Aplicabilidade do Global Media Transmission Protocol (GMTP)

Wendell Silva Soares, Mário André de Menezes Costa, Joilnen Batista Leite Leandro Melo de Sales, André Lage Freitas

[1]Instituto de Computação (IC)
Universidade Federal de Alagoas (UFAL) – Maceió, AL - Brasil

{wss,leandro,andre.lage}@ic.ufal.br

Abstract. *Current Internet live streaming systems require various middlewares as they need specific network services. As a result, this increases network resource consumption as well as it is difficult to implement interoperable live streaming systems. The Global Media Transmission Protocol (GMTP) addresses this problem by proposing a live media distribution protocol for the Internet. GMTP uses P2P algorithms at socket level in order to build a network of favors among cooperative routers. The network of favors is managed by streaming servers, which orchestrate router partnerships based on the capacity of each router transmission channel. This paper presents an applicability study of GMTP by implementing GMTP in a real Operating System and assessing GMTP deployment and performance.*

Resumo. *Os sistemas para transmissão de mídias ao vivo na Internet atuais dependem de vários middlewares, pois eles precisam de serviços de rede específicos. Isso favorece o aumento do consumo de recursos de rede e dificulta a implementação de sistemas de streaming ao vivo que possam cooperar entre si. O Global Media Transmission Protocol (GMTP) aborda este problema, propondo um protocolo de distribuição de mídias ao vivo pela Internet. No GMTP, utilizam-se algoritmos P2P a nível de socket, a fim de construir uma rede de favores constituída entre roteadores. A rede de favores é gerenciada pelos servidores de streaming, que orquestram parcerias entre roteadores com base na capacidade de cada canal de transmissão. Neste trabalho apresentam-se um estudo da aplicabilidade do GMTP através da implementação do protocolo em um sistema operacional e uma avaliação da implantação e desempenho do GMTP.*

1. Introdução

A demanda dos usuários por conteúdo multimídia na Internet tem crescido a cada ano. A popularização do acesso à banda larga resulta em um aumento de demanda para os sistemas de distribuição de mídias ao vivo, elevando-se os custos com largura de banda dos distribuidores de conteúdo [James F. Kurose 2010]. No cenário atual, os desafios relacionados à escalabilidade, custos econômicos e do nível de satisfação do consumidor das transmissões de conteúdo multimídia na Internet ainda são uma questão em aberto [Barbosa and Soares 2014]. Por isto, é necessário pesquisar e propor novas soluções para utilizar eficientemente as redes de computadores, a fim de distribuir mídias ao vivo através da Internet em escala e com qualidade.

Atualmente, os sistemas para transmissão de mídias ao vivo na Internet executam diversas funções que foram implementadas para suprir a ausência de serviços de rede para este fim. A disseminação de diferentes sistemas e protocolos com este propósito tem levado à pulverização de soluções para transporte de dados multimídia, gerando

uma falta de padronização na forma como tais sistemas transportam seus dados na Internet [Sales 2014], principalmente por tais soluções serem implementadas na camada de aplicação [Liu et al. 2008]. Como consequência, não há a possibilidade de que dois ou mais nós, conectados em sistemas distintos, cooperem entre si, compartilhando o mesmo fluxo de dados transmitido por um servidor. Idealmente, os nós deveriam compartilhar o mesmo fluxo de dados e cada um apresentar o conteúdo da forma definida pela aplicação, desacoplando a forma de transportar os dados da forma como estes são apresentados pela aplicação aos seus respectivos usuários, evitando o consumo desnecessário de recursos de rede.

Nesse contexto, [Sales 2014] propôs um protocolo de rede denominado *Global Media Transmission Protocol* (GMTP), que considera uma arquitetura híbrida P2P/CDN para distribuição de mídias ao vivo com base na constituição de uma rede de favores entre roteadores. No GMTP, constituem-se redes de favores formadas por roteadores de rede, que cooperam entre si a fim de entregar o conteúdo multimídia de interesse comum aos seus usuários. As parcerias entre os roteadores são formadas com base na capacidade dos canais de transmissão já em uso. No GMTP, os clientes não necessariamente recebem os pacotes de dados diretamente dos servidores, mas podem recebê-los de roteadores já envolvidos na transmissão da mídia, evitando-se múltiplas conexões ao servidor.

Este documento está organizado como se segue. Na Seção 2, apresenta-se o funcionamento do GMTP. Na Seção 3, apresenta-se uma análise da implementação do GMTP. Na Seção 4, apresentam-se os experimentos, metodologia e resultados obtidos sobre o GMTP em um cenário de rede. Por fim, na Seção 5, apresentam-se as considerações finais, como conclusões e trabalhos futuros.

2. *Global Media Transmission Protocol* (GMTP)

O *Global Media Transmission Protocol* (GMTP) [Sales 2014] é um protocolo projetado para operar na Internet em sistemas de distribuição de mídias ao vivo. Com o GMTP, busca-se otimizar a distribuição de conteúdos multimídia ao vivo em larga escala, abstraindo-se a complexidade e promovendo-se a interoperabilidade entre os sistemas.

O GMTP é um protocolo *cross-layer* atuando nas camadas de transporte e de rede da pilha TCP/IP. Trata-se de um protocolo baseado em uma arquitetura de rede híbrida P2P/CDN, através da qual transmitem-se e compartilham-se datagramas de um ou mais sistemas independente do controle da aplicação. No GMTP, constituem-se redes de favores formadas por roteadores de rede, que cooperam entre si a fim de entregar o conteúdo multimídia de interesse comum aos seus clientes.

Nesse cenário, os servidores atuam como super nós para os nós da rede P2P, orquestrando as parcerias entre os roteadores da rede. Uma aplicação cliente reproduz o conteúdo multimídia ao usuário final à medida que recebe os pacotes de dados contendo partes da mídia, gerada por um ou mais servidores. Os servidores instruem os roteadores a efetivarem as parcerias com outros roteadores que possuem clientes também interessados no mesmo conteúdo multimídia. As parcerias entre os roteadores são formadas com base na capacidade dos canais de transmissão já em uso e as informações sobre a capacidade de transmissão de cada roteador são atualizadas periodicamente, por meio de uma versão adaptada do algoritmo controle de congestionamento assistido pela rede chamado *Rate Control Protocol* (RCP) [Dukkipati et al. 2005].

O princípio básico do GMTP é a introdução do conceito de *socket* P2P, onde a aplicação envia um pedido de conexão ao servidor, similar ao que ocorre com protocolos de transporte tradicionais. A diferença é que, se um roteador presente no trajeto de um fluxo de dados entre um cliente requisitante e o servidor já estiver repassando a mídia

requisitada, este pode interceptar o referido pacote de requisição e responder ao cliente, como se fosse o próprio servidor.

2.1. Visão Geral

O protocolo GMTP é composto de quatro etapas básicas, conforme ilustra-se na Figura 1:

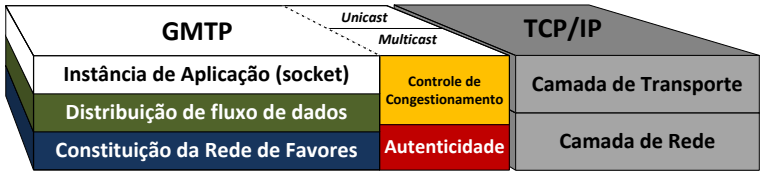
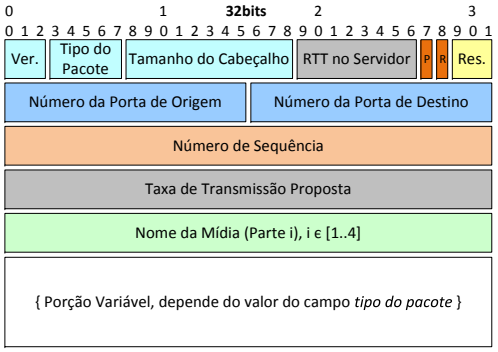


Figura 1. Blocos funcionais do GMTP e suas relações com a pilha de protocolos TCP/IP.

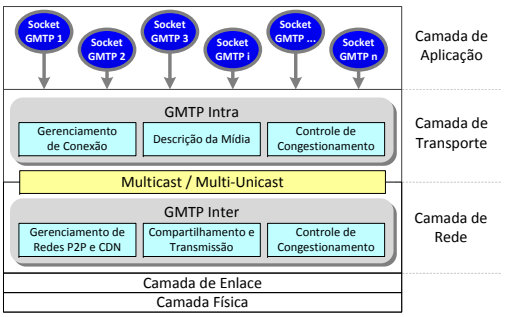
Fonte: [Sales 2014]

1. **Constituição da rede de favores**: descobrir, definir, efetivar e desfazer parcerias entre os roteadores de acordo com o evento ao vivo a ser transmitido;
2. **Distribuição de fluxos de dados através de uma camada de *socket***: conectar os clientes interessados em receber um fluxo de dados de um evento ao vivo, bem como transmitir tal fluxo de dados através das redes de favores;
3. **Controle de congestionamento**: controlar a taxa de transmissão dos fluxos de dados distribuídos e utilizar as informações sobre a capacidade de transmissão de um canal para sugerir favores entre roteadores;
4. **Autenticidade do conteúdo**: verificar a autenticidade do fluxo de dados antes de repassá-los aos clientes, evitando-se ataques de poluição e modificação indevida do conteúdo.

O protocolo GMTP é composto por dois módulos chamados de *GMTP-Intra* e *GMTP-Inter*, que operam nas camadas de transporte e de rede, respectivamente, definindo assim sua arquitetura, ilustrada na Figura 2(b). A principal responsabili-



(a) Porção fixa do cabeçalho dos pacotes GMTP.



(b) Arquitetura do Protocolo GMTP.

Figura 2. Arquitetura do Protocolo GMTP e a porção fixa do seu cabeçalho.

Fonte: [Sales 2014]

dade do GMTP-Intra é fornecer serviços às aplicações de rede a fim de abstrair a complexidade na execução de tarefas comuns a qualquer sistema final, tais como conexão multi-ponto, multiplexação/demultiplexação de segmentos IP entre as camadas de

transporte/rede/aplicação e controle de congestionamento. Este módulo compreende a instância do GMTP em execução no sistema operacional do cliente, acessível através de uma API de *socket* GMTP. Já a responsabilidade do GMTP-Inter é de constituir as redes de favores P2P compostas por roteadores, os quais funcionam como pontes de acesso aos servidores de uma rede CDN [Sales 2014].

No contexto dos tipos de nós do GMTP, definem-se 4 tipos, definidos a seguir:

1. **Cliente**: Sistema final capaz de reproduzir e gerar conteúdos multimídia ao vivo, executa um processo em nível de sistema operacional, representando uma aplicação manipulada pelo usuário final.
2. **Relator**: Um *cliente* com habilidades de enviar relatórios periódicos ao nó *repassador* sobre o estado da transmissão.
3. **Repassador**: Roteadores que participam efetivamente da rede de favores, com a responsabilidade de repassar os datagramas, originados em um ou mais *servidores*, para outros *repassadores* até que os datagramas alcancem os *clientes*.
4. **Servidor**: Sistema final que participa de uma rede CDN e obtém a mídia a ser transmitida.

2.1.1. Fluxo Básico

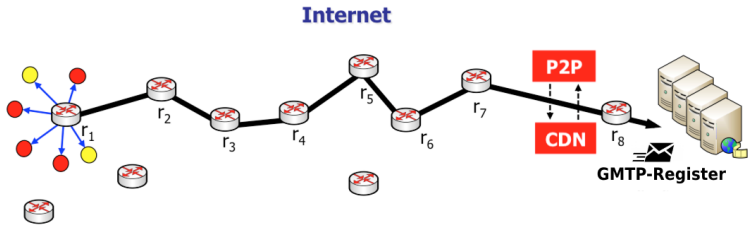
O fluxo básico de comunicação do GMTP ocorre quando um cliente deseja reproduzir um conteúdo multimídia. No protocolo GMTP executam-se 6 passos principais:

1. **Registro de Participação**: É o primeiro passo do processo de funcionamento do GMTP. O cliente envia uma requisição em direção ao servidor que está transmitindo o conteúdo multimídia de interesse. Em seguida, um repassador intercepta a requisição durante seu trajeto até o servidor, que é capaz de determinar os melhores parceiros para atendê-la. Em geral, isto ocorre já no roteador de borda do cliente, que funciona como repassador de origem. Caso o repassador não encontre nenhum repassador parceiro capaz de transmitir a mídia de interesse, a mensagem de requisição alcança o servidor que transmite a mídia correspondente.
2. **Transmissão de Dados**: Caso o servidor aceite o pedido de conexão do cliente, este inicia a transmissão do fluxo de dados solicitado ao cliente.
3. **Formação de Rede de Favores e Cache**: Em vez de aceitar o pedido de conexão do cliente, o servidor pode delegar tal requisição a um repassador, com base no conhecimento de quais repassadores já estão encaminhando o conteúdo de interesse para alguma rede.
4. **Verificação de Autenticidade de Dados**: Baseia-se na assinatura digital dos dados transportados nos pacotes e na capacidade que os repassadores têm de validar se o conteúdo foi intencionalmente alterado por usuários maliciosos no trajeto até o cliente, com base no certificado digital emitido pelo servidor – esta ação é opcional e decidida pela aplicação no início da transmissão.
5. **Controle de Congestionamento**: Executa-se um algoritmo de controle de congestionamento unicast entre os repassadores e os servidores. Nele, o repassador calcula sua capacidade de transmissão atual com base apenas no tamanho da fila de repasse e no atraso fim-a-fim marcado em pacote de dados a ser roteado. Entre os repassadores e os clientes, executa-se uma versão adaptada do algoritmo de controle de congestionamento *TCP-Friendly Multicast Congestion Control* (TFMCC) [Widmer and Handley 2006] para fluxo de dados *multicast*, regulando-se a taxa de transmissão na rede local com base nos relatórios transmitidos por nós eleitos como relatores.

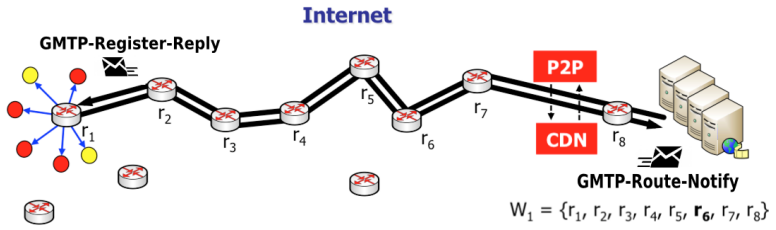
2.2. Constituição da Rede de Favores

No GMTP, a constituição da rede de favores ocorre por meio do registro de participação de um ou mais nós repassadores de um fluxo de mídia, a um ou mais nós servidores de mídia. O registro de participação permite que um nó repassador se registre no servidor para sinalizar interesse em funcionar como repassador de um fluxo de dados.

Seja r um nó repassador e s um nó servidor de mídias. Para realizar o registro de participação, o nó r deve enviar uma mensagem para o nó s utilizando um pacote do tipo *GMTP-Register*, conforme ilustra-se na Figura 3(a). Este, por sua vez, responde com um pacote do tipo *GMTP-Register-Reply*. O uso do pacote do tipo *GMTP-Register-Reply* permite a descoberta de um caminho entre s e r. Isto porque todos os nós repassadores existentes no caminho entre s e r devem adicionar seu identificador no pacote *GMTP-Register-Reply*, antes de roteá-lo para o próximo salto da rota em direção ao nó r. Quando o pacote *GMTP-Register-Reply* alcança o nó r, este envia a lista ordenada de repassadores entre r e s de volta ao nó s, utilizando um pacote do tipo *GMTP-Route-Notify*, conforme ilustra-se na Figura 3(b). A partir desse ponto, o nó s conhece o caminho que utilizará para enviar qualquer fluxo de dados até alcançar r, organizando-o em uma estrutura de dados do tipo grafo. No GMTP, as parcerias ocorrem entre os nós repassadores, e não entre os



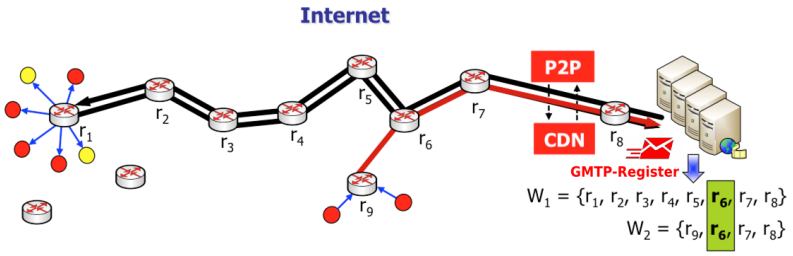
(a) Para realizar o registro de participação, o nó r_1 envia uma mensagem para o nó servidor utilizando um pacote do tipo *GMTP-Register*.



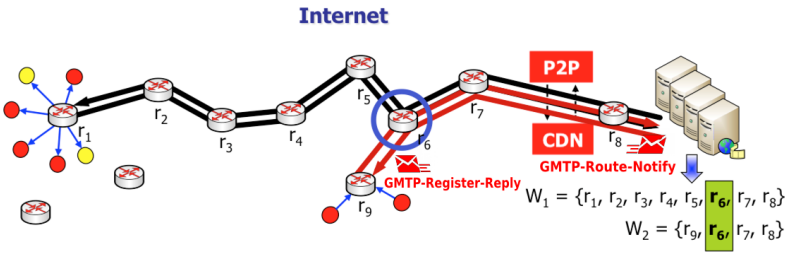
(b) Quando o pacote *GMTP-Register-Reply* alcança o nó r_1, este envia o caminho W_1 de volta ao nó servidor, utilizando um pacote do tipo *GMTP-Route-Notify*.

Figura 3. Registro de participação do repassador.

nós clientes. A formação de parcerias consiste em determinar intersecções de caminhos entre repassadores e servidores. Dessa forma, se um nó repassador r for um nó comum entre dois caminhos conhecidos por s, será necessário apenas enviar um fluxo de dados até r, que repassará o referido fluxo de dados para os demais nós parceiros, mais distantes de s, utilizando *multicast*. Conforme ilustra-se na Figura 4(b), é possível que um repassador atue somente encaminhando conteúdos multimídia entre duas ou mais redes distintas, mesmo que este não tenha demandas explícitas dos seus clientes por tal conteúdo. Na referida figura, o nó r_6 atua dessa forma. Sendo assim, quanto mais nós repassadores se registrarem em nós servidores, mais caminhos serão conhecidos. Quanto mais caminhos forem conhecidos, mais parcerias poderão ser formadas entre os nós repassadores. Quanto



(a) O nó r_9 realiza o registro de participação no servidor. Após o registro, o servidor conhece o caminho W_2 até r_9.



(b) Através do algoritmo de formação de parcerias do GMTP, identifica-se uma possível parceria entre r_6 e r_9, que passam a colaborar entre si.

Figura 4. Formação de parcerias entre repassadores.

mais parcerias forem formadas, maior será o número de nós clientes capazes de receber um fluxo de dados originado nos servidores [Sales 2014].

2.2.1. Envio e recebimento do fluxo de mídia

Após o estabelecimento de conexão, os nós repassadores trocam dados entre si em modo *unicast* a fim de distribuir os pacotes de dados. De forma similar, os nós repassadores repassam os dados para os nós clientes, porém em modo *multicast*. No GMTP, executa-se um algoritmo para controle de congestionamento híbrido, cujo comportamento dependerá do modo de transmissão sendo utilizado para transportar os pacotes de dados (*unicast* ou em *multicast*).

2.2.2. Controle de congestionamento *unicast*

O controle de congestionamento *unicast* do GMTP (GMTP-UCC) funciona de forma similar ao protocolo RCP, porém com alguns diferenciais. O RCP é um protocolo para controle de congestionamento assistido pela rede que tenta emular um Comutador Compartilhado (*Processor Sharing* - PS), por exemplo, um roteador [Dukkipati et al. 2005].

Seja r um nó repassador, s um nó servidor de mídias e W o caminho entre r e s. Cada nó r mantém uma única taxa de transmissão $R(\hat{t})$ que é atribuída no cabeçalho de todos os pacotes transmitidos do nó s aos nós repassadores em W. À medida que o pacote passa em cada repassador em W, se a taxa atual $R(\hat{t})$ no roteador for menor do que R_p informada no pacote sendo processado, $Rp \leftarrow R(\hat{t})$. Quando o pacote alcançar o último nó do caminho, este envia para s a o valor de R_p, que é a máxima taxa de transmissão suportada no caminho W. Ao receber o valor de R_p, s a atualiza $R(\hat{t})$ para transmitir os

próximos pacotes de dados. Este procedimento se repete a cada intervalo de tempo H.

O RCP utiliza uma abordagem fim-a-fim para determinar a taxa de transmissão de um nó, o que pode limitar a taxa de transmissão em um caminho. Utilizando o RCP, dado um determinado caminho W, a taxa de transmissão $R(t)$ de W será igual à menor taxa máxima de transmissão informada por um nó em W. Visando contornar as limitações do RCP, o GMTP-UCC segmenta o caminho, não limitando a taxa de transmissão de um nó capaz de receber um fluxo de dados em uma taxa de transmissão maior.

O GMTP permite descrever a mídia em múltiplas codificações, ao mesmo tempo que os nós repassadores podem acessar tal informação para implementar a solução discutida anteriormente. Como resultado dessa estratégia, permite-se que a rede se seja capaz de selecionar fluxos codificados pelo servidor de forma segmentada, de acordo com a capacidade de transmissão de sub-caminhos entre observados entre os nós repassadores e o servidor.

2.2.3. Controle de congestionamento *multicast*

O controle de congestionamento *multicast* do GMTP (GMTP-MCC) tem como objetivo determinar uma taxa de transmissão equânime entre os fluxos de dados transmitidos pelo GMTP e por outros protocolos, como o TCP, em modo de transmissão *multicast*. Trata-se de uma versão adaptada do algoritmo de controle de congestionamento *TCP-Friendly Multicast Congestion Control* (TFMCC) [Widmer and Handley 2006] para fluxo de dados *multicast*. No GMTP-MCC, elegem-se alguns dos nós clientes como relatores que devem calcular a taxa de transmissão baseado na taxa de eventos de perda de pacotes. Assim, regula-se a taxa de transmissão na rede local com base nos relatórios transmitidos por nós eleitos como relatores.

3. Implementação do GMTP

O protocolo GMTP está sendo implementado em um sistema Linux padrão, integrado à pilha de protocolos do *kernel* do Linux. É importante salientar que o GMTP está em fase de desenvolvimento. Portanto, a maior parte dos comportamentos do GMTP encontra-se funcionando, em fase de experimentação, excetuando-se aqueles módulos assinalados no texto a seguir.

3.1. GMTP-Intra

O módulo GMTP-Inter está sendo implementado em dois *Linux Kernel Modules* (LKM) distintos, denominados *gmtp.ko* e *gmtp_ipv4.ko*. No primeiro módulo encontram-se as funções e variáveis de estado relativas ao GMTP que são independentes da versão do *Internet Protocol* (IPv4 ou IPv6). O segundo diz respeito às funções e variáveis de estado relativas ao GMTP que são dependentes do IPv4. O LKM do GMTP específico para o IPv6 ainda não foi desenvolvido.

Uma vez que os módulos do GMTP estejam habilitados no sistema, o desenvolvedor de aplicações que deseja prover o suporte ao GMTP em uma aplicação multimídia, seja ela cliente ou servidora, deve abrir um *socket* do tipo GMTP utilizando qualquer linguagem de programação que forneça uma API compatível com as especificações de *socket* BSD e POSIX.

No GMTP, um fluxo de dados P tem um nome único que o identifica em qualquer nó. Na prática, cada fluxo de dados P corresponde a uma mídia gerada a partir de um evento ao vivo por exemplo, a transmissão de um jogo de futebol ou uma corrida de Fórmula 1. Portanto, define-se um nome de um fluxo de dados P por um código de *hash*

MD5 no formato UUID (*Universally Unique IDentifier*) de 128 bits [Leach et al. 2005]. Para determinar o nome de um fluxo de dados P, disponibilizado por um nó servidor s_a, utiliza-se MD5(IP_{sa} + $PORTA_{sa}$). Opcionalmente, o nó servidor pode divulgar o nome do fluxo de dados através do serviço DNS [Sales 2014]. O GMTP automaticamente gera o nome do fluxo de dados (nome da mídia) a partir do IP e porta fornecidos no momento da criação do *socket*. Opcionalmente, o nome da mídia pode ser definido pelo desenvolvedor através da função *setsockopt*.

É importante salientar que toda transferência de pacotes de controle entre nós do GMTP ocorre com garantia de entrega. Isso inclui os pacotes de solicitação de conexão e suas respectivas confirmações.

3.1.1. Clientes

Quando a aplicação cliente executa a chamada *connect*, o GMTP envia um pacote do tipo *GMTP-Request* com endereço IP de destino configurado como o endereço do servidor e com o campo "Nome da Mídia"definido adequadamente. Assim que a requisição alcança um repassador na rede local, este assume a requisição do cliente, intermediando a sua comunicação com o servidor. Como resposta ao pedido de conexão, o cliente recebe do repassador um pacote do tipo *GMTP-RequestReply* contendo o código *OK*, indicando que a conexão foi aceita e informando o endereço *multicast* pelo qual a mídia será ou está sendo transmitida, ou contendo o código *WAIT*, indicando que o repassador ainda aguarda a resposta a seu pedido de registro de participação no servidor. Assim que a resposta chegar, caso a conexão seja aceita, o repassador deve enviar um pacote *GMTP-RequestReply* com código *OK* para o cliente. Há ainda o código *ERROR*, indicando que a conexão foi recusada pelo servidor por algum motivo.

Caso não haja nenhum nó repassador no caminho entre o cliente e o servidor, a requisição alcançará o servidor. Caso o servidor aceite a conexão, enviará um pacote do tipo *GMTP-Register-Reply* de volta ao cliente.

Quando um cliente se conecta diretamente ao servidor, sem intermédio de um repassador, ele recebe os pacotes de mídia que são diretamente endereçados a ele pelo servidor, através do seu IP e porta de requisição. Alternativamente, quando o cliente se conecta ao servidor através de um repassador, o cliente recebe os pacotes de mídia endereçados ao endereço *multicast* definido pelo repassador. Em ambos os casos, o GMTP localiza o *socket* GMTP correspondente ao pacote recebido e o envia à aplicação.

A forma que o GMTP utiliza para localizar o *socket multicast* correspondente ao pacote de mídia difere da forma usual de busca de *socket* no Linux, porque (1) O endereço de conexão do cliente GMTP é sempre definido como o endereço do servidor (*unicast*) e portanto, é essa conexão *unicast* que o Linux armazena em suas estruturas internas, e (2) no Linux, os protocolos orientados à conexão como por exemplo o TCP e o DCCP não permitem a conexão de *sockets multicast*.

Os clientes GMTP armazenam em uma tabela *hash* cuja chave é o nome da mídia, uma estrutura que contém o endereço *multicast* e porta definidos pelo repassador e o endereço IP e uma lista de IPs e portas de todos os clientes GMTP daquele *host* que requisitaram a mídia ao servidor. Assim, ao receber um pacote de dados destinado a um endereço *multicast*, o cliente GMTP faz a busca pelo *socket* informando os endereço IP e porta *unicast* da requisição inicial ao servidor. Ao realizar essa busca, a função retorna o *socket* de conexão entre o cliente e o servidor, e o GMTP pode, finalmente, enviar o pacote à aplicação. Se no mesmo *host*, mais de um cliente GMTP solicitou a mídia, a função retorna o *socket* correspondente para cada cliente.

Um nó cliente sinaliza periodicamente sua participação na rede através de um método *keep-alive*, comumente utilizado em outros protocolos de rede consolidados, como o TCP. Um nó cliente pode sinalizar explicitamente sua desistência de participação quando não desejar continuar recebendo um fluxo de dados enviando um pacote do tipo *GMTP-Close*. Se o nó cliente estiver conectado diretamente ao servidor, este deve interromper o envio de dados ao cliente, finalizando o procedimento de desconexão com o envio do pacote do tipo *GMTP-Reset*. Caso o cliente esteja conectado através de um repassador, o cliente deverá sinalizar a sua participação ou desistência apenas ao nó relator que lhe foi designado.

Caso o nó cliente receba um pacote do tipo *GMTP-Close* seja do servidor ou do repassador, deve finalizar o procedimento de desconexão com o envio do pacote do tipo *GMTP-Reset*.

3.1.2. Relatores

Um relator um *cliente* com habilidades de enviar relatórios periódicos ao nó *repassador* sobre o estado da transmissão. Os clientes são escolhidos como relatores através de seus repassadores.

No algoritmo de controle de congestionamento *TCP-Friendly Rate Control protocol* (TFRC) [Handley et al. 2003], o nó receptor envia para o transmissor relatórios sobre as perdas observadas e, com base nesse relatório, o transmissor calcula a nova taxa de transmissão. O TFRC é categorizado com um protocolo de controle de congestionamento baseado em uma equação matemática (*Equation Based Congestion Control*) e algoritmos desse tipo são adotados em diversos protocolos, como no DCCP [Sales 2014].

O algoritmo do TRFC não é eficiente em transmissões *multicast*, devido ao problema conhecido por implosão de confirmações (*feedback implosion*). Esse problema ocorre quando há muitos receptores enviando relatórios de perdas para o mesmo transmissor, o que resulta em uma inundação de relatórios, os quais o transmissor é incapaz de processar em tempo hábil [Sales 2014]. Visando contornar esse problema, [Widmer and Handley 2006] propuseram o *TCP-Friendly Multicast Congestion Control* (TFMCC), uma versão do TFRC para fluxo de dados *multicast*, que propõe que apenas um subconjunto dos receptores envie relatórios ao transmissor, a fim de evitar a implosão de confirmações.

Nos relatores, executa-se o GMTP-MCC, que é uma versão adaptada do algoritmo de controle de congestionamento TFMCC. A implementação do GMTP-MCC nos relatores foi realizada através da adaptação da implementação do TRFC já disponível no código fonte do DCCP. No GMTP-MCC, o próprio relator (que é um receptor), baseado no relatório de perdas, calcula a taxa de transmissão adequada e envia o relatório ao receptor. Cabe ao repassador, após receber os relatórios dos relatores, definir a nova taxa de transmissão, através da média das taxas de transmissão recebidas em um determinado intervalo de tempo.

O nó relator, ao enviar seu relatório ao repassador, deve informar: (1) a taxa de transmissão proposta, (2) quantos de seus *slots* estão ocupados e a marca de tempo do último pacote de dados recebido antes do cálculo da taxa de transmissão proposta. Os relatórios dos relatores também servem como mecanismos de *keep-alive* para os seus repassadores.

Um nó relator pode sinalizar explicitamente sua desistência de participação de forma semelhante a um nó cliente, com a diferença que ele precisa continuar enviando

os relatórios até que o repassador lhe envie um pacote do tipo *GMTP-Reset*, confirmando que outro relator foi escolhido pelo repassador. Esse procedimento de substituição de um relator por outro ainda não foi implementado, estando em fase de desenvolvimento.

3.1.3. Servidores

Quando o servidor atende um requisição, seja de um cliente ou de um repassador, envia um pacote do tipo *GMTP-Register-Reply*. O servidor espera receber de volta um pacote do tipo *GMTP-Ack*, caso a requisição venha de um cliente, ou do tipo *GMTP-RouteNotify*, caso a requisição venha de um repassador.

Através do pacote do tipo *GMTP-Ack*, sinaliza-se o estabelecimento de uma conexão bem-sucedida entre o servidor e o cliente. Através do pacote do tipo *GMTP-RouteNotify*, não somente sinaliza-se estabelecimento de uma conexão bem-sucedida, como também informa-se ao servidor o caminho de repassadores que existe entre o cliente e o servidor. Com este procedimento, o nó servidor passa a conhecer o caminho W entre o cliente e o servidor, que pode ser utilizado para determinar futuras parcerias, conforme discutido na Seção 2.2.

O nó repassador pode enviar periodicamente um pacote do tipo *GMTP-RelayQuery* para o nó servidor a fim de descobrir melhores parceiros e aumentar o número de parcerias. O nó servidor constrói uma lista de nós parceiros e envia como resposta ao nó repassador. Esse procedimento de indicação de parcerias entre repassadores ainda não foi implementado, estando em fase de desenvolvimento.

O servidor armazena os caminhos em uma estrutura que combina o uso de tabelas *hash* com listas encadeadas. As funções e estrutura de armazenamento dos caminhos está em fase de desenvolvimento. A cada caminho W é definida uma taxa de transmissão através do GMTP-UCC, conforme discutido na Seção 2.2.2. Ao receber a nova taxa de transmissão informada pelos repassadores, o servidor atualiza a sua taxa de transmissão de acordo.

Os nós clientes e repassadores devem notificar sua presença periodicamente para continuar a receber o fluxo de mídia. Eles também podem sinalizar explicitamente sua desistência enviando um pacote do tipo *GMTP-Close*. O nó servidor deve finalizar o procedimento de desconexão com o envio do pacote do tipo *GMTP-Reset*.

3.2. GMTP-Inter

O módulo GMTP-Inter está sendo implementado como um *Linux Kernel Module* (LKM), denominado *gmtp_inter.ko*. Para a construção do referido módulo utiliza-se a ferramenta denominada *Netfilter*.

Netfilter [Welte and Ayuso 1999] é um *framework* dentro do *kernel* do Linux, que possui uma série de *hooks* que oferecem flexibilidade para várias operações relacionadas com a rede. Através do Netfilter, torna-se possível que módulos do *kernel* do Linux possam registrar funções de *callback* a fim de interceptar e manipular os pacotes de rede. O Netfilter oferece várias opções para filtragem de pacotes, tradução de endereços de rede e de portas. Estas funções geralmente são aplicadas ao tráfego na forma de regras de filtragem e de modificação, e são chamadas para cada pacote que atravessa o respectivo *hook* dentro da pilha de rede, fornecendo as funcionalidades necessárias para modificar pacotes, direcioná-los através de uma rede, bem como proibi-los de alcançar determinados locais. No GMTP-Inter, apenas os pacotes GMTP são manipulados pelos *hooks* do Netfilter. Registram-se as funções de ingresso de paco-

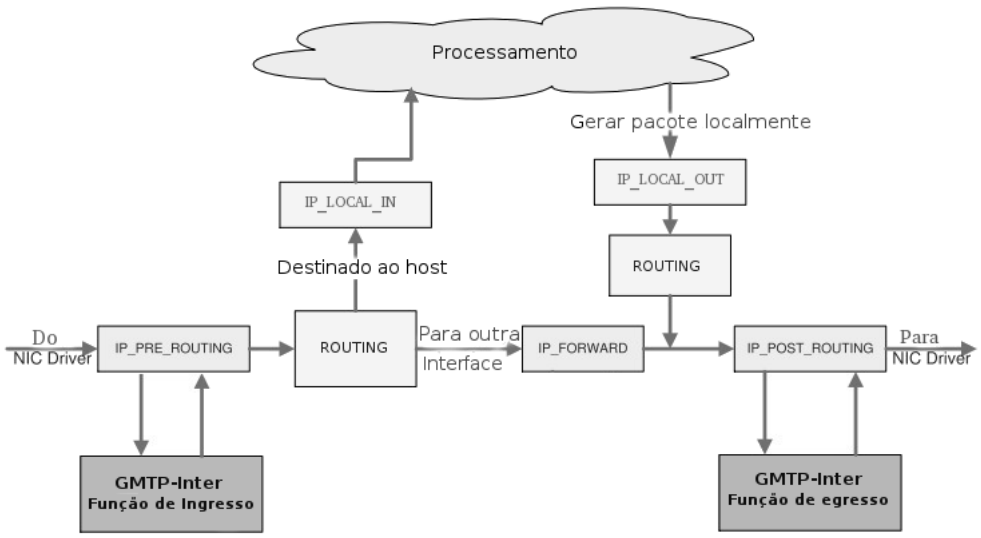


Figura 5. Linux Netfilter: Caminhos possíveis no processamento de pacotes e locais onde os *hooks* do GMTP-Inter foram configurados.

tes através do *hook* NF_INET_PRE_ROUTING e as funções de egresso através do *hook* NF_INET_POST_ROUTING, conforme ilustra-se na Figura 5

Ao receber a primeira requisição de um cliente por um fluxo de mídia P, o repassador deve defini-lo como um relator daquele fluxo. Por padrão, um em cada seis clientes que solicitarem o fluxo de midia P deve ser designado como relator. A proporção clientes/relator padrão é de 6:1, mas deve ser configurável. Para cada relator, o repassador guarda uma estrutura com o seu endereço IP, porta e número de *slots* ocupados. Se a proporção clientes/relator for *K:1*, o número máximo de *slots* de um relator deve ser *K-1*. Por exemplo, como a proporção padrão clientes/relator é de 6:1, por padrão, o número de *slots* disponíveis por relator é de 5 *slots*. Para o relator, cada *slot* corresponde a um cliente que lhe foi designado.

Caso um relator que não tenha nenhum *slot* ocupado saia da rede, não é necessária a realização de uma eleição de novo relator. Porém, caso um relator que tenha pelo menos um de seus *slots* ocupado saia da rede, o repassador deve executar o algoritmo de eleição de um novo relator, através de um sorteio entre os clientes associados ao relator que saiu. Para tanto, o pacote *GMTP-Close* oriundo do relator que está saindo deve ter a lista dos clientes que lhe foram designados, pois o repassador não armazena a lista de clientes de cada relator. Esse procedimento de eleição de um novo relator ainda não foi implementado, estando em fase de desenvolvimento.

Os relatórios dos relatores também servem como mecanismos de *keep-alive* para os seus repassadores. Após determinado período sem receber relatórios, o repassador deve considerar que não há mais nenhum cliente recebendo o fluxo de mídia, e deve encerrar a transmissão, podendo inclusive, sinalizar explicitamente sua desistência de participação na rede de favores diretamente ao nó servidor. O repassador não precisa armazenar informações de cada cliente conectado, mas necessita manter uma lista dos relatores de cada fluxo de mídia. Cabe ao repassador, após receber os relatórios dos relatores, definir a nova taxa de transmissão, através da média das taxas de transmissão recebidas em um determinado intervalo de tempo.

Caso o repassador receba uma mensagem do tipo *GMTP-Close* oriunda do ser-

vidor, deve repassá-la via canal *multicast*, apagar a entrada correspondente na tabela de repasse e finalizar a conexão, retornando ao servidor um pacote do tipo *GMTP-Reset*.

4. Experimentos

Visando avaliar a corretude e eficiência da implementação proposta, definiu-se uma série de experimentos a serem realizados.

4.1. Metodologia

Visando avaliar os resultados dos experimentos, definiu-se que a métrica a ser estudada seria a taxa recepção de dados total percebida nos clientes, medida em *bytes*/segundo. A **taxa de recepção** é a razão entre a quantidade total de dados recebidos, medida em *bytes*, pelo tempo total do experimento, medido em segundos. O ambiente do experimento foi definido da forma que se segue:

1. **Servidor**: Aplicação executada em uma máquina *servidor de mídia* capaz de transmitir dados através de um dos seguintes protocolos de transporte: TCP, UDP e GMTP. Os dados são transmitidos a uma taxa fixa de 50.000 *bytes*/segundo.
2. **Clientes**: Aplicação executada em uma máquina cliente capaz de receber dados através de um dos seguintes protocolos de transporte: TCP, UDP e GMTP. Cada cliente registra a sua taxa de recepção de dados em arquivos de texto (*log*).
3. **Roteador**: Máquina virtual que realiza o papel de roteador, permitindo a interconexão entre a sub-rede do servidor e a sub-rede dos clientes. A máquina funciona como um roteador comum quando o módulo GMTP-Inter está desativado. Quando ativa-se o módulo GMTP-Inter, a máquina passa a funcionar como um repassador GMTP.

No ambiente proposto acima, quatro máquinas virtuais foram utilizadas para realizar a simulação, cada uma com seu papel definido: uma máquina hospeda o servidor, estando alocada em uma sub-rede, duas máquinas hospedam os clientes, estando alocadas em outra sub-rede, e uma máquina realiza o papel de roteador. A comunicação entre o servidor e os clientes se dava através desse roteador. Todas as máquinas virtuais estão hospedadas na mesma máquina real. Logo, todas as transmissões de dados são locais, sem interferências externas. Todos os clientes requisitam o mesmo fluxo de dados. Em cada experimento, os clientes receberam 6500 pacotes de dados. Cada simulação foi repetida 10 vezes, e os dados foram agregados utilizando-se a média aritmética dos resultados obtidos.

Utilizando a estrutura descrita acima, foram organizadas três topologias básicas, conforme ilustrado na Figura 6. Cada experimento foi realizado em uma dessas topologias.

Os experimentos realizados foram os seguintes:

1. **Experimento A**: realizado utilizando a topologia ilustrada na Figura 6(a). Apenas um Cliente GMTP requisitando o fluxo de dados do Servidor GMTP.
2. **Experimento B-TCP**: realizado utilizando a topologia ilustrada na Figura 6(b). Dez Clientes TCP, sendo cinco em cada máquina cliente, requisitando o fluxo de dados do Servidor TCP.
3. **Experimento B-UDP**: realizado utilizando a topologia ilustrada na Figura 6(b). Dez Clientes UDP, sendo cinco em cada máquina cliente, requisitando o fluxo de dados do Servidor UDP.
4. **Experimento B-GMTP**: realizado utilizando a topologia ilustrada na Figura 6(b). Dez Clientes GMTP, sendo cinco em cada máquina cliente, requisitando o fluxo de dados do Servidor GMTP.

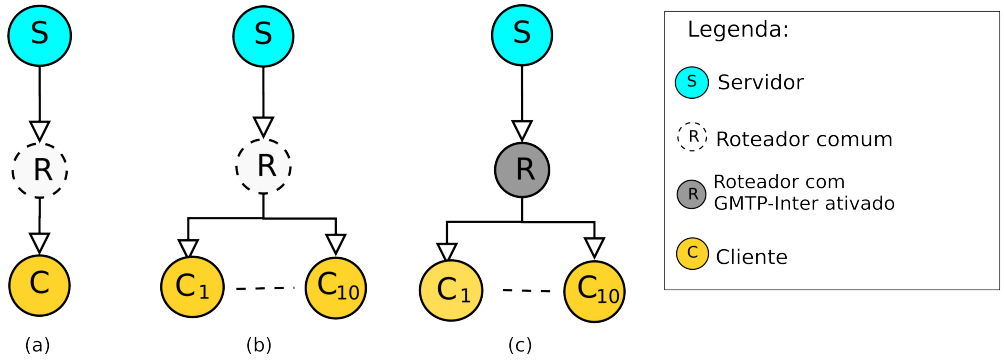


Figura 6. Topologias disponíveis para os experimentos.

5. **Experimento C**: realizado utilizando a topologia ilustrada na Figura 6(c). Dez Clientes GMTP, sendo cinco em cada máquina cliente, requisitando o fluxo de dados Servidor. Neste experimento, o roteador encontra-se com o módulo GMTP-Inter ativado.

4.2. Resultados

Segue abaixo a análise dos resultados obtidos. Um resumo do resultado das simulações pode ser visto através da Tabela 1 e da Figura 7.

Tabela 1. Taxa de recepção (bytes/segundo)

Simulação	Mínima	Média	Máxima	Coeficiente de Variação
Experimento A	47.600	47.770	47.920	0.0012
Experimento B-TCP	27.520	28.550	29.050	0.0146
Experimento B-UDP	37.710	38.140	40.010	0.0136
Experimento B-GMTP	38.110	39.530	41.180	0.0146
Experimento C	45.240	47.580	48.120	0.0091

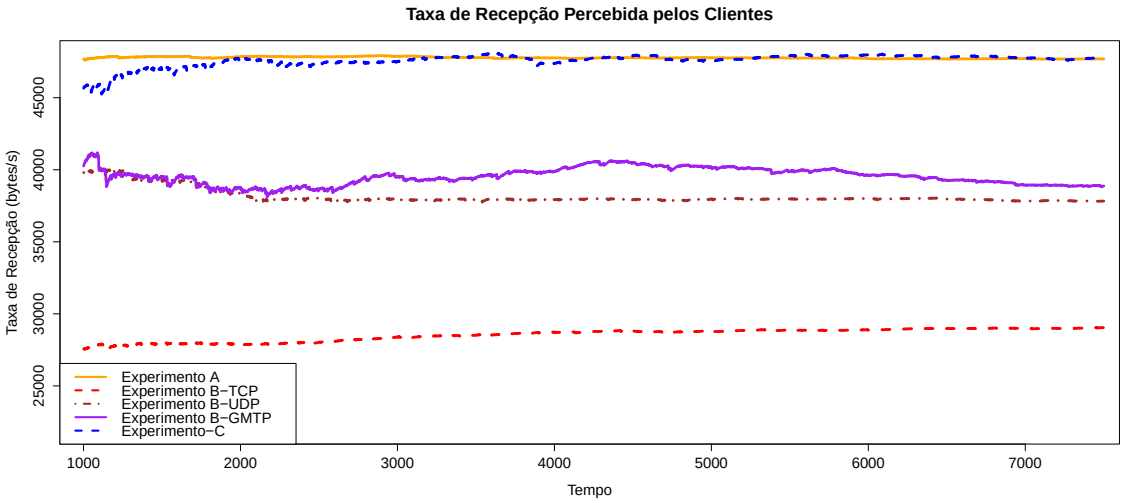


Figura 7. Evolução da taxa de recepção em relação ao tempo.

No Experimento A, a taxa de recepção de dados permaneceu muito próxima à taxa de envio de 50.000 *bytes*/s fornecida pelo servidor.

Nos experimentos B (B-TCP, B-UDP e B-GMTP), a taxa de recepção de dados dos clientes foi inferior à taxa de envio fornecida pelo servidor. Os possíveis motivos para a redução da taxa de recepção do clientes são a sobrecarga de processamento no servidor, que precisa tratar múltiplas requisições de diversos clientes. e a sobrecarga na rede, devido à quantidade de fluxos de dados trafegando. Para cada requisição, o servidor criou um fluxo de dados, o que ocasionou o crescimento na fila de pacotes do roteador e o aumento do atraso na recepção dos pacotes percebido pelos clientes. O algoritmo de controle de congestionamento do TCP impactou negativamente o desempenho da transmissão no experimento B-TCP.

No Experimento C, a taxa de recepção de dados dos Clientes GMTP se manteve próxima à taxa obtida na simulação simples, mesmo havendo vários clientes requisitando o fluxo de dados. Na simulação de sobrecarga com o GMTP-Inter habilitado, os clientes apresentaram taxas mínimas e máximas de recepção, próximas às taxas fornecidas pelo servidor.

Através do GMTP-Inter, os clientes não necessariamente recebem os pacotes de dados diretamente dos servidores, mas podem recebê-los de roteadores já envolvidos na transmissão da mídia, evitando-se múltiplas conexões ao servidor. Os benefícios dessa abordagem são:

- Como o servidor recebe apenas uma requisição ao servidor, evita-se o consumo excessivo de recursos computacionais do servidor.
- O servidor envia apenas um fluxo de dados ao roteador, o que evita o crescimento desnecessário da fila de pacotes do roteador. Este, por sua vez, transmite os dados recebidos aos clientes utilizando *multicast*.

5. Conclusões

Neste trabalho, apresentou-se a metodologia e implementação de um novo protocolo de distribuição de mídias ao vivo chamado *Global Media Transmission Protocol* (GMTP). Com base nos resultados de projeto e análise de desempenho do GMTP, a seguir, apresentam-se as principais conclusões deste trabalho.

A utilização de soluções para transporte de dados multimídia implementadas apenas na camada de aplicação impossibilita que dois ou mais nós conectados em sistemas distintos cooperem entre si, compartilhando o mesmo fluxo de dados transmitido por um servidor. Nessas soluções, ocorre a duplicação de fluxos de dados da transmissão de um evento, mesmo que este seja destinado a um sub-conjunto de redes contendo nós receptores interessados em tal evento. Esta duplicação acarreta o consumo desnecessário de recursos de rede.

Através da utilização do GMTP (com seu módulo GMTP-Inter), os nós interessados no mesmo fluxo de dados compartilham recursos, evitando a duplicação de fluxos de dados e o consumo desnecessário de recursos de rede. Os clientes não necessariamente recebem os pacotes de dados diretamente dos servidores, mas podem recebê-los de roteadores já envolvidos na transmissão da mídia, evitando-se múltiplas conexões ao servidor.

Por isto, conclui-se também que o conceito de *socket P2P* empregado no GMTP permite uma abstração de distribuição P2P com suporte transparente ao uso de *multicast* por parte da aplicação. A utilização de informações explícitas sobre o congestionamento da rede para permitir a um servidor determinar parcerias torna possível não apenas compartilhar os pacotes de dados entre múltiplos sistemas interessados por um mesmo conteúdo, como também reduzir o consumo de recursos de rede e melhorar a qualidade de serviço oferecida às aplicações.

5.1. Trabalhos futuros

Como trabalhos futuros, pretende-se conduzir estudos para explorar outras topologias de rede, incluindo redes com diferentes padrões de mobilidade, como redes veiculares, bem como entender o impacto do tráfego de dados gerados por outros protocolos sobre o GMTP.

Deve-se também fazer uso de um arcabouço próprio para a construção de experimentos de rede, acoplando a implementação do GMTP no *Kernel* do Linux em um simulador de rede. Assim, será possível realizar simulações com cenários mais complexos, com maior número de nós envolvidos, utilizando a implementação real do protocolo no núcleo de um sistema operacional.

Referências

Barbosa, F. R. N. and Soares, L. F. G. (2014). Towards the Application of WebRTC Peer-to-Peer to Scale Live Video Streaming over the Internet. In *Simpósio Brasileiro de Redes de Computadores (SBRC)*.

Dukkipati, N., Kobayashi, M., Zhang-Shen, R., and McKeown, N. (2005). Processor Sharing Flows in the Internet. In *Quality of Service–IWQoS 2005*, pages 271–285. Springer.

Handley, M., Floyd, S., Padhye, J., and Widmer, J. (2003). TCP Friendly Rate Control (TFRC): Protocol Specification. `http://www.ietf.org/rfc/rfc3448.txt`. Último acesso: 25 de junho de 2015.

James F. Kurose, K. W. R. (2010). *Redes de computadores e a Internet: uma abordagem top-down*. Addison Wesley, São Paulo, 5 edition.

Leach, P., Mealling, M., and Salz, R. (2005). A Universally Unique IDentifier (UUID) URN Namespace. http://www.ietf.org/rfc/rfc4122.txt. Último acesso: 25 de junho de 2015.

Liu, Y., Wang, H., Lin, Y., Cheng, S., and Simon, G. (2008). Friendly P2P: Application-level congestion control for peer-to-peer applications. In *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, pages 1–5. IEEE.

Meskovic, M., Bajric, H., and Kos, M. (2012). Content Delivery Architectures for Live Video Streaming: Hybrid CDN-P2P as the best option. In *The Fifth International Conference on Communication Theory, Reliability, and Quality of Service*, pages 26–32.

Sales, L. M. d. (2014). *GMTP: Distribuição de Mídias Ao Vivo através de uma Rede de Favores Constituída entre Roteadores*. PhD thesis, Centro de Engenharia Elétrica e Informática - Universidade Federal de Campina Grande, Campina Grande, PB, Brasil.

Welte, H. and Ayuso, P. N. (1999). The netfilter.org project.

Widmer, J. and Handley, M. (2006). TCP-friendly Multicast Congestion Control (TFMCC): Protocol Specification. *Internet Engineering Task Force, RFC*, 4654.