmetros necessários para proceder à reserva de largura banda (hora e data de início de fim da reserva, quantidade de largura de banda desejada e os endereços IP de origem e destino do fluxo de dados). Uma vez que estes dados podem ser obtidos através da análise das mensagens SIP, o procurador SIP foi modificado para extrair a informação necessária dos cabeçalhos das mensagens de início de sessão SIP e utilizá-la para elaborar o pedido de reserva de recursos. Quanto à identificação de SLA e senha do utilizador, o procurador foi modificado para utilizar os dados introduzidos pelo utilizador quando este se regista no procurador SIP. Assim sendo, basta ao utilizador registar-se no procurador e iniciar uma sessão SIP. Todo o processo de reserva de largura de banda decorre então automaticamente e de forma transparente para ele.

Adicionalmente foi ainda elaborada uma configuração de domínio *DiffServ* para testar a arquitectura implementada.

4.2 Descrição da Arquitectura

Nesta secção pretende-se apresentar o funcionamento global da arquitectura proposta, assim como uma descrição mais pormenorizada dos seus elementos constituintes.

4.2.1 Funcionamento Global da Arquitectura

Através da Figura 4.2, é possível obter uma perspectiva global do funcionamento da arquitectura implementada, dentro de um domínio *DiffServ* comum a todos os utilizadores (intra-domínio) e entre vários domínios *DiffServ* (inter-domínio). Cada domínio possui uma única entidade BB, que é composta pelo BBServ e o PEPClient, ligada a uma base de dados. O BBServ actua como PDP, baseando as suas decisões na informação contida na base de dados. Esta interacção é estabelecida através do uso de *MySQL*. As decisões são então enviadas para o seu PEPClient, que por sua vez, configura os encaminhadores de forma a satisfazer o pedido de reserva de recursos. Se a comunicação estabelecida apenas envolve utilizadores dentro de um domínio comum, então somente os encaminhadores internos (CRs) são reconfigurados. Caso a comunicação ocorra entre utilizadores em domínios distintos, torna-se necessário configurar também os encaminhadores de fronteira que estabelecem a ligação com domínios adjacentes. A interface entre os utilizadores e o BB é feita de forma indirecta através do procurador SIP.

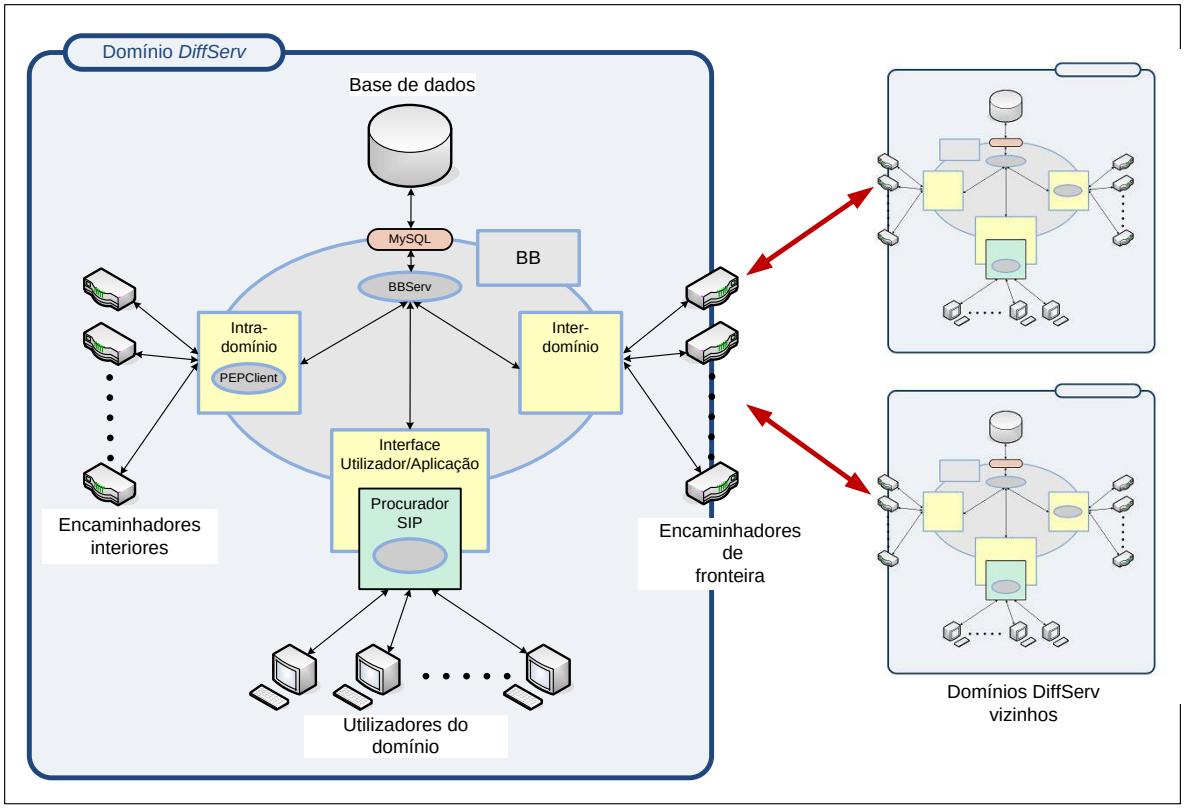

Figura 4.2– Diagrama da arquitectura implementada

4.2.1.1 Intra-domínio

No nível intra-domínio, o BB necessita de comunicar tanto com os encaminhadores interiores como com os de fronteira, de forma a transmitir decisões de políticas. É segundo estas decisões que os encaminhadores são configurados, de forma a providenciarem qualidade de serviço (QoS). Existem vários protocolos que sustentam esta comunicação, como o COPS ou o SNMP, ficando a escolha de qual usar a cargo do administrador da rede, uma vez que esta decisão só tem importância a nível local do domínio *DiffServ*. Nesta dissertação, optou-se por utilizar a versão do protocolo COPS incluída na arquitectura original, pois serve o objectivo proposto, uma vez que não foi necessário alterar a comunicação entre o BB e os encaminhadores.

O PEP tem a capacidade de lidar com tráfego IP e implementa controlo de admissão baseado em políticas para os fluxos de dados, enquanto o PDP tem uma visão global da rede e configura os PEPs de acordo com a política de rede. O BB actua como PDP enquanto os encaminhadores são configurados como PEPs. Existe uma ligação cliente/servidor entre PDP (servidor) e PEP (cliente) através de uma ligação TCP, sobre a qual é utilizado o protocolo COPS para a comunicação.

O BB funciona como um ponto de decisão de políticas (PDP) que se liga aos encaminhadores do seu próprio domínio de forma a configurá-los de acordo com a política de domínio predefinida. Quando o BB aceita um pedido, os encaminhadores relacionados, tanto os interiores como os de fronteira (se necessário), são contactados utilizando COPS-PR. Um encaminhador interior necessita de ser reconfigurado quando é o primeiro ponto de passagem para o fluxo. Esta reconfiguração é necessária para a marcação e modelação dos pacotes do fluxo. A marcação é necessária para a classificação do pacote e a modelação é necessária para manter o fluxo dentro dos limites pretendidos. Um encaminhador de fronteira é contactado pelo BB quando o destino ou a origem do fluxo se encontra num domínio *DiffServ* possibilitando que este filtre/modele/marque os pacotes do fluxo de acordo com o seu SLA.

A interacção do BB com o utilizador é feita de forma indirecta através do procurador SIP. Os dados necessários para a identificação do utilizador são obtidos a partir do registo do utilizador no procurador e os parâmetros que compõem o pedido de reserva de recursos são obtidos a partir da análise dos pacotes de inicio da sessão SIP estabelecida. Na Figura 4.3 é apresentado o estabelecimento de uma sessão SIP entre dois utilizadores no mesmo domínio *DiffServ*. É possível observar também a interacção entre o procurador SIP e o BB, através do seu servidor (BBServ). O procurador recolhe os dados necessários e envia-os para o BB sob a forma de um pedido de reserva de recursos e recebe a respectiva resposta. Quando a sessão termina, é também o procurador SIP o responsável por terminar a reserva de recursos. Do ponto de vista do utilizador, todo o processo é mantido transparente. Para obter uma sessão SIP com QoS, o utilizador necessita apenas de se registar no procurador e estabelecer a sessão. No âmbito desta dissertação foi utilizado um cenário de uma chamada telefónica VoIP entre dois utilizadores no mesmo domínio *DiffServ*, que se processa utilizando SIP. O cenário é análogo ao apresentado na Figura 4.3, com os dois utilizadores a comunicarem através do procurador SIP, que faz um pedido de reserva de recursos ao BB para garantir QoS para a chamada.

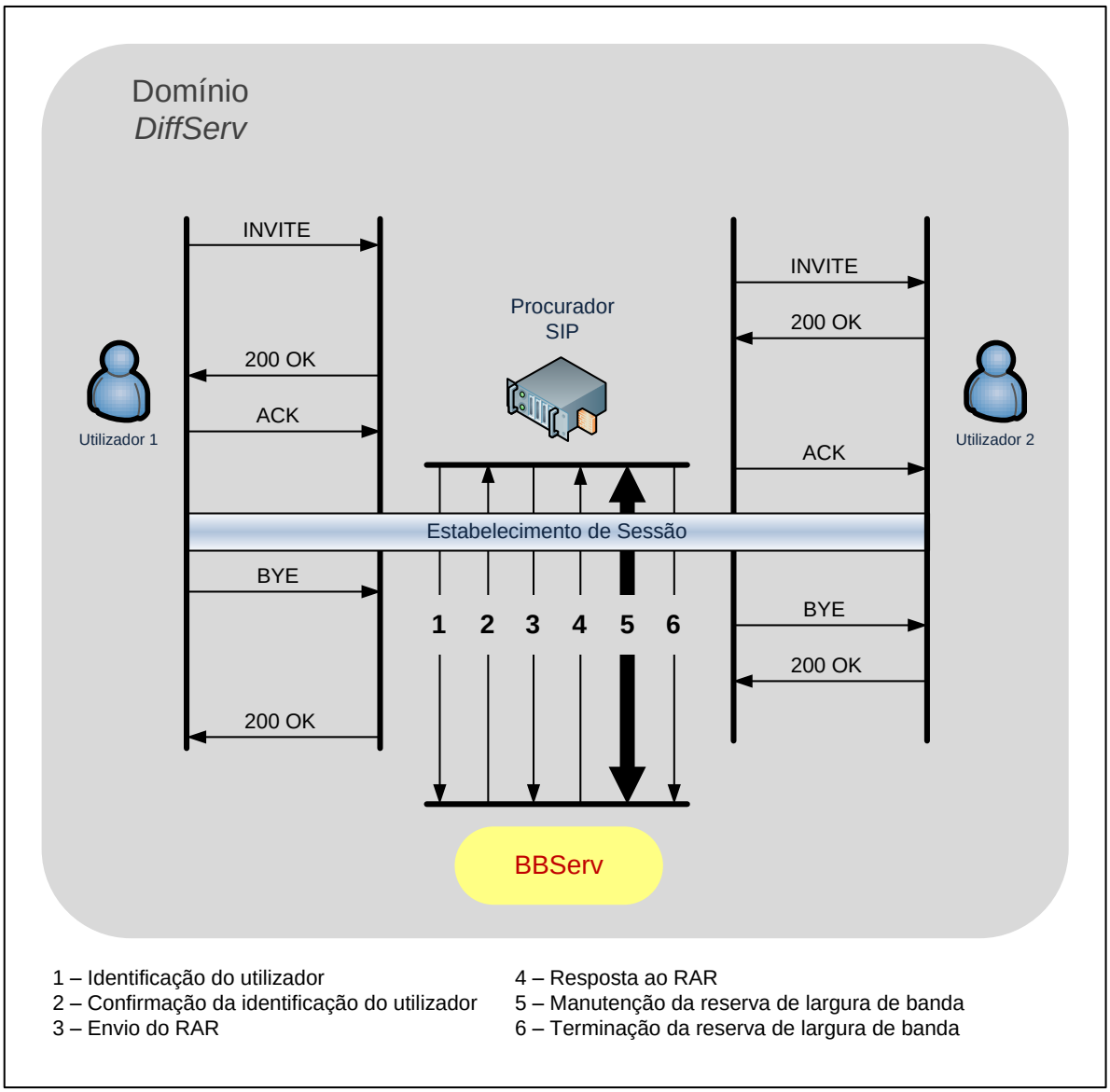

Figura 4.3– Interacção entre procurador SIP, utilizadores e BB

4.2.1.2 Inter-domínio

A funcionalidade do BB a nível inter-domínio consiste na sua comunicação com BBs vizinhos, de forma a reservar recursos em outros domínios. Isto é necessário quando o destino do fluxo de dados do utilizador se encontra num domínio *DiffServ* diferente. O protocolo utilizado nesta comunicação está incluído na estrutura do BB e é baseado no SIBBS, proposto pela comissão de aconselhamento sobre BBs, da Internet2 QBone. Este protocolo segue um modelo de pedido-resposta entre BBs e é orientado para o emissor. Os BBs estabelecem ligações TCP de longa duração, com o TCP a assegurar a fiabilidade básica e o controlo de fluxo ao protocolo. Quando um BB recebe um pedido de reserva de recursos (RAR) de um BB vizinho, este verifica a autenticação do emissor, a rota, o encaminhador de saída (*"egress*

router") do fluxo, o SLA associado ao utilizador e as políticas associadas ao fluxo. Caso o destino do fluxo não se encontre no seu domínio *DiffServ*, o BB propaga o RAR que lhe foi enviado para o próximo BB vizinho que se encontre no caminho do fluxo. O RAR é assim propagado até atingir o BB que contém o destinatário do fluxo no seu domínio. A resposta ao pedido de recursos (RAA) é em enviada ao BB de origem através do caminho inverso.

As especificações de projecto do protocolo SIBBS não definem explicitamente a forma como o BB recolhe informações acerca dos seus BBs vizinhos ou dos seus domínios. No caso do protótipo implementado, o protocolo utilizado recolhe a informação da sua base de dados, que contém um mapeamento da rede, permitindo ao BB identificar qual o vizinho que deve contactar para satisfazer o RAR do utilizador.

Após enviar a RAA, caso aceite o pedido, o BB configura os seus encaminhadores de fronteira utilizando o seu protocolo intra-domínio de forma a reservar os recursos de rede necessários para o fluxo aceite.

Na Figura 4.4 é apresentado um cenário de comunicação entre dois utilizadores localizados em domínios *DiffServ* distintos. O cenário é análogo ao descrito na secção 4.2.1.1 para a comunicação intra-domínio mas neste caso, o destinatário da comunicação encontra-se num domínio *DiffServ* diferente. O procurador envia o RAR para o servidor do BB do seu domínio (BBServ 1) mas este não tem capacidade para garantir QoS nos dois terminais da comunicação. Necessita, por isso, estabelecer uma ligação ao servidor do BB do domínio *DiffServ* vizinho (BBServ 2), onde está localizado o Utilizador 2. O RAR é então propagado para o BBServ 2. Em conjunto, os BBs dos dois domínios *DiffServ*, garantem QoS para a comunicação em ambos os terminais. Se o Utilizador 2 estivesse num domínio *DiffServ* que não fosse adjacente ao domínio onde se localiza o Utilizador 1, o BBServ 1 propagaria o RAR para os BBs vizinhos, que o propagariam para os seus vizinhos, até o RAR alcançar o BB do domínio *DiffServ* do Utilizador 2.

