Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Ciência da Computação

GMTP: Um Protocolo Multi-Camada para Distribuição de Conteúdos Multimídia Ao Vivo com Suporte à Redes Centradas no Conteúdo

Leandro Melo de Sales

Tese de Doutorado submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Doutor em Ciências, domínio da Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Redes de Computadores e Sistemas Distribuídos

Angelo Perkusich e Hyggo Almeida
(Orientadores)

Campina Grande, Paraíba, Brasil

Resumo

O transporte de conteúdos multimídia em tempo real pela Internet é primordial em aplicações como voz sobre IP (VoIP), videoconferência, jogos e WebTV. Ao longo dos anos, observa-se um intenso crescimento de utilização das aplicações cujo cenário é caracterizado por um único nó transmissor e milhares de nós receptores. Por exemplo, o Youtube Live transmitiu os jogos da copa américa 2011 para 100 milhões de usuários. Um ponto crítico nessas aplicações é a sobrecarga de utilização de recursos computacionais nos servidores e canais de transmissão. Isto tem demandado investimentos exorbitantes na construção de Redes de Distribuição de Conteúdos por empresas como Google, Netflix etc. Porém, as aplicações continuam à mercê de protocolos de transportes tradicionais (TCP, UDP, DCCP e SCTP), que não foram projetados para transmitir dados multimídia em larga escala. Para suprir as limitações desses protocolos, os desenvolvedores implementam mecanismo para utilizar os recursos de rede de forma mais eficiente, destacando-se o uso do modelo de serviço P2P (Peer-to-Peer). Todavia, estas soluções são paleativas porque são disseminadas em forma de sistemas ou protocolos de aplicação, sendo impossível evitar a pulverização e fragmentação das mesmas, aumentando-se a complexidade de implantação em larga escala na Internet. Neste trabalho propõe-se um protocolo de transporte multimídia denominado *Global Media Transmission Protocol* (GMTP). O GMTP constrói dinamicamente uma rede de sobreposição entre os nós de uma transmissão (P2P), sem a influência direta da aplicação e com suporte à transmissão multicast e controle de congestionamento. Resultados preliminares apontam que o GMTP utiliza de forma eficiente os recursos de redes e melhora a escalabilidade das aplicações multimídia, evitando-se o retrabalho de desenvolvimento ao concentrar os principais mecanismos em um único protocolo de transporte.

Abstract

The transport of multimedia content in real time over the Internet is essential in applications such as voice over IP (VoIP), video conferencing, games and WebTV. In the last years, there has been an increasing number of applications with a single transmitting node and thousands of receiving nodes. For example, YouTube Live broadcasted games of the American Cup 2011 to 100 million users. A critical aspect in these applications is the overhead of using computing resources on servers and transmission channels. This has demanded exorbitant investment in building Content Delivery Networks (CDNs) by companies like Google, Netflix etc. However, the applications are still at the mercy of traditional transport protocols (TCP, UDP, SCTP and DCCP), which were not designed to transmit multimedia data in a large scale. To address the limitations of these protocols, developers implement a mechanism to use network resources more efficiently, specially using P2P (Peer to Peer) architectures. However, these solutions are palliative because they are disseminated in the form of systems or application protocols, where it is impossible to avoid scattering and fragmentation of them, increasing the complexity of large-scale deployment in the Internet. In this work it is proposed a multimedia transport protocol called Global Media Transmission Protocol (GMTP). The GMTP dynamically builds an overlay network between nodes in a P2P fashion, without the direct influence of the application and supporting multicast transmission and congestion control. Preliminary results indicate that the GMTP efficiently utilizes network resources and improves scalability of multimedia applications, avoiding the rework development by concentrating the main mechanisms in a single transport protocol.

Conteúdo

1 Global Media Transmission Protocol (GMTP) 1

1.1 Visão Geral do GMTP 3

1.1.1 Terminologias e Convenções 6

1.1.2 Arquitetura e Principais Características 8

1.1.3 GMTP Intra e GMTP Inter 10

1.1.4 Canais de Comunicação 11

1.1.5 Tipos de Pacotes 13

1.2 Definições, Relações e Restrições do GMTP 15

1.3 Constituição da Rede de Favores η 19

1.3.1 Registro de participação de r_d em η 19

1.3.2 Seleção de Nós 22

1.3.3 Sobre o melhor caminho W_j 27

1.4 Distribuição de P em η 28

1.4.1 Indexação de Conteúdo 29

1.4.2 Estabelecimento de conexão e compartilhamento para obter P . . . 30

1.4.3 Fase 1: primeira requisição a um fluxo P 30

1.4.4 Fase 2: próximas requisições para obter P 31

1.4.5 Fase 3: busca por mais parceiros r_d para obter P 32

1.4.6 Compartilhamento de P entre s_a 34

1.4.7 Envio e recebimento de $p_x \in P$ em η 35

1.5 Controle de Congestionamento em η 41

1.5.1 Controle de Congestionamento Unicast 42

1.5.2 Controle de Congestionamento Multicast 46

1.6 Outras funções no GMTP 49
1.6.1 Procedimentos para desconexão de nós c_f, l_w e r_d 49
1.6.2 Eleição de nós l_w 50
1.6.3 Segurança 51
1.7 Implemetação e Implantação 52
1.8 Benefícios, Aplicabilidade e Justificativas 52
1.8.1 Benefícios e Aplicabilidade 52
1.8.2 Justificativas 54
1.9 Sumário do Capítulo 55

Lista de Símbolos

3WHS - *Three Way Hand Shake*

BSD - *Berkley Software Distribution*

CCID - *Congestion Control IDentifier*

CPM - *Cooperative Peer Assists and Multicast*

DCCP - *Datagram Congestion Control Protocol*

ECN - *Explicit Congestion Notification*

GMTP - *Global Media Transport Protocol*

HySAC - *Hybrid Delivery System with Adaptive Content Management for IPTV Networks*

IANA - *Internet Assigned Numbers Authority* IETF - *Internet Engineering Task Force*

PDTP - *Peer Distributed Transfer Protocol*

POSIX - *Portable Operating System Interface*

PPETP - *Peer-to-Peer Epi-Transport Protocol*

PPSP - *P2P Streaming Protocol*

RTO - *Retransmission Timeout*

RTT - *Round Trip Time*

SCTP - *Stream Control Transmission Protocol*

Swift - *The Generic Multiparty Transport Protocol*

TCP - *Transport Control Protocol*

TTL - *Time-To-Live*

UDP - *User Datagram Protocol*

Lista de Figuras

1.1 Princípio da Cooperação de Brigadas utilizado no GMTP. 3
1.2 Cenário global de atuação do GMTP. 4
1.3 Rede de sobreposição construída pelo GMTP 6
1.4 Tipos de Nós e modos de conexões do GMTP. 7
1.5 Arquitetura do Protocolo GMTP. 9
1.6 Ferramenta de administração do OpenWRT [1] com suporte ao GMTP. Tela de configuração para permitir registro em um GMTP Inter a uma rede CDN. 11
1.7 Canais de Comunicação do GMTP. 12
1.8 Um usuário precisa descobrir e selecionar seus parceiros. 22
1.9 Cenário e passos para seleção de nós intra caminhos W_j. 23
1.10 Cenário e passos para seleção de nós por interseção de caminhos W_j. 24
1.11 Cenário e passos para seleção de nós por combinação de caminhos W_j. . . 25
1.12 Processo Básico de Estabelecimento de Conexão do GMTP. 31
1.13 Fase 3 de conexão do GMTP (Passo 1). 33
1.14 Fase 3 de conexão do GMTP (Passo 2). 34
1.15 Exemplo da estrutura do buffer de envio e recepção de um nó GMTP com tamanho de 17 p_x. 36
1.16 Exemplo do mapa de buffer de um nó GMTP com tamanho de 17 p_x. 37
1.17 Organização do algoritmo de controle de congestionamento no GMTP. . . . 42
1.18 Um nó r_d mal-intencionados podem poluir o sistema com conteúdos alterados. 51

Lista de Tabelas

1.1 Tipos de Pacotes do protocolo GMTP. 14

Capítulo 1

Global Media Transmission Protocol (GMTP)

O *Global Media Transmission Protocol* (GMTP) é um protocolo de rede multi-camada projetado para operar na Internet em sistemas de distribuição de fluxos dados ao vivo. O GMTP é baseados em uma arquitetura híbrida P2P/CDN e opera nas camadas de rede e transporte da pilha de protocolos TCP/IP. Isto ocorre através da constituição de uma rede de favores P2P, onde os nós cooperam entre si a fim de obterem um conteúdo multimídia de interesse, a ser reproduzido por um processo em execução na camada de aplicação, conectando-se a outros processos através da camada de transporte. O GMTP permite a interação dos nós receptores com servidores de uma ou mais redes CDNs, que atuam como super nós da rede P2P, auxiliando-os no envio e recebimento dos fluxos de dados de eventos ao vivo.

As trocas de dados entre nós GMTP ocorrem por meio do envio e recebimento de pequenas partes do conteúdo de uma mídia, que são transmitidas por diferentes nós da rede, constituindo um fluxos de datagramas IP. Estes fluxos são transmitidos em modo *multicast* ou em múltiplos fluxos *unicast* compartilhados (multi-unicast), realizando-se controle de congestionamento desses fluxos sem garantia de entrega dos datagramas. A escolha do modo de transmissão utilizado para disseminar um determinado conteúdo ocorre automaticamente, de acordo com o suporte oferecido pela rede, priorizando-se o uso do modo *multicast*. Este processo ocorre sem a influência da aplicação, de modo que os sistemas utilizam o GMTP através de uma API compatível com as especificações de socket BSD e POSIX.

O GMTP permite o estabelecimento de conexões entre diversas aplicações, executadas

de forma distribuída em cada sistema final, tornando-as compatíveis entre si, uma vez que o protocolo desacopla a forma como os dados são transportados da forma como estes são exibidos ao usuário final. Isto promove a integração do GMTP em aplicações já existentes, quando se considera futuras adoções, ao tempo que se permite a utilização dos novos recursos introduzidos no protocolo, abstraindo-se a complexidade de construção dos sistemas de transmissão de fluxos de dados de eventos ao vivo.

Nas próximas seções deste capítulo, detalham-se o funcionamento do GMTP, conforme a seguinte organização:

- Na Seção 1.1, apresenta-se uma visão geral do protocolo, como cenário de atuação, arquitetura, canais de comunicação e tipos de nós e pacotes.
- Na Seção 1.2, formaliza-se as definições e restrições do protocolo, que serão utilizadas nas seções subsequentes.
- Na Seção 1.3, descreve-se o processo de constituição da rede de favores, bem como aspectos de conexão multi-ponto através da introdução de um novo conceito de sockets P2P. Aspectos inerantes à constituição de uma rede P2P, como o registro de participação de um nó na rede e o processo de seleção de nós parceiros.
- Na Seção 1.4, discute-se sobre aspectos de transmissão e recepção de fluxos de dados, com os algoritmos utilizados para compartilhar um fluxos de dados e as estratégias de disponibilização e obtenção das partes de uma mídia. Além disso, apresenta-se um arcabouço para dar suporte aos algoritmos de controle de congestionamento, bem como os algoritmos atualmente adotados de acordo com os modos de transmissão suportados.
- Na Seção **??**, apresentam-se outros aspectos relacionados ao GMTP, tais como finalização de conexão, tolerância à desconexão, segurança e integração com outros protocolos de rede.
- Na Seção 1.7, discute-se brevemente sobre aspectos de implementação e implantação do GMTP.
- E, por fim, na Seção 1.8, apresenta-se um resumo sobre os benefícios e as justificativas de decisões de projeto do protocolo GMTP.

1.1 Visão Geral do GMTP

O GMTP é composto por dois módulos chamados de *GMTP Intra* e *GMTP Inter*, que operam na camada de transporte e de rede, respectivamente. O *GMTP Intra* fornece serviços às aplicações de rede a fim de abstrair a complexidade na execução de tarefas comuns a qualquer sistema, tais como conexão multi-ponto, multiplexação/demultiplexação de segmentos IP e controle de congestionamento. O *GMTP Inter* é responsável por constituir uma rede de sobreposição P2P composta por roteadores, os quais funcionam como pontes de acesso aos servidores de uma rede CDN.

Para viabilizar a disseminação de conteúdos multimídia, emprega-se o Princípio da Cooperação de Brigadas (*Bucket Brigade Principle*), onde cada nó repassa o conteúdo recebido para outro nó e assim sucessivamente, análogo a ilustração da Figura 1.1.

Figura 1.1: Princípio da Cooperação de Brigadas utilizado no GMTP.

Na Figura 1.2, observa-se o cenário global de atuação do protocolo GMTP, onde ilustram-se nós *Clientes GMTP* interessados em obter o conteúdo de um determinado evento ao vivo. Neste caso, observa-se também um *Servidor GMTP*, que está conectado a uma rede CDN e atua como fonte geradora de dados. Na prática, os nós *Clientes GMTP* são aplicações de rede capazes de iniciar uma sessão GMTP, que transmitem, recebem e reproduzem dados multimídia de um determinado evento. Os nós *Clientes GMTP* estão conectados a um nó *Repassador GMTP*, que é executado em um roteador de rede e, junto com outros nós *Repassadores GMTP*, efetivamente constituem a rede de sobreposição P2P. Os *Repassadores GMTP* também podem se conecta a um ou mais *Servidores GMTP*. Os *Servidores GMTP* são as fontes de conteúdos multimídia, obtidos através de três formas: i) diretamente a partir de uma unidade geradora de conteúdo (filmadora e/ou microfone); ii) a partir de um *Cliente GMTP*; e/ou iii) a partir de outro *Servidor GMTP* (troca de dados entre os servidores da CDN). Os *Servidores GMTP* recebem sinalizações de controle contendo requisições dos

nós *Repassadores GMTP*, que ao receberem uma resposta correspondente a sua requisição, atendem a demanda de um nó *Cliente GMTP*.

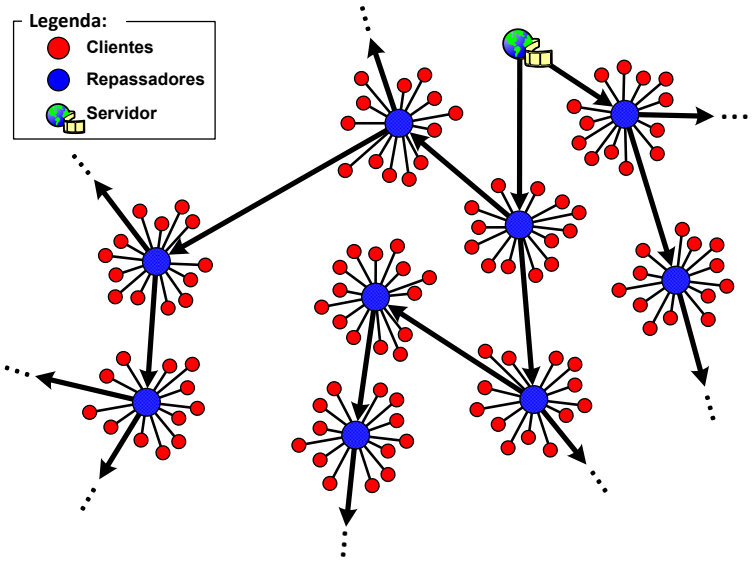

Figura 1.2: Cenário global de atuação do GMTP.

Quando um nó *Cliente GMTP* deseja reproduzir um determinado evento, este envia uma requisição destinada ao nó *Servidor GMTP* que está transmitido o conteúdo de interesse, como atualmente acontece em qualquer conexão na Internet. A diferença é que um pedido de conexão é interceptado por algum nó *Repassador GMTP* durante o trajeto do pedido de conexão até o nó *Servidor GMTP*, que então determina os melhores parceiros para atendê-la. Em geral, isto ocorre já no roteador de borda do *Cliente GMTP*, que funciona como nó *Repassador GMTP* de origem. Caso o nó *Repassador GMTP* não encontre nenhum nó parceiro capaz repassar a mídia de interesse, este encaminha tal requisição ao nó *Servidor GMTP* que transmite a mídia correspondente. Em todo caso, sempre o nó *Repassador GMTP* de origem assumirá o controle da requisição do *Cliente GMTP*, habilitando-se como candidato a parceiro para outros nós *Repassadores GMTP*, quando motivados por requisições originadas pelos seus *Clientes GMTP*.

O posicionamento dos nós *Repassadores GMTP* e suas habilidades permitem a redução do número de fluxos de dados correspondente a um mesmo evento e ainda uma maior escalabilidade do número de nós *Clientes GMTP* interessado em receber um mesmo fluxo de dados. Por este mesmo motivo, o protocolo GMTP é flexível para permitir que um nó *Repassador GMTP* atue somente encaminhando conteúdos multimídias entre duas ou mais

redes distintas, mesmo que este não esteja conectado a nenhum nó *Cliente GMTP* interessado por tal conteúdo. Desta forma, maximiza-se o uso de canais de transmissão ociosos, em particular das redes residenciais, as quais seus usuários muitas vezes estão ausentes e portanto sem fazer uso dos recursos disponíveis, não necessitando, inclusive, manter um determinado computador da sua rede interna ativo (ligado), como é obrigatório em todas as outras soluções para este fim.

Pelo princípio da cooperação de brigadas empregado no GMTP, as requisições de conexão podem ser originados não apenas por nós *Clientes GMTP* para seu respectivo nó *Repassador GMTP*, mas também as requisições podem ocorrer entre nós *Repassadores GMTP* que, motivados pelos interesses dos seus nós *Clientes GMTP*, podem formar parcerias entre si. Isto significa que um nó *Repassador GMTP* pode agir como se fosse um nó *Servidor GMTP*, respondendo às requisições originadas por seus nós *Clientes GMTP* ou de outros nós *Repassadores GMTP*, como se a requisição estivesse alcançado o *Servidor GMTP* que oficialmente transmite o conteúdo, o que ocorre de forma transparente para a aplicação.

As estratégias discutidas até aqui e adotadas no GMTP são diferenciais que permitem o protocolo disseminar mais rapidamente um determinado fluxo de dados originado no *Servidor GMTP*. Como a rede de favores e os dados são trocados de forma distribuída, ou seja, nem sempre com a participação de um *Servidor GMTP*, empregou-se também um mecanismo para validação dos dados transmitidos pelo servidor, evitando-se ataques de poluição, por exemplo. Além disso, incorporou-se um mecanismo no GMTP chamado de *registro de participação* que, após um nó *Repassador GMTP* se registrar em um nó *Servidor GMTP*, permite-se que os *Servidores GMTP* determinem quais são os candidatos a perceiros de um nó *Repassador GMTP*, o que ocorre periodicamente. A vantagem disso é que se permite, *a priori*, que os nós *Repassadores GMTP* avaliem seus parceiros sem necessariamente um nó estar recebendo um fluxo de dados de um determinado evento. Com isto, um nó *Repassador GMTP* pode repassar um fluxo de dados para um outro nó *Repassador GMTP* sem que o primeiro tenha interesse no referido fluxo, mas devido ao seu posicionamento na rede e sua capacidade computacional e de rede, pode melhorar sobremaneira no processo de disseminação de um determinado fluxo de dados.

Na Figura 1.3, observam-se detalhes do cenário supracitado, introduzindo-se o conceito de um grupo especial de nós chamados de nós *Relatores GMTP*. Estes nós são responsáveis

por enviar relatórios periódicos sobre o estado da transmissão ao seu nó *Repassador GMTP*, que os utiliza para regular a taxa de transmissão de um ou mais fluxos de dados, impedindo-se que a rede entre em colapso de congestionamento. Na prática, os nós *Repassadores GMTP* são nós *Clientes GMTP*, eleitos automaticamente pelo seu nó *Repassador GMTP*, dentre os nós *Clientes GMTP* interessados em obter um determinado fluxo de dados.

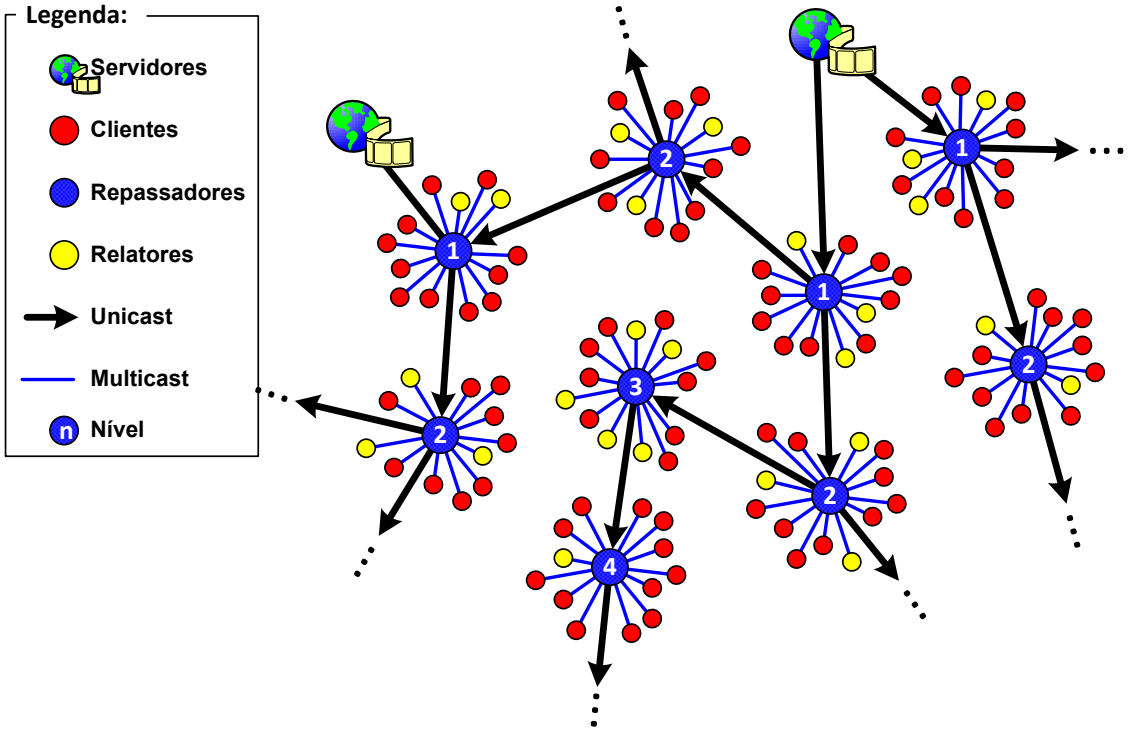

Figura 1.3: Rede de sobreposição construída dinamicamente pelo GMTP com a presença de nós repassadores e relatores.

1.1.1 Terminologias e Convenções

Nesta seção, apresentam-se algumas definições, terminologias e convenções utilizadas no restante deste documento, de acordo com a Figura 1.4.

Tipos de Nós:

- **Nó GMTP ou Processador GMTP:** qualquer processador de rede que implementa o protocolo GMTP. É um sistema computacional que implementa parte ou todo do protocolo GMTP, sendo capaz de interpretar os cabeçalhos dos pacotes definidos pelo

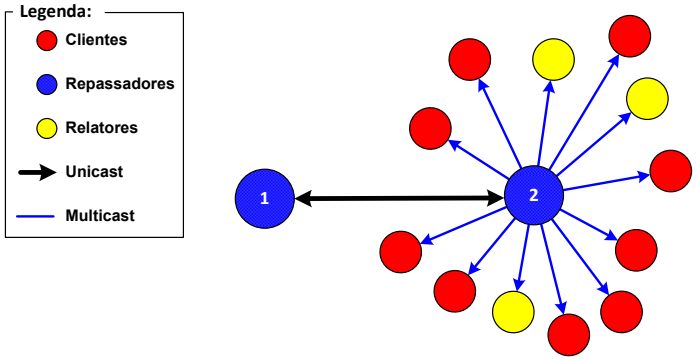

Figura 1.4: Tipos de Nós e modos de conexões do GMTP.

GMTP e realizar ações pré-definidas. Não há restrições de qual tipo de processador de rede pode implementar qual(is) parte(s) do GMTP.

- **Cliente GMTP:** é um *nó GMTP* capaz de reproduzir e gerar conteúdos multimídia ao vivo. Em geral, um *Cliente GMTP* é um sistema final que executa um processo a nível de sistema operacional, representando uma aplicação manipulada pelo usuário final. A maioria dos *Clientes GMTP* funciona apenas de forma passiva, recebendo o fluxo de dados de um conteúdo multimídia e entregando para um processo em execução.

- **Servidor GMTP:** é um *nó GMTP* capaz de gerar ou receber, de um *Cliente GMTP*, conteúdos multimídia ao vivo. Em geral, um *Servidor GMTP* é um sistema final que compõe uma rede CDN.

- **Repassador GMTP:** é um *Nó GMTP* com habilidades de repassar os fluxos de dados originados de um ou mais *Servidores GMTP* ou de um outro nó *Repassador GMTP*.

- **Relator GMTP:** é um *Cliente GMTP* com habilidades de enviar relatórios periódicos sobre o estado da transmissão ao repassador.

Modos de Transmissão:

- ***Unicast*:** toda comunicação que ocorre entre dois nós *Repassadores GMTP*, com a interpretação do conceito definido por *unicast*, no contexto de redes de computadores, em sua forma tradicional.

- ***Multi-unicast*:** é um conjunto formado por dois ou mais canais de Canal Transmissão *unicast*.

- ***Multicast*:** toda comunicação que ocorre entre um nó *Repassador GMTP* e seus respectivos *Clientes GMTP*, com a interpretação do conceito definido por *multicast*, no contexto de redes de computadores, em sua forma tradicional.

O modo multicast sempre é utilizado, porém quando este modo não é suportado pela rede, o modo multi-unicast do protocolo é executado. É requerido que o modo multicast seja utilizado para transmissões de um salto, ou seja, em redes locais. O modo unicast é utilizado para que *Clientes GMTP* estabeleçam uma conexão com um *Servidor GMTP* ou um *Repassador GMTP* e passe a distribuir o conteúdo de dados multimídia em sua rede local.

Deste ponto em diante, os termos *Nó GMTP*, *Cliente GMTP*, *Servidor GMTP*, *Repassador GMTP* e *Relator GMTP* serão utilizados em sua forma simplificada, ou seja, *nó*, *cliente*, *servidor*, *repassador* e *relator*, respectivamente. Além disso, estes termos não serão mais formatados em itálico, bem como os termos *socket*, *unicast*, *multi-unicast* e *multicast*. Ademais, quando o termo *transmissão* ou *transmissão de um evento* for mencionado, denotar-se-á a transmissão de um fluxo de datagramas IP correspondente a um evento ao vivo, utilizando-se o protocolo GMTP.

As palavras “deve”, “não deve”, “requerido”, “pode”, “não pode”, “recomendado” e “opcional”, incluindo suas variações morfológicas, devem ser interpretadas como descrito na RFC 2119 [2], em inglês.

1.1.2 Arquitetura e Principais Características

Na Figura 1.5, ilustra-se a arquitetura geral do GMTP, seguida de uma breve descrição das suas principais funcionalidades.

- Registro de participação de um nó repassador em um nó servidor. Isto permite o suporte a pré-seleção de nós parceiros filtrados por métricas que influenciam na qualidade de experiência do usuário ao assistir um evento ao vivo, como atraso fim-afim.

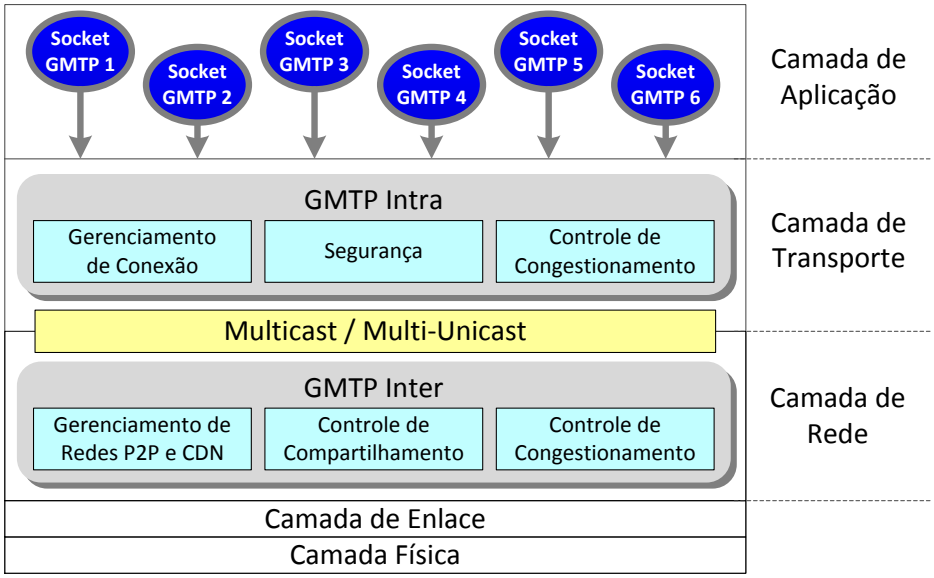

Figura 1.5: Arquitetura do Protocolo GMTP.

- Acesso a uma transmissão através de um processo de conexão em três-vias (*3WHS*), com a requisição de conexão transmitida ao servidor e podendo ser interceptada por um nó repassador em seu trajeto ao servidor, com suporte automático de detecção e uso dos modos de transmissão suportados pelo nó repassador.

- Descoberta de nós parceiros entre redes distintas e negociação de parcerias, com suporte a formação de parcerias baseadas em métricas que influenciam na qualidade de experiência do usuário.

- Envio e recebimento de fluxos de dados compartilhados entre nós da mesma rede através de multicast e uso de fluxos unicast entre redes distintas, porém sem a relação de uma conexão por cliente e assim evitando o fenômeno da tragédia dos bens comuns, discutido na Seção **??**.

- Suporte a algoritmos de controle de congestionamento assistidos pela rede e de fluxos multicast. Troca de relatórios periódicos entre os nós repassadores sobre a transmissão para dar suporte aos algoritmos de controle de congestionamento, além de ajudar na otimização do processo de seleção de nós parceiros.

- Eleição de nós relatores com suporte a tolerância a desconexões de nós, com notifica-

ção e reeleição de novos nós.

- Garantia de autenticidade das partes de uma mídia por meio do uso de certificados digitais para impedir ataques de poluição.

1.1.3 GMTP Intra e GMTP Inter

- **GMTP Intra:** é executado na camada de transporte da pilha de protocolos TCP/IP e corresponde à parte interna de uma rede, composta por vários nós clientes que possuem um nó repassador. É executado em um sistema final, operado por um usuário que deseja assitir a um evento ao vivo, através de uma aplicação que cria um socket GMTP. Um socket GMTP é a representação de uma instância do protocolo GMTP em execução, sendo responsável por gerenciar todas as atividades de comunicação da aplicação correspondente com o meio externo (outros processos GMTP). No contexto de uma conexão, o GMTP Intra mantém diversas variáveis de estado que representam uma instância do GMTP em execução, que executa algoritmos para: eleição de nós relatores, controle de congestionamento, validação de partes de uma mídia e multiplexação e demultiplexação de datagramas IP.

- **GMTP Inter:** é executado na camada de rede da pilha de protocolos TCP/IP e corresponde à parte externa de uma rede, composta por vários nós repassadores que cooperam entre si. É executado por um roteador de rede e aceita conexões oriundas de um nó cliente ou de um nó repassador. No contexto de uma conexão, o GMTP Inter mantém diversas variáveis de estado relacionadas às funções de sua responsabilidade: estabelecimento de conexão com nós servidores ou repassadores, seleção de nós repassadores parceiros, eleição de nós relatores, controle de congestionamento assistido pela rede, controle de compartilhamento de fluxos multimídia e validação da autenticidade das partes de uma mídia.

  Além disso, no GMTP Inter, permite-se a configuração de parâmetros iniciais de configuração da rede de cooperação e da integração com servidores de uma ou mais CDN, como ilustrado na Figura 1.6. Nesse caso, o usuário pode configurar seu roteador como nó repassador para se autenticar em uma ou mais redes CDN, definir a largura de banda (*download* e *upload*) que deseja compartilhar, o período (dias da semana e horas) que

permitirá seu roteador funcionar como repassador, dentre outras opções. É possível também que um repassador GMTP detecte automaticamente o suporte do GMTP em um servidor remoto, portanto sem a necessidade de uma configuração prévia.

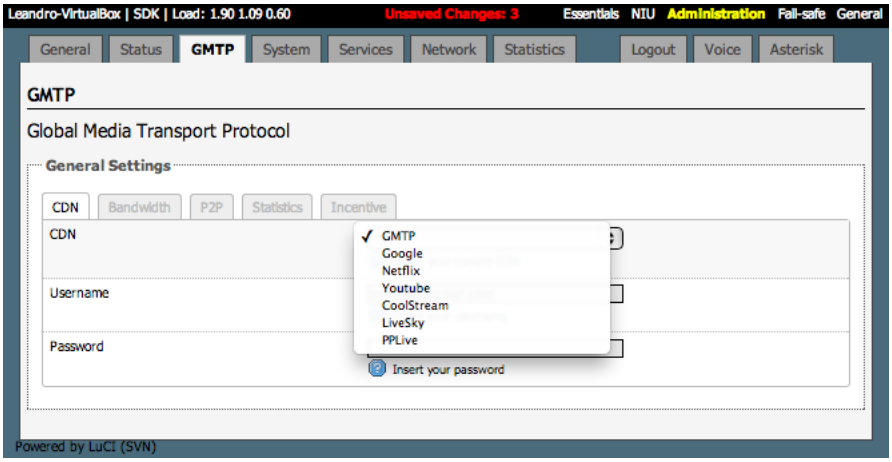

Figura 1.6: Ferramenta de administração do OpenWRT [1] com suporte ao GMTP. Tela de configuração para permitir registro em um GMTP Inter a uma rede CDN.

1.1.4 Canais de Comunicação

No protocolo GMTP, utilizam-se três canais de comunicação para implementar suas funcionalidades (Figura 1.7). Embora alguns autores considerem os termos "repasse" e "roteamento" como conceitos distintos, neste trabalho ambos os termos são considerados sinônimos e devem ser interpretados como a capacidade que um nó GMTP tem de receber dados em uma interface de rede de entrada e encaminhar estes dados através de uma interface de rede de saída, permitindo-se que uma mesma interface de rede seja utilizada como entrada e saída ao mesmo tempo.

Canal de Controle:

Todo nó ao iniciar uma instância do protocolo GMTP deve criar um socket multicast no endereço IP 238.255.255.250 e na porta 1900. Através desse socket, um nó GMTP é capaz de enviar e receber pacotes de controle utilizados para implementar as funcionalidades do protocolo. Este socket, denominado *canal de controle* é suma importância no processo de conexão do GMTP porque é utilizado para enviar e receber sinais de controle, principalmente entre um nó repassador e um nó cliente.

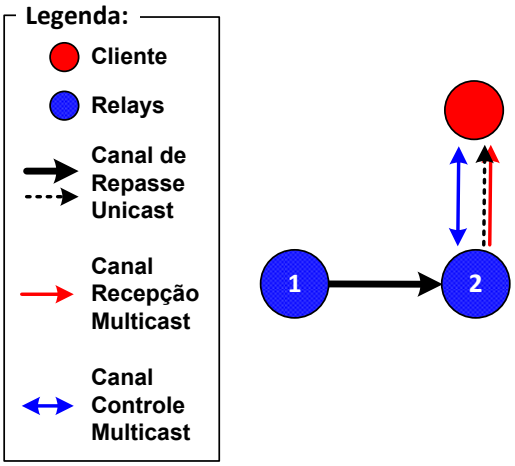

Figura 1.7: Canais de Comunicação do GMTP.

O canal de controle é utilizado para permitir que o nó repassador realize o processo de eleição e reeleição de nós relatores, procedimentos de estabelecimento de conexão, descoberta de nós e notificações de desconexões, envio e recebimento de relatórios para controle de congestionamento, entre outros. Tais recursos são fundamentais para o funcionamento do GMTP, sendo umas das justificativas de sua originalidade, diferentemente de qualquer outro protocolo encontrados na literatura.

A decisão do uso do endereço IP multicast 238.255.255.250 foi baseada na RFCs 2365 [3], que define o escopo administrativo do uso dos endereços multicast entre 239.0.0.0 e 239.255.255.255. O endereço 238.255.255.250 é definido no escopo de uso global e por este motivo esse endereço foi o escolhido. Com a padronização do protocolo GMTP e a publicação da sua RFC, será necessário registrar o uso desse endereço IP multicast ao *Internet Assigned Numbers Authority* - IANA[1].

Canal de Repasse:

Além do canal de controle, define-se no protocolo GMTP um canal de repasse utilizado por um nó repassador para encaminhar datagramas vindos de um servidor ou de outro repassador para a rede local. Esse canal de repasse, na prática, é um socket multicast criado pelo repassador para transmitir os datagramas para todos os seus clientes com interesse em reproduzir o mesmo fluxo de dados de um evento ao vivo.

O *socket de repasse dos dados* deve ser criado quando um repassador passa a obter um

[1]IANA: http://www.iana.org/

determinado fluxo de dados correspondente a um determinado evento de interesse de pelo menos um dos seus clientes. Na prática, quando isto acontece, o repassador deve criar um socket multicast em um endereço IP e número de porta escolhida aleatoriamente para repassar os dados vindos do servidor ou de outro repassador para dentro de sua rede. Isto permitirá alcançar um número maior de clientes em uma rede local, melhorando a escalabilidade do protocolo do ponto de vista da quantidade de clientes. A faixa de endereços IP multicast que um cliente deve utilizar para criar seu socket de repasse é a de escopo local 239.192.0.0/14, definida na RFC 2365 [3]. Como é uma faixa de endereços IP multicast de domínio local, não se faz necessário registrar o uso dessa faixa de endereços.

Um outro canal de repasse pode ser criado por um repassador A caso um outro repassador B esteja interessado por um fluxo de dados disponível no repassador A. Nesse caso, o repassador A negocia um canal de repasse unicast e informar ao repassador B o endereço IP e porta que este deve se conectar para obter os dados.

Canal de Recepção de Dados:

O canal de recepção de dados é um socket multicast criado por um cliente para receber um fluxo de dados transmitindo por um nó repassador em algum endereço IP da faixa 239.192.0.0/14. Alternativamente, um canal de recepção de dados será um socket unicast quando existir apenas um cliente em uma rede interessado por um fluxo de dados, utilizando-se o endereço da sua própria interface de rede local. O endereço IP e o número de porta que o cliente deve se conectar é determinado pelo nó repassador no momento da conexão.

1.1.5 Tipos de Pacotes

Toda comunicação entre dois ou mais nós GMTP ocorre através da troca de datagramas IP, que carregam sinalizações de controle e/ou dados da aplicação.

No cabeçalho dos pacotes GMTP existe um campo denominado *tipo do pacote*. Este campo determina que informação está contida em um determinado pacote GMTP e, ao processá-los, um nó GMTP executa uma determinada ação. Na Tabela 1.1 são apresentados os possíveis valores desse campo, nome do pacote e sua função. Nas próximas seções serão descritas todas as funções do GMTP e os diferentes tipos de pacotes serão utilizados

para descrever os algoritmos no tocante às ações realizadas pelo GMTP de acordo com um determinado evento.

Tabela 1.1: Tipos de Pacotes do protocolo GMTP.

#	Tipo	Descrição
0	Request	Pedido de estabelecimento de conexão multicast
1	Response	Resposta ao pedido de estabelecimento de conexão multicast
2	Data	Contém dados da aplicação
3	Ack	Confirmação de recebimento de pacote
4	DataAck	Dados da aplicação e confirmação de recepção
5	Elect	Inicia o processo de eleição de um nó em relay ou reporter
6	ElectReply	Sinaliza o interesse de um nó em se transformar em relay ou reporter
7	ElectAck	Confirmação do nó eleito para relay ou reporter
8	RelayQuery	Transmitido por um nó para consultar a lista de relays
9	RelayReply	Resposta ao pedido de consulta da lista de relays
10	AdvConn	Utilizado por um nó relay ou reporter para anunciar que está ativo na rede
11	Reservado	Uso futuro e ignorado pelo receptor
12	Reservado	Uso futuro e ignorado pelo receptor
13	CloseReq	Servidor ou Relay solicita término de conexão sem TIMEWAIT
14	Close	Servidor/Cliente/Relay solicita término da conexão
15	Reset	Determina, incondicionalmente, o final da conexão

No Apêndice **??** são apresentados detalhes acerca do uso dos tipos de pacotes do GMTP, sendo seu teor completamente técnico. Desta forma, tal apêndice é dedicado aos leitores interessados em sua implementação.

FALAR: - O conteúdo do relatório e como esses dados são obtidos, qual a frequencia que são enviados - dizer que o no repassador pode repassar pacotes mesmo se não houver nó interessado pelo fluxo correspondente - Falar que os protocolos não são apropriados para os cenários considerados - Estudar e escrever sobre o RCP (base de CC para a rede intra); XCP; LEDBAT e ConEx. Arrumar justificativa para não usar o ConEx. Dizer que o GMTP pode englobar o uso de todos estes, caracterizando uma justificativa de que o GMTP é preparado

para atendê-los após uma vasta revisão bibliográfica! - Declaração da tese: não esquecer de falar sobre uso do roteador e a unificação dos sistemas, além de que as rotas não mudam sempre porque as origens (servidores da cdn) são poucas

- GMTP * Escrever o algoritmo de CC para a rede P2P/roteador * Algoritmo de seleção de peer e parcerias * Escrever algoritmo para o mapa de buffer * Escrever sobre o Inter o Intra * Reorganizar como nazareno mencionou

ver: http://peerstreamer.org

Anotações após defesa: - Ler o survey: http://www.aicit.org/jcit/ppl/- Simuladores mais robustos: SimGrid [23], OPSS [26, 27], ChunkSim [29], 3LS [30], OPNET [30] (não sei se vai ser bom!) - - Resultados * Comentar gráficos * Lembrar de falar sobre redes móveis * Sobre churn (roteador pode reduzir churn) - Implementação - Trabalho futuro: * Avaliar com o ConEx??? * Terminar a implementação no kernel do Linux * RFC - Corrigir capítulo de fundamentação (rever comentários do Nazareno) - Fazer Apendice explicando onde obter os códigos de simulação e implementação - Trabalho futuro tb é discutir aspectos de padrão de tráfego por dia/hora, por exemplo, e formar parcerias nesse sentido.

1.2 Definições, Relações e Restrições do GMTP

Nesta seção, descrevem-se formalmente definições, relações e restrições do protocolo GMTP. Para isto, faz-se uso de fundamentos da algebra booleana, lógica proposicional, teoria de conjuntos e teoria dos grafos [4–7].

1. Seja $B = \{b_1, b_2, b_3, \ldots, b_e\}$, onde $e \in \mathbb{N}$, o conjunto dos roteadores de uma rede de computadores, por exemplo, a Internet, tal que existe uma relação $R \rightarrow B$ que determina que os nós repassadores $r_d \in R$ podem sobrepor os roteadores em B (*overlay network*).

2. Seja $S = \{s_1, s_2, s_3, \ldots, s_a\}$, onde $a \in \mathbb{N}$, o conjunto finito de todos os nós servidores.

3. Seja $R = \{r_1, r_2, r_3, \ldots, r_d\}$, onde $d \in \mathbb{N}$, o conjunto finito de todos os nós repassadores.

4. Seja $C = \{c_1, c_2, c_3, \ldots, c_f\}$, onde $f \in \mathbb{N}$, o conjunto finito de todos os nós clientes.

5. Seja $(\mathbb{P}, \prec) = \{p_1, p_2, p_3, \ldots, p_h\}$, onde $h \in \mathbb{N}$, o conjunto *totalmente ordenado (toset)*, sem repetição, dos pacotes de dados gerados pelos nós $s_a \in S$ durante a transmissão de um evento ao vivo $\mathcal{E}$. Note que o símbolo $\prec$ é utilizado para representar precedência entre dois elementos diferentes.

6. Seja $\mathcal{T} = \{S, R, C_i, P\}$, as relações e restrições estabelecidas entre os diferentes tipos de nós participantes de uma transmissão de um evento ao vivo $\mathcal{E}$, tal que:

   - Seja $C_i : r_d \rightarrow 2^C$, $\forall r_{d_1}, r_{d_2} \in R$, $C_i(r_{d_1}) \cap C_i(r_{d_2}) = \{\emptyset\}$, uma função tal que $C_i(r_d)$ denota todos os clientes relacionados a um nó repassador r_d, de modo que nenhum nó $c_f \in C$ pode estar relacionado com dois ou mais nós r_d;
   - $(P, \prec) = \{p_1, p_2, p_3, \ldots, p_x\}$, onde $x \in \mathbb{N}$. Isto é, o conjunto *parcialmente ordenado (poset)* dos pacotes de dados p_x transmitidos por um nó s_a ou por um nó r_d, também chamado de fluxo de pacotes de dados ou apenas fluxo de dados, de modo que:
     - Um fluxo de dados P é dito *fluxo completo*, representado por $P^\bullet$, se e somente se $P \leftrightarrow \exists \mathbb{P}_\theta$ (relação bijetora), tal que $\mathbb{P}_\theta \in 2^{\mathbb{P}}$ e $\mathbb{P}_\theta \neq \{\emptyset\}$. Ou seja, um *fluxo completo* $P^\bullet$ é um conjunto *toset* e portanto não apresenta lacunas;
     - $\rho : (p_x, P) \rightarrow (P^\lhd, \prec) = \{p_x, p_{x+1}, p_{x+2}, p_{x+3}, \ldots\}$, tal que $P^\lhd \subset P$, uma função $\rho(p_x, P)$ que define um sub-fluxo de pacotes de dados P a partir de um determinado pacote $p_x \in P$. Neste caso, como $P^\lhd \subset P$, se $P^{\lhd\bullet} \rightarrow P^\bullet$.

7. Seja $L = \{l_1, l_2, \ldots, l_w\}$, o conjunto finito de todos os nós relatores de uma transmissão $\mathcal{T}$, onde:

   - $L_\theta \subset L$, tal que $L_\theta \in 2^{C_i(r_d)}$ e $L_\theta \neq \{\emptyset\}$, o conjunto de nós relatores de um nó repassador r_d. Ou seja, qualquer nó cliente $c_f \in C_i(r_d)$ pode atuar como um nó relator l_w de um nó repassador r_d, pois $C_i(r_d) \cup L_\theta = C_i(r_d)$.

8. Seja $T = \{t_u \mid t_u \in Z \text{ e } \varphi(s_a, P) = 1\}$, tal que $Z = S \cup R$ e $u \in \mathbb{N}$, o conjunto dos nós s_a e r_d que transmitem ou repassam os pacotes de dados $p_x \in P$, onde:

- $\varphi : (t_u, P) \rightarrow \{0, 1\}, \forall (t_u, P) \in \{T \times \{P\}\}$, uma função booleana que determina se um nó $t_u \in T$ transmite o fluxo de dados $p_x \in P$ de um evento $\mathcal{E}$, onde 0 e 1 denotam, respectivamente, *falso* e *verdadeiro*.

9. Seja $\eta = G(Z, W)$, um grafo formado com vértices de nós servidores e repassadores, que podem estar interligados entre si por um conjunto de diferentes caminhos W, onde:

   - $W = \bigcup_{m=1}^{j} W_j$, onde $j \in \mathbb{N}$ é a quantidade de todos os possíveis caminhos W_j. Isto é, W é o conjunto de todos os caminhos por onde o fluxo de dados P, pode ser transmitido, sendo um caminho definido por um conjunto $(W_j, \prec)$ totalmente ordenado (*toset*):
     - $(W_j, \prec) = \{w_m \mid s_a, r_1, r_2, r_3, \ldots, r_d\}$, tal que $s_a, r_d \in Z$, tal que $\forall w_m, w_{m+1} \in W_j : w_m \prec w_{m+1}$ e $W_j \neq \{\emptyset\}$. Isto é, W_j é um conjunto de nós que denota um dos possíveis caminhos por onde um fluxo de dados $p_x \in P$ pode ser transmitido, obrigatoriamente a partir de um nó servidor s_a, chamado de pivô ou raiz, até um nó r_1;
     - Um caminho W_j é dito *caminho semi-completo*, representado por W_j°, se e somente se $W_j \leftrightarrow \exists B_\theta$ (bijetora), tal que $B_\theta \in 2^B$ e $B_\theta \neq \{\emptyset\}$. Isto é, todos os roteadores $b_e \in B$ são sobrepostos por um nó $r_d \in W_j^{\circ}$;
     - Um caminho W_j é dito *caminho completo*, representado por $W_j^{\bullet}$, se e somente se tal caminho for W_j° e se $(W_j^{\circ} - \{s_a\}) \subset T$. Isto é, um caminho é completo se para todo nó $\forall w_m \in W_j$, desconsiderando o nó s_a, todos os nós r_d são transmitem o fluxo de dados P.

   - Seja $\sim : (W_j, \prec) \rightarrow (W_j, \succ)$, uma função reversa $\sim(W_j)$ de um conjunto *toset*. Isto é, se $(W_j, \prec) = \{w_m \mid s_a, r_1, r_2, \ldots, r_d\}$, então $\sim(W_j) = \{w_m \mid r_d, r_{d-1}, r_{d-2}, \ldots, r_1, s_a\}$;

   - Seja $\delta : (t_u, W_j) \rightarrow (W_j^{\triangleleft}, \prec) = \{t_u, t_{u-1}, t_{u-2}, \ldots, t_2, t_1\}$, tal que $W_j^{\triangleleft} \subset W_j$, uma função $\delta(t_u, W_j)$ que define um sub-caminho de W_j a partir de um nó $t_u \in W_j$ até um nó $t_1 \in W_j$. Neste caso, como $W_j^{\triangleleft} \subset W_j$, tem-se que $W_j^{\circ} \rightarrow W_j^{\triangleleft\circ}$ ou $W_j^{\bullet} \rightarrow W_j^{\triangleleft\bullet}$;

- O custo total para transmitir um pacote $p_x \in P$, através de um caminho W_j, é denotado pela função $\zeta : W_j = \sum_{j=1}^{|W_j|} \gamma(w_m, w_{m+1})$, tal que γ é uma função que determina o custo para transmitir o pacote p_x entre dois nós distintos $\forall w_m, w_{m+1} \in W_j$: $w_m \prec w_{m+1}$. No GMTP, utiliza-se a métrica atraso entre dois nós distintos para realizar o cálculo de custo, podendo-se utilizar outras métricas, como número total de saltos no caminho W_j;
- Conjectura 1: $\forall r_d \in R$ e $\forall c_f \in C$, r_d é mais estável que qualquer c_f com relação a sua disponibilidade e participação em uma rede de favores η. No ponto de vista das redes de computadores, um roteador torna-se indisponível com menos frequência (desconecta-se) se comparado com seus nós clientes. Como se sabe, nas transmissões de dados na Internet, a participação de um roteador no processo de transmissão de um fluxo de dados qualquer é fundamental, mesmo que seja apenas para rotear os pacotes de tal fluxo. Assim, dado que para um cliente receber um determinado pacote primeiramente este pacote deve passar pelo seu roteador padrão (de aborda), quando um roteador se desconecta, nenhum de seus clientes será capaz de receber os pacotes de dados, porém a recíproca não é verdadeira. Ou seja, caso um cliente se desconecte, outros clientes em sua mesma rede continuarão sendo capazes de receber um fluxo de dados encaminhados pelo seu roteador;
- Conjectura 2: as tabelas de roteamento dos nós $w_m \in W_j$ não mudam frequentemente e são independentes umas das outras. No mundo real (Internet), as rotas entre quaisquer nós c_{f_1} e $c_{f_2} \in C$ não se alteram com um nível de frequência que desestabilize a comunicação entre estes, apesar de se tratar de uma rede comutada por datagramas IP. Mesmo se estas mudanças ocorrerem em uma rota de um caminho W_j, o impacto causado é temporário e insignificante no ponto de vista da qualidade de experiência do usuário ao assistir a um evento $\mathcal{E}$.

Desta forma, η representa formalmente a rede de sobreposição constituída pelo GTMP (informalmente ilustratada na Figura 1.3), definindo-se as relações e restrições estabelecidas em $\mathcal{T}$ e as conjecturas consideradas para a execução de tal protocolo.

Nas próximas seções, descrevem-se as funcionalidades do GMTP com base em η, explicando-se os mecanismos computacionais empregados no protocolo para a transmissão de um evento ao vivo $\mathcal{E}$. Para isto, utiliza-se uma abordagem *top-down*, a partir do processo de constituição de η, em seguida discutindo-se aspectos inerantes ao processo de distribuição de conteúdos multimídia, que envolvem os nós em $\mathcal{T}$, C e S, e suas funções específicas.

1.3 Constituição da Rede de Favores η

A constituição da rede de favores η ocorre por meio do registro de participação de um ou mais $r_d \in R$ a um ou mais $s_a \in S$, direta ou indiretamente por meio de outros repassadores $r_q \in R$. Todo esforço realizado nesse processo é visando otimizar a transmissão de um fluxo de pacotes $p_x \in P$ a um cliente $c_f \in C$, podendo ser distribuído pelos nós r_d por meio de diferentes caminhos $W_j \in W$. Por este motivo, o GMTP busca determinar um caminho sub-ótimo W_θ para que a experiência de um usuário ao assistir um evento $\mathcal{E}$ seja a melhor possíveis, segundo as métricas estabelecidas anteriormente. Em outras palavras, busca-se sempre o melhor caminho W_θ tal que $\zeta(W_\theta) < \zeta((\forall W_j \in W \text{ e } W_\theta \neq W_j))$ e, sempre que possível, com $W_\theta^\bullet$.

1.3.1 Registro de participação de r_d em η

O procedimento de registro de participação de um nó r_d em uma rede η é o primeiro passo, e um dos mais importantes. Um nó r_d envia uma mensagem de registro de participação a um nó s_a utilizando o pacote GMTP_REGISTER, que constituirá um caminho W_j. Para isto, todos os nós r_d no caminho entre r_d e s_a alteram o pacote GMTP_REGISTER, adicionando seu identificador único que, na prática, é o endereço IP. Quando o pacote GMTP_REGISTER alcança seu destino (s_a), nele conterá a lista ordenada de todos os nós repassadores até s_a, que de fato é um caminho W_j entre r_d até s_a. Tal informação é enviada de volta ao nó r_d por meio do pacote GMTP_REGISTER_REPLY. Posteriormente, W_j é utilizado no processo de formação de parcerias, a ser discutido mais adiante na Seção 1.3.2.

