

SmartBand: Sistema de auditoria para transmissões Live Streaming

Maurício Bento Ghem

Resumo—O Live Streaming trouxe novas funcionalidades para a interação e comunicação entre usuários, como a videoconferência. O presente trabalho tem como objetivo relatar o andamento do projeto SmartBand que consiste na criação de uma aplicação para auditoria da transmissão Live Streaming. Será apresentado o projeto, o embasamento teórico e a metodologia para concretizá-lo. Por fim, será descrito o andamento do trabalho.

Index Terms—Gerenciamento de banda, Live Streaming, SmartBand

1 INTRODUÇÃO

O *Live Streaming* [1] é todo conteúdo audiovisual capturado em tempo real e disponibilizado para inúmeros espectadores via Internet. Esse conceito é aplicado em diversas áreas, tais como a de videoconferência, entretenimento e educação à distância (EaD), que vem crescendo ao longo dos anos por proporcionar muitos benefícios se comparada ao ensino presencial [2]. Hoje, o meio mais comum para se utilizar essa modalidade é um ambiente de ensino que contempla apostilas, fóruns e conversações. Muitos desses ambientes carecem do contato ao vivo com o instrutor/professor [3].

O projeto Convergência Digital, no qual este trabalho de conclusão de curso está inserido, trata de permitir a maior interatividade possível do usuário com um meio de acesso. Nesse contexto e na atual etapa na qual o projeto se encontra, percebeu-se a necessidade de se criar um sistema Web para interação aluno-professor utilizando-se um ambiente baseado na videoconferência. É importante destacar que a EaD é um exemplo de aplicação de transmissão de audiovisual em tempo real (*Live Streaming*) [4].

Em sua concepção, o SmartBand tinha como objetivo ser um protocolo de gerenciamento de transmissão das informações que funcionaria proativamente selecionando a melhor *stream* baseado na largura de banda que o usuário possui. Uma grande limitação na criação desse protocolo seria a dificuldade em acoplá-lo a uma plataforma de transmissão *Live Streaming* em funcionamento. Para que esse protocolo pudesse funcionar, seria necessário criar uma aplicação específica que tivesse as interfaces necessárias para permitir seu acoplamento. Como este projeto se tornaria inviável, o escopo do SmartBand foi alterado para o atual.

Após a revisão de escopo, este trabalho tem como objetivo criar a aplicação SmartBand (SmB), que fará auditoria na transmissão de informações por meio da Internet visando à melhor qualidade de serviço para o usuário na recepção de áudio e vídeo em tempo

real. O SmB possibilitará a visualização em tempo real de estatísticas relacionadas à utilização da banda, ao atraso na transmissão e outros dados relevantes que possam ser, posteriormente, analisados e utilizados para proporcionar aos usuários um serviço melhor, tanto na transmissão quanto na conectividade com a rede.

2 REVISÃO BIBLIOGRÁFICA

Nesta seção serão abordados os principais temas que compõem a revisão bibliográfica do trabalho até o presente momento. Também, serão apresentados tópicos que serão aprofundados posteriormente para a total completude do trabalho.

2.1 Live Streaming

A tecnologia de *streaming* consiste na quebra de um fluxo de dados em pequenos pedaços para enviá-los um a um sucessivamente, sendo que o receptor decodificará cada parte e a executará sem ter que esperar por todo o fluxo de dados [5]. Essa tecnologia existe desde que as transmissões por rádio e televisão analógica foram inventadas. O equipamento do espectador, desde aqueles tempos, recebia dados continuamente, sendo que simultaneamente esses dados eram apresentados, seja na tela ou nas caixas de som [6]. Hoje em dia, esse conceito de *streaming* foi migrado para a Internet, preservando alguns princípios de antigamente, como o de oferecer conteúdo sob demanda para o usuário.

Em relação ao conteúdo estático, que é necessário ter em sua totalidade para ser íntegro, o *streaming* tem uma grande vantagem, pois é possível acessar um conteúdo sob demanda. Conforme o conteúdo é executado as partes seguintes estão sendo recebidas, executadas, e assim sucessivamente. Hoje em dia, grandes *websites* utilizam esta tecnologia como um serviço. O Youtube, por exemplo, recebe uma mídia estática e a torna disponível como um fluxo de dados para ser acessada sob demanda [7].