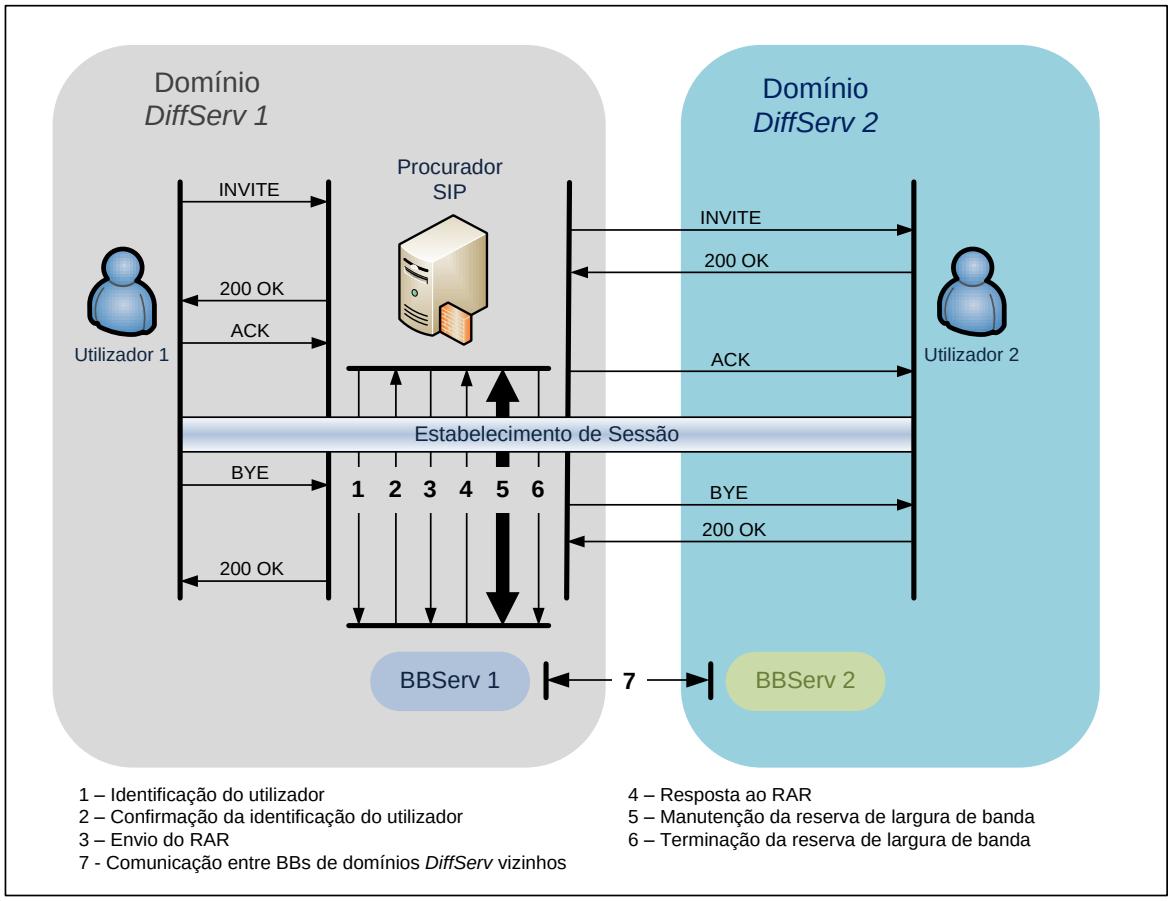

Figura 4.4– Comunicação entre utilizadores em domínios distintos

4.2.2 Servidor do BB (BBServ)

A actividade do BB é iniciada com o arranque do módulo BBServ.

Um dos principais requisitos para o funcionamento do BB é a capacidade de permitir a ligação por parte de vários clientes em simultâneo e processar os seus pedidos. Esta comunicação é efectuada através de uma interacção do tipo cliente/servidor, utilizando o protocolo COPS através de ligações TCP. O servidor do BB também aceita ligações de encaminhadores Linux (PEPs), uma vez que o BB actua como PDP, como definido pelo protocolo COPS. Estas ligações também são efectuadas utilizando TCP.

Outra função do BBServ é remover da base de dados os pedidos à medida que estes vão expirando e proceder à reconfiguração dos encaminhadores da rede, para que estes às novas necessidades de tráfego.

Após o estabelecimento da ligação do cliente ao BB, é necessário estabelecer uma troca de dados entres as duas entidades. Esta é efectuada através de dois fluxos de dados (um de entrada e outro de saída). Os dados são transferidos entre o servidor do BB e o cliente em mensagens compostas por *"strings"* de texto concatenadas. Ao condensar os dados do

utilizador numa única mensagem é possível reduzir o tráfego na rede, pois a interface do cliente condensa todos os dados necessários numa única *"string"* e envia-a ao servidor, em vez de enviar cada um individualmente. Segundo o protocolo proposto, as mensagens são formatadas de acordo com um padrão pré-definido. Os parâmetros que compõem a mensagem são separados pelo carácter ';', tendo este sido escolhido por não ser utilizado na composição dos mesmos e a ordem dos parâmetros é pré-estabelecida, para possibilitar a extracção da informação por parte do BB. De acordo com o protocolo proposto, um pedido de reserva de recursos (RAR) apresenta a seguinte forma:

Pedido de recursos (RAR):

tipo;sla;data de início; hora de início; data de fim;hora de fim;largura de banda;fonte;destino

Exemplo:

"requestbw;1;2008-08-02;00:00:01;2008-10;23:59:59;1000;129.94.231.23;129.24.232.41"

- tipo – indica ao servidor o tipo da mensagem,
- sla – identificação do SLA.
- data de início – data de início da validade do RAR.
- hora de início – hora a partir da qual o RAR entra em vigor.
- data de fim - data de expiração do RAR.
- hora de fim - hora a que expira o RAR,
- largura de banda – largura de banda requisitada.
- fonte - endereço IP da fonte do pedido.
- destino - endereço IP do destino do pedido.

São também especificados outros dois tipos de mensagem: "*exit*" e "*shutdown*". As mensagens do tipo "*exit*" são utilizadas para terminar a ligação entre o cliente e o servidor (neste caso, entre o procurador SIP e o BBServ). As mensagens do tipo "*shutdown*" são utilizadas para desligar o BB. Na implementação original [Pham 2003], são disponibilizados outros tipos de mensagem, que permitem ao cliente requisitar informações e editar SLAs e RARs. No entanto, como o conceito básico desta tese é a transparência do serviço em relação

ao utilizador, a interface original que permite o acesso a estas funcionalidades não foi utilizada.

Ao receber a mensagem, o servidor do BB procede à sua desconstrução, extraindo os parâmetros relevantes. Efectua então a sua autenticação, verificando a identificação do utilizador assim como a sua senha de acesso ao serviço do BB, através da comparação dos dados com a informação contida na sua base de dados. Uma vez que não é utilizada encriptação na transmissão de dados entre o servidor e o cliente, apresenta-se aqui uma falha de segurança. Mas no âmbito desta dissertação, como prova de conceito, este funcionamento simplificado é suficiente para alcançar o objectivo proposto Caso o resultado da validação seja negativo, a ligação ao cliente é terminada. Quando o cliente é validado com sucesso, é-lhe atribuída permissão para efectuar pedidos de recursos ao servidor, sendo a sua identificação utilizada para determinar que serviços lhe estão disponíveis, pois cada identificação está associada a um SLA negociado previamente.

Para que o BB possa operar em modo inter-domínio, é necessário que este se possa ligar a outros BB em domínios *DiffServ* adjacentes. Esta funcionalidade foi definida pela *"QBone Signaling Design Team"* [Chimento 2002]

4.2.3 Comunicação COPS-PR

O COPS é utilizado para enviar decisões de políticas de um ponto de decisão de políticas (PDP) para um ponto de decisão implementação de políticas (PEP). Estes protocolo tem também algumas funcionalidades que o tornam particularmente indicado para ser usado com BBs. Uma destas funcionalidades é o mecanismo de manutenção de actividade (*"keep alive"*) que permite ao PEP determinar se o seu PDP está activo ou não. Permite ainda ao PDP redireccionar um utilizador caso não suporte o seu tipo ou para balanceamento de carga na rede. Para protecção contra ataques de negação de serviço e de roubo de serviços, o COPS recorre ao IPSec para providenciar a segurança necessária.

Para o suporte de aprovisionamento baseado em políticas, foi introduzida uma extensão do COPS, o COPS-PR. Este protocolo é independente do tipo de política aprovisionada, podendo ser QoS ou segurança. Suporta ainda um mecanismo de comunicação por eventos em tempo real.

4.2.4 Interface com o utilizador

Na arquitectura original, estão disponíveis dois tipos de cliente, um para uso geral em termos de acesso ao BB (BBClient) e outro para executar serviços administrativos, apenas disponíveis para gestores de rede (BBClientAdmin). Devido ao objectivo desta dissertação de estabelecer uma configuração de QoS de forma transparente para o utilizador, a aplicação de interface com o utilizador BBClient foi removida da arquitectura, sendo as suas funcionalidades integradas no procurador SIP utilizado. Quanto ao BBClientAdmin, o uso desta aplicação é compatível com a arquitectura implementada, embora não faça parte do âmbito da dissertação. Com esta aplicação, um utilizador com privilégios administrativos ou um gestor de rede pode aceder a funções que permitem a criação e edição de SLAs, consultar informação acerca de SLAs e RARs em vigor e terminar a actividade do BB, através de uma interface de texto na consola Linux.

Como já foi dito, a interacção BBClient/BBServ é substituída por procurador SIP/BBServ, tornando todo o processo transparente ao utilizador. Ele simplesmente efectua o registo no procurador e inicia uma sessão SIP. As mensagens que o BBServ necessita receber do BBClient são enviadas pelo procurador. Para estabelecer a troca de mensagens, o procurador necessita criar uma ligação TCP com BB. Para isso, precisa saber o endereço IP e porto do BBServ. Esta informação é configurável no procurador (mediante algumas alterações à sua versão original) pelo administrador da rede. A identificação do SLA e a senha de validação são obtidas pelo procurador no registo do utilizador. Basta ao administrador da rede colocar a entrada correspondente na base de dados, associando o nome do utilizador à identificação do SLA e o IP à senha ou então utilizando a opção do procurador de executar o *"login"* do utilizador mediante a introdução de uma senha e utilizar os dados introduzidos para efectuar essa mesma associação.

A aplicação BBClient da arquitectura original foi integrada no procurador SIP mantendo as funcionalidades originais de comunicação com o BB e de envio de pedidos de reserva de recursos, mas outras funcionalidades foram descartadas. Uma vez que se pretendia manter todo o processo de reserva de recursos o mais transparente possível para o utilizador, foi eliminada a interface de texto através da qual o utilizador poderia solicitar informações acerca dos SLAs e RARs em vigor, modificar e apagar RARs ou seleccionar a quantidade de largura de banda desejada. Quando o utilizador inicia uma ligação VoIP através do procurador SIP modificado é este que efectua o pedido de reserva de largura de banda ao BB, utilizando os dados do utilizador extraídos do cabeçalho da mensagem SIP de início de sessão.

Caso a ligação seja aceite pelo servidor do BB, este fica a aguardar a identificação do SLA do utilizador e a correspondente senha. Nesta fase do processo, o BBClient original solicitava estes dados ao utilizador através da sua interface de texto. Na versão modificada, é enviada de forma automática ao BB a identificação de utilizador e correspondente senha utilizadas quando o utilizador fez o seu registo no procurador SIP. Como esta é uma operação simples e facilmente configurável no telefone de *"software"* utilizado, não põe em causa o conceito de transparência defendido nesta tese. No entanto, torna-se necessário alterar as entradas correspondentes na base de dados para que o BB consiga estabelecer uma relação entre o utilizador, o SLA que lhe pertence e a senha de verificação.

Após a recepção dos dados, o BB verifica então se a identificação está ou não correcta. O resultado desta operação é retornado ao procurador SIP pelo BB mas não é apresentado ao utilizador. Em caso negativo, o procurador acaba por estabelecer a ligação VoIP mesmo sem QoS assegurado. Caso a identificação seja positiva, o pedido é composto pelo procurador obedecendo aos critérios de formatação das mensagens aceites pelo BB, sendo então enviado.

4.2.5 PEPClient

O módulo PEPClient tem como função estabelecer o ambiente *DiffServ* no domínio. Está em contacto com o servidor BB e recebe deste as informações de configuração de encaminhamento necessárias ao tratamento adequado dos pacotes que recebe e executa-as.

Basicamente, este módulo é um encaminhador Linux que funciona como um PEP, segundo a especificação do protocolo COPS-PR. O PEPClient complementa assim o BBServ (que funciona como PDP) na provisão de serviço *DiffServ*.

Após o PEPClient se ter inicializado para a sua operação em *DiffServ*, procura estabelecer ligação com o PDP que monitora o seu domínio (neste caso, o BBServ). Para tal, acede à base de dados para encontrar o endereço IP e o porto do BBServ e estabelece uma ligação TCP com esses parâmetros. Se a ligação for aceite, o PEPClient actualiza a base de dados, indicando que se encontra em actividade e disponível para receber ordens de configuração. O BBServ envia-lhe uma mensagem COPS do tipo 'Decision', contendo a informação de configuração em vigor. Posteriormente, o BBServ enviará mensagens deste tipo sempre que for necessário proceder a alterações na configuração de *DiffServ*. Isto pode ser observado na Figura 4.5.

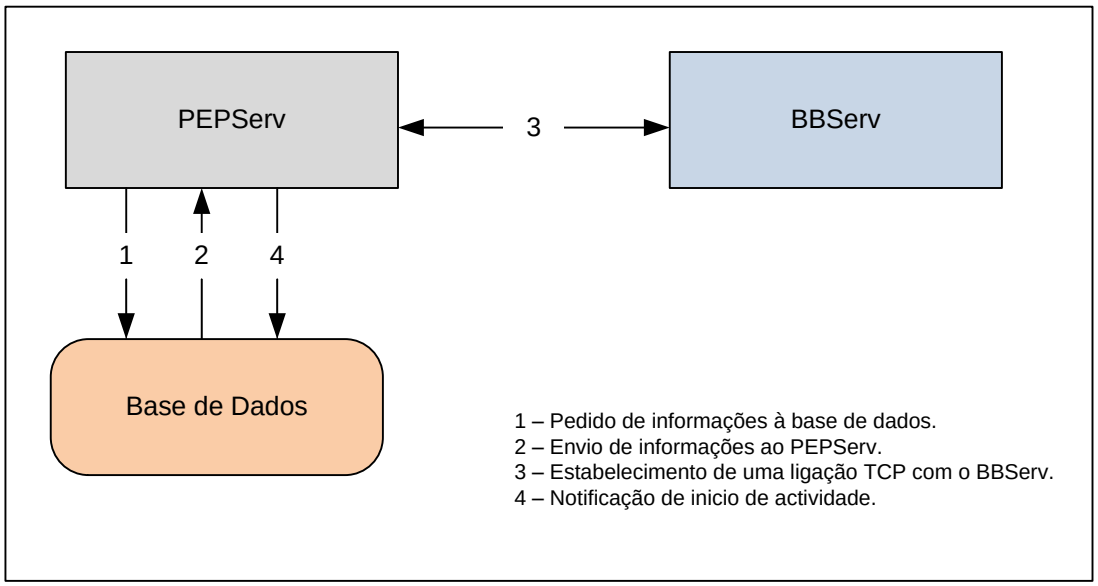

Figura 4.5 – Interacção BBServ/PEPClient

A forma como a informação é extraída da mensagem COPS faz parte do pacote COPS utilizado e informação adicional acerca deste processo pode ser encontrada em [Pham 2003].

A utilização do protocolo COPS permite a servidores remotos enviarem informações de configuração ao PEPClient, o que oferece a flexibilidade necessária para alterações dinâmicas no encaminhamento.

4.2.6 Base de Dados

A base de dados utilizada pelo BB é uma base de dados *MySQL* e é onde está armazenada toda a informação crítica para a viabilidade do BB. A informação nela contida pode ser dividida em três partes distintas: utilizador, BB e rede. A parte da base de dados referente ao utilizador consiste nos SLAs do utilizador, a sua senha (*"password"*) de acesso e a informação acerca dos seus pedidos de recursos. A parte referente ao BB contém informação acerca de outros BBs na rede e dos SLAs destes BBs. A última parte da base de dados contém toda a informação acerca da rede, essencial para que seja possível determinar que encaminhadores necessitam de ser reconfigurados quando o BB aceita um pedido. Esta informação é também essencial para determinar que BB vizinho deve ser contactado quando o pedido de recursos inclui recursos de outros domínios.

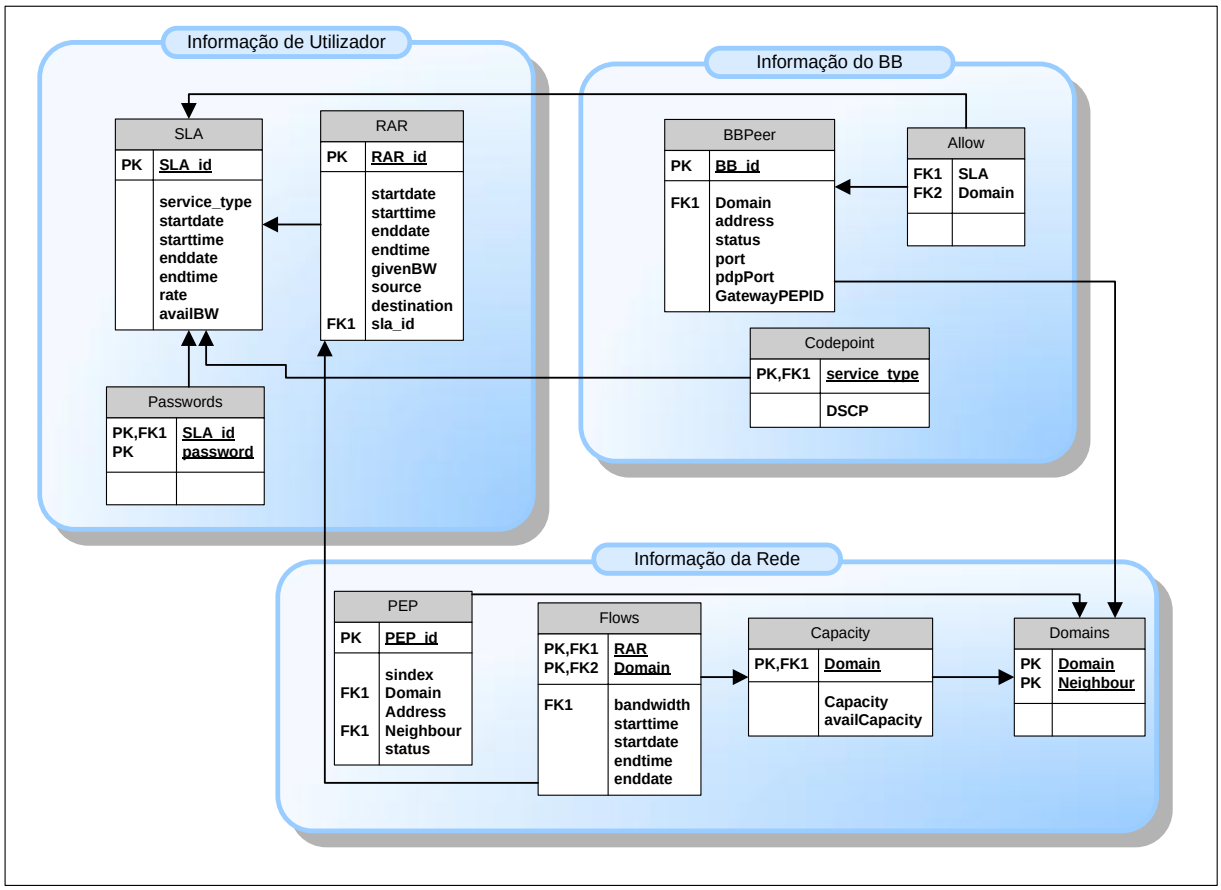

Figura 4.6 – Esquema relacional da base de dados

- ## Informação de Utilizador

  - **Tabela: SLA**

A tabela SLA contém a descrição do SLS acordado com o cliente, ou seja, as características técnicas do SLA. A chave primária utilizada para esta tabela é o 'sla_id' atribuído ao cliente. Uma vez que cada utilizador conhece o seu 'sla_id', o acesso ao BB torna-se mais simples para os utilizadores se for feito através desta identificação ao invés de ter uma identificação separada para SLS e SLA. Nesta implementação, a tabela SLA possui condições suficientes para lidar com pedidos de reserva de recursos simples. No entanto, a constituição da tabela é flexível, sendo possível adicionar mais campos, como a altura do dia em que o utilizador pode efectuar pedidos de reserva de recursos, caso surja essa necessidade.

Campo	Tipo	Comentário
sla_id(chave primária (PK))	INTEGER	Identificação numérica atribuída ao SLA negociado
service type (tipo de serviço)	VARCHAR	Texto representativo do tipo de serviço *DiffServ* (exemplo: EF, AF, BE)
Startdate	DATE	Data de início do SLA
Starttime	TIME	Hora de início do SLA
Enddate	DATE	Data de expiração do SLA
Endtime	TIME	Hora de expiração do SLA
Rate (taxa de débito)	INTEGER	O total de largura de banda reservada para este SLA (em kbs)
AvailBW	INTEGER	Largura de banda disponível restante para este SLA

Tabela 4 – Descrição dos campos da tabela SLA

- **Tabela: RAR**

Após um pedido ser aceite, as condições do RAR são armazenadas na base de dados. Uma vez que as condições do RAR estão dependentes das especificações do SLA, os campos desta tabela derivam da tabela de SLA. Tal como para a tabela de SLA, existe flexibilidade na sua construção e organização, permitindo modificá-la para melhor servir outro tipo de parâmetros. A grande diferença entre a tabela de RAR e a tabela de SLA é que a primeira contém informação referente aos endereços de origem e de destino. Esta informação é necessária para que o BB consiga determinar a que domínio pertence o pedido efectuado e se é necessário reenviá-lo para outro BB, caso o RAR envolva mais do que um domínio.

Campo	Tipo	Comentário
rar_id (chave primária (PK))	INTEGER	Identificação numérica, designada automaticamente e de forma incremental a cada RAR aceite pelo BB.
service type (tipo de serviço)	VARCHAR	Texto representativo do tipo de serviço *DiffServ* (exemplo: EF, AF, BE)
Startdate	DATE	Data de início do RAR
Starttime	TIME	Hora de início do RAR
Enddate	DATE	Data de expiração do RAR

Endtime	TIME	Hora de expiração do RAR
GivenBW (largura de banda atribuida)	INTEGER	A quantidade de largura de banda atribuída com sucesso ao RAR (em kbps)
Source (origem)	VARCHAR	Texto representativo do endereço IP de origem do fluxo
Destination (destino)	VARCHAR	Texto representativo do endereço IP de destino do fluxo
sla_id	INTEGER	Referencia ao campo sla_id da tabela de SLA que associa o RAR ao SLA respectivo

Tabela 5 – Descrição dos campos da tabela RAR

- **Tabela: 'Passwords' (senhas de identificação)**

Cada 'sla_id' está associado a uma senha de identificação que permite ao BB autenticar os pedidos de cada utilizador. Desta forma, o BB controla quais os utilizadores autorizados a efectuarem pedidos e a que SLAs têm acesso.

Campo	Tipo	Comentário
sla_id (chave primária (PK))	INTEGER	Referência ao campo 'sla_id' da tabela de SLA
password (senha de identificação)	VARCHAR	Texto representativo da senha de identificação correspondente ao SLA em causa

Tabela 6 – Descrição dos campos da tabela 'Passwords'

- **Informação do BB**

- **Tabela: BBPeer**

Os dados da tabela BBPeer são essenciais ao funcionamento do BB, pois é nesta tabela que está contida a informação necessária para iniciar o seu funcionamento. Sempre que é iniciado um novo BB, este recebe uma identificação e os parâmetros necessários para comunicar com os encaminhadores do seu domínio de acordo com os dados contidos nesta tabela. O campo referente ao estado (*"status"*) indica o estado de cada BB (activo ou inactivo). Este campo é necessário para a comunicação inter-domínio pois cada BB verifica o estado dos BBs vizinhos quando determina qual o BB a contactar quando recebe um pedido de reserva de recursos referente a tráfego inter-domínio. Este campo é alterado pelo BB sempre que inicia e termina a sua actividade.

Campo	Tipo	Comentário
BB_id (chave primária (PK))	INTEGER	Identificação numérica atribuída a cada BB
Address (endereço)	VARCHAR	Texto representativo do endereço IP do BB
Domain (domínio)	VARCHAR	Texto representativo do domínio *DiffServ* controlado pelo BB
Status (estado)	VARCHAR	Estado do BB (ON ou OFF)
Port (porto)	INTEGER	Número do porto associado ao BB no endereço IP especificado
pdpPort (porto de PDP)	INTEGER	Número do porto utilizado pela componente PDP do BB para comunicação COPS-PR
GatewayPEPID (identificação do portal PEP)	VARCHAR	Texto representativo da identificação do encaminhador de fronteira do domínio, por omissão

Tabela 7 – Descrição dos campos da tabela BBPeer

- **Tabela: 'Allow' (Permissão)**

É uma tabela de permissões que estabelece uma relação entre os SLAs e os domínios onde estes são autorizados. O BB verifica os valores desta tabela quando processa os pedidos dos utilizadores.

Campo	Tipo	Comentário
Sla_id (chave primária (PK))	INTEGER	Referência ao campo sla_id da tabela SLA
Domain (domínio)	VARCHAR	Texto representativo do domínio *DiffServ*

Tabela 8 – Descrição dos campos da tabela Allow

- **Tabela: 'Codepoint'**

Armazena as relações entre os tipos de serviço definidos nos SLAs com os respectivos valores de *"codepoint"* de *DiffServ*.

Campo	Tipo	Comentário
service_type (tipo de serviço) (chave primária (PK))	VARCHAR	Texto representativo do tipo de serviço *DiffServ* (ex: EF, AF, BE)
DSCP	VARCHAR	Frase de bits representativa do DSCP para o tipo de serviço correspondente

Tabela 9 – Descrição dos campos da tabela Codepoint

- **Informação da Rede**

- **Tabela: 'Capacity' (Capacidade)**

É necessário para o funcionamento do BB que este tenha acesso à informação acerca da largura de banda disponível no seu domínio, desde a largura de banda total disponível à largura de banda disponível após satisfazer pedidos de reserva. Esta informação é armazenada nesta tabela. Para a realização desta dissertação, foi assumido que a largura de banda total para um domínio é determinada pela ligação mais lenta dentro do domínio, pois este será um ponto de estrangulamento da rede. Embora outros pontos da rede possam ter uma maior capacidade em termos de largura de banda, a ligação no ponto de estrangulamento não será capaz de lidar com débitos de tráfego superiores à sua capacidade, resultando no descarte de pacotes e em retransmissões, o que seria contraproducente em termos da gestão da largura de banda do domínio.

Campo	Tipo	Comentário
Domain (domínio)	VARCHAR	Texto representativo do domínio *DiffServ*
Capacity (Capacidade)	INTEGER	Débito de tráfego máximo com o qual que o domínio consegue lidar, sendo restrito pela capacidade da ligação mais lenta do domínio. (Em kbps)

Tabela 10 – Descrição dos campos da tabela 'Capacity'

- **Tabela: 'Domains' (Domínios)**

Quando a topologia de rede considerada envolve domínios múltiplos, surge a necessidade do BB conseguir localizar os seus BBs vizinhos. A informação necessária para que isto seja possível é armazenada na tabela 'Domains'. Pela forma como a arquitectura foi implementada, o BB apenas necessita de contactar os BBs de domínios *DiffServ* vizinhos.

Campo	Tipo	Comentário
Domain (domínio)	VARCHAR	Texto representativo do domínio *DiffServ*
Neighbour (vizinho)	VARCHAR	Texto representativo do domínio *DiffServ* vizinho

Tabela 11 – Descrição dos campos da tabela 'Domains'

- **Tabela: 'Flows' (Fluxos)**

A tabela 'Flows' armazena a informação referente a todos os pedidos válidos em actividade. O BB utiliza esta informação para determinar quando os níveis de recursos necessitam de ser alterados para todos os domínios associados a um determinado RAR.

Campo	Tipo	Comentário
RAR	INTEGER	Referência ao rar_id da tabela RAR
Domain (domínio)	VARCHAR	Texto representativo do domínio *DiffServ*
Bandwidth (largura de banda)	INTEGER	Largura de banda reservada para o RAR
Startdate	DATE	Data de início do RAR
Starttime	TIME	Hora de início do RAR
Enddate	DATE	Data de expiração do RAR
Endtime	TIME	Hora de expiração do RAR

Tabela 12 – Descrição dos campos da tabela 'Flows'

- **Tabela: PEP**

O BB necessita de conhecer os encaminhadores e PEPs do seu domínio, para conseguir enviar a informação necessária para os configurar. Na tabela PEP é armazenada a informação necessária para este fim. Os PEPs são referenciados por uma identificação única, independentemente do domínio a que pertençam. O campo 'sindex' é necessário para indiciar a lista de PEPs para um determinado domínio, pois nesta implementação específica de BB, a lista é armazenada num vector.

Campo	Tipo	Comentário
pepID	VARCHAR	Identificação textual do PEP (encaminhador)
Sindex	INTEGER	Valor numérico para o índice do vector que contém o numero de *socket* TCP correspondente a este PEP
Domain (domínio)	VARCHAR	Texto representativo do domínio *DiffServ*
Neighbour (vizinho)	VARCHAR	Texto representativo do domínio *DiffServ* adjacente a que o PEP está ligado
Status (estado)	VARCHAR	Estado do PEP (ON ou OFF)

Tabela 13 – Descrição dos campos da tabela PEP

4.3 Implementação

O BB é implementado em Java e segue um modelo cliente/servidor. Tem a capacidade de configurar encaminhadores (*"routers"*) baseados em Linux. Esta escolha de sistema operativo fica a dever-se ao facto do Linux possuir ferramentas inerentes ao seu núcleo (*"kernel"*) que fornecem suporte para *DiffServ* e permitir a utilização de uma estação de trabalho como encaminhador virtual. Isto possibilita a implementação dos cenários de teste necessários para a avaliação do desempenho da arquitectura, assim como a sua implementação sem ter de recorrer ao uso de encaminhadores reais. Outra vantagem para o uso de Linux é o facto de o seu código-fonte ser aberto (*"open source"*), o que o torna numa alternativa económica (visto a sua distribuição ser gratuita) e prática, pois é possível personalizar a sua instalação, de forma a optimizar o seu funcionamento especificamente para uma determinada aplicação. É, no entanto, necessário que as máquinas utilizadas tenham o seu suporte de *DiffServ* activo. Esta funcionalidade está incluída no núcleo do Linux desde a versão 2.4. Embora não tenha sido realizado, a arquitectura proposta também funciona com encaminhadores reais desde que se desenvolva uma interface para os programar, por exemplo utilizando comandos IOS. A funcionalidade remota cliente/servidor é obtida através de ligações *"sockets"* TCP. É possível estabelecer várias ligações, tanto a encaminhadores como a clientes, em simultâneo

O BB é uma entidade complexa. A implementação foi realizada em cinco módulos, implementados em separado e posteriormente integrados na arquitectura final. Estes cinco módulos são:

- Servidor do BB (BBServ)
- Comunicação COPS-PR (implementação em Java do COPS-PR);
- BBClient (integrado no procurador SIP);
- PEPClient;
- Base de Dados.

4.3.1 Servidor do BB (BBServ)

O funcionamento do BB é inicializado com o lançamento da classe *BBServ*. Esta classe é responsável pela criação das diferentes *"threads"* que efectuam as funcionalidades necessárias para a operação do BB.

As ligações TCP com os clientes e com os encaminhadores, sobre as quais é utilizado o COPS para a comunicação entre estes e o BB, são suportadas pela linguagem Java através das classes *ServerSocket* e *Socket* incluídas no seu pacote '*net'* [J2SE 2003]. Após a criação de um *"socket"* de servidor através da classe *ServerSocket*, este fica pronto a receber ligações TCP de clientes através do endereço IP e porto definidos na identificação do BB. Sendo necessário que o BB aceite ligações por parte de vários clientes em simultâneo, a classe *BBServ* lança uma nova *"thread" BBMultiServerThread* para cada ligação detectada, que irá lidar com os pedidos do cliente. As ligações TCP com os encaminhadores Linux (PEPs) são feitas através de uma instância adicional da classe *ServerSocket*.

O código do protocolo COPS foi implementado como um pacote em separado, logo a classe *BBServ* apenas inicializa o código para um servidor PDP.

Durante o inicio da sua actividade, o servidor do BB também cria uma *"thread"* temporizada que irá remover automaticamente os pedidos à medida que estes vão expirando. Este processo requer que as suas entradas correspondentes a esses pedidos sejam removidas de todas as tabelas relevantes da base de dados, assim como a reconfiguração dos encaminhadores da rede para que estes deixem de considerar esses fluxos de tráfego nas suas tabelas de encaminhamento. É possível ajustar o intervalo de tempo entre verificações no ficheiro *'autoDelete.java'*, de forma a optimizar o seu funcionamento. Uma vez que cada verificação requer a utilização do processador e gera tráfego adicional no acesso à base de dados, torna-se necessário encontrar um ponto de equilíbrio entre verificações frequentes, que podem levar a uma sobrecarga dos recursos e verificações esporádicas, que levam a que recursos já libertados ainda sejam vistos pelo BB como ocupados. Este processo de optimização pode ser levado a cabo pelo administrador da rede.

Grande parte da actividade do BB baseia-se na informação contida na base de dados, sendo utilizado um servidor *MySQL* para efectuar a gestão dessa informação. Esta implementação utiliza um *"driver"* desenvolvido especificamente para estabelecer a ligação entre o código Java e o servidor *MySQL* [Pham 2003]. Após a inicialização do *"driver"*, o acesso à base de dados é feito utilizando as classes e métodos incluídos no pacote *java.sql*.

4.3.2 Comunicação COPS-PR

A implementação de COPS/COPS-PR utilizada no desenvolvimento desta arquitectura foi originalmente realizada por alunos da Universidade de Nova Gales do Sul [Pham 2003].

A integração entre o código COPS (que utiliza um PIB como repositório de dados) e o BB (que utiliza uma base de dados *MySQL*) é feita através da classe '*IpFilterEntry*' do pacote COPS. Esta classe é utilizada para armazenar os parâmetros necessários à criação de um encaminhador Linux num PIB.

Os parâmetros utilizados para a configuração dos encaminhadores Linux (PEPs) são o endereço IP da fonte e do destino do fluxo de dados e a especificação da acção a efectuar. Existem dois tipos de acção possível: *'add'* (adicionar), utilizada para adicionar uma nova entrada à tabela de encaminhamento do encaminhador; *'del'* (apagar), utilizada para remover entradas da tabela de encaminhamento. No desenvolvimento desta dissertação não houve necessidade de acrescentar novos parâmetros. No entanto, tal poderia ser feito fazendo as necessárias alterações à classe '*IpFilterEntry*' presente no pacote de COPS utilizado.

Para enviar mensagens COPS para um PEP é necessário criar uma instância da classe '*IpFilterEntry*' de forma a fornecer os dados necessários ao PIB. Uma classe adicional, '*RARcops*', foi criada para inicializar o processo de envio de decisões baseadas no protocolo COPS para os PEPs. Sempre que um pedido de reserva recursos ou uma terminação de reserva é aceite pelo servidor do BB, é utilizada a classe '*RARcops*' para enviar os dados de configuração relevantes para os encaminhadores. Estes valores são obtidos a partir da base de dados *MySQL*, sendo armazenados por esta classe no PIB, antes de enviar ao encaminhador a mensagem COPS correspondente.

4.3.3 Interface com o utilizador

Quando um utilizador inicia uma sessão SIP, ao fazer uma chamada VoIP através do procurador, este lança uma *"thread"* em Java que procura estabelecer uma ligação TCP ao BBServ. O porto e o endereço IP utilizados para estabelecer essa ligação são configurados no procurador pelo administrador de rede, tendo sido acrescentada a opção 'Admin' ao menu de opções da interface gráfica do procurador SIP para esse efeito.

Se a ligação for aceite pelo BBServ, o procurador envia-lhe a identificação do SLA assim como a senha de validação. A identificação do SLA e a senha de validação são obtidas pelo procurador no registo do utilizador. Basta ao administrador da rede colocar a entrada correspondente na base de dados, associando o nome do utilizador à identificação do SLA e o

IP à senha ou então utilizando a opção do procurador de executar o *"login"* do utilizador mediante a introdução de uma senha e utilizar os dados introduzidos para efectuar essa mesma associação. Se o BBServ aceitar o *"login"* como correcto, então o procurador envia os parâmetros requeridos para a configuração do BB num formato de mensagem aceite pelo BBServ. Na arquitectura original, eram enviadas a data e a hora de início e de fim da reserva de largura de banda. Na arquitectura proposta nesta dissertação, são enviadas a data e a hora actual incrementada de um minuto (este será o prazo de validade do RAR). A classe *cRAR*, responsável originalmente pela extracção dos dados inseridos pelo utilizador na interface de texto e pela composição das mensagens a enviar ao BB, foi modificada para obter esses dados a partir do cabeçalho da mensagem SIP de início de sessão e do registo do utilizador no procurador.

Durante o período de duração da chamada (até ser detectada uma mensagem de fim de sessão), o procurador vai enviando pedidos idênticos ao primeiro (excepto pela hora de fim) periodicamente. O período de envio é de trinta segundos, para que o novo pedido chegue ao BBServ antes do anterior expirar, quando a hora actual for superior à hora final. Outra alteração efectuada é que quando o BBServ recebe um pedido do mesmo utilizador para o mesmo fluxo, este substitui o pedido antigo pelo mais recente na base de dados, actualizando o RAR. Quando a sessão acaba, é enviado um pedido desactualizado, isto é, com a hora final inferior à hora actual. O novo RAR substitui o antigo, sendo apagado em seguida, pois os RARs expirados são apagados automaticamente.

O procurador não envia a quantidade de largura de banda desejada para o BBServ. Ao invés disso, o BBServ verifica qual o SLA associado ao utilizador e atribui-lhe a largura de banda máxima possível para esse utilizador. Esta abordagem tem como vantagem manter o processo de pedido de recursos transparente ao utilizador. De outra forma, o utilizador teria de introduzir manualmente a largura de banda desejada nalgum tipo de interface. No entanto, pode resultar num desperdício de recursos, caso a largura de banda a ser reservada, acordada no SLA seja superior à largura de banda utilizada.

O uso de *"threads"* permite que vários utilizadores possam aceder ao procurador SIP simultaneamente, possibilitando-lhe exercer as funções de BBClient sem que haja interferência com as suas funcionalidades originais.

4.3.4 PEPClient

A implementação do PEPClient é feita utilizando a classe *PEPClient*, sendo necessário executar uma instância desta classe em cada máquina destinada a funcionar como encaminhador.

Quando o encaminhador inicia a sua actividade necessita de se inicializar a si próprio para operar segundos os critérios de *DiffServ*. Para implementar o ambiente *DiffServ* pretendido, é utilizada a seguinte série de comandos *tc*:

```
  ### //  Definição da largura de banda total de saída
  tc qdisc add dev eth0 root handle 1: htb default 50

  ### // Definição da hierarquia de suavizadores de tráfego HTB
  tc class add dev eth0 parent 1: classid 1:1 htb rate 10mbps ceil 10mbps
  tc class add dev eth0 parent 1:1 classid 1:10 htb rate 20kbps ceil 20kbps
  tc class add dev eth0 parent 1:1 classid 1:20 htb rate 5mbps ceil 10mbps
  tc class add dev eth0 parent 1:1 classid 1:30 htb rate 2mbps ceil 10mbps
  tc class add dev eth0 parent 1:1 classid 1:40 htb rate 980kbps ceil 10mbps
  tc class add dev eth0 parent 1:1 classid 1:50 htb rate 2mbps ceil 10mbps

  ### // Definição das filas individuais como SFQ
  tc qdisc add dev eth0 parent 1:10 handle 100: sfq perturb 10
  tc qdisc add dev eth0 parent 1:20 handle 200: sfq perturb 10
  tc qdisc add dev eth0 parent 1:30 handle 300: sfq perturb 10
  tc qdisc add dev eth0 parent 1:40 handle 400: sfq perturb 10
  tc qdisc add dev eth0 parent 1:50 handle 500: sfq perturb 10

  ### // Definição de filtros tc para diferentes flowid
  tc filter add dev eth0 protocol ip parent 1:0 prio 1 handle 1 fw flowid 1:10
  tc filter add dev eth0 protocol ip parent 1:0 prio 1 handle -j DSCP --dscp EF
fw flowid 1:20
  tc filter add dev eth0 protocol ip parent 1:0 prio 1 --dscp AF1 fw flowid 1:30
```

```
tc filter add dev eth0 protocol ip parent 1:0 prio 1 --dscp AF2 fw flowid 1:40

tc filter add dev eth0 protocol ip parent 1:0 prio 1 --dscp BE fw flowid 1:50
```

Esta série de comandos foi incorporada no código do BB. Mais concretamente, foi criada a classe *LinuxRoute* que, através do método '*setupDiff()*', executa estes comandos na consola através da classe *Runtime* do Java. Um exemplo do uso desta função é:

```
String DiffServ="tc qdisc add dev eth0 root handle 1: htb default 50";

Runtime.getRuntime().exec(DiffServ);
```

A classe *Runtime* permite a uma aplicação Java interagir com o ambiente onde esta de encontra a correr. Neste exemplo é executado o comando representado pela *"string"*'*DiffServ*' como se este tivesse sido introduzido manualmente numa consola Linux. Desta forma, o PEPClient acede à tabela de encaminhamento e às suas condições de controlo de tráfego.

Quando um PEPClient inicia a sua actividade é iniciada uma instância da classe *LinuxRoute* e é chamado o método *setupDiff()*. Desta forma, é configurado o ambiente *DiffServ* na máquina onde está a ser executado o PEPClient. O PEPClient recorre também à classe *LinuxRoute* sempre que surja a necessidade de proceder a alterações na tabela de encaminhamento, ou seja, sempre que necessite de introduzir ou apagar uma entrada na tabela.

O código Java do ficheiro *LinuxRoute.java* (mais concretamente o código referente ao método *setupDiff()*) pode ser alterado conforme as características do domínio *DiffServ* que se pretende implementar.

O pacote de COPS utilizado fornece uma classe para uso como uma implementação de PEP denominada *CopsprPepImpl* e é através desta classe que é efectuada a comunicação COPS com o BBServ, que actua como PDP. A informação de configuração em vigor é enviada pelo BBServ ao PEPServ através de uma mensagem COPS, do tipo 'Decision'. Esta informação é processada utilizando o método *processDEC()*, segundo o código de comando contido na mensagem.

É possível adaptar o PEPClient para qualquer acção de controlo de tráfego ou de encaminhamento desejada, através da alteração da classe *LinuxRoute.*.

4.3.5 Captura de pacotes SIP

A captura de pacotes SIP é feita pelo procurador SIP inserido na arquitectura. Através da análise do cabeçalho das mensagens SIP, o procurador determina se é necessário estabelecer uma nova reserva de largura de banda ou terminar uma reserva realizada anteriormente. Para tal, o procurador verifica o campo do cabeçalho relativo ao tipo da mensagem. Se for do tipo 'INVITE', o procurador lança uma *"thread"* associada a uma nova sessão iniciada pelo utilizador. Verifica então os campos relativos à identificação do utilizador (endereço IP) e dos pontos de origem e destino do fluxo de dados pretendido para a sessão. Com estes dados é possível ao procurador compor um RAR válido para a sessão iniciada. Mas antes de enviar o pedido ao BBServ, o procurador aguarda por uma mensagem do tipo '220_OK' por parte do receptor. Desta forma, garante que o pedido só é feito caso a sessão seja estabelecida com sucesso. Após a recepção da mensagem '200_OK', o procurador envia então o RAR ao BBServ e aguarda pela resposta deste. Caso o RAR seja aceite e validado pelo BBServ, o procurador recebe uma mensagem de confirmação. A partir deste ponto, o procurador vai renovando o RAR para manter a reserva de recursos activa durante a sessão, ou seja, até interceptar uma mensagem do tipo 'BYE' enviada por um dos intervenientes da comunicação. Isto significa que a reserva de largura de banda deixou de ser necessária, podendo ser terminada através do envio de um RAR com data e hora de início e de fim iguais à data e hora de encerramento da sessão SIP.

Alternativamente ao uso de um procurador SIP modificado, foi considerada a hipótese de implementar um interceptor/analisador de pacotes baseado em 'Jpcap'. O 'Jpcap' é uma biblioteca em Java com métodos que possibilitam a captura e envio de pacotes de rede. Com esta biblioteca é possível desenvolver aplicações em Java para capturar pacotes que circulem através de uma interface de rede e visualizar/analisar o seu conteúdo. Esta biblioteca foi testada em Linux (Fedora, Mandriva, Ubuntu), MAC OSX (Darwin), Microsoft Windows (98/2000/XP/Vista), FreeBSD e Solaris, ficando demonstrada a sua compatibilidade com estas plataformas. O 'Jpcap' é licenciado em código aberto segundo o GNU LGPL e é possível descarregar esta biblioteca assim como obter informações e um tutorial acerca do seu uso em [Jpcap 2007].Ambas as opções são gratuitas e compatíveis com diversos sistemas operativos (devido à sua implementação em Java). No entanto, o interceptor/analisador de pacotes implicaria a implementação de um mecanismo de encaminhamento de pacotes que seleccionasse o tráfego na rede, de forma a redireccionar os pacotes SIP para a aplicação. A alternativa seria interceptar todos os pacotes circulantes na rede. Quanto à utilização de um

procurador, embora possa ser implementado um mecanismo para o encaminhamento do tráfego SIP para o procurador, este pode ser substituído pelo registo dos utilizadores no procurador quando activam a sua aplicação telefónica. Uma vez que esta é uma operação simples e com a qual muitos utilizadores de VoIP estão familiarizados, não foi considerado que interferisse no conceito de transparência do processo para o utilizador, defendido no âmbito desta dissertação. Adicionalmente, o utilizador pode usufruir dos serviços normais fornecidos por um procurador SIP, pois todo o processo de estabelecimento e manutenção da reserva de recursos é feito paralelamente ao funcionamento normal do procurador, graças à utilização de *"threads"*. Ao comparar as duas soluções, optou-se pela solução composta pelo uso de um procurador SIP implementado em Java e licenciado em código aberto.

4.3.6 Implementação de domínios *DiffServ*

Para efectuar os testes é necessária a implementação de um ou mais domínios *DiffServ*, dependendo do tipo de cenário (intra-domínio ou inter-domínio). O sistema operativo Linux possui funcionalidades inerentes que permitem o estabelecimento de domínios *DiffServ*, razão pela qual foi escolhido como plataforma de suporte para a arquitectura implementada. Utilizando a ferramenta '*tc'* é possível criar e configurar funcionalidades de suavização de tráfego e de reserva de largura de banda, que permitem estabelecer um domínio *DiffServ* adequado às necessidades dos testes. Esta ferramenta faz parte do pacote 'iproute2'e faz parte do Fedora Core 6.

O HTB encontra-se incluído como módulo no *"kernel"*do Linux desde a versão 2.4.20 mas a sua utilização pode necessitar de uma recompilação do *"kernel"* caso não esteja activo por omissão. No caso da distribuição do Linux utilizada (Fedora Core 6), foi necessário activar o módulo 'sch_htb' através do comando:

```
# modprobe sch_htb
```

Para carregar o módulo automaticamente no início do sistema é necessário adicionar a linha acima ao ficheiro '/etc/rc.d/rc.local' ou então ao ficheiro '/etc/rc.d/rc.modules'.

O modelo de fila de espera escolhido foi o de repositório de testemunhos hierárquico (HTB – *"Hierarchic Token Bucket"*) [HTB 2003]. Através do HTB é possível definir a forma como a largura de banda é distribuída pelos diversos fluxos de dados que compõem o tráfego

na rede. Esta partilha é feita de modo hierárquico, estabelecendo diferentes ordens de prioridade para tipos de tráfego distintos. Desta forma, é possível distribuir a largura de banda disponível por várias classes de serviço, garantindo que cada classe recebe largura de banda necessária para desempenhar as funções a que é destinada sem problemas (latência, *"jitter"*, perda de pacotes).

No modelo definido, são utilizadas duas placas de rede cuja capacidade máxima de débito de tráfego é de 10Mbits. Este valor é o limite da largura de banda máxima disponível na rede considerada, na qual estes elementos servem de ponto de estrangulamento. Mesmo que os outros elementos da rede possuam uma maior capacidade, não é possível garantir QoS para débitos de tráfego superiores à capacidade máxima do elemento com menor débito.

Estabelecendo o limite máximo de largura de banda disponível em 10Mbps, definiu-se então a forma como esta seria distribuída pelas classes de serviço desejadas. Para este protótipo foram definidas as seguintes classes de serviço:

- **Controlo** – Esta classe de serviço foi criada para garantir o que a troca de pacotes entre BBs, procurador SIP e encaminhadores da rede é feita sem latência ou perdas, independentemente das reservas de recursos em vigor. Esta classe abrange também os pacotes de controlo de sessão SIP. Em testes feitos durante a implementação da arquitectura, verificou-se a necessidade de garantir largura de banda para este tipo de tráfego, pois em situações de sobrecarga o utilizador ficava impossibilitado de estabelecer ligação com o procurador SIP, caso o ponto de estrangulamento da rede se encontrasse entre os dois. Isto levava a que o utilizador não conseguisse estabelecer ligação com o BB para efectuar o pedido de reserva de recursos. O mesmo problema ocorria na comunicação entre BBs e com o procurador SIP, caso esta se efectuasse através do ponto de estrangulamento. A criação deste serviço foi adoptada como solução para este problema;

- **EF** – Esta classe de serviço foi criada para fornecer uma garantia de baixa latência, baixo *"jitter"* e perdas reduzidas. Estas características são indicadas para tráfego de voz, vídeo ou outros serviços de tempo real. É a classe com maior prioridade , a seguir à classe de controlo e é aquela que possui os requisitos de QoS mais elevados;

- **AF1**, e **AF2** – São duas classes de serviço com prioridade e requisitos de QoS, sendo AF1 superior a AF2 em ambos os aspectos;

- **BE** (melhor esforço) – Dentro desta classe enquadra-se todo o restante tráfego que não é abrangido pelas classes anteriores. Não fornece garantia de QoS, procurando satisfazer as necessidades dos utilizadores de acordo com os recursos disponíveis em determinado instante. O tráfego gerado por utilizadores que não têm acesso a nenhum SLA em vigor também é incluído nesta classe.

Foi atribuída a cada classe de serviço uma parcela da largura de banda correspondente à sua prioridade. Esta atribuição é feita segundo determinados limites mínimos e máximos. O limite mínimo define a largura de banda garantida para cada classe, independente da carga na rede e a soma dos limites mínimos corresponde à capacidade total da ligação. A largura de banda que não é utilizada por cada classe é redistribuída pelas restantes classes, optimizando assim o aproveitamento dos recursos da rede. Esta redistribuição é definida pelo limite máximo. Na figura seguinte são apresentadas as classes de serviço e os seus limites máximos e mínimos.

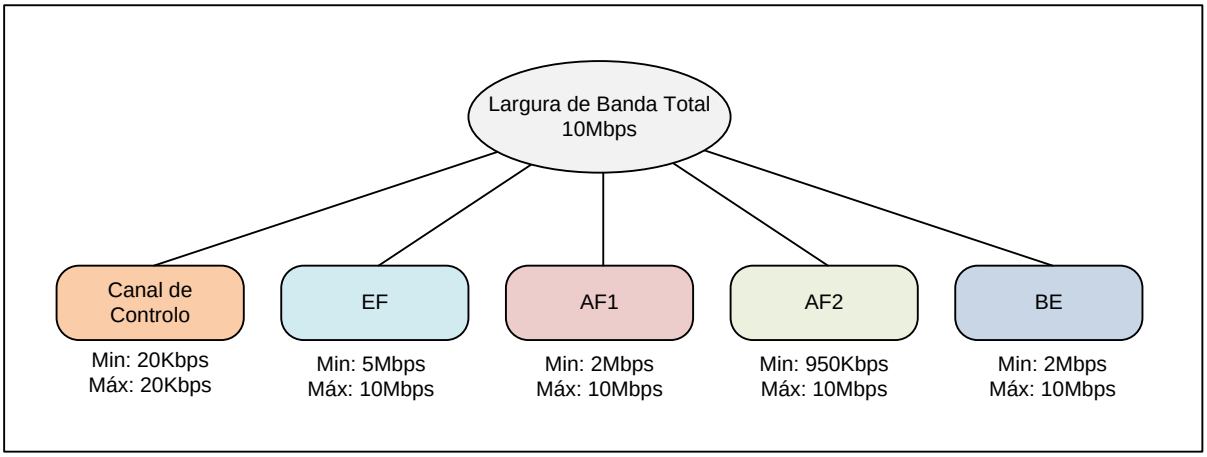

Figura 4.7 – Distribuição de largura de banda pelas classes de serviço

CAPÍTULO 5.

RESULTADOS

5.1 Cenário de Testes

Para realizar os testes à arquitectura foi montado um cenário utilizando uma LAN composta por dois computadores e com uma largura de banda máxima disponível de 10Mbits/s. Este limite foi obtido através da utilização de uma placa rede Realtek RTL8029 com capacidade máxima de 10 Mbit/s instalada no computador 1. Esta placa desempenhou o papel de ponto de estrangulamento na rede, uma vez que a placa instalada no computador 2 tem uma capacidade máxima de 100 Mbit/s. Como a largura de banda máxima que se pode garantir numa rede é determinada pelo nó de menor capacidade, conseguiu-se assim definir o limite de 10 Mbit/s desejado para os testes. O *"codec"* de áudio utilizado foi o G.711, que , como foi apresentado na secção 2.4, utiliza um ritmo binário médio de 64 kbit/s. A classe de serviço atribuída ao tráfego VoIP de teste para ambos os utilizadores foi EF (segundo a configuração apresentada na secção 4.3.6.

Utilizando esta rede foi possível avaliar o comportamento da arquitectura implementada tanto em cenários de comunicação dentro de um domínio *DiffServ* comum a ambos os intervenientes (cenário intra-domínio) como em cenários de comunicação entre domínios (cenário inter-domínio). Em ambos os casos foram feitos testes em condições ideais (sem

tráfego adicional na rede) e em condições extremas (com tráfego intenso na rede). Mais concretamente, foram consideradas três situações de teste para cada cenário:

- **Teste 1** – Foi estabelecida a ligação telefónica VoIP entre dois utilizadores, sem qualquer tráfego adicional na rede e sem a utilização da arquitectura implementada. O objectivo deste teste foi obter uma referência de comparação.

- **Teste 2** – Ainda sem a utilização da arquitectura implementada, foi estabelecida a ligação VoIP entre os dois utilizadores mas agora com a rede sobrecarregada com tráfego UDP.

- **Teste 3** – Foi estabelecida a ligação VoIP entre os dois utilizadores, agora sobre a arquitectura implementada e com a rede sobrecarregada com tráfego UDP.

Para gerar o tráfego intenso necessário para sobrecarregar a ligação foi utilizada uma aplicação de *software* que consiste num emissor (*'gen_send'*) e num receptor (*'gen_recv'*) de pacotes UDP. O emissor gera pacotes UDP de tamanho configurável, enviando-os para o endereço IP/porto do receptor. O ritmo de envio de pacotes é também configurável, de forma a obter a intensidade de tráfego desejada. Esta aplicação foi desenvolvida pelo Centro de Integração de Tecnologias de Informação (CITI – *"Center for Information Technology Integration"*) da Universidade do Michigan [CITI 2008], com a vista a testar a funcionalidade de QoS em redes. O motivo pelo qual se optou pela utilização de tráfego UDP para sobrecarregar a rede, foi o facto de este protocolo não reduzir o ritmo de pacotes quando ocorrem perdas, ao contrário do TCP.

No cenário de testes intra-domínio, foi considerado que ambos os computadores operavam no mesmo domínio *DiffServ*. É possível observar um esquema deste cenário na Figura 5.1. O computador 1, equipado com a placa de 10Mbit/s, desempenhou as funções de encaminhador e de um dos extremos na comunicação VoIP. Foi também nesta máquina que foram lançadas as aplicações referentes ao BB (BBServ e PEPClient), onde foi armazenada a respectiva base de dados. O procurador SIP, que estabelece a interface entre os utilizadores do domínio e o respectivo BB também foi executado nesta máquina. O computador 2 serviu como segundo terminal da ligação VoIP.

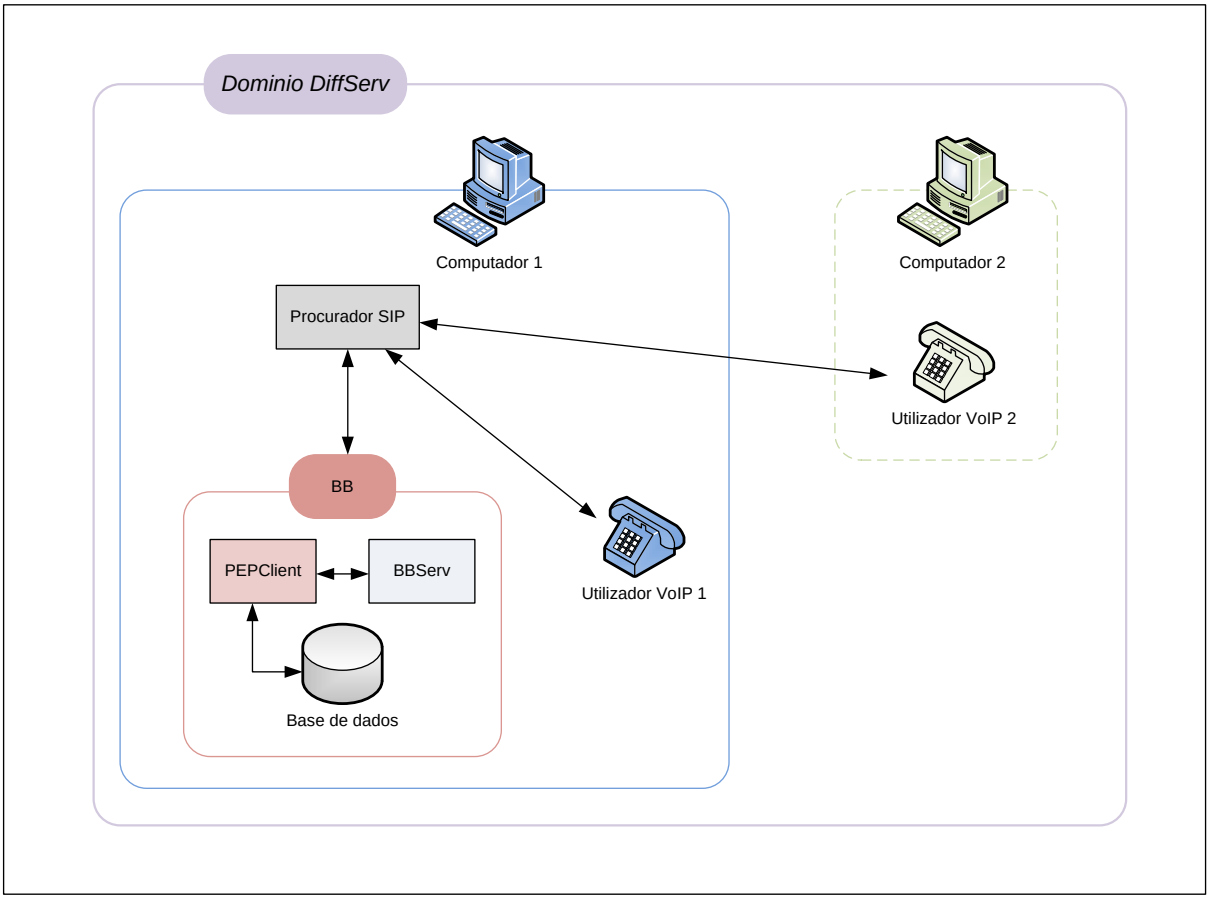

Figura 5.1 – Cenário de testes intra-domínio

Na situação de tráfego intenso na rede, foram realizados testes alternando o sentido do tráfego adicional com o sentido da ligação telefónica e utilizando tráfego adicional em ambos os sentidos em simultâneo, de forma a determinar quais os efeitos no desempenho da comunicação.

Para o cenário de comunicação inter-domínio, foram considerados três domínios *DiffServ*. Um dos domínios funcionava no computador 2 (domínio 1) enquanto os outros dois funcionavam no computador 1 (domínio 2 e 3). Neste teste foi necessário correr uma instância do BB para cada domínio. Uma consideração adicional foi que o domínio 1 e o domínio 3 não são vizinhos embora sejam ambos vizinhos do domínio 2. Desta forma, para estabelecer uma reserva de recursos entre os domínios 1 e 3 é necessário envolver o BB do domínio 2. Um esquema deste cenário é apresentado na Figura 5.2.

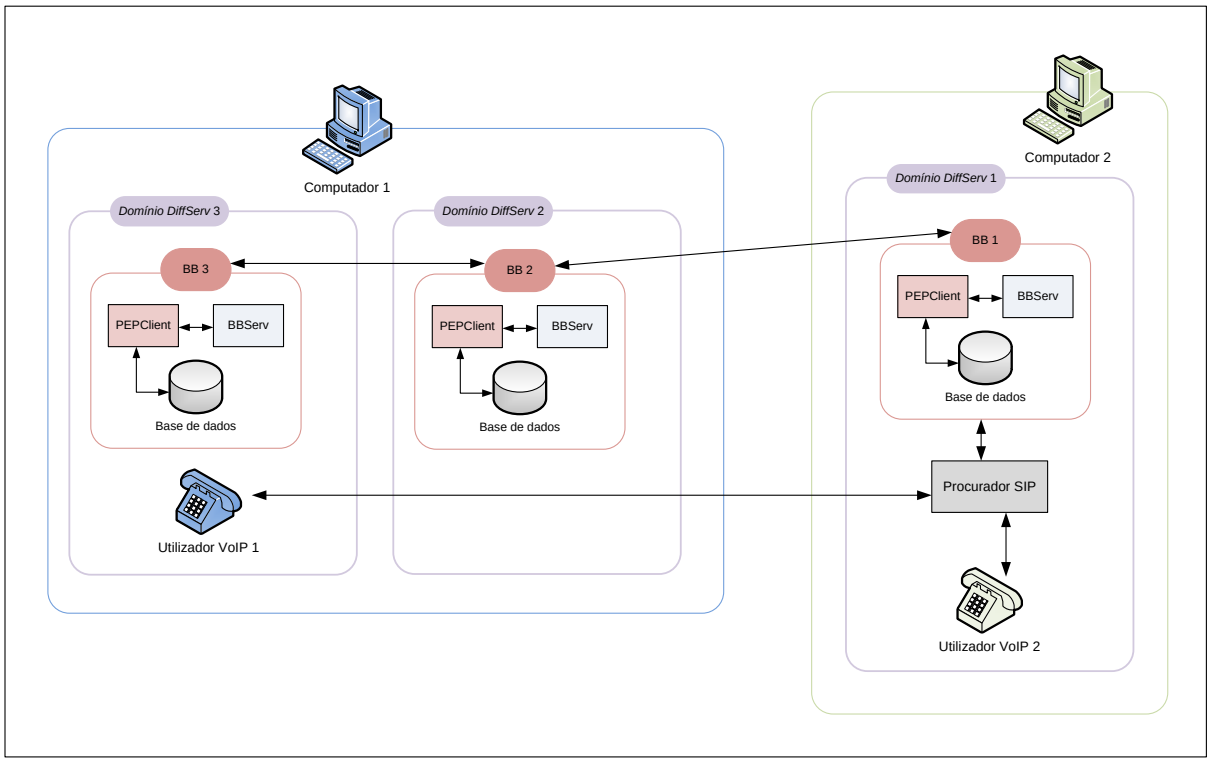

Figura 5.2 – Cenário de testes intra-domínio

Tal como para o cenário intra-domínio, na situação de tráfego intenso foram testadas as várias configurações de tráfego adicional possíveis de forma a verificar os seus efeitos no desempenho da rede.

A experiência levada a cabo e repetida para todos os cenários de teste considerados consistiu no estabelecimento de uma chamada telefónica VoIP entre dois utilizadores em máquinas distintas. Esta chamada foi efectuada utilizando um procurador SIP como mediador. A experiência foi repetida alternando o papel de emissor e receptor entre os dois utilizadores. Os resultados dos ensaios levados a cabo foram recolhidos utilizando a aplicação *Wireshark*, um analisador de protocolos de rede.

O desempenho da comunicação estabelecida durante os ensaios de teste foi avaliado segundo os seguintes critérios de QoS:

- Latência – Este é o tempo que um pacote demora no percurso entre dois terminais de uma ligação. Devido às características físicas do meio de transporte e à capacidade de processamento de dispositivos intermédios na rede, é esperada sempre alguma latência na chegada de pacotes mas este deve ser minimizada. Uma vez que a audição humana tem uma tolerância normal à latência de cerca de 150 milissegundos, a norma ITU G.144 recomenda que o atraso numa comunicação de voz não exceda esse valor.

- *"Jitter"*–. O *"jitter"* consiste na variação do atraso ao longo de uma comunicação. Embora não afecte chamadas em redes de PSTN (cuja largura de banda é fixa), este fenómeno pode ser problemático em redes VoIP, em que o tráfego se pode processar em rajadas. Uma possível solução para este problema é a utilização de armazenamento (*"buffering"*).

- Perda de pacotes – Numa comunicação de voz é importante que os pacotes enviados cheguem ao seu destino de forma ordenada para não distorcer a comunicação. O principal efeito da perda de pacotes é introdução de falhas no áudio na recepção.

5.2 Resultados dos Testes

Os gráficos apresentados nesta secção foram elaborados com base nos resultados recolhidos com o auxílio do *Wireshark* com durante as experiências levadas a cabo. Na sua elaboração foi considerada uma amostra sequencial de 1000 pacotes para cada teste.

5.2.1 Cenário Intra-Domínio

Começou-se por analisar o funcionamento da arquitectura no cenário de funcionamento mais simples - o cenário intra-domínio em que ambos os utilizadores se encontram inseridos do mesmo domínio *DiffServ*. Por esse motivo utilizou-se apenas uma instância do BB e do procurador SIP.

5.2.1.1 Teste 1

No teste 1 pretendeu-se obter uma referência de comparação para determinar a qualidade do desempenho da arquitectura implementada. O teste foi executado com as duas máquinas ligadas directamente através de um cabo cruzado e sem tráfego adicional na rede. Desta forma, os resultados obtidos foram os correspondentes a uma situação ideal, sendo este o desempenho desejado (ou o mais próximo possível) para uma situação de uma chamada VoIP com tráfego intenso em paralelo, sobre a arquitectura implementada.

O resultado foi o esperado. Foi possível estabelecer a ligação sem problemas e a comunicação de voz não apresentou latência ou interferências audíveis. Também não se registaram perdas de pacotes.

5.2.1.2 Teste 2

O Teste 2 também foi efectuado para se obter uma referência de comparação mas, desta vez, para uma situação extrema de tráfego intenso na rede. O modelo de rede foi o mesmo utilizado no Teste 1 mas desta vez com o gerador de pacotes UDP a funcionar em paralelo. Inicialmente tentou-se estabelecer a ligação com o gerador de pacotes já a correr mas isto revelou-se extremamente difícil pois os pacotes de início de sessão SIP apresentaram taxas de perdas muito elevadas quando o destino este se encontrava a jusante do ponto de estrangulamento, com o computador 2 a iniciar a chamada. Na maior parte dos ensaios, o telefone de *"software"*utilizado esgotava as tentativas de ligação sem conseguir estabelecê-la com sucesso. Experimentou-se inverter o sentido da comunicação utilizando o computador 1 a iniciar a chamada. Desta forma, o computador 1 recebia o pedido de início de chamada e activava o toque indicativo da mesma mas ao atender não era possível estabelecer a ligação pois o comutador 2 não recebia a mensagem de aceitação da chamada. Chegou-se então à conclusão que a melhor forma de obter os resultados desta experiência seria iniciar a ligação e posteriormente activar o gerador de pacotes. Assim sendo, foi possível recolher os dados experimentais desejados com o *Wireshark*. Em relação ao desempenho, este começou a decair imediatamente após a activação do gerador de pacotes, registando-se inicialmente falhas na recepção áudio, com cortes cuja duração foi aumentando conforme o aumento da carga na rede, até ao ponto da recepção de som se tornar nula exceptuado alguns fragmentos esporádicos de áudio.

5.2.1.3 Teste 3

Com este teste pretendeu-se avaliar o desempenho da arquitectura implementada. A chamada foi estabelecida com sucesso mesmo com o gerador de pacotes em funcionamento. Durante a chamada não se verificaram falhas ou latência audíveis na recepção de áudio.

5.2.1.4 Análise de Resultados

Os gráficos apresentados nesta secção referem-se à experiência efectuada utilizando o computador 2 a iniciar a chamada telefónica e com tráfego intenso nos dois sentidos. Os gráficos que se referem a "tráfego directo" correspondem ao tráfego composto pelo fluxo da chamada VoIP efectuada, cujo sentido vai do emissor para o receptor. Embora a comunicação seja bidireccional e ambos os utilizadores enviem e recebam pacotes, considerou-se como emissor o utilizador que inicia a chamada. Analogamente, os gráficos do "tráfego inverso" correspondem ao fluxo do receptor para o emissor. Registaram-se algumas diferenças no comportamento da comunicação entre o "tráfego directo" e o "tráfego inverso". Invertendo o sentido da comunicação verificou-se que os resultados obtidos eram similares aos registados anteriormente mas agora o "tráfego directo" apresentava o comportamento do "tráfego inverso" e vice-versa. Uma vez que o canal de comunicação é afectado de igual forma em ambos os sentidos da comunicação, esta diferença pode ficar a dever-se ao *"hardware"* utilizado, em especial a placa de rede utilizada como ponto de estrangulamento no computador 1, pois foram registados piores desempenhos quando esta foi utilizada para recepção.

- **Latência**

Na Figura 5.3 pode comparar-se o comportamento médio do atraso registado durante os três testes, tanto para o "tráfego directo" como para o "tráfego inverso". A análise deste gráfico não é conclusiva pois os valores nele apresentados não revelam a diferença registada na qualidade da chamada durante os testes efectuados. Embora os valores registados durante o Teste 2 sejam mais elevados que nos outros testes, a diferença é muito pouca, tendo em conta que se observou uma extrema degradação na qualidade da chamada durante este ensaio. De facto, com a sobrecarga de tráfego na rede, apenas se registaram alguns fragmentos de áudio durante a chamada.

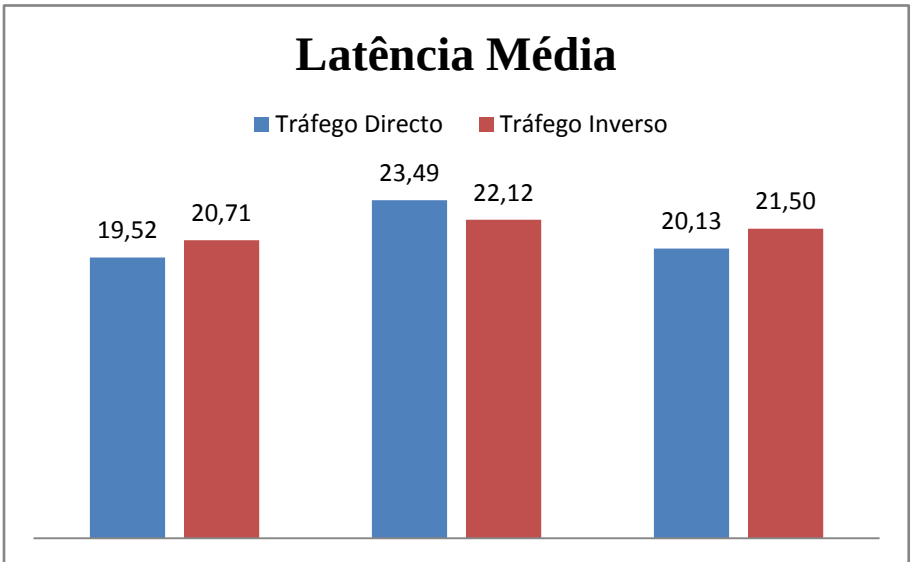

Figura 5.3 - Comparação da latência média nos três testes (em milissegundos)

Na Figura 5.4 é possível observar uma representação instantânea da latência para cada pacote. É possível constatar que, apesar de em média os valores registados no Teste 2 serem semelhantes aos dos outros dois testes, registaram-se picos em que a latência é muito superior à média.

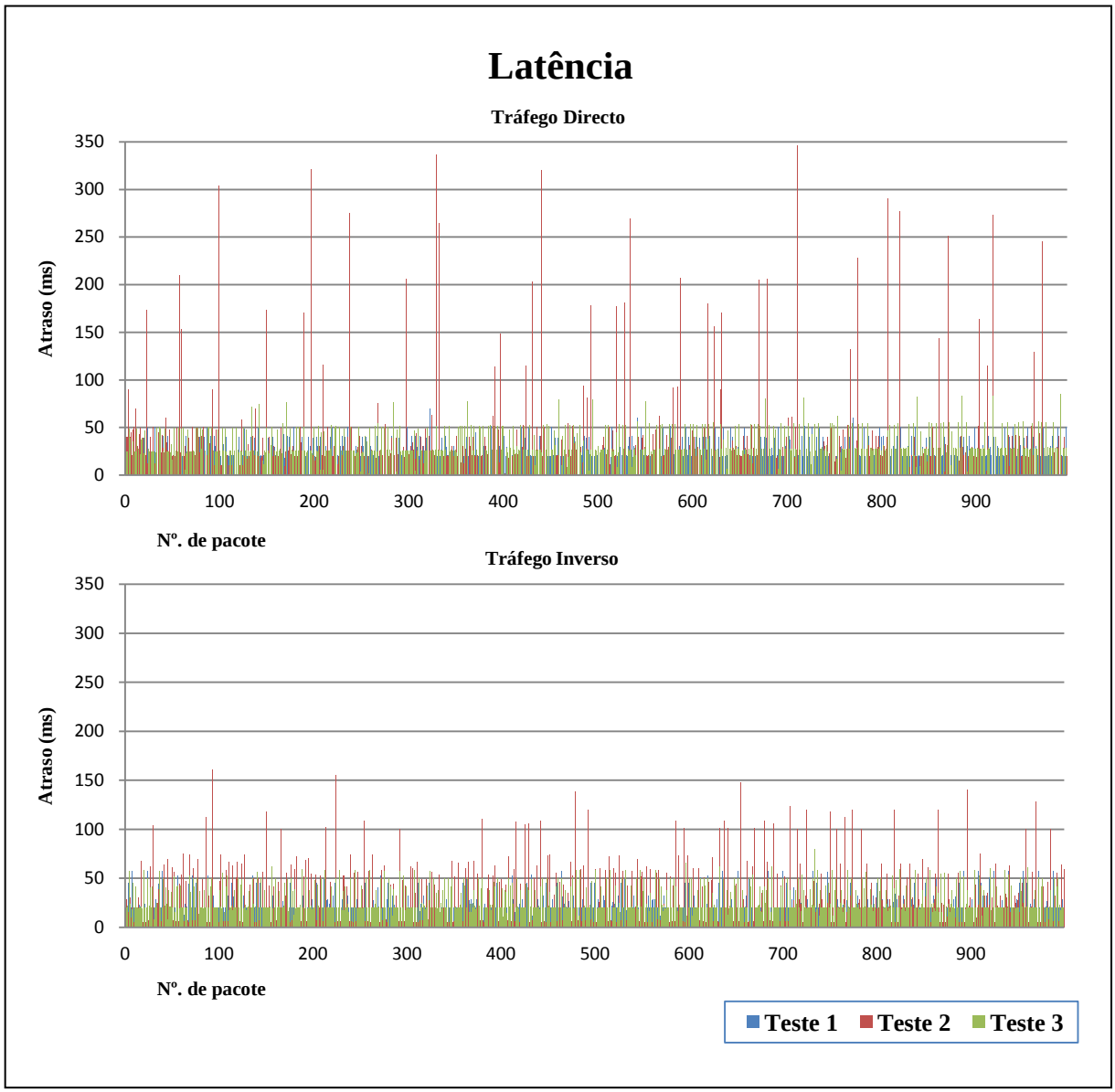

Figura 5.4 - Gráfico da latência do tráfego nos três testes

No entanto, para o "tráfego directo" estes picos apresentam uma maior amplitude.

- ***"Jitter"***

É possível observar nos gráficos referentes ao *"jitter"*, tanto para o "tráfego directo" como "inverso", que no teste 2 os valores foram superiores aos registados nos outros dois testes.

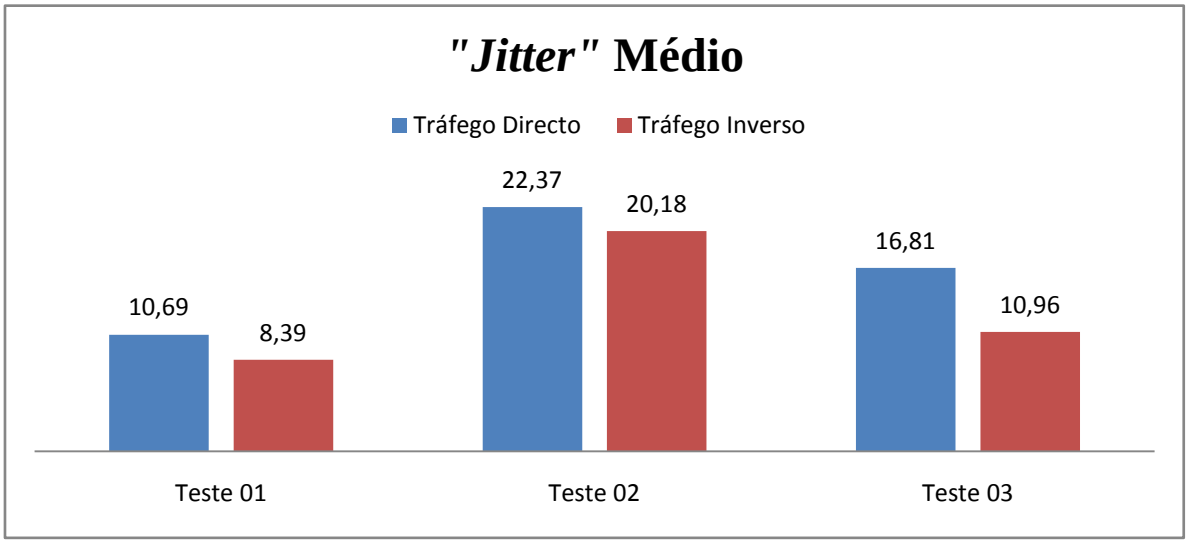

Figura 5.5 - Comparação do *"jitter"* médio (em milissegundos)

Tal como para a latência, o comportamento observado para o "tráfego directo" apresenta algumas diferenças relativamente ao "tráfego inverso", tendo sido registados picos mais pronunciados para o primeiro caso. Isto pode ser observado na Figura 5.6.

Tal como era esperado, o teste 2 apresenta picos bastante elevados em relação à referência obtida no teste 1, assim como um valor médio bastante mais elevado. Em relação ao teste 3, notou-se uma melhoria significativa no desempenho da comunicação, embora não se tenham alcançado os valores do teste de referência. No entanto, como já foi referido anteriormente, a comunicação processou-se sem se registarem falhas audíveis durante o teste 3, o que leva a crer que os valores registados se encontram dentro do exigido para se obter uma QoS aceitável relativamente às exigências do VoIP, o que leva a crer que os valores registados se encontram dentro do exigido para se obter uma QoS aceitável relativamente às exigências do VoIP.

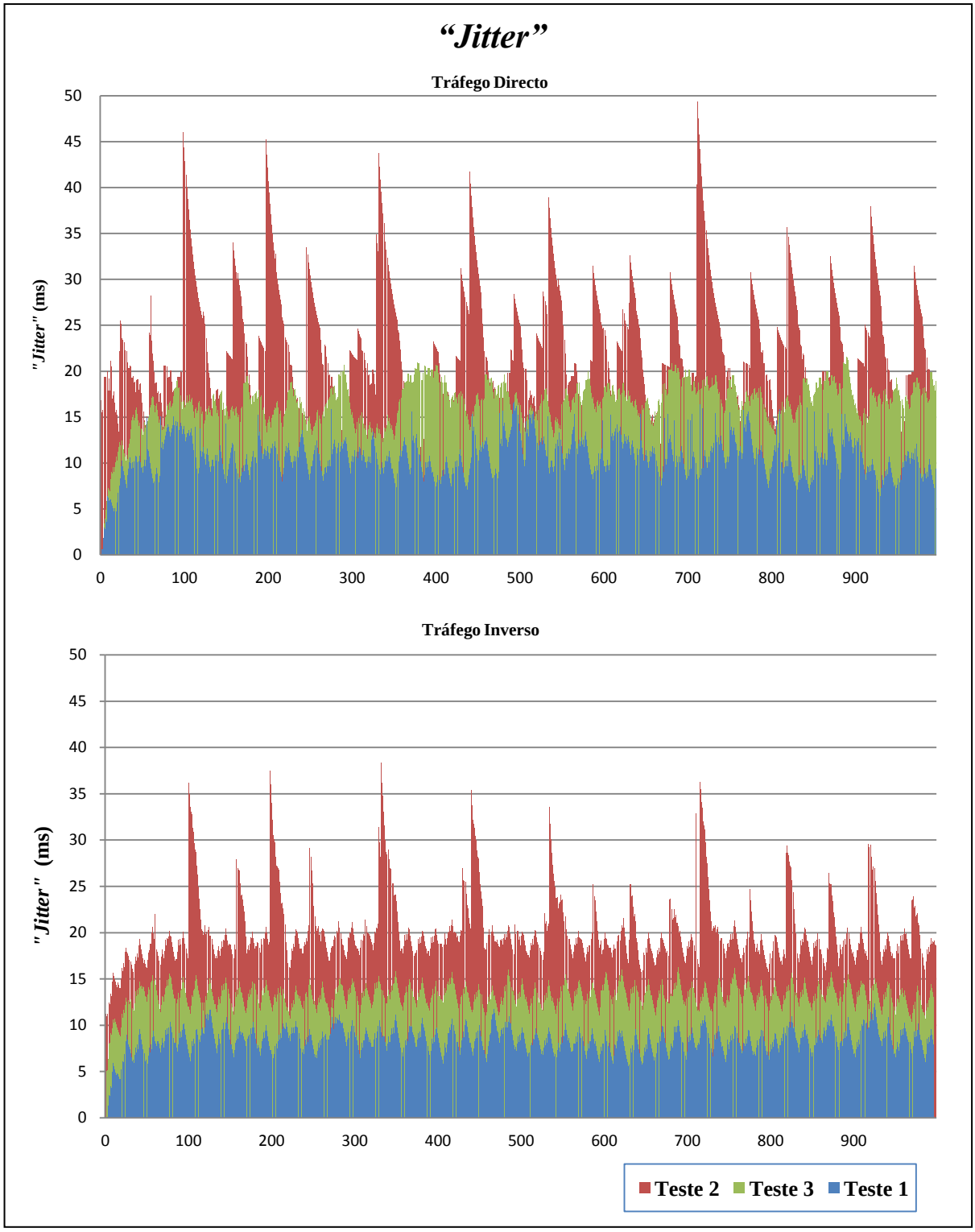

Figura 5.6- Gráfico do *"jitter"* do tráfego nos três testes

- **Perda de pacotes**

Quanto à perda de pacotes, é possível constatar, a partir da observação do gráfico abaixo, que o teste 2 apresenta a maior taxa de perda de pacotes relativamente aos outros ensaios. No teste 1 não se registou qualquer perda, enquanto para o teste 3 registaram-se algumas perdas de pacotes (0,28%). No entanto, estas perdas não influenciaram de forma perceptível a qualidade do áudio durante a comunicação. Durante o teste 2, a comunicação apresentou-se extremamente deteriorada, sendo apenas registados alguns sons esporádicos em ambos terminais da mesma, isto apesar da taxa de perda de pacotes não ser muito elevada.

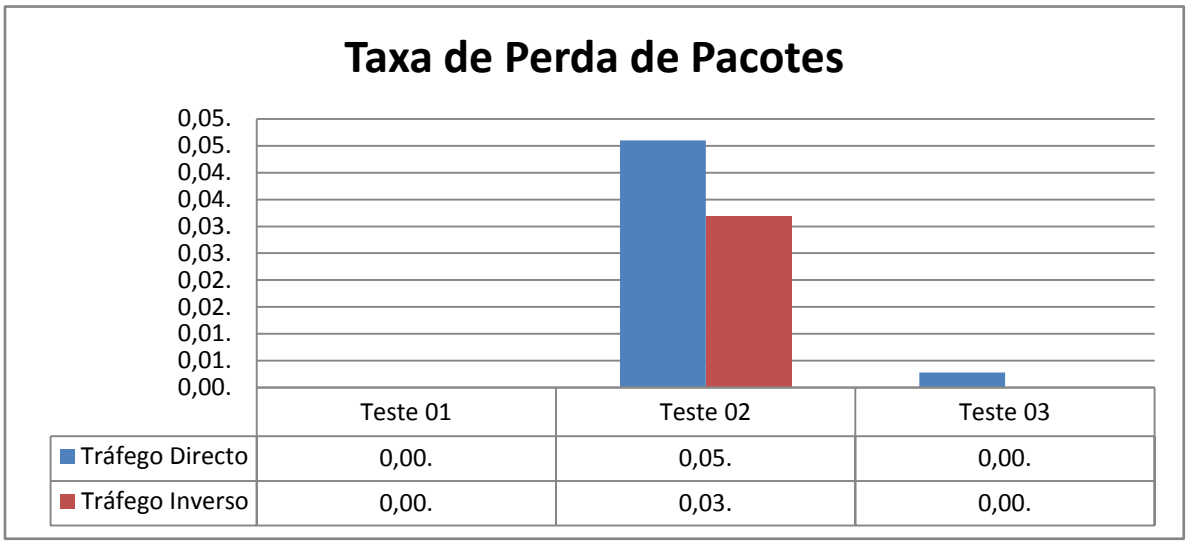

Figura 5.7 - Taxa de perda de pacotes para os três testes.

5.2.2 Cenário Inter-Domínio

Neste cenário foram repetidos os testes efectuados para o cenário intra-domínio com a diferença de que os utilizadores nos pontos terminais da ligação se encontravam em domínios *DiffServ* distintos. Pretendia-se testar nesta situação a eficácia da arquitectura em situações que envolvem mais do que um BB, a comunicação entre BBs e a sua capacidade em configurar encaminhadores de fronteira de mais do que um domínio.

5.2.2.1 Análise de Resultados

- **Latência**

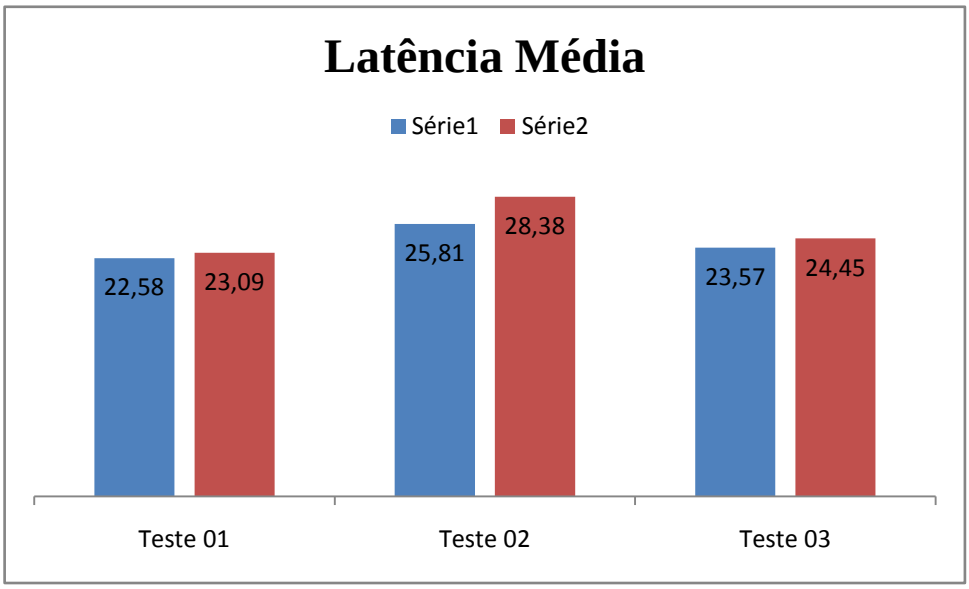

Figura 5.8– Comparação da latência média nos três testes (em milissegundos)

- ***"Jitter"***

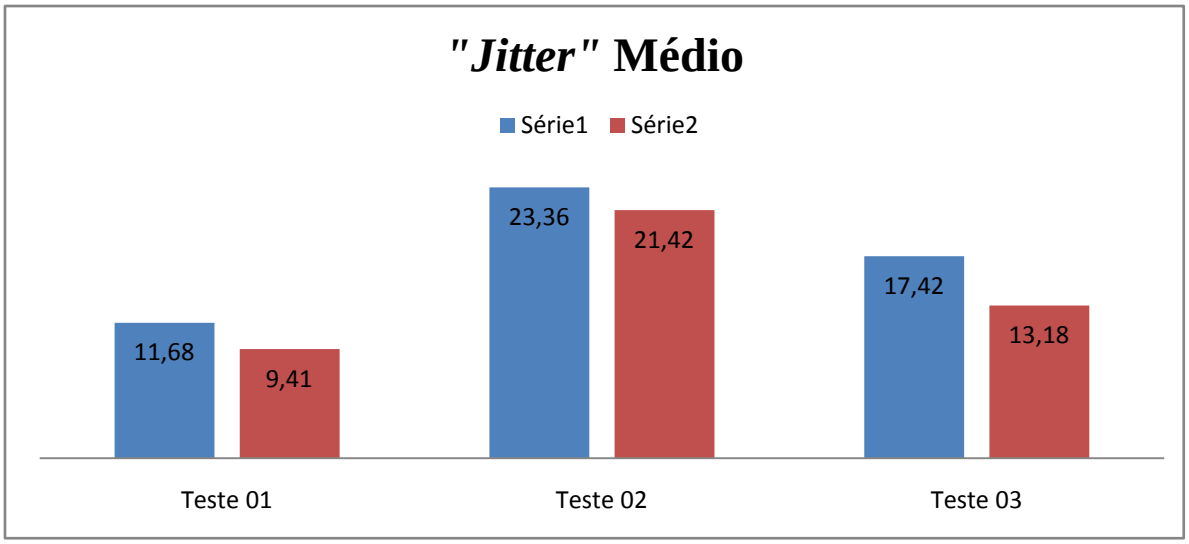

Figura 5.9 - Comparação do *"jitter"* médio (em milissegundos)

- **Perda de pacotes**

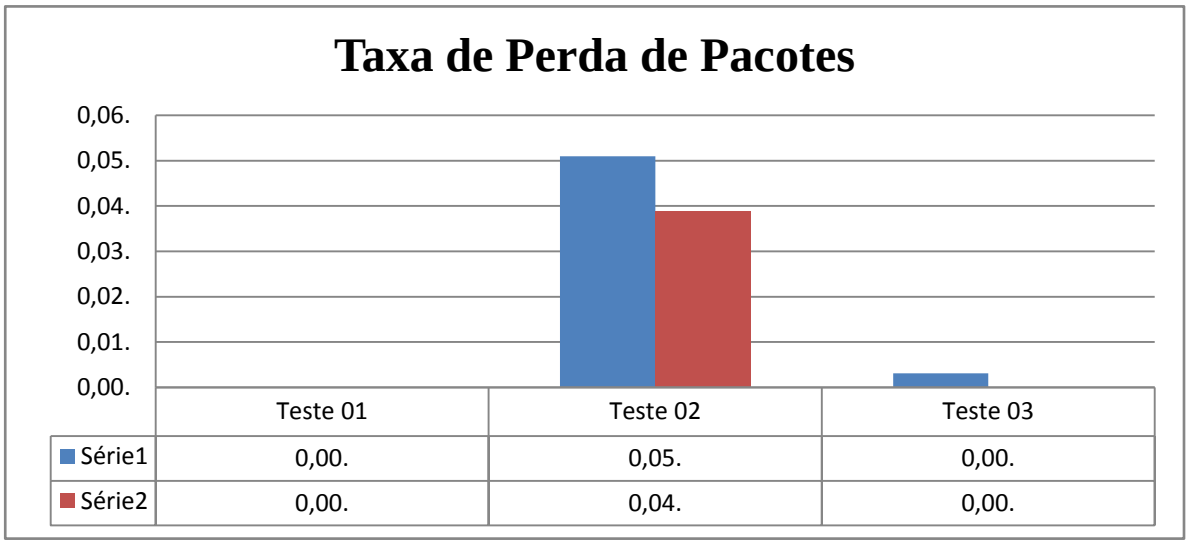

Figura 5.10 - Taxa de perda de pacotes para os três testes.

5.2.2.2 Análise de Resultados

Em relação aos resultados obtidos neste cenário, estes foram semelhantes aos obtidos com o cenário inter-domínio, o que pode ser observado nos gráficos da secção 5.2.2.2. Mas isto já era esperado pois o ponto de estrangulamento utilizado para limitar a largura de banda disponível na rede criada foi o mesmo. Os domínios adicionais foram criados na mesma

máquina, sendo-lhes atribuídos endereços IP e portos distintos. A comunicação adicional entre os BBs necessária para a propagação dos pedidos de largura de banda e para a reconfiguração dos encaminhadores processou-se utilizando o canal de dados criado para o efeito e cuja largura de banda é reservada na configuração *DiffServ* dos domínios. Assim sendo, este tráfego adicional não interfere no desempenho do tráfego reservado para voz.

Devido a estes resultados, a grande relevância deste teste reside a nível da sinalização.

Concluindo, as pequenas diferenças nos resultados devem-se sobretudo ao tempo de processamento adicional necessário para as instâncias dos BBs utilizadas processarem os pedidos e estabelecerem a comunicação entre si. A análise feita aos resultados apresentada para o cenário intra-domínio é por isso válida também para este cenário.

Quanto à comunicação entre os BBs, é possível verificar a sua ocorrência e a forma como se processa através da impressão de mensagens de texto na consola ou utilizando o *Wireshark* para verificar o fluxo da troca de mensagens, assim como o seu conteúdo. Verificou-se a ocorrência da troca de mensagens apresentada na Figura 5.2 da secção 5.1. Quando o BB verifica que um dos utilizadores não se encontra no seu domínio *DiffServ* propaga o RAR para o seu BB vizinho. Como este também não tem o segundo utilizador no seu domínio, propaga o RAR para o seu outro vizinho. Quando o RAR chega ao BB que gere a QoS do domínio do segundo utilizador, este verifica que pode satisfazer o pedido e envia a respectiva resposta, que é propagada seguindo o caminho inverso até ao procurador.

CAPÍTULO 6.

CONCLUSÕES

6.1 Síntese da Dissertação

Nesta dissertação procurou-se apresentar uma solução para o problema da gestão de largura de banda numa rede. Procurou-se encontrar uma solução que, mantendo intactas as funcionalidades de reserva de recursos e de garantia de qualidade de serviço, pudesse ser o mais transparente e fácil de utilizar possível, para o utilizador comum.

No segundo capítulo são apresentados alguns conceitos que, apesar de não serem o foco desta dissertação, são fundamentais para obter uma melhor compreensão do trabalho realizado.

No terceiro capítulo apresentam-se alguns esforços de pesquisa e desenvolvimento na área da garantia de qualidade de serviço e na utilização de procuradores de largura de banda, que serviram de inspiração e apoio para a realização desta dissertação.

A solução adoptada é descrita em pormenor no quarto capítulo. Esta descrição engloba a arquitectura que serviu como base para a elaboração do protótipo elaborado, assim como as modificações que lhe foram efectuadas para atingir o objectivo proposto no início da sua realização. Apesar de parte da arquitectura não ter sido implementada durante a realização da dissertação a descrição do funcionamento de todos os módulos é necessária para a compreensão total do seu funcionamento global.

Os cenários utilizados para testar o funcionamento da arquitectura e os resultados obtidos com estes testes são apresentados no quinto capítulo.

6.2 Conclusões

Partindo de uma arquitectura *DiffServ*, baseada na utilização de procuradores de largura de banda, previamente implementada, obteve-se uma nova arquitectura que, mantendo as funcionalidades de QoS originais, dispensa a interacção directa com o utilizador. Como todo o processo de autenticação e elaboração de pedidos de recursos é feito de forma transparente, torna-se mais fácil de utilizar pelo utilizador comum.

Foram efectuados testes de desempenho, simulando os principais casos de uso para os quais esta arquitectura pode ser potencialmente utilizada. Os resultados obtidos e apresentados no capítulo anterior provam a capacidade do sistema continuar a oferecer o serviço de voz sobre IP em condições de saturação. O desempenho da arquitectura mostrou ter resultados próximos dos obtidos com os ensaios de referência em condições ideais (sem tráfego adicional na rede).

A arquitectura implementada provou ser capaz de lidar com o problema da optimização da utilização da largura de banda através da sua gestão e distribuição pelos utilizadores, conforme os seus privilégios atribuídos pelo administrador de rede.

Quanto ao foco principal desta dissertação, a transparência para o utilizador de todo o processo de pedido e reserva de largura de banda, este objectivo também foi alcançado com sucesso. Embora seja necessária uma identificação por parte do utilizador junto do BB, esta processa-se de forma simples, através do *"login"* no procurador SIP. Foi abolida a necessidade de especificar os endereços IP dos terminais da comunicação, necessários para a identificação o fluxo de dados correspondente e a largura de banda desejada, sendo-lhe atribuída automaticamente aquela que foi previamente acordada com a entidade responsável pela rede. O visionamento de mensagens de erro ou de confirmação de pedidos é omisso ao utilizador, mantendo o processo o mais transparente possível para o utilizador.

A solução adoptada de criar um canal com largura de banda reservada para a comunicação SIP e para a troca de mensagens entre BBs garante o funcionamento da arquitectura mesmo em circunstâncias de carga elevada na rede. Adicionalmente, pode ser também uma solução para o problema de escalabilidade, em que as mensagens entre BBs e entre encaminhadores possam gerar um excesso de carga que interfira no desempenho do tráfego de voz. Desta forma, este tráfego adicional é mantido em separado e a largura de banda que lhe é reservada pode ser facilmente adaptada para corresponder às necessidades de uso. Basta ao administrador de rede alterar a configuração por omissão de *DiffServ* do BB.

Com o uso de HTB, esta largura de banda não é desperdiçada quando não está em uso pois é redistribuída pelas restantes classes criadas.

6.3 Perspectivas de Trabalho Futuro

Embora tenha sido alcançado o objectivo de implementar uma arquitectura funcional, em termos de garantia de QoS, mantendo o seu funcionamento transparente para o utilizador, existem ainda possibilidades para desenvolvimento futuro da mesma.

Como foi apresentado na descrição da arquitectura, as mensagens que são trocadas entre o BB e as aplicações de cliente assim como entre os seus pares são simples mensagens de texto, convertidas em tramas de bytes para fins de transmissão. Isto apresenta uma falha de segurança que poderia ser colmatada através da codificação das mensagens, anteriormente ao seu envio.

Ainda no capítulo da segurança, a identificação e autentificação dos utilizadores perante o BB é feita de forma básica, sendo requisitados apenas o nome de utilizador e a correspondente senha. Estes dados são enviados ao BB pela aplicação de cliente sob a forma de texto, podendo ser facilmente interceptadas e analisadas. Isto aumenta o potencial para o uso fraudulento de recursos da rede, em que um utilizador não autorizado poderia aceder ao BB e fazer uma reserva de largura de banda à qual não tem direito. Embora este tipo de autenticação seja suficiente para fins de teste e para o propósito desta dissertação, é claramente insuficiente para o uso desta arquitectura numa rede pública ou numa hipotética aplicação comercial. O uso de encriptação e de autenticação baseada em chaves são hipóteses a considerar para solucionar este problema.

A robustez da arquitectura poderia ser reforçada através da implementação de um mecanismo de descoberta de rotas mais avançado. Nesta implementação, o mecanismo utilizado é estático, dependendo da informação acerca da topologia da rede armazenada na base de dados *MySQL*. Em caso de uma possível falha na rede, o reencaminhamento através de uma rota alternativa não é suportado. A adição de um protocolo de encaminhamento dinâmico como Abrir Caminho Mais Curto Primeiro (*"Open Shortest Path First"* – OSPF) [Moy 1991] poderia ser uma solução a ter em conta para este problema.

BIBLIOGRAFIA

[3GPP 2007] *"Quality of Service (QoS) – Concept and Architecture (Release 7) – Technical Specification 23.107"*; 3GPP, Setembro 2007.

[Ananthanarayanan R.; Parashar M.; *"Active Resource Management for the Differentiated Services Enviroment."* em *"Third Annual Workshop on Active Middleware Services in Conjuction With the Tenth IEEE International Symposiumon High Performance Distributed Computing (HDPC-10)"*, São Francisco, Agosto 2001.

[Aukia 2000] Aukia P., Kodiamlam M., Koppol V., Lakshman T., Sarin T., Sutter B., *"RATES: A Server for MPLS Traffic Engineering"*, IEEE Network Magazine, pág. 34-41, Abril 2000.

[BCIT 2008] http://www.bcit.ca/

[Braun 1999] Braun T., Gunter M., Khalil I.; *"Virtual Private Network Architecture"*; Relatório Técnico IAM-99-001, CATI, Abril 1999.

[CANARIE 2008] http://www.canarie.ca/

[Chimento 2002] Chimento P. et al; *"QBone Signaling Design Team – Final Report"*, Outubro 2002, http://www.internet2.edu/qos/wg/documents-informational/20020709-chimento-etal-qbone-signaling/

[CITI 2008] http://www.citi.umich.edu/projects/qbone/generator.html

[Clayton 1998] Clayton P., Poulton A., "Internet Quality Of Service" – Em *"1st South African Telecommunications, Networks and Applications Conference (SATNAC'98)"*, University of Cape Town, África do Sul, Setembro 1998.

[HTB 2003] http://luxik.cdi.cz/~devik/qos/htb/

[Hwang 2000] Hwang J., Weiss M.; *"On the Economics of Interconnection Among Hybrid QoS Networks in the Next Generation Internet"* – Em *"XIII Biennial Conference of the International Telecommunications Society (ITS)"*, Buenos Aires, Julho 2000.

[J2SE 2003] *"Java 2 Platform Standard Edition, v1.4.1 API Specification"*, Sun Microsystems 2003, http://java.sun.com/j2se/1.4.1/docs/api/

[Jha 2002] Jha S., Hassan M.; *"Engineering Internet QoS"*; Artech House 2002.

[Jpcap 2007] http://netresearch.ics.uci.edu/kfujii/jpcap/doc/index.html

[Karagiannis 2000] Karagiannis G.; *"Qos in GPRS – Open Report 5/0362-FCB NB 10288"*, Dezembro 2000.

[Liljebladh 1999] Liljebladh M., Gateva R.; *"Integration Problems Between IP Multicast an DiffServ Concepts in Mobile Networks"* – Tese de Mestrado, Umea University e Chalmers University of Technology, Dezembro 1999.

[Moy 1991] Moy J.; *"OSPF Version 2 – Request For Comments 1247"*; Grupo de Trabalho de Rede (NWG - *"Network Working Group"*), Julho 1991.

[NIST 2008] http://www.nist.gov/

[Pham 2003] Pham K., Nguyen R.; *"Implementation Of A Bandwidth Broker In Java"*; Tese de Bacharelato em Engenharia de Telecomunicações, University of New South Wales, Junho 2003.

[Ramanthan 2001] Ramanathan A., Parashar M., *"Active Resource Management for the Differentiated Services Environment."* – Em *"Third annual international workshop on active middleware services in conjunction with the Tenth IEEE International Symposium on High Performance Distributed Computing (HPDC-10)"*, São Francisco, Agosto 2001.

[RFC2474 1998] Nichols K., Blake S., Baker F., Black D.; *"Definition Of The Differentiated Services Field (DS Field)in the IPv4 and IPv6 Headers – Request For Comments 2474"*; NWG, Dezembro 1998.

[RFC2475 1998] Blake S., Black D., Carlson M., Davies E., Wang Z., Weiss W.; *"An Architecture for Differentiated Services – Request For Comments 2475"*; NWG, Dezembro 1998.

[RFC2597 1999] Heinanen J., Baker F., Weiss W., Wroclawski J:; *"Assured Forwarding PHB – Request For Comments 2597"*; NWG, Junho 1999.

[RFC2598 1999] Jacobson V., Nichols K., Poduri K.; *"An Expedited Forwarding PHB – Request For Comments 2598"*; NWG, Junho 1999.

[RFC2638 1999] Nichols K., Jacobson V., Zhang L.; *"A Two-bit Differentiated Services Architecture for the Internet"*; NWG, Julho 1999.

[RFC2748 2000] Durham D., Boyle J., Boyle J., Cohen R., Herzog S., Rajan R., Sastry A.; *"The COPS (Common Open Policy Service) Protocol – Request For Comments 2748"*; NWG, Janeiro 2000.

[RFC2753 2000] Yavatkar R., Pendarakis D., Guerin R.; *"A Framework for Policy-based Admission Control – Request For Comments 2753"*; NWG, Janeiro 2000.

[RFC3084 2001] Chan K., Seligson J., Durham D., Gai S., McCloghrie K., Herzog S., Reichmeyer F., Yavaktar R., Smith A.; *"COPS Usage for Policy Provisioning (COPS-PR) – Request For Comments 3084"*; NWG, Março 2001.

[RFC3261 2002] Rosenberg J., Schulzrinne H., Camarillo G., Johnston A., Peterson J., Sparks R:, Handley M., Schooler E.; *"SIP: Session Initiation Protocol – Request For Comments 3261"*; IETF, Junho 2002.

[RFC3550 2003] Schulzrinne H., Casner S., Frederick R. Jacobson V.; *"RTP: A Transport Protocol For Real-Time Applications – Request For Comments 3550"*; NWG, Julho 2003.

[RFC4566 2006] Handley M., Jacobson V, Perkins C.; *"SDP: Session Description Protocol – Request For Comments 4566"*; NWG, Julho 2006.

[Zhang 2001] Zhang Z., Dunn Z., Hou Y

.;*"On scalable design of bandwidth brokers"*, Em *"IEICE Transaction on Communications, E84-B"*, Agosto 2001.