O interessante no GMTP é que o registro de participação ocorre quando um nó deseja participar da rede de sobreposição, não necessariamente quando se deseja obter um fluxo de dados multimídia para assistir a um evento $\mathcal{E}$. Em ambos os casos, o algoritmo de registro

de participação é o mesmo. A diferença primordial é que se um r_d solicitar previamente um registro de participação a um s_a, inicialmente sem interesse por um evento $\mathcal{E}$ qualquer, será possível mapeá-lo mais rapidamente para seleção de um subconjunto de possíveis nós parcerios r_q. Neste caso, pode-se utilizar r_d para repassar pacotes de dados p_x mesmo quando $C_i(r_d) = \{\emptyset\}$. De forma similar, caso $\exists c_f \in C_i(r_d)$ interessado em $\mathcal{E}$ e $\varphi(r_d, P) = 1$, pode-se reduzir o tempo de início de reprodução do fluxo de dados P, bastando para isto r_d começar a encaminhar o fluxo de dados P para o nó c_f, o que ocorre em modo multicast. Este assunto será retomado na Seção **??**.

No Trecho de Código 1, apresenta-se o pseudo-algoritmo utilizado por um nó repassador r_d para se registrar a um nó servidor s_a. Note que o processo de registro não requer que um nó repassador r_d registre qualquer interesse sobre um evento $\mathcal{E}$. Além disso, no final do procedimento do registro de participação, o nó r_d passa a conhecer sobre W_j através do pacote GMTP_REGISTER_REPLY (linha 8). Note também que no GMTP toda transferência de pacotes de controle é feita com garantia de entrega, com os nomes das funções representadas com o sufixo *Rdt* (*reliable data transfer*).

Uma outra decisão importante tomada no GMTP é que um nó repassador r_d deve periodicamente sinalizar sua participação na rede de favores (*keep-alive*), ou seja, após o registro de participação, o repassador r_d precisa enviar periodicamente sinalizações de controle (*polling*) sobre sua participação na rede de favores. Um repassador r_d também sinaliza sua desconexão a s_a quando não desejar mais participar da rede de favores η.

Algorithm 1: registerRelay(s_a, c_f = nulo)

```
1  servs ← getCurrRegedServers (); /* servs ⊂ S */
2  pkt ← makePkt (GMTP_REGISTER, s_a); /* register request */
3  if pkt = OK then
4      sendPktRdt (pkt); /* send request pkt of r_d to s_a */
5      pkt ← recvPktRdt (GMTP_REGISTER_REPLY);
6      if pkt = OK then
7          servs[length (servs) ] ← s_a;
8          W_j ← parsePath (pkt); /* get the W_j until s_a */
9          if c_f ≠ NULL then
               /* notify c_f that r_d is registered in s_a */
10             pkt ← makePkt (GMTP_CONN_REPLY(1), c_f);
11             sendPktRdt (pkt);
12         end
13     else
           /* Notify admin or try again or... */
14         if c_f ≠ NULL then
               /* impossible to register r_d in s_a. Notify c_f. */
15         end
16     end
17 end
18 return servs;
```

Assim, o registro de participação de um nó r_d gera um conjunto de caminhos W por onde um fluxo de dados P pode ser transmitido. Sendo assim, quanto mais nós r_d se registrarem em nós s_a, mais caminhos W_j serão conhecidos e portanto quanto mais caminhos forem conhecidos, mais parcerias poderão ser formadas entre os nós r_d. Consequentemente, quanto mais parcerias formadas, maior será o número de nós c_f capazes de receber um fluxo de dados P originado em s_a, mas disponíveis através dos seus respectivos nós r_d ou de seus parceiros.

1.3.2 Seleção de Nós

Nesta seção, discute-se sobre o processo de seleção de nós parceiros r_q, os quais auxiliam r_d a obter um fluxo de dados P (Figura 1.8).

Figura 1.8: Um usuário precisa descobrir e selecionar seus parceiros.

No GMTP, implementam-se os seguintes métodos para formação de parcerias, que são executados em paralelo, durante a transmissão de um fluxo de dados P de um evento $\mathcal{E}$.

1. Formação de parcerias intra W_j;
2. Formação de parcerias por intersecção de W_j; e
3. Formação de parcerias por combinação de W_j.

Formação de parceria intra W_j:

No protocolo GMTP, os nós $w_m \in W_j$ são automaticamente considerados parceiros entre si e por isto qualquer w_m pode repassar um fluxo de dados P para o nó r_d em questão. Dessa forma, qualquer nó pode agir como se fosse um nó s_a, sendo este o método mais simples e direto que um nó r_d pode fazer para obter um fluxo de dados P. Na Figura 1.9, Passos 1 e 2, ilustra-se essa abordagem.

No Passo 1, ilustra-se um cenário de rede com 19 nós r_d e dois nós s_a. No Passo 2, ilustra-se o pedido de registro de participação do nó r_9 ao nó s_2, o que constitui o caminho W_1. Este processo pode se repetir ao longo da rede, sendo necessário apenas armazenar o caminho W_j sem precisar manter conexão entre os nós de um dado caminho W_j – vide Conjecturas 1 e 2, apresentadas na Seção 1.2. Tal abordagem é interessante porque quando

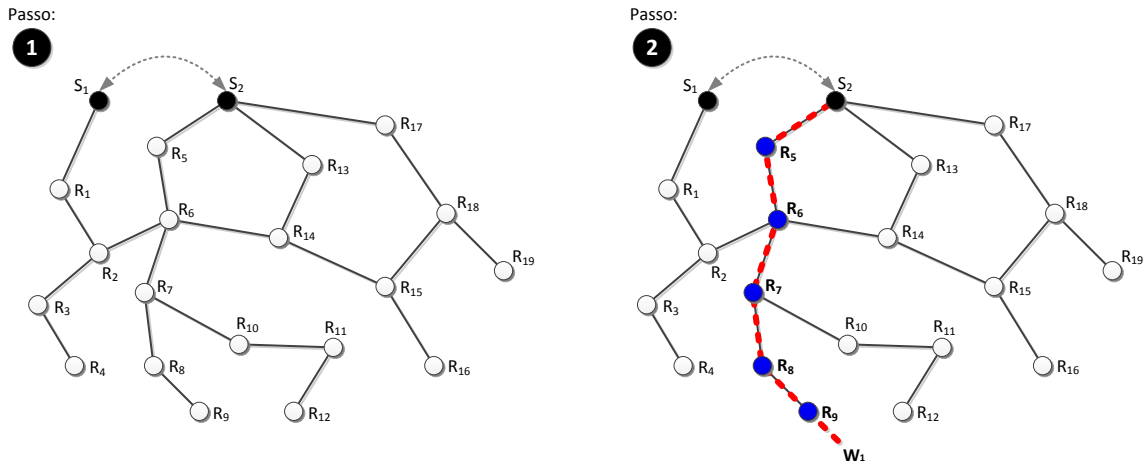

Figura 1.9: Cenário e passos para seleção de nós intra caminhos W_j.

um nó r_d solicita o fluxo de dados P de um evento $\mathcal{E}$, motivado por algum nó $c_f \in C_i(r_d)$, um nó w_m pode enviar uma resposta a r_d como se fosse o servidor s_a, se $\varphi(w_m, P) = 1$.

Dessa forma, permite-se que r_d descubra mais rapidamente outros candidatos a parceiros r_q, tal que $\varphi(r_q, P) = 1$, efetivando-se parcerias quando ocorrer $C_i(r_d) \neq \{\emptyset\}$, incluindo possíveis parcerias com os próprios nós em $W_j^{\triangleleft} = \sim(\delta(r_{q+1}, W_j))$, em caso de desconexão prematura de r_q, por exemplo. Neste exemplo, o nó r_d é adicionado ao caminho $W_j^{\triangleleft}$, tal que $r_d \prec r_{q+1}$. Isto permite reduzir o tempo de (re)início de reprodução do evento $\mathcal{E}$ por um nó c_f, contribuindo na melhoria na qualidade de experiência do usuário. Aspectos sobre desconexões e as ações realizadas pelo GMTP nesse contexto são discutidos na Seção 1.6.1.

Formação de parceria por intersecção de W_j:

O processo de formação de parcerias por intersecção ocorre pela identificação de um ou mais nós comuns a dois caminhos W_j. Esse caso é ilustrado nos Passos 1 e 2 na Figura 1.10.

Após o registro de participação de r_9 e a consequente constituição de W_1 (Figura 1.9), se o nó r_{11} também solicitar o registro de participação (Passo 1 da Figura 1.10), r_7 se torna candidato a parceiro de r_{11}, $W_1 \cap W_2 = \{s_2, r_5, r_6, r_7\}$ (Passo 4). Assim, para este exemplo, o método de formação de parcerias por interseção de W_j cobrirá o seguinte caso: após o primeiro nó $c_f \in \sum_{d=8}^{12} C_i(r_d)$ solicitar o fluxo de dados P, o nó r_7 será o nó comum entre os caminhos W_1 e W_2 e o mais próximo de qualquer nó abaixo dele, tal que $\varphi(r_7, P) = 1$. Dessa forma, para as próximas requisições para obter o fluxo de dados P, originadas por

$\forall c_f \in \sum_{d=8}^{12} C_i(r_d)$, motivará algum nó $r_{d=8..12}$ a enviar tal requisição para s_2, que passará obrigatoriamente por r_7, que a interceptará e responderá para o nó r_d correspondente, como se fosse o nó s_2. Aspectos relacionados ao envio de dados nesse contexto serão discutidos na Seção 1.4.

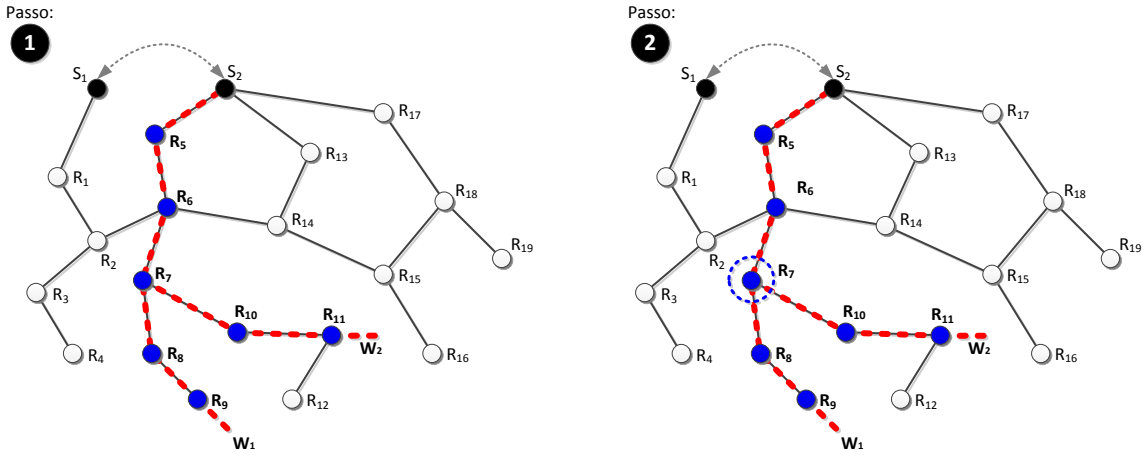

Figura 1.10: Cenário e passos para seleção de nós por interseção de caminhos W_j.

Formação de parceria por combinação de W_j:

O método de seleção de nós por meio da combinação de caminhos W_j considera um nó *pivot* s_a e o conjunto W de caminhos conhecidos, mas que não se interceptam (Passo 1 na Figura 1.11). Tais caminhos também são constituídos pelo processo de registros de participação de um nó r_d, porém o nó s_a é responsável por determinar candidados a parcerios de um determinado nó r_d. Note que este é o único caso que o nó s_a participa do processo de formação de parcerias. Isto é possível porque um nó s_a conhece um conjunto de caminhos $W_j \subset W$ gerados no processo de registro de participação de cada nó r_d em s_a (Passo 2). No exemplo da Figura 1.11, s_2 conhece os caminhos W_1 e W_3 e pode compartilhar essa informação com o nó s_1.

Em geral, o algoritmo funciona da seguinte forma: s_a processa os caminhos W_j conhecidos e determina quais são os melhores parceiros para um determinado nó r_d, sendo tal informação enviada para r_d. No caso do cenário ilustrado na Figura 1.11, Passo 2, é possível combinar os caminhos W_1 e W_2 pelos nós r_6 e o r_{15}. Nesse caso, existem duas opções, ilustradas nos Passos 3 e 4. A primeira opção é o nó r_6 repassar o fluxo de dados P para o

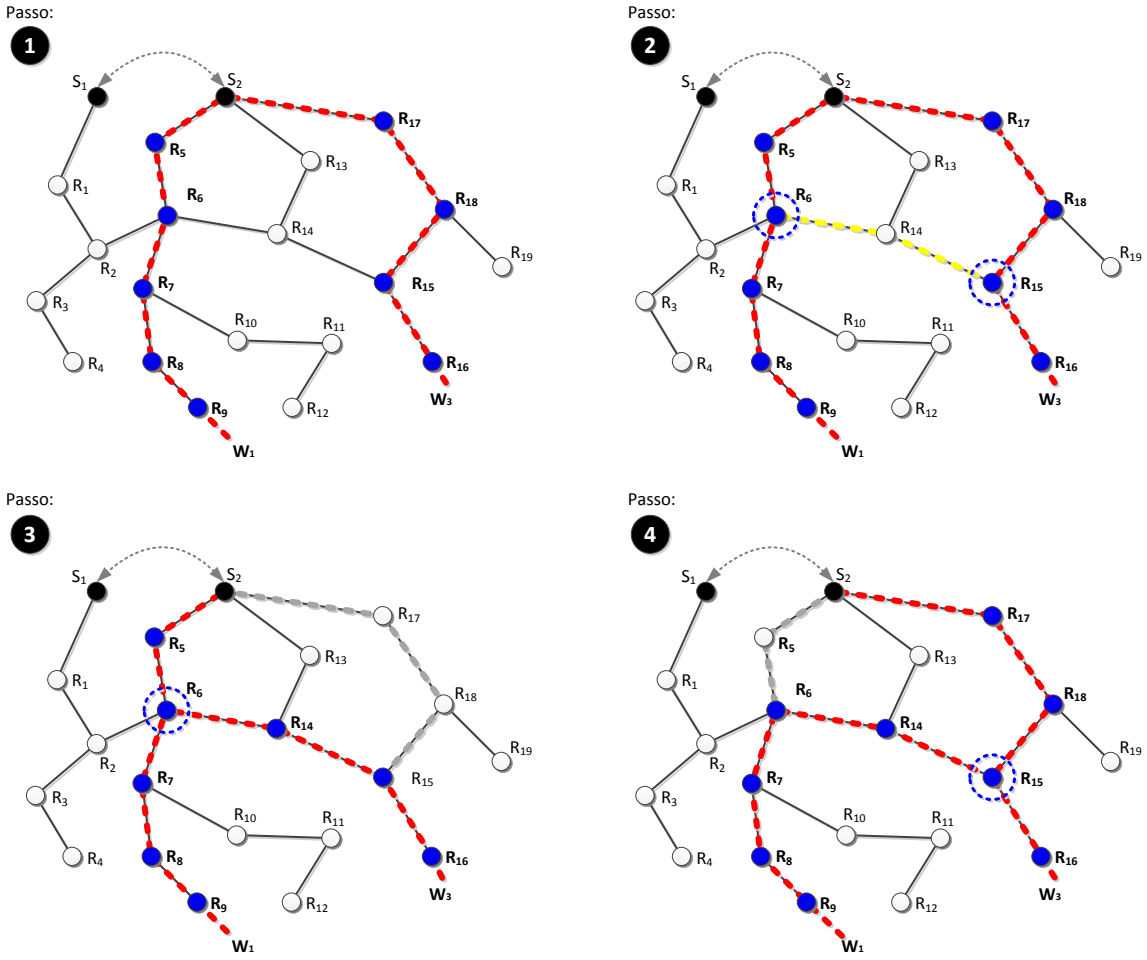

Figura 1.11: Cenário e passos para seleção de nós por combinação de caminhos W_j.

nó r_{15}, através do nó r_{14} (Passo 3). A segunda opção é o nó r_{15} repassar o fluxo de dados P para o nó r_6.

No Trecho de Código 2, apresenta-se um pseudo-algoritmo sobre como esse mecanismo funciona, executado em s_a.

Algorithm 2: matchPartnersByPathIntersection(W_j, r_d)

Data: *relayPartners* ← [ ]

```
1  mspf ← 0.4;      /* paths are considered similar if similarity
   level is equal or above mspf value */
2  pathSet ← getKnownPaths();        /* get W known in this s_a */
3  foreach W_x ∈ W do
4      if matchSimilarPath(W_x, W_j) >= mspf then
           /* Get the closest partner in the path (intersection
              between W_x and W_j) and add to the list of
              prospective partners for r_d.                    */
5          prosRelay = NULL;
6          foreach w_m ∈ W_x do
7              if w_m ∈ W_j then
8                  relayPartners[length(relayPartners)] ← prosRelay;
9              end
10         end
11     end
12 end
13 pkt ← make_pkt(GMTP_ADV_RELAY(relayPartners), r_d);
14 send_pkt_rdt(pkt);
```

Note que, os nós s_a podem trocar informações sobre os caminhos W através da rede CDN, tornando-as disponíveis através da chamada da função *getKnownPaths()*. Este compartilhamento pode otimizar, ainda mais, o processo de formação de parcerias entre os nós em R. Apesar disso ser possível, em geral, isso não é necessário, principalmente considerando transmissão de eventos em tempo real com suporte a um rede CDN. Isto porque o tráfego de fluxos de dados P termina se concentrando em uma região devido ao mecanismo de balanceamento de carga empregado na CDN e portanto o servidor alocado pela infraestrutura da CDN é sempre o ideal para o nó c_f interessado em P. Na prática, o GMTP não considera a troca de caminhos W entre servidores s_a. Em vez disso, no GMTP, caso nenhum nó $r_q \in W_j$ seja capaz de interceptar uma requisição de $r_d \in W_j$, naturalmente tal requisição

alcançará o nó $s_a \in W_j$, pois o endereço de destino da requisição sempre é o de s_a. A partir desse ponto, cria-se a possibilidade de algum nó $w_m \in W_j$ interceptar requisições de outros nós r_d.

Note que isto não significa que r_d fará parcerias com todos os nós r_d indicados por s_a. As parcerias serão efetivadas apenas de acordo com os interesses individuais de cada nó r_d por um fluxo de dados P. Este aspecto é discutido em mais detalhes quando a função de requisição de um fluxo for apresentada na Seção 1.4.

Com base na Conjectura 1, se um nó r_d realizar seu registro de participação antes de uma solicitação de um nó c_f, interessado por um evento $\mathcal{E}$, pode-se formar parcerias de melhor qualidade para transmitir que o fluxo de dados P. Por exemplo, considere $r_q \in T - S_s$, definido para uma transmissão $\mathcal{T}$. Pode-se determinar previamente um sub-caminho $\delta(r_q, W_j)$, tal que r_d possa realizar uma parceria com r_q, quando um nó $c_f \in C_i(r_d)$ estiver interessado em assistir a $\mathcal{E}$. Este procedimento pode ser realizado através da interseção de dois ou mais caminhos W_j, baseando-se em um nó pivô $s_a \in W_j$, que origina o fluxo de dados P. Além disso, periodicamente os nós r_d avaliam a disponibilidade de cada W_j conhecido e atualiza uma tabela que armazena informações relacionadas as métricas de qualidade de serviço, como o atraso fim-a-fim.

1.3.3 Sobre o melhor caminho W_j

De acordo com os métodos empregados de seleção de nós, é possível obter diferentes caminhos W_j, partindo-se de um nó r_d para um nó s_a. Por este motivo, é importante definir, a partir de um conjunto de caminhos possíveis, qual é o melhor caminho a utilizar e ordená-los de acordo com a prioridade de uso. Com isto, é possível obter P a partir de múltiplos r_d e usar caminhos alternativos em caso de falha de algum caminho, por exemplo, por desconexão. No GMTP, a ordem de prioridade para uso de cada caminho W_j é determinada de acordo com dois critérios:

1. Menor atraso fim-a-fim entre r_d e s_a;
2. Maior tempo de disponibilidade dos nós $w_m \in W_j$;
3. Menor número de nós $w_m \in W_j$;

4. Se o caminho for $W_j^{\bullet}$;

5. Escolha aleatória de W_j entre os W conhecidos.

O critério 1 é determinado através da medição do RTT. O critério 2 é obtido através de tal informação compartilhada por cada nó $w_m \in W_j$. O critério 3 é determinado pela contagem simples do número de $w_m \in W_j$. O critério 4 é determinado através da verificação da condição $|W_j| = ttl(r_d,W_j)$, onde *ttl* é uma função que determina o número de saltos entre o nó r_d até o último nó $w_m \in W_j$, que é um nó s_a. Na prática, este valor pode ser obtido através do valor do campo TTL (*Time-to-Live*), disponível no cabeçalho IP de qualquer pacote. O critério 5 é utilizado em caso de não determinação do melhor W_j até o critério anterior.

Note que no GMTP é possível que um nó r_d tenha simultaneamente mais de um nó r_d parceiro, porém não mais do que uma certa qualidade configurável devido ao fato de que os pacotes p_x dos fluxos Pserem transientes, portanto não faz sentido realizar muitas parcerias. No caso do GMTP, a quantidade máxima padrão de parcerias que um nós r_d realiza é 5, valor praticado em outros soluções similares para transmissão de fluxos de dados ao vivo baseados em arquitetura P2P.

1.4 Distribuição de P em η

No GMTP, a transmissão de dados é feita utilizando uma estratégia híbrida *pull/push* para obtenção do fluxo de dados P. O método *push* é adotado como padrão, onde os nós s_a iniciam a transmissão de $p_x \in P$ para os demais nós $w_m \in W_j$, onde $w_1 = s_a$. Já o método *pull* é utilizado quando um nó c_f precisa obter parte de uma mídia que está prestes a ser reproduzida e ainda não foi repassada por um nó r_d via *push*, de acordo com o seu mapa de *buffer*. Os nós r_d e c_f mantêm seus próprios mapas de *buffer*, sendo que um no r_d sempre terá um mapa de *buffer* mais atualizado do que os mapas de *buffer* dos seus clientes.

Nessa seção, apresentam-se detalhes sobre como o GMTP realiza a disseminação de P e como os nós c_f recebem tal conteúdo para reprodução, discutindo-se aspectos sobre indexação de conteúdos, requisição e recepção de uma mídia, compartilhamento e controle de congestionamento.

1.4.1 Indexação de Conteúdo

No GMTP, um fluxo de dados P tem um nome único que o identifica em qualquer nó, seguindo o princípio das redes centradas no conteúdo. Sendo assim, todo evento $\mathcal{E}$ é identificado por um nome que é utilizado por qualquer nó para solicitar o fluxo de dados P corresponte a $\mathcal{E}$.

No caso do GMTP, um nome para o evento é definido por um UUID (*Universally Unique IDentifier*) de 128 bits [8]. Na sua forma canônica, um evento $\mathcal{E}$ é representado por uma sequência de 32 dígitos hexadecimal, exibidos em cinco grupos separados por hífen, na forma de {8}-{4}-{4}-{4}-{12}. Por exemplo, $\mathcal{E}$ = 641f931f-d3ac-50e3-b625-537574541f1f.

O identificador de um evento $\mathcal{E}$ é criado pelo nó s_a que gera a mídia, e o divulga através de um serviço similar ao DNS ou por meio de uma busca de diretório. Além disso, um nó GMTP poderá requisitar uma lista dos fluxos transmitidos por s_a. De posse de um identificador para um evento $\mathcal{E}$, um nó GMTP poderá solicitar P de $\mathcal{E}$ aos seus nós parceiros ou diretamente a um nó s_a. A divulgação dos identificadores de todos os eventos transmitidos por um nó s_aé feito no seguinte formato textual e consultado pelo registro do tipo SID (*Streaming IDentifier*) do DNS. Por exemplo, suponha um serviço de distribuição de conteúdos multimídia da Rede Globo de Televisão, uma requisição poderá ocorrer como ilustrado no Trecho de Código 3, utilizando qualquer ferramenta de resolução de nomes por tipo de registro a um servidor DNS. Note que três eventos estão registrados, identificados pelos seus respectivos nomes.

Algorithm 3: Requisição da lista de eventos de um distribuidor de conteúdo.

```
1  dig -t SID globo.com
2  QUESTION SECTION:
3      globo.com.    IN    SID
4  ANSWER SECTION:
5      globo.com.    IN    SID    "111f931f-d3ac-10e3-b62f-f17f74541f1f"
6      globo.com.    IN    SID    "72c44591-7d82-427c-825f-722f015787c1"
7      globo.com.    IN    SID    "0bb0b9f5-f57d-4da5-8a6c-13acf1965188"
8  SUMMARY:
9      Query time: 4 msec
10     SERVER: 192.168.1.252:53(192.168.1.252)
11     WHEN: Tue Jul 16 15:44:25 2013
```

1.4.2 Estabelecimento de conexão e compartilhamento para obter P

O processo de conexão do protocolo GMTP é separado em três fases. A primeira acontece quando um nó qualquer c_1 deseja obter P transmitido por um nó s_a e não existe nenhum outro nó c_f em sua rede local recebendo P (passos 1 e 2 da Figura 1.12) através de um nó r_d, tal que $c_1 \in C_i(r_d)$. A segunda fase acontece quando um novo nó $c_2 \in C_i(r_d)$ deseja obter o mesmo fluxo P do nó c_1 (passos 3 e 4 da Figura 1.12). E, por fim, a terceira fase acontece quando o nó r_d começa a buscar novos parceiros a fim de obter P de mais outros nós parceiros.

1.4.3 Fase 1: primeira requisição a um fluxo P

Na primeira fase, o nó c_1 envia um pedido de conexão para o nó s_a, que é interceptado por r_d, cuja principal responsabilidade é repassar P para quaisquer outros nós $c_f \in C_i(r_d)$.

O estabelecimento de uma conexão GMTP ocorre quando um nó c_f cria um socket e envia um pacote do tipo *GMTP-Request* contendo o nome do fluxo P(camada de transporte), destinado ao nó s_a(camada de rede). Como na fase 1 assume-se que não existe nenhum nó c_f recebendo o fluxo P , o pacote *GMTP-Request* alcança obrigatoriamente seu nó repas-

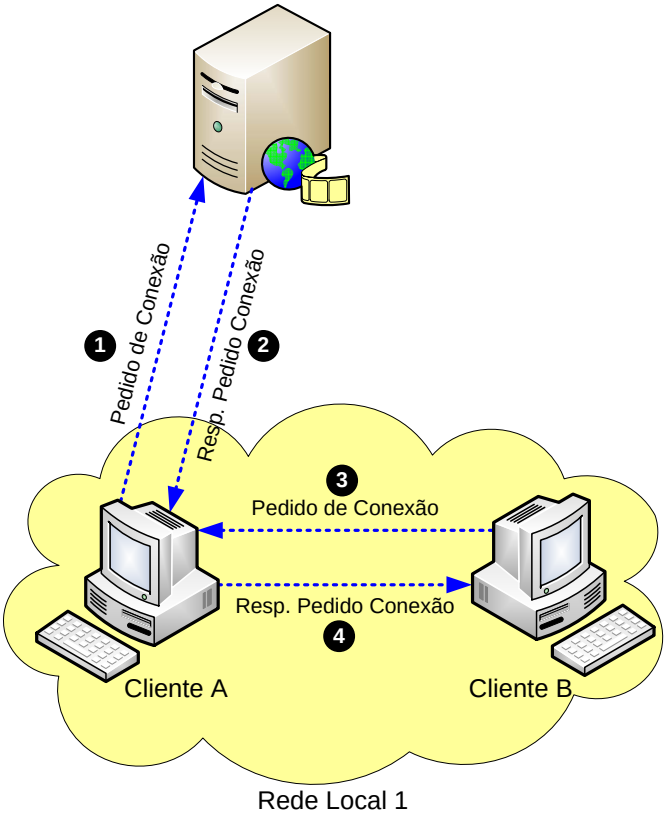

Figura 1.12: Processo Básico de Estabelecimento de Conexão do GMTP.

sador r_1, que verifica a inexistência do registro de recepção do fluxo P e por isso roteia o pacote *GMTP-Request* com destino ao nó s_a correspondente. À medida que o pacote *GMTP-Request* é repassado através dos nós $r_d \in W$, selecionado conforme discutido na Seção 1.3.2, cada nó r_d que o processa verifica se existe registro de recepção do fluxo P. Se a verificação for válida, o respectivo nó r_d envia um pacote do tipo *GMTP-Response* para r_1 que, ao recebê-lo, registra a recepção de P, usa *GMTP-RequestNotify* para notificar c_1 que o fluxo Pserá iniciado e envia um pacote do tipo *GMTP-ResponseAck* para o nó r_d. Neste ponto, conclui-se a fase 1 do processo de estabelecimento de conexão e o fluxo Pcomeça a ser transmitido entre os nós r_1 e r_d.

1.4.4 Fase 2: próximas requisições para obter P

A fase 2 de conexão ocorre quando futuras requisições a Pocontecerem após a fase 1, originadas por qualquer nó $c_f \in C_i(r_1)$. Tais solicitações são também interceptadas por r_1, que

confirma a existência de recepção de fluxo P e o repassa também para o nó c_f solicitante, notificando c_f , por exemplo, c_2, usando o pacote do tipo *GMTP-RequestNotify*, tal como ocorreu na fase 1, porém com uma diferença: a configuração de um canal de transmissão multicast na rede local.

O canal de transmissão multicast é configurado para atender ao mesmo tempo os nós c_1 e c_2, ao permitir o envio do fluxo de dados P para um grupo multicast configurado dinamicamente pelo nó r_d. A configuração automática consiste em gerar aleatoriamente um número de porta e enviar um pacote do tipo *GMTP-RequestNotify* contendo tal informação e com a flag binária chamada multicast ativada. Após a crição de tal pacote, r_1 o envia para o nó c_2 e também notifica c_1, que então realiza as configurações devidas para receber o fluxo em modo *multicast*. Além do bit de sinalização multicast, o pacote do tipo *GMTP-RequestNotify* também contém um campo de endereço IP (32 *bits*), que especificará qual endereço IP o nó $r_d$passará a transmitir os dados (canal de repasse), e mais um campo para especificar o número de porta (16 *bits*), que especifica a porta correspondente ao canal de repasse.

1.4.5 Fase 3: busca por mais parceiros r_d para obter P

Na fase 3, o nó r_d inicia um processo de aumentar suas parcerias a fim de obter mais rapidamente os pacotes $p_x \in P$ e caminhos W alternativos em caso de falha e/ou desconexões de algum nó parceiro r_d. Ao considerar os aspectos discutido na Seção 1.3.2, nota-se que na Fase 1 e 2 utiliza-se os modos de formação de parcerias intra W e por intersecção, porém ainda resta fazer uso do modo de formação de parceria por combinação de W(Figura 1.11). Na fase 3 de conexão, o GMTP explora tal recurso.

Nesse contexto, seja um nó r_3 que esteja recebendo P originado em um nó s_a. Para conseguir mais nós parceiros, o nó r_3 envia uma requisição do tipo *GMTP-RelayQuery* para s_a e obtém um subconjunto de nós r_d candidatos a parceiro de r_3, como ilustrado na Figura 1.13. Note que a lista de nós parceiros enviada pelo nó s_a é construída usando o algoritmo 2 e portanto os nós s_a funcionam como um indexador (*tracker*) de nós parceiros, executando uma pré-seleção de nós parcerios para r_3. Esta pré-seleção ajuda o nó r_3 a selecionar os melhores parcerios disponíveis, de acordo com os critérios definidos em 1.3.3.

Diante do exposto, faz-se necessário registrar três procedimentos importantes realizados pelo GMTP na Fase 3:

1. um nó r_d pode enviar periodicamente requisições do tipo *GMTP-RelayQuery* para o servidor a fim de descobrir melhores parceiros e aumentar seu leque de opções. Apesar disso, a quantidade de possíveis parcerios de um nó r_d não significa, necessariamente, que tal nó mantém a mesma quantidade de parcerias efetivas para obter um fluxo de dados P. Os parâmetros de periodicidade de requisições do tipo *GMTP-RelayQuery* e a quantidade máxima de parcerias efetivas pode ser alteradas pelo administrador de r_d e tem valores padrões de 10 minutos e 5 nós, respectivamente;

2. como ilustra-se na Figur 1.14, apenas na Fase 3, permite-se requisições do tipo *GMTP-Request* partindo de um nó r_{d_3} em direção a outro nó r_{d_2}, que irá enviar um resposta do tipo *GMTP-Response* se r_{d_1} enviar uma chave secreta aceita por r_{d_2} e encaminhada para r_{d_1} pelo nó s_a, que a obteve de r_{d_2} no processo de registro de participação discutido na Seção 1.3.1. Note que nesse caso, mesmo se nó r_{d_2} não estiver recebendo o fluxo de dados P de interesse de r_{d_1}, o nó r_{d_2} deve estabelecer uma conexão (Fase 1) para obtê-lo e então repassar P para r_{d_1};

3. como se considera uma arquitetura híbrida P2P/CDN, o nó s_a pode facilmente realizar um mecanismo de balanceamento de carga, incluindo na lista, como se fosse um nó r_d, um outro nó s_a, levando-se em consideração, inclusive, todos os critérios estabelecidos na Seção 1.3.3.

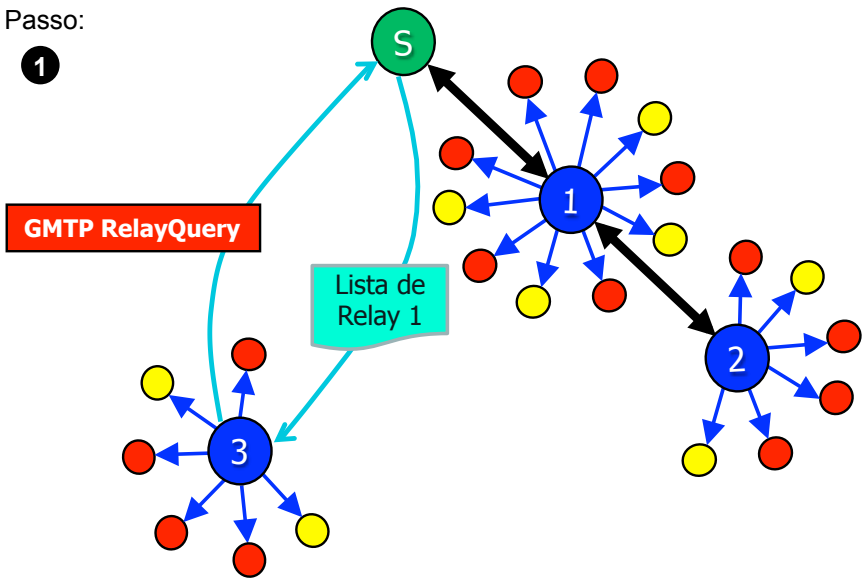

Figura 1.13: Fase 3 de conexão do GMTP (Passo 1).

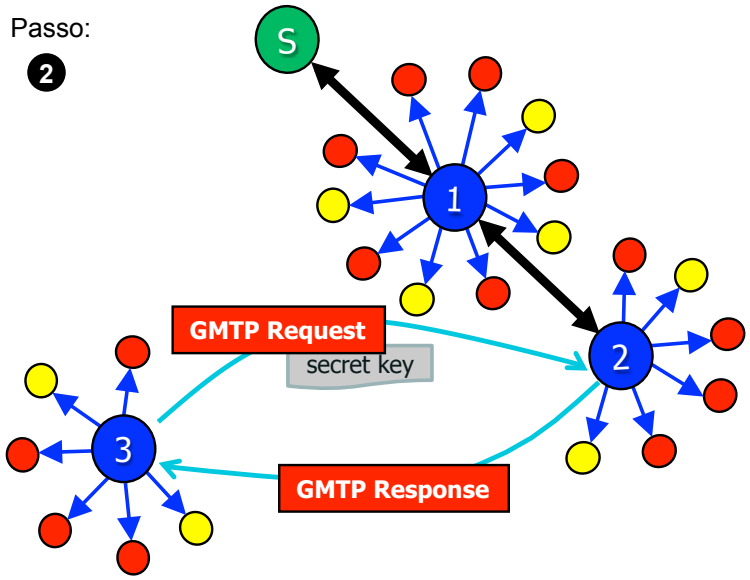

Figura 1.14: Fase 3 de conexão do GMTP (Passo 2).

1.4.6 Compartilhamento de P entre s_a

Além do processo transparente para se obter um fluxo de dados Pempregado no GMTP, como os nós s_a constituem uma rede CDN, estes podem negociar entre si o envio e a recepção de um fluxo de dados P de acordo com as requisições submetidas aos nós r_d. Desta forma, se um nó r_d enviar uma requisição para obter P de um evento $\mathcal{E}$ a um nó s_a e este não esteja recebendo tal fluxo, s_a poderá solicitá-lo a outros nós s_a da CDN que participa. A partir desse ponto, o nó s_a passará a servir o nó r_d normalmente. Como no GMTP se faz uso indireto dessa função das redes CDNs, a qual já está consolidada, resolveu-se suprimir maiores detalhes a respeito deste assunto. Para maiores informações sobre a função de distribuição de conteúdos ao vivo entre os servidores de uma rede CDN, o leitor pode consultar as referências [9–11].

Desta forma, o processo de conexão do GMTP é fundamental para a efetiva distribuição de mídias ao vivo, pois permite-se que as aplicações compartilhem fluxos de dados entre si, mesmo que tais aplicações não tenham sido desenvolvidas pela mesma equipe. Esta unificação ajuda no processo de distribuição do fluxo de dados P, pois, na prática, até mesmo uma aplicação *standalone* e um objeto de vídeo imbutido em uma página Web podem obter o mesmo fluxo de dados sem que estas conheçam um a outra. Como resultado, reduz-se para 1 o número de transmissões para um mesmo fluxo de dados P destinados a uma mesma rede

ou para um sub-conjuntos de redes adjacentes. Além dessa diferença substancial, a forma de conexão do GMTP supre uma antiga deficiência das soluções tradicionais de transmissão multicast, as quais os nós clientes, camada de aplicação, tinham que se adaptar às configurações estáticas dos canais multicast definidos pelo administrador de rede, e até os próprios administradores de rede tinham que fazer tal configuração de forma manual, que obrigatoriamente tem que ser realizada em todos os nós roteadores de um determinado caminho. Até o presente momento, não se conhece nenhuma solução que permita configuração dinâmica de canais multicast da forma que foi explicada nesta seção, com benefícios diretos para a aplicação e para a rede, fazendo-se uso dos recursos computacionais e de rede de forma mais apropriada, como será discutido no Capítulo **??**.

1.4.7 Envio e recebimento de $p_x \in P$ em η

Após o estabelecimento de conexão, os nós r_d trocam dados entre si em modo unicast a fim de obter P, constituído de pacotes p_x do tipo *GMTP-Data* e *GMTP-DataAckVec*. De forma similar, os nós r_d utilizam os mesmos tipos de pacotes para enviar $p_x \in P$ para os nós c_f, porém em modo multicast. Em ambos os casos, o GMTP realiza controle de congestionamento, sendo tais aspectos introduzidos nas Seções 1.1.3 e 1.1.4. Nesta seção, detalha-se como o GMTP executa tais funções.

Após o processo de estabelecimento de conexão, o GMTP entra no estado de transmissão de dados. Se o GMTP estiver em funcionamento em um nó s_a ou em um r_d, o estado é o de *transmitindo dados*, ao passo que quando executado em um cliente o estado é o de *recepção de dados*. Nesta seção, discute-se o funcionamento do mecanismo de transmissão e recepção de dados no GMTP.

Para o transporte de dados da aplicação, um s_a ou um r_d deve criar pacotes do tipo GMTP-Data ou o GMTP-DataAck e enviá-los aos nós c_f através do socket correspondente à conexão estabelecida. Embora o protocolo GMTP transmite dados sem garantia de entrega, em alguns casos, dados de controle podem ser transmitidos de forma confiável. Nestes casos, durante a transmissão de dados, um nó GMTP utiliza-se do pacote do tipo GMTP-Data para enviar dados, ao passo que utiliza-se pacote GMTP-Ack para confirmar a recepção de pacotes, ou ainda, utiliza-se GMTP-DataAck para enviar pacotes de dados e ao mesmo tempo confirmar a recepção de pacotes de dados vindos da direção oposta (*piggyback*).

Buffer de Envio e Recepção:

A transmissão de um evento $\mathcal{E}$ consiste no processo de disseminação dos pacotes $p_x \in P$ através dos nós interessados em obtê-lo. Para isto, cada nó GMTP controla um buffer de envio e recepção no formato de uma estrutura de dados do tipo array, onde cada posição é utilizada para armazenar um pacote p_x (Figura 1.15). Ao receber p_x, um nó GMTP armazena-o no buffer e posteriormente o entrega para a aplicação, que o reproduz para o usuário final. Para o envio ou repasse de um pacote, o nó GMTP consome os pacotes p_x do buffer e transmite para o(s) nós interessados, seja em modo unicast e/ou em modo multicast. Isto porque é possível que um nó r_{d_1} repasse p_x para um outro nó r_{d_2} (unicast) ao mesmo tempo que r_{d_1} pode repassar P para seus nós c_f (multicast).

O buffer de envio e recepção do GMTP tem seu tamanho definido no processo de estabelecimento de conexão, sendo determinado um valor mínimo e um valor máximo, sendo estes permanecendo fixos durante todo o ciclo de vida de uma conexão GMTP. Essa decisão é importante porque permite um nó r_d alocar previamente o recurso necessário para um determido fluxo de dados P. O tamanho do buffer é especificado pelo nó s_a e sempre é propagado para os demais nós no cabeçalho do pacote do tipo *GMTP-MediaDesc*, como discutido a seguir. Este aspecto é muito importante, pois a aplicação que deve tomar tal decisão, de acordo com o tipo e formato da mídia a ser transmitida. Para o GMTP, é importante apenas ter conhecimento sobre o tamanho do buffer para executar ações de descarte de p_x.

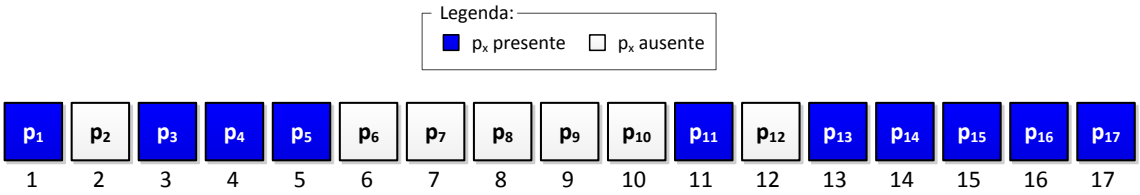

Figura 1.15: Exemplo da estrutura do buffer de envio e recepção de um nó GMTP com tamanho de 17 p_x.

Mapa de *buffer*:

O mapa de buffer do GMTP descreve o estado atual do buffer de envio e recepção de um nó GMTP. Como ilustrado na Figura 1.16, trata-se de uma estrutura de dados binária que

determina se um pacote p_x está ou não presente no buffer de um respectivo nó GMTP.

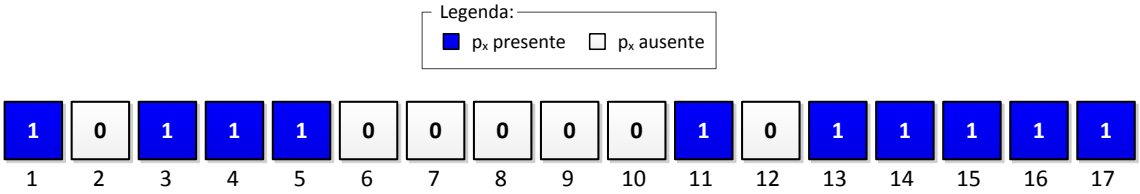

Figura 1.16: Exemplo do mapa de buffer de um nó GMTP com tamanho de 17 p_x.

O mapa de buffer é utilizado por um nó GMTP para sinalizar seu atual estado com relação a um determinado fluxo de dados P. Um nó GTMP pode enviar o mapa de buffer completo, como ilustrado na Figura 1.16, ou o mapa de buffer apenas dos p_x presentes ou ausentes. Na prática, um nó GMTP envia para um nó GMTP parceiro o mapa de buffer dos p_x presentes quando deseja indicar a sua atual disponibilidade; ao passo que envia o mapa de buffer dos p_x ausentes quando deseja obtê-los;

As trocas do mapa de buffer entre os nós GMTP ocorrem sob demanda, utilizando o método *pull*, uma vez que o método *push* é utilizado por padrão. Neste caso, quando um nó r_d percebe a falta de um ou mais pacotes p_x, este pode solicitar a um ou mais nós r_d os pacotes p_x ausentes e então obtê-los usando o método *pull*. Para isso, um nó r_d enviar aos seus nós parceiros r_d o mapa de buffer dos pacotes p_x ausentes e aguarda as respostas sobre tal disponibilidade. Essa sinalização ocorre através do uso do pacote do tipo *GMTP-DataPullRequest*, que é preenchido com o mapa de buffer dos pacotes ausentes e transmitido aos respectivos nós parceiros. Ao receber esse tipo de requisição, um nó parceiro avalia seu conteúdo e responde com o pacote do tipo *GMTP-DataPullResponse*, o qual contém o mapa de buffer dos pacotes disponíveis, seguido dos pacotes p_x do tipo *GMTP-Data*. Note que os pacotes do tipo *GMTP-DataPullRequest* e *GMTP-DataPullResponse* são transmitidos com garantia de entrega, ou seja, caso sejam perdidos, o GMTP garante sua retransmissão. Para isto, o GMTP utiliza o mecanismo básico de envio e confirmação utilizando o pacote do tipo *GMTP-DataPullAck*. No caso de falha na execução de uma requisição utilizando o método *pull*, o nó GMTP pode reavaliar a necessidade de retransmitir o pedido, pois é possível que os p_x ausentes já tenham sido expirados e requisitá-los novamente não fará mais sentido.

Na prática, o mapa de buffer utilizado para sinalizar a presença ou ausência de p_x é

representado por faixas de acordo com o índice do buffer. Por exemplo, para representar o mapa de buffer dos pacotes ausentes ilustrados na Figura 1.16, o nó GMTP preenche o pacote do tipo *GMTP-DataPullRequest* com a sequencia *2;6-10;12*. Ao receber esta sequência, o nó r_d parceiro responde com o pacote do tipo *GMTP-DataPullResponse* contendo o mapa de buffer de quais pacotes serão enviados e começa a transmití-los nessa ordem.

Descarte de pacotes:

O descarte de pacotes p_x ocorre sempre no nó r_d e em duas situações:

1. **Por transbordo do buffer:** descartar os primeiros pacotes p_x recebidos se o buffer alcançou seu limite, mesmo que ainda não tenham sido repassados. Uma otimização não explorada neste trabalho, mas que é possível de ser realizada, é o descarta seletivo de pacotes, primeiro os que tenham menos impacto na qualidade da mídia, por exemplo, pacotes de dados contendo quadros B (codificação mpeg). Isto não impede que o vídeo seja reproduzido, porém com perda de qualidade, ao passo que permite-se a transmissão de conteúdo com os recursos disponíveis;

2. **Por duplicação:** ocorre quando o pacote p_x já foi recebido anteriormente. Tal verificação é feita de acordo com o número de sequência presente em cada pacote p_x.

Descrição do fluxo P:

O GMTP é um protocolo de transporte e por isto não precisa conhecer o tipo da mídia a ser transmitida. Porém, levando-se em consideração que uma das principais motivação do GMTP é promover a compatibilidade entre diferentes aplicações que o utiliza, faz-se necessário que as aplicações conheçam o tipo da mídia e assim permitir que qualquer aplicação consiga reproduzir o fluxo de dados P.

Nesse contexto, incorporou-se no GMTP um mecanismo para descrever Pe permitir que a camada de aplicação receba tal descrição. Para isto, utiliza-se o padrão SDP (*Session Description Protocol*), definido na RFC 2327 [12], uma vez que já é um padrão utilizado pela maioria das aplicações multimídia e portanto facilita a adaptação destas para o uso do GMTP. Apesar de ter um propósito geral para descrever sessões multimídia, no GMTP, o

SDP tem o propósito de descrever o conteúdo de Ppara permitir que os nós c_f interessados em Pinterpretem seu conteúdo e sejam capazes e reproduzi-lo.

Desta forma, o GMTP utiliza o pacote do tip *GMTP-MediaDesc* para encapsular o conteúdo de descrição SDP. O conteúdo SDP é gerado pelo nó s_a e disseminado para os nós r_d, que os repassam para os nós c_f no processo de estabelecimento de conexão, descrito na Seção 1.4.2. Com isto, o nó c_f obtém as seguintes informações sobre P:

- Identificador do fluxo de dados P;
- A codificação e endereçamento da mídia:
  - O formato da mídia (H.261 video, MPEG video, etc.);
  - O endereço e porta para obter a mídia (multicast); e
- Informação para validação de cada pacote $p_x \in P$. Este assunto será retomado na Seção 1.6.3.

Um exemplo de uma mensagem SDP transmitida pelo GMTP é apresentado no Trecho de Código 4, onde:

- *v*, a versão do SDP;
- *o*, a lista de nós s_a que a distribui;
- *s*, o nome da mídia, como discutido na Seção 1.4.1;
- *i*, o título da mídia;
- *u*, a URI que descreve detalhes sobre a mídia;
- *c*, as informações de conexão, como o tipo da rede, a versão do protocolo de rede e o endereço do nó r_d;
- *k*, a chave de criptografia se os pacotes $p_x \in$ estiverem criptografados;
- *f*, o certificado digital emitido pelo nó s_a para validação do conteúdo de p_x , se desejado;

- *m*, o tipo da mídia, a porta de conexão e protocolo de transporte; e
- *a*, atributos adicionais sobre a mídia como, por exemplo, qualidade, idioma, taxa de bits mínima e máxima necessária para transmitir a mídia, em bytes.

Algorithm 4: Exemplo de uma mensagem SDP no pacote *GMTP-MediaDesc*.

```
1  v=0
2  o=- IN IP4 177.135.177.241, IP4 186.192.82.163, IP6 2001:0db8:85a3::7344
3  s=72c44591-7d82-427c-825f-722f015787c1;        /* ver Seção 1.4.1 */
4  i=An Introduction about Global Media Transmission Protocol (GMTP).
5  u=http://www.ic.ufal.br/projects/gmtp/introduction.ps
6  c=IN IP4 200.17.113.100
7  k=base64:aGVsbG8gd29ybGQK
8  f=x509:http://www.gmtp.org/certs/cert.crt;          /* ver Seção 1.6.3 */
9  m=audio 49170 GMTP/RTP/AVP 16000-20000
10 m=video 51372 GMTP/RTP/AVP 163840-655360
11 a=type:multicast
12 a=sendrecv
13 a=quality:10;                                    /* ver Seção ?? */
14 a=lang:en;                                     /* ver RFC1766 [13] */
15 a=framerate:23.0
```

Nesse exemplo do Trecho de Código 4, utiliza-se a primeira versão do protocolo SDP e descreve-se a transmissão de dois fluxos P (Linhas 10 e 11), sendo um deles de áudio e outro de vídeo. Os fluxos estão sendo distribuídos por três nós s_a (Linha 2), dos quais dois são acessíveis através de endereços IPv4 e um através de um endereço IPv6. Os dois fluxos de áudio e vídeo P são repassados por um nó r_d acessível por um endereço IPv4 (Linha 6) através das portas 49170 e 51372, respectivamente (Linhas 9 e 10). O conteúdo transmitido é criptografado utilizando base64 e a chave especificada deve ser utilizada para decifrar o conteúdo dos pacotes p_x (Linha 7). Nesse contexto de segurança, determina-se também que é possível realizar a verificação da autenticidade do conteúdo de cada pacote p_x através do certificado digital disponível na URL especificada na Linha 8. Alguns parâmetros adicionais

da mídia são especificados entre as Linhas 13 e 17, como a qualidade da mídia, que varia entre 1 e 10. Além disso, informações importantes relacionadas as mídias são as taxas de bits sendo, neste exemplo, o áudio variando entre 16000 *Bytes* à 20000 *Bytes* e o vídeo entre 156250 *Bytes* e 625000 *Bytes* (Linhas 10 e 11).

É importante salientar que os nós r_d utilizam as informações de taxa de bits para determinar o tamanho do buffer necessário para permitir a transmissão da mídia, como discutido anteriormente. Note que o tamanho do buffer é definido em consonância com os parâmetros determinados pelo algoritmo de controle de congestionamento executado no módulo GMTP-Inter, a ser discutido em detalhes na Seção 1.5.

1.5 Controle de Congestionamento em η

No GMTP, disponibiliza-se um arcabouço para adição de novos algoritmos de controle de congestionamento de forma modularizada. Desta forma, permite-se a adição e remoção de novos algoritmos de controle de congestionamento. Atualmente, o GMTP oferece dois algoritmos, um voltado para transmissões em modo unicast e outro voltado para transmissões em modo multicast.

Na prática, definiu-se um algoritmo para controle de congestionamento híbrido, cujo comportamento dependerá se o nó que o executa está transmitindo em modo unicast ou em multicast. Em modo de transmissão unicast, utilizado na comunicação entre os nós r_d, defini-se a taxa de transmissão de um nó GMTP através de um algoritmo de janela deslizante baseado em uma equação cúbica, com suporte aos protocolos RCP [1] e ConEx [1]. Já em modo de transmissão multicast, executa-se um algoritmo baseado em relatórios transmitidos pelos nós l_w, eleitos em cada rede controlado por um nó r_d, tal que $l_w \in C_i(r_d)$. Como ilustrado na Figura 1.17, para a parte do algoritmo que funciona em modo unicast, dar-se o nome de *GMTP Unicast Congestion Control* (GMTP-UCC), ao passo que para a parte do algoritmo que funciona em modo multicast, dar-se o nome de *GMTP Multicast Congestion Control* (GMTP-MCC).

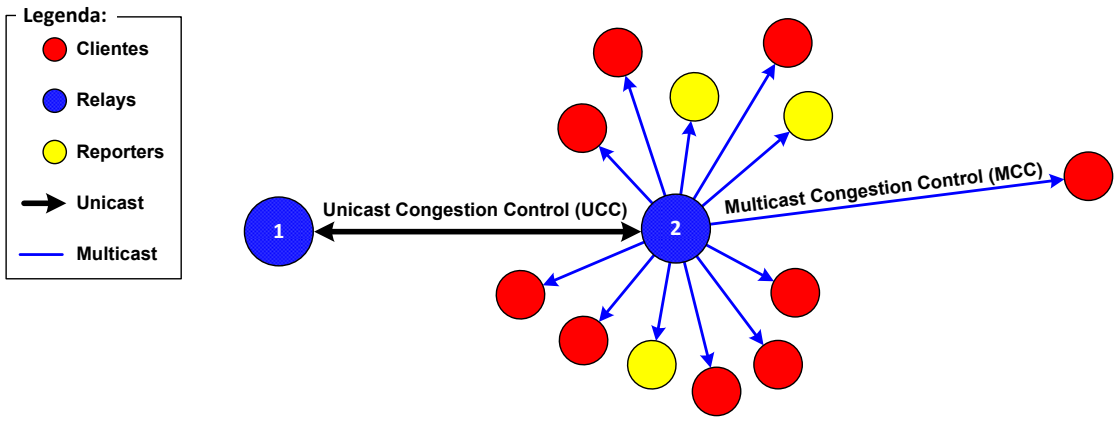

Figura 1.17: Organização do algoritmo de controle de congestionamento no GMTP.

1.5.1 Controle de Congestionamento Unicast

O GMTP-UCC funciona de forma similar ao protocolo RCP (Rate Control Protocol) [14], porém com alguns diferenciais a serem discutidos a seguir. O RCP é um protocolo para controle de congestionamento assistido pela rede que tenta emular um *Processor Sharing* (PS), como um roteador. Nesse caso, se um roteador pudesse obter a informação exata sobre o número de fluxos de entrada em um dado instante t, a taxa de transmissão ideal para cada fluxo de dados seria $R_{ps}(t) = \frac{C}{N(t)}$, onde C corresponde à capacidade do link e $N(t)$ ao número de fluxos no instante t.

Dessa forma, Nandita et. al [14] argumenta que para um roteador funcionar de forma equânime, tal roteador deve oferecer a mesma taxa de transmissão para todos os fluxos transmitidos através dele, mantendo-se a fila de roteamento perto de zero a fim de evitar que apenas os fluxos que tem pacotes na fila de repasse compartilhem a largura de banda disponível. Com base nisso, Nandita et. al [14] determinou a Equação 1.1, onde $R(t)$ é a taxa de transmissão que deve ser oferecida para cada fluxo de dados que passa pelo roteador. Com base na Equação 1.1, estima-se a largura de banda disponível em um determinado canal, representado pela porção $\alpha(C - y(t)) - \beta\frac{q(t)}{d_0}$ (mudança agregada) e a dividir por $N(t)$. Como é impossível determinar o valor exato de $N(t)$, estima-se $\hat{N}(t) = \frac{C}{R(t-T)}$ e para atualizar $R(t)$ com mais frequência do que no tempo de um RTT, escala-se a mudança agregada por $\frac{T}{d_0}$, resultando na Equação 1.2, onde:

$$R(t) = R(t - d_0) + \frac{\alpha(C - y(t)) - \beta\frac{q(t)}{d_0}}{\hat{N}(t)} \tag{1.1}$$

$$R(t) = R(t-T)\left[1 + \frac{\frac{T}{d_0}\left(\alpha(C - y(t)) - \beta\frac{q(t)}{d_0}\right)}{C}\right] \tag{1.2}$$

- d_0, é a média móvel dos valores de RTT_s, calculada através da Equação 1.3, onde θ é o ganho e corresponde a 0.02. Note que quanto maior o valor de θ, mais rápida será a convergência de d_0 ao valor de RTT_s.

$$d_0 = \theta \times RTT_s + (1 - \theta) \times d_0 \tag{1.3}$$

- $T = min(RTT_{user}, d_0))$, sendo RTT_{user} um tempo definido pelo usuário caso seja necessário atualizar $R(t)$ mais rápido do que o tempo de d_0;
- $R(t - T)$, é a última taxa de transmissão medida;
- $y(t)$, é a taxa de tráfego de entrada medida no intervalo entre a última atualização da taxa de transmissão e d_0;
- $q(t)$, é o tamanho instantâneo da fila de repasse, em bytes;
- α e β, são parâmetros pré-definidos que determinam a estabilidade e o desempenho;
- C, é a capacidade do link.

No GMTP-UCC, o algoritmo para controle de congestionamento, adaptado do RCP, funciona da seguinte forma:

1° Todo nó r_d mantém uma única taxa de transmissão $R(t)$, que é oferecida para todos os fluxos de dados passando por r_d em um certo instante t. Cada nó r_d atualizada $R(t)$ aproximadamente a cada RTT.

2° Todo pacote dos tipos *GMTP-Ack*, *GMTP-Data* ou *GMTP-DataAck* carrega duas informações de controle (campo no cabeçalho):

- *taxa de transmissão proposta* (R_p): corresponde à taxa de transmissão necessária para transmitir um fluxo de dados P, em geral, calculada pelo nó s_a;

- ***RTT na fonte*** (RTT_s): corresponde ao RTT estimado entre quaisquer nós t_u,t_{u+1} $\in W_j$, ou seja, o RTT entre dois nós t_u e t_{u+1} que processam o respectivo pacote p_x de um fluxo de dados P para repassar aos seus nós $c_f \in C_i(t_u)$ e em $C_i(t_{u+1})$, respectivamente.

3° No início de uma transmissão de um fluxo de dados P, o nó s_a transmite um pacote p_x com o valor de R_p correspondente à taxa de transmissão desejada para transmitir o referido fluxo P, com o valor para $RTT_s = \infty$. A taxa de transmissão desejada R_p deve ser calculada pela aplicação, de acordo com a taxa de bits da mídia a ser transmitida e repassada à instância do GMTP no nó s_a.

4° Todo nó $w_m = r_d$ que receber um pacote p_x, se $R(t) < R_p$, então $R_p \leftarrow R(t)$, caso contrário nenhuma modificação é realizada nesse campo. Nesse ínterim, se existir pelo menos um nó $\in C_i(w_m)$ interessado em obter os pacotes $p_x \in P$ (Seção 1.4.2), w_m executa as seguintes ações:

   (a) repassa p_x para seus nós c_f em modo multicast (Seção 1.4.7);

   (b) cria um pacote ***GMTP-Ack*** contendo R_p e o envia de volta para seu nó parceiro w_{m-1}. O pacote ***GMTP-Ack*** também carrega um campo de RTT_s. Quando w_m receber um pacote ***GMTPAck***, deve-se utilizar RTT_s para atualizar a média móvel do RTT para os fluxos que passam por ele, representado por d_0.

5° O nó w_m deve usar R_p como a nova taxa de transmissão para enviar os próximos pacotes de dados p_x para seu nó parceiro w_{m+1}. Na prática, R_p é a menor taxa de transmissão oferecida ao longo do caminho W_j.

6° Todo nó r_d atualiza periodicamente sua taxa de transmissão local $R(t)$ de acordo com a Equação 1.2.

Sendo assim, no caso do GMTP, a ideia básica é a seguinte: para quaisquer dois nós t_1,$t_2 \in W_j$, a taxa de transmissão a ser utilizada por t_1 e t_2 será definida pela menor taxa de transmissão oferecida pelos nós $w_m \in W_j$ posicionados entre t_1 e t_2. Com isto, se existir largura de banda disponível entre t_1 e t_2, ou seja, $C - y(t) > 0$, então o GMTP compartilhará igualmente o canal entre todos os fluxos, inclusive para o fluxo entre t_1 e t_2. Caso contrário,

ou seja, se $C - y(t) < 0$, o canal é considerado saturado e o GMTP reduzá a taxa de transmissão igualmente para todos os fluxos, inclusive para o fluxo entre t_1 e t_2. Especificamente, no intervalo de tempo T, a largura de banda necessária para repassar todos os pacotes p_x que estão na fila de repasse em um certo instante t corresponde à $\frac{q(t)}{d_0}$.

Escolha do algoritmo RCP em detrimento ao XCP:

Tanto o RCP quanto o XCP são os protocolos mais famosos do estado da arte que tentam emular um PS entre os fluxos que passam por ele, e por este motivo as equações de controle tanto do RCP quanto do XCP são similares. O grande dilema foi decidir qual dos dois poderia ser adotado no GMTP-UCC. A diferença entre eles é o modo que cada um tenta convergir $R_{rcp}(t)$ e $R_{xcp}(t)$ para $R_{ps}(t)$. Especificamente, a diferença está no tipo de informação enviada para um nó transmissor de um fluxo de dados P para atualizar o valor de $R_{rcp}(t)$ ou de $R_{xcp}(t)$. O XCP continuamente tenta convergir a taxa de transmissão para um ponto de equilíbrio onde todos os transmissores transmitirão pacotes de dados a uma taxa de transmissão $R_{xcp}(t)$, ao passo que o RCP calcula uma única taxa de transmissão que deve ser utilizada por todos os nós transmissores.

No caso do XCP, o protocolo aumenta ou diminui a janela de congestionamento de um fluxo P de acordo com o tamanho atual da sua janela de congestionamento. Isto ocorre porque o XCP reduz gradativamente os tamanhos da janela de congestionamento dos fluxos com $R_{xcp}(t)$ maior do que $R_{ps}(t)$ e gradativamente aumenta o tamanho das janelas de congestionamento dos fluxos com $R_{xcp}(t)$ menor do que $R_{ps}(t)$. Porém, o tamanho da janela de congestionamento é sempre menor para os fluxos iniciados mais recente. Assim, em qualquer momento, os fluxos XCP podem ter diferentes tamanhos de janela de congestionamento e de RTTs, portanto diferentes taxas de transmissão $R_{xcp}(t)$.

No RCP, todos os fluxos (novos e antigos) recebem a mesma taxa de transmissão $R_{rcp}(t)$ baseada no estado atual do nó r_d com menor largura de banda disponível em um certo instante t. Isto permite que um fluxo de dados de curta duração termine o mais rápido possível ao passo que os fluxos de dados mais longos não influenciam diretamente no compartilhamento equânime do PS, alocando para estes também um taxa de transmissão sem permitir que parte da largura de banda disponível fique ociosa por muito tempo.

O XCP é computacionalmente mais complexo do que o RCP, uma vez que define diferen-

tes valores de *feedback* para cada fluxo, envolvendo operações matemáticas (multiplicação e soma) para cada pacote, o que torna um XCP mais lento que o RCP. Pela estratégia de mudança no tamanho da janela de congestionamento, o XCP pode leva múltiplos RTTs para a maioria dos fluxos alcançarem a taxa de transmissão equânime entre eles, mas que mudam com o passar do tempo à medida que novos fluxos são injetados na rede e outros são finalizados, devido à natureza dinâmica das redes. No caso do RCP, esses problemas não ocorrem porque mantém-se uma única taxa de transmissão para todos os fluxos, não envolvendo qualquer computação adicional por pacote p_x que passa por r_d.

Desta forma, os aspectos que determinam o funcionamento do RCP são fundamentais quando se trata de transmissão de conteúdos multimídia ao vivo, aliado às outras estratégias adotadas no GMTP para a distribuição de conteúdos multimídia ao vivo. Isto porque, ao tempo que o RCP define uma taxa de transmissão equânime para todos os fluxos, sua reação é rápida às mudanças circunstanciais na rede, tanto para uma super-utilização de um canal quanto para a sua sub-utilização. Como o RCP escala naturalmente com relação à capacidade de transmissão do canal e ao RTT, o seu desempenho é invariante com relação ao tamanho de um fluxo, portanto não importa qual tipo de fluxo as aplicações geram. Isto permite que fluxos de dados GMTP+RCP e TCP+RCP coexistam na Internet de forma equânime, além do GMTP evitar sobrecarga nos nós s_a devido às outras funções de distribuição de fluxos de dados empregadas, explicadas anteriormente.

1.5.2 Controle de Congestionamento Multicast

Da mesma forma que no GMTP-UCC, o objetivo principal do GMTP-MCC é determinar uma taxa de transmissão equânime entre os fluxos de dados transmitidos pelo GMTP e por outros protocolos, como o TCP, porém em modo de transmissão multicast. No caso GMTP-MCC, trata-se de um algoritmo responsável pelo controle de congestionamento em uma rede local constituída por η_{sub} = $r_d \cup C_i(r_d)$. Na prática, os nós da rede η_{sub} formam um grupo multicast para a transmissão e recepção de um ou mais fluxos de dados P, onde o nó r_d sempre será o transmissor e os nós $c_f \in C_i(r_d)$ os receptores. A estratégia é que o valor da taxa de transmissão para um fluxo de dados P seja tão próximo ao valor da taxa de transmissão que o fluxo TCP utilizaria caso fosse transmitido na rede, portanto um algoritmo *TCP-Friendly*. Um fluxo de dados é considerado *TCP-Friendly* quando este não degrada a

taxa de transmissão de um fluxo de dados TCP mais do que outro fluxo TCP degradaria se começasse a ser transmitido na rede.

O GMTP-MCC foi inspirado em um protocolo publicado pela IETF chamado *TCP-friendly Rate Control protocol (TFRC)* (RFC 3448 [15]). O TFRC é um mecanismo para controle de congestionamento de fluxos unicast que tenta prevê a taxa de transmissão de um fluxo TCP e utilizá-la em protocolos diferentes do TCP [16]. Trata-se de uma abordagem diferente da utilizada em algoritmos baseados em janela deslizante e que utilizam pacotes de confirmação para determinar a taxa de transmissão de uma conexão, como acontece no TCP. No TFRC, o 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*). Algoritmos desse tipo são adotados em diversos protocolos, como é o caso dos CCIDs 3 e 4 do DCCP [17, 18]. Em linhas gerais, o algoritmo TFRC funciona da seguinte forma:

1° o receptor mede a taxa de perda de pacotes e envia essa informação para o transmissor;

2° o transmissor usa esse relatório para medir o RTT até o receptor;

3° o transmissor utiliza a Equação 1.4 para determinar qual será a sua próxima taxa de transmissão em função do relatório de perdas e o RTT obtidos anteriormente;

4° o transmissor então ajusta sua taxa de transmissão para o valor calculado no passo anterior.

$$R(RTT, p) = \frac{s}{RTT \times (\sqrt{\frac{2\times p}{3}} + (12 \times \sqrt{\frac{3\times p}{8}}) \times p \times (1 + 32 \times p^2))} \tag{1.4}$$

Na Equação 1.4 [19], T é a taxa de transmissão medida em bytes/segundo definida em função de s, que é o tamanho do pacote medido em bytes; RTT, é o RTT entre o nó transmissor e o receptor, medido em segundos e p, a taxa de perda de pacotes observado pelo nó receptor.

Apesar de ser uma estratégia interessante e funcionar em conexões unicast, em transmissões multicast o algoritmo descrito anteriormente não é eficiente. O algoritmo é limitado devido a um problema conhecido por *explosão de retorno* (*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.

Nesse contexto, para evitar o problema da *explosão de retorno*, determinou-se que apenas alguns nós c_f são obrigados a enviar tais relatórios ao nó r_d. Estes nós são chamados de nós GMTP Relatores e representados por l_w. No GMTP-MCC, a versão original do TFRC foi alterada e funciona da segunte forma:

1° O nó r_d executa um algoritmo de eleição de nós relatores $l_w \in C_i(r_d)$. Na Seção 1.6.2, descreve-se o procedimento para eleger os nós l_w.

2° Os nós l_w calculam a taxa de transmissão utilizando a Equação 1.4, ao invés do transmissor realizar este cálculo, como na versão original do TFRC;

3° Os nós l_w determinam a taxa de eventos de perda, e não todos os receptores do grupo multicast. Para calcular o evento de perda p, utiliza-se o mesmo procedimento feito pelo TFRC [15, 19], onde um intervalo de perda é determinado por consecutivas perdas de pacotes, desde do primeiro pacote perdido até o último pacote perdido, seguido de um pacote recebido com sucesso;

4° O RTT é calculado entre o nó l_w e o nó r_d, com o temporizador controlado pelos nós l_w e não pelo nó r_d. Isto evita que o nó r_d tenha que manter estado de temporizador para cada fluxo de dados P transmitido para os nós $c_f \in C_i(r_d)$. Para determinar o valor do parâmetro RTT e calcular a taxa de transmissão através da Equação 1.4, o GMTP-MCC utiliza a Equação 1.3, com $\theta = 0.25$, padrão do TCP;

5° A taxa de transmissão a ser utilizada pelo nó r_d é a média aritmética de todas as taxas enviadas pelos nós l_w;

6° Repete-se todos os passos a partir do passo 2 a cada intervalo igual ao RTT ou quando um intervalo de perda p é determinado.

Teoricamente, o GMTP-MCC seria um protocolo *TCP-Friendly* se $R(RTT, p)$ fosse o valor máximo entre as taxas de transmissão relatadas pelos nós l_w. Porém, optou-se por utilizar a média aritimética dos valores relatados pelos nós l_w porque, na prática, diversos fatores podem alterar o estado da rede no instante da transmissão usando o valor máximo da taxa de transmissão reportada pelos nós l_w. Com esta decisão, define-se uma margem de segurança evitando-se que o GMTP-MCC alcance o limite superior para o valor da taxa de transmissão de um fluxo transmitido com TCP. Além disso, a média aritimética suaviza os valores subsequentes para a taxa de transmissão a ser utilizada pelo nó r_d.

Um aspecto importante na medição do RTT está relacionado com o início de uma conexão GMTP, pois não se sabe o valor para inicial para RTT até o final do processo de estabelecimento de uma conexão. Nesse caso, deve-se utilizar um valor consideravelmente alto para evitar taxas de transmissões maiores do que a rede tem capacidade de suportar. No GMTP, utiliza-se o valor inicial de RTT igual a 150 *ms*. Quando um nó c_f envia um pedido de conexão utilizando o pacote do tipo *GMTP-Request*, o mesmo deve realizar a sua primeira medição do valor de RTT, iniciando-se o marcador de tempo para o cálculo do RTT quando enviar o primeiro GMTP-Request e parando-o quando receber o pacote do tipo GMTP-Response. Em seguida, deve-se acionar o mecanismo de cálculo da taxa de transmissão através da Equação 1.4, caso o respectivo nó c_f seja eleito um nó relator.

1.6 Outras funções no GMTP

Nesta seção, apresentam-se brevemente outras funcionalidades do GMTP, tais como o procedimento de desconexão, adaptação de fluxo, eleição de nós relatores, segurança, implementação e implantação.

1.6.1 Procedimentos para desconexão de nós c_f, l_w e r_d

O processo de finalização de uma conexão GMTP ocorre com algumas diferenças se comparado com outros protocolos orientados à conexão. Para sinalizar uma desconexão, um nó c_f transmite um pacote do tipo *GMTP-Close* pelo canal de controle, contendo o nome do fluxo que deseja se desconectar. Ao receber este tipo de pacote, o nó r_d transmite ao nó c_f um pacote do tipo *GMTP-Reset*, sinalizando que está ciente do fechamento da conexão. Nesse

interim, os nós desalocam recursos relacionados à respectiva conexão. Este procedimento é suficiente para o pedido de finalização de uma conexão de um cliente GMTP, porém para finalizar uma conexão de um nó l_we r_d outros procedimentos são necessários.

Desconexão de um nó l_w

Como apresentado na Seção 1.5.2, um nó l_w é responsável por relatar ao nó r_d as condições de recepção de pacotes $p_x \in P$em uma transmissão multicast e assim determinar a taxa de transmissão que deve ser utilizada para repassar o referido fluxo de dados. Sem os nós l_w, tal procedimento não seria possível. Sendo assim, deve-se realizar um procedimento para eleger um novo nó l_w quando um nó com tal responsabilidade solicite desconexão. São candidatos a nó l_w os nós c_f já recebendo o fluxo de dados P, e o nó l_w em procedimento de desconexão deve esperar que o procedimento de nova eleição seja concluído. Nesse interim, o nó l_w em processo de desconexão deve continuar enviando pacotes do tipo *GMTP-Ack* para o nó r_d.

Desconexão de um nó r_d

Um nó r_d realiza o procedimento de desconexão não por intervenção da aplicação, mas sim quando $C_i(r_d) = 0$ para um determinado fluxo de dados P. Neste caso, pode ocorrer uma situação crítica para todos os nós parceiros de r_d, pois teoricamente estes não poderão mais receber os pacotes de dados $p_x \in P$. Para evitar um período de instabilidade na recepção de Ppor parte dos nós parceiros de r_d, o GMTP define um parâmetro chamado de período de carência para novas parcerias (*grace period for new partnerships*). Este parâmetro determina o tempo que um nó r_d continuará repassando o fluxo de dados P para seus parceiros. O valor para esse tempo é transmitido para os nós parceiros de r_d, que por sua vez deve iniciar o procedimento de realizar outras parcerias a fim de continuar recebendo o fluxo de dados P. Opcionalmente, um nó r_d pode aceitar receber de seus nós parceiros tal valor para o referido parâmetro, desde que não ultrapasse um limite máximo definido pelo administrador de r_d.

1.6.2 Eleição de nós l_w

Para um fluxo de dados P, o primeiro nó l_w será o nó c_f que iniciar a primeira conexão unicast para obter o referido fluxo. Os seguintes nós l_w serão os próximos nós c_f que se

conectar para receber o fluxo de dados P, até atingir um parâmetro que determinará a quantidade máxima de nós l_w por fluxo de dados P. Tal parâmetro pode ser determinado pelo administrador do nó r_d.

Sendo assim, à medida que um nó $r_d$recebe pacotes do tipo *GMTP-Request*, no pacote de resposta *GMTP-Response*, o nó r_d ativa um indicador sinalizando que o referido nó c_f em processo de conexão deverá se comportar como um nó l_w, passando a enviar relatórios da taxa de transmissão calculada por ele. Note que este modo de transmissão deve ser implementado com garantia de entrega, ou seja, com a confirmação de recepção de pacotes e retransmissão caso este tipo de pacote seja perdido. Assim, um nó r_d poderá ter controle sobre a quantidade de nós l_w e receber relatórios apenas dos nós $l_w \in L$.

Uma outra situação que se faz necessária a eleição de nós l_w é no procedimento de desconexão, como explicado na Seção 1.6.1. Para esse caso, quando o nó r_d receber o pacote do tipo *GMTP-Close*, este deve verificar se o referido nó c_f é um nó l_w. Em caso afirmativo, o nó r_d deve transmitir para um dos nós c_f que também recebe o referido fluxo de dados P(se houver), um pacote do tipo *GMTP-Elect* e aguardar por um GMTP-ElectReply. Este procedimento deve ocorrer com garantia de entrega.

1.6.3 Segurança

Em uma solução baseada em um modelo de serviço P2P, é possível que nós mal-intencionados poluam o sistema com conteúdos alterados (Figura 1.18. Isto porque um nó r_d ao processar um pacote do tipo $p_x \in P$ pode injetar um conteúdo multimídia diferente do originalmente transmitido pelo nó s_a, comprometendo o sistema.

Figura 1.18: Um nó r_d mal-intencionados podem poluir o sistema com conteúdos alterados.

Para evitar esta situação, empregou-se no GMTP um mecanismo para validação do con-

teúdo de p_x antes que o mesmo seja transmitido para os nós $c_f \in C_i(r_d)$.

1.7 Implemetação e Implantação

PROVER API PARA SETAR AS INFOS DO SDP

O GMTP não necessita explicitamente da instalação de um nó na rede para encaminhar o conteúdo de uma rede externa para uma rede interna (*proxy*). Além disso, o GMTP mantém a *interface* de programação com a camada de aplicação inalterada, apenas adicionando uma extensão na API padrão de socket BSD para preservar a compatibilidade com as aplicações multimídia existentes e, ao mesmo tempo, permitir que as aplicações façam uso dos novos recursos do GMTP. Esta decisão pode ajudar em uma rápida adoção do GMTP nas aplicações multimídia, permitindo-se simples alterações das aplicações existentes e, ao mesmo tempo, a efetiva padronização da forma como algumas funcionalidades hoje em dia são implementadas.

1.8 Benefícios, Aplicabilidade e Justificativas

Nesta seção, apresentam-se os benefícios e justificativas para diversas tomadas de decisões realizadas no processo de especificação do GMTP.

1.8.1 Benefícios e Aplicabilidade

O uso do GMTP nas aplicações de distribuição de mídia ao vivo fomenta benefícios em três vertentes, para o desenvolvedor da aplicação; para os usuários interessados em assistir/ouvir uma mídia ao vivo e para a rede.

Considerando-se as discussões realizadas nas Seção **??** e os requisitos dos sistemas de transmissão de mídia em tempo real, o GMTP atende-os na medida em que:

- possibilita que os sistemas de transmissão de mídia ao vivo obtenham um melhor desempenho quanto a escalabilidade do número de usuários e possivelmente na qualidade da experiência do usuário (*Quality of Experience* – QoE). Nesse aspecto, o GMTP é capaz de identificar a melhor forma que o conteúdo será transportado pela

rede, permitindo-se o uso do modo de transmissão multicast sempre que possível ou de múltiplos fluxos unicast caso contrário, mas não proporcional ao número de clientes interessados;

- permite que as aplicações façam uso de soluções estáveis e largamente testadas, uma vez adicionadas ao protocolo em questão. Neste caso, grupos diferentes de desenvolvimento podem adicionar e testar suas propostas em um protocolo de rede que, uma vez consideradas estáveis podem ser compartilhadas entre os diferentes sistemas. Até mesmo soluções já implementadas nos sistemas existentes podem ser portadas para o GMTP;

- padroniza a forma como os fluxos de dados multimídia gerados pelos sistemas considerados são transportados na Internet, incluindo aspectos de controle de congestionamento e compartilhamento desses fluxos de dados entre os nós participantes de uma transmissão, além de um arcabouço que permite estender o protocolo por meio da adição de novos algoritmos;

- indiretamente diminui os fluxos de dados na rede sem qualquer controle de congestionamento, pois aplicações para distribuição de mídia ao vivo atualmente fazem uso do protocolo UDP ou variantes;

- flexibilidade no acesso a rede por parte dos nós participantes, pois eles podem entrar e sair da rede a qualquer momento, realizando parcerias com um subconjunto de nós participantes a fim de receber o conteúdo multimídia interessado, caso esteja-se utilizando o GMTP;

- disponibiliza uma solução unificada que permite o uso da API padrão de *sockets* BSD, o que facilita a migração das aplicações existentes para utilizam este novo protocolo. Além disso, permite-se que as aplicações continuem utilizando outros padrões de redes definidos pela IETF, tais como o RTP e o RSTP;

- elimina ou pelo menos inibe a presença de nós *free-riders* na rede, uma vez que todo nó GMTP é obrigado a compartilhar conteúdo sem a influência da aplicação;

- unifica as aplicações clientes no ponto de vista do processo de conexão e obtenção dos dados, uma vez que todo esse processo ocorre na camada de transporte sem qualquer influência da aplicação. Isto significa que se existir um nó executando um cliente do sistema PPLive e um outro cliente do sistema SopCast, desenvolvido por equipes diferentes, ambos ainda sim serão compatíveis e capazes de cooperar entre si na obtenção do fluxo de mídia de interesse comum.

Dentre os diversos sistemas P2P para transmissão de mídia ao vivo que podem se beneficiar com o uso do GMTP, destacam-se os baseados em uma arquitetura em malha e sem organização rígida dos nós participantes do sistema. Isto também se aplica a todas as variantes dessa arquitetura como, por exemplo, híbrido por encaminhamento automático e pedido explícito (*Push-Pull*) e híbrido árvore-malha (vide Capítulo **??**). Especificamente, os sistemas de distribuição de mídia ao vivo mais conhecidos e que podem se beneficiar diretamente com o uso do GMTP são o Sopcast [20], o PPLive [21] e o GridMedia [22, 23].

1.8.2 Justificativas

, principalmente no tocante aos frequentes questionamentos realizados durante as palestras sobre o GMTP ministradas pelo autor

- Por que um protocolo de rede, invés de um middleware? 1 - porque visualizei o GMTP como a forma primitiva de acesso, podendo ser acomplado em aplicações simples como sistemas mais avançados, que podem fazer uso de middlewares já existentes 2 - a necessidade de executar algoritmos na camada de rede obriga uma interface de comunicação disponível na camada de transporte

- Por que utilizar RCP?

- Por que utilizar roteadores?

- Qual o processo ideal para o deploy do GMTP na Internet?

- Quais são as similares e diferenças entre o GMTP e o LibSwift?

- O GMTP faz computação Per-Flow, se sim, quanto?

- Por que não usar diretamente o traceroute? e o tcptraceroute?

- Por que não manter um único RTT s? R: porque todos os acks teriam que ser repassados para o nó originador do fluxo de dados, o que geraria uma explosão de acks e além disso, qual

valor deveria ser considerado, já que um fluxo transmitido pelo servidor pode ser recebido indiretamente por muitos outros nós? Além disso, isto limitaria a taxa de transmissão R(t) entre um sub-caminho

- O GMTP Inter pode ser exeutado em um cliente?

- Em um mesmo domínio administrativo, contendo multiplos roteadores, sendo um de borda, é possível que ter apenas um nó repassador instalado em um roteador e os demais roteadores funcionarem como cliente/repassador?

- Por que um cliente não pode ter mais de um repassador? R: o tráfego sempre passará pelo roteador padrão, onde tem o gmtp inter rodando

- ... E considerando um rede com múltiplos roteadores? R: limitação do GMTP? tem que escolher um roteador como rota padrão

- Por que usou a equação do reno? R: melhor seria usar cubic-like, interações com o pesquisador, contratado pela samsung, trabalhos futuros

- Qual a vant de um r repassar?

- tem que ler header de transporte? (pode colocar uma flag no campo opt do ip)

- Por que tipos de pacotes, invez de flags

========== PAREI AQUI

1.9 Sumário do Capítulo

Neste capítulo, apresentou-se uma visão geral do *Global Media Transmission Protocol* (GMTP), um protocolo de transporte baseado em uma arquitetura P2P para distribuição de fluxos de dados multimídia de aplicações com um nó transmissor e muitos nós receptores (1 $\rightarrow n$), desenvolvido para operar principalmente na Internet. O GMTP permite a transmissão de pacotes de dados com suporte a controle de congestionamento de fluxos não confiáveis, operando em modo de transmissão multicast ou múltiplos fluxos unicast compartilhados entre os nós participantes da transmissão, através de uma rede de favores constituída dinamicamente a fim de evitar a relação de uma conexão por cliente ao servidor, como acontece em protocolos unicast de transporte de dados multimídia disponíveis na literatura.

O GMTP possui um mecanismo de conexão separado em duas fases, onde a primeira fase acontece quando o primeiro nó em uma rede local deseja estabelecer uma conexão com um

servidor que está transmitindo um determinado fluxo multimídia. Ao perceber que nenhum outro nó em sua rede local está recebendo o fluxo de dados desejado, o cliente estabelece uma conexão unicast com o servidor e se auto promove a um nó especial chamado de relay. A segunda fase do processo de conexão do GMTP acontece quando um segundo nó cliente deseja obter o mesmo fluxo de dados multimídia que o primeiro nó cliente, considerado o nó relay daquela rede. No momento em que isto acontece, o nó cliente é capaz de perceber a presença de um relay e passa a receber o fluxo de dados através do nó relay em modo multicast, evitando assim um novo pedido de conexão ao nó servidor de dados.

Um aspecto importante do GMTP é seu mecanismo de controle de congestionamento de fluxos não confiáveis. O controle de congestionamento empregado no GMTP funciona de forma híbrida, a depender do modo de conexão utilizado por um determinado nó. Quando o protocolo GMTP está operando em modo unicast, utiliza-se o GMTP-UCC, um algoritmo de controle de congestionamento baseado no TCP Cubic e escolhido para operar no GMTP porque tem alta capacidade de convergência no compartilhamento do canal de transmissão entre os diferentes fluxos de dados. Existe também o GMTP-MCC, que é o algoritmo para controle de congestionamento utilizado quando um nó GMTP opera em transmissões multicast. Tal algoritmo é baseado em uma equação TFRC (*TCP Friend Rate Control*) que faz uso de nós especiais para determinar a próxima taxa de transmissão que um nó transmissor GMTP deverá utilizar. Esses nós especiais são chamados de reporters.

Em seguida, discutiu-se sobre outras funcionalidades do protocolo GMTP, tais como seu mecanismo para finalização de conexão, eleição, monitoramento e desconexão de nós relays e reporters, assim como possíveis mecanismos para adaptação de fluxos de dados multimídia de acordo com a capacidade do canal, ainda em definição e estudo no contexto deste trabalho. Por fim, apresentou-se os benefícios trazidos pelo GMTP às aplicações que o utiliza.

No próximo capítulo, continua-se com as discussões sobre o protocolo GMTP, porém apresentando-o de forma mais técnica e com discussões sobre a sua implementação.

Bibliografia

[1] TBE. Tbe, 3 2008.

[2] S. Bradner. Key words for use in rfcs to indicate requirement levels, 3 1997. `http://www.ietf.org/rfc/rfc2119.txt`. Último acesso: 10 de Novembro de 2013.

[3] D. Meyer. Administratively scoped ip multicast, 7 1998. `http://www.ietf.org/rfc/rfc2365.txt`. Último acesso: 10 de Novembro de 2013.

[4] Jonathan L. Gross and Jay Yellen. *Handbook of Graph Theory (Discrete Mathematics and Its Applications)*. CRC Press, 2 2003. ISBN: 978-1584880905.

[5] R Séroul. *Programming for Mathematicians*. Springer-Verlag, 2 2000. ISBN: 978-3540664222.

[6] R. Courant and H. Robbins. *The Algebra of Sets. What Is Mathematics?: An Elementary Approach to Ideas and Methods*. Oxford University Press, 7 1996. ISBN: 978-0195105193.

[7] K. J. Devlin. *Fundamentals of Contemporary Set Theory*. Springer, 9 1979. ISBN: 978-0387904412.

[8] P. Leach, M. Mealling, and R. Salz. Congestion exposure (conex) concepts and use cases, 7 2005. `http://www.ietf.org/rfc/rfc4122.txt`. Último acesso: 10 de Novembro de 2013.

[9] Erik Nygren, Ramesh K. Sitaraman, and Jennifer Sun. The akamai network: a platform for high-performance internet applications. *SIGOPS Oper. Syst. Rev.*, 44(3):2–19, August 2010.

[10] A. Vakali and G. Pallis. Content delivery networks: status and trends. *Internet Computing, IEEE*, 7(6):68–74, nov.-dec. 2003.

[11] Mukaddim Pathan, Rajkumar Buyya, and Athena Vakali. Content delivery networks: State of the art, insights, and imperatives. In Rajkumar Buyya, Mukaddim Pathan, and Athena Vakali, editors, *Content Delivery Networks*, volume 9 of *Lecture Notes Electrical Engineering*, pages 3–32. Springer Berlin Heidelberg, 2008.

[12] M. Handley and V. Jacobson. Sdp: Session description protocol, 4 1998. `http://www.ietf.org/rfc/rfc2327.txt`. Último acesso: 10 de Novembro de 2013.

[13] H. Alvestrand. Tags for the identification of languages, 2 1995. `http://www.ietf.org/rfc/rfc1766.txt`. Último acesso: 10 de Novembro de 2013.

[14] Nandita Dukkipati. *Rate control protocol (rcp): congestion control to make flows complete quickly*. PhD thesis, Stanford, CA, USA, 2008. AAI3292347.

[15] M. Handley, S. Floyd, J. Padhye, and J. Widmer. Tcp friendly rate control (tfrc): Protocol specification, 1 2003. `http://www.ietf.org/rfc/rfc3448.txt`. Último acesso: 10 de Novembro de 2013.

[16] Sally Floyd, Mark Handley, Jitendra Padhye, and Jörg Widmer. Equation-Based Congestion Control for Unicast Applications. *ACM SIGCOMM Computer Communication Review*, 30(4):43–56, October 2000.

[17] Eddie Kohler, Mark Handley, and Floyd. Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2: TCP-like Congestion Control. In *IETF Online RFC*, 3 2006. `http://www.ietf.org/rfc/rfc4341.txt`. Último acesso: 10 de Novembro de 2013.

[18] Eddie Kohler and Mark Handley. Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC), 3 2006. `http://www.ietf.org/rfc/rfc4342.txt`. Último acesso: 12/04/2011.

[19] Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose. Modeling TCP throughput: a simple model and its empirical validation. In *SIGCOMM '98: Proceedings of*

the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication, pages 303–314, New York, NY, USA, 1998. ACM Press.

[20] B. Fallica, Yue Lu, F. Kuipers, R. Kooij, and P. Van Mieghem. On the quality of experience of SopCast. In *The Second International Conference on Next Generation Mobile Applications, Services and Technologies, 2008. NGMAST '08*, pages 501–506. IEEE, September 2008.

[21] X Hei, C Liang, J Liang, Y Liu, and KW Ross. Insights into PPLive: a measurement study of a Large-Scale P2P IPTV system. In *In Proc. of IPTV Workshop, International World Wide Web Conference*, 2006.

[22] Li Zhao, Jian-Guang Luo, Meng Zhang, Wen-Jie Fu, Ji Luo, Yi-Fei Zhang, and Shi-Qiang Yang. Gridmedia: A practical Peer-to-Peer based live video streaming system. In *2005 IEEE 7th Workshop on Multimedia Signal Processing*, pages 1–4. IEEE, November 2005.

[23] Meng Zhang, Jian-Guang Luo, Li Zhao, and Shi-Qiang Yang. A peer-to-peer network for live media streaming using a push-pull approach. In *Proceedings of the 13th annual ACM international conference on Multimedia*, MULTIMEDIA '05, pages 287–290, Hilton, Singapore, 2005. ACM. ACM ID: 1101206.