A evolução desta tecnologia é a chamada *Live Streaming* que possibilita a transmissão ao vivo de uma mídia. Com esta tecnologia é possível enviar, diretamente, para a Internet um vídeo capturado por uma câmera sem ter que armazená-lo. Esta evolução permite que as emissoras transmitam via Internet eventos, programas e rádios em tempo real, permitindo o aumento no número de espectadores que assistem ou escutam seus programas. Porém, a transmissão em tempo real não foi a maior aplicação para esta tecnologia, e sim a videoconferência que foi amplamente difundida por meio de programas como Skype [8]. A videoconferência trouxe uma enorme quebra de paradigma, possibilitando que pessoas conversem frente a frente estando em qualquer lugar do mundo.

Tanto o *streaming* quanto o *live streaming* podem ser implementados de muitas maneiras. Uma das mais eficientes, e não proprietária é utilizando um codec para comprimir a mídia e transmiti-la para os usuários via protocolo RTP (*Real-time Transport Protocol*) [9] que roda em cima do protocolo de transporte UDP (*User Datagram Protocol*), que não é orientado a conexão. Aliado ao RTP, o protocolo RTCP (*Real-time Control Protocol*) permite controle e geração de estatísticas da transmissão. Existem implementações que utilizam o TCP (*Transmission Control Protocol*) como protocolo de transporte, multiplexando na mesma conexão o protocolo de controle como o Skype.

2.2 Protocolo RTP/RTCP

O protocolo RTP teve sua especificação inicial em 1996 pela RFC (*Request for comments*) 1889 [10], mas ao longo do tempo foi aprimorado e em 2003 foi criada a RFC 3550 [9]. Este protocolo permite o transporte fim-a-fim de aplicações que transmitem dados em tempo real, como áudio e vídeo, por meio de *multicast* ou *unicast*. Este protocolo não garante a qualidade do serviço (QoS) para transmissões em tempo real, por isso funciona em conjunto com o RTCP que permite monitorar a entrega, gerar estatísticas dos dados, e identificar os usuários que os recebem - numa rede *multicast* - proporcionando dados para o administrador analisar e realizar uma melhora no serviço.

Existem diversas aplicações que entregam conteúdo multimídia em tempo real que rodam sob o protocolo RTP, tais como: Cisco IPTV, Asterisk Video Call, Quicktime Streaming Server e Shoutcast Webradio. Em contrapartida, outras grandes empresas optaram por desenvolver um protocolo de transporte proprietário, algumas aplicações são: Skype e Microsoft Windows Media.

Por ser aberto e por ter uma RFC que padronize o protocolo RTP, é possível incorporar uma aplicação que faça interpretação dos dados gerados pelo protocolo RTCP, o de controle. Este protocolo foi a principal evolução desde a primeira RFC, devido aos algoritmos utilizados para calcular *jitter* [11] e atraso, e do cálculo dos tempos

para envio e recebimento das mensagens. O RTCP é bidirecional, ou seja, pacotes são enviados da origem de conteúdo para o destino e vice-versa. O pacote RTCP tem um formato padrão, este é resumido na figura 1, porém recebe diferentes conteúdos dependendo do papel do host na rede. Se ele é a origem dos dados, é incluído no pacote um *sender report*, como pode ser visto na figura 2. Caso o papel do host seja de destinatário, este enviará um *receiver report* com diferentes tipos de dados, como apresentado na figura 3. Os dados contidos neste pacote serão de extremo valor para os cálculos e estatísticas gerados pelo sistema SmartBand.

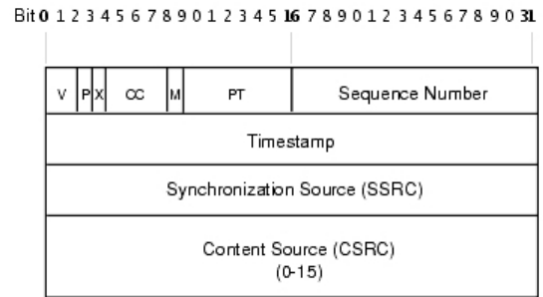


Figura 1. O pacote RTCP.

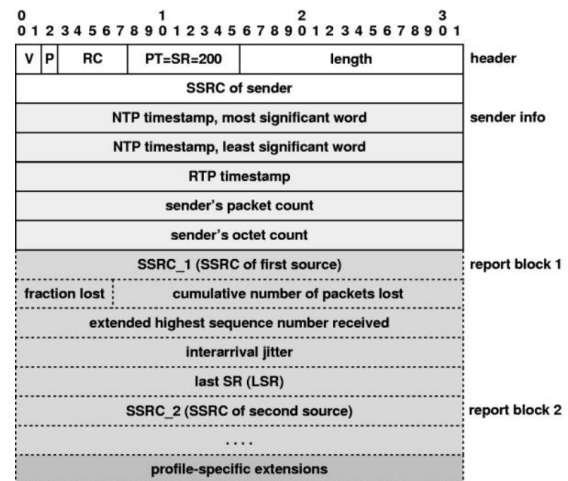


Figura 2. O *sender report* inserido dentro do pacote RTCP.

2.3 NS-2 e RTPUP

O NS-2 é uma ferramenta baseada no simulador de redes REAL, criado em 1989. Ele é um simulador de redes orientado a eventos discretos, direcionado à pesquisa científica. Com este simulador é possível simular protocolos de diversas camadas do modelo OSI, tais como: HTTP e Telnet (camada de aplicação); TCP, UDP e RTP (camada de transporte); protocolos de roteamento e QoS (camada de rede); e CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*) da camada de enlace. Além disso, suporta o desenvolvimento de extensões e

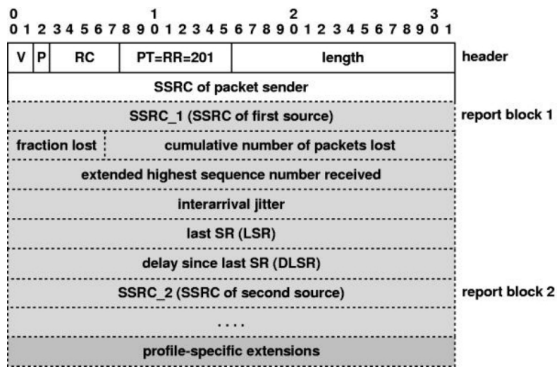


Figura 3. O *receiver report* inserido dentro do pacote RTCP.

novos protocolos que são incorporados a esta ferramenta para serem validados cientificamente [12].

Devido às limitações que o protocolo RTP nativo do NS-2 possui - como não incorporar o protocolo de controle (RTCP) - a Universidade de Patras criou uma extensão do protocolo RTP contida no NS-2, chamado de RTPUP [13], incluindo grande parte dos conceitos explicitados na RFC 3550, incluindo o RTCP. Desta maneira, é possível utilizar o NS-2 em conjunto com esta implementação do RTP para gerar dados fictícios e verificar o comportamento do SmartBand.

3 METODOLOGIA

Nesta seção será apresentada a metodologia modificada desde que o escopo do projeto foi alterado. Esta metodologia é tal que permite atingir os objetivos propostos na introdução.

Anteriormente, este trabalho era dividido também em três etapas, sendo elas, respectivamente: estudo, definição, implementação. Ao longo da primeira etapa foi descoberto um grande problema de viabilidade, diante do qual o projeto teve seu escopo e objetivos redefinidos.

Após essas modificações, o presente trabalho foi dividido em quatro grandes etapas. A primeira é responsável pela coleta de dados oriundos do protocolo de controle, o RTCP, que roda em paralelo com o RTP. Na segunda etapa, com base nos conceitos estudados e no resultado da etapa anterior, serão definidos o tipo e a estrutura que a informação terá para ser apresentada na aplicação. Na terceira etapa, ocorrerá o desenvolvimento da aplicação, que fará o monitoramento para gerar estatísticas e dados relevantes para auditoria da transmissão. A quarta e última etapa contemplará a validação do sistema como um todo.

Cada uma destas etapas será descrita a seguir.

3.1 Coleta de Dados

Para se desenvolver a aplicação final, será necessário interagir com os pacotes gerados pelo protocolo RTCP, que atua em conjunto com o RTP. Devido à quantidade de informação contida no pacote RTCP, será feita

uma coleta de dados proveniente de múltiplos tipos de aplicações para verificar os dados e elaborar uma maneira de dar sentido à informação no sistema SmartBand. Como esses protocolos são definidos por meio de RFCs e as aplicações que os implementam seguem essa proposta de padrão, o processo de obtenção de dados será facilitado.

A maneira proposta para se coletar os dados parte da utilização de um *sniffer* de rede que irá capturar pacotes de três diferentes tipos de aplicações, permitindo a heterogeneidade de dados. A aplicação utilizada para se capturar os pacotes é o Wireshark [14], e os tipos de aplicações a serem monitoradas são: videoconferência, transmissão de dados via simulador de redes e fluxo de dados aleatório via gerador de pacotes TCP/IP. A seguir, serão especificados os programas e métodos utilizados para atingir o objetivo da coleta de dados.

O método utilizado para se adquirir dados da aplicação de videoconferência será o de captura de dados por meio do *sniffer* Wireshark, que interceptará pacotes entre dois ou mais hosts que participam da conversa. O programa que suporta videoconferência é o Asterisk [15], um sistema de telefonia IP e PSTN completo que permite alta customização, que está inserido numa solução IP-PBX open-source completa chamada Trixbox [16]. Essa suíte será instalada em um servidor, e dois ou mais clientes estarão executando *soft-phones*, um telefone baseado em IP [17]. Um cliente fará uma ligação para o outro, e será iniciada a videoconferência, enquanto que o *sniffer* captura as mensagens trocadas entre os hosts.

O simulador de redes empregado no segundo tipo de aplicação a monitorar é o NS-2, acompanhado da extensão RTPUP [13], que implementa, conforme a RFC 3550, o protocolo RTP. O método para se capturar tráfego desse simulador será um pouco diferente em relação ao anterior devido à maneira como esse simulador foi concebido. Será necessário programar as transmissões entre hosts com base nos exemplos que a extensão proporciona. Devido à falta de dinamismo na captura dos dados, esse método terá a menor prioridade na coleta e uso dos dados.

O último método para coleta de dados utilizará um gerador de pacotes a ser definido. Esse método é semelhante ao anterior, porém os pacotes trafegam numa rede TCP/IP de fato, diferentemente do ambiente simulado. Com esse método será possível verificar o comportamento do protocolo RTCP em diversos tipos de transmissão.

3.2 Estruturação da informação

Nesta etapa será feito um estudo de que dados serão relevantes apresentar na aplicação SmartBand. Para serem apresentados, os dados deverão ser tratados para terem relevância para a pessoa que está acompanhando a ferramenta e fazendo auditoria. Baseado nos dados coletados na etapa anterior e no estudo da RFC 3550 serão criadas equações de tratamento ou algoritmos para cada um dos campos contido no pacote RTCP.

Atualmente, percebe-se a necessidade de monitorar os seguintes itens, todos contidos implícita ou explicitamente no pacote RTCP:

- Número de pacotes RTP trafegados;
- Perda de pacotes decorrente de instabilidade na conexão;
- Latência do link;
- *Jitter*, a variância na latência entre os pacotes recebidos;
- Uso instantâneo de banda, decorrente da transmissão *live streaming*.

Alguns itens podem ser derivados do pacote RTCP, em contrapartida outros deverão ter um método de cálculo diferenciado, utilizando artifícios de programação.

3.3 A aplicação SmartBand

Nesta etapa é descrito o produto deste trabalho, a aplicação SmartBand. Para atingir os objetivos especificados na introdução, como base, será utilizada a linguagem de programação Java com uma API que permite a interpretação de pacotes RTP/RTCP. Deste modo, será possível obter dados de uma aplicação que está rodando sem interferir em sua execução. Se fosse mantido o escopo anterior, a intervenção seria necessária tornando o projeto inviável para um ambiente de produção. Nos parágrafos a seguir será detalhado o funcionamento esperado da aplicação, a maneira que ela será concebida e método de execução para se obter os dados e gerar as estatísticas propostas.

O funcionamento do sistema SmartBand será dado em três fases: (1) identificação automática de tráfego RTP para verificação dos dados, (2) processamento e (3) apresentação visual da informação resultante na tela, isso tudo em tempo real. Para a primeira fase, a aplicação fará uma varredura constante de portas até encontrar um padrão RTP que possa ser monitorado. A segunda, responsável pelo processamento, será estruturada conforme a etapa anterior, descrita na seção 3.2. Por fim, a terceira fase apresentará os dados tratados pela fase anterior de maneira visual através de gráficos.

A aplicação SmartBand será desenvolvida na linguagem de programação Java acoplada à API Java.net.RTP desenvolvida pela Universidade de Columbia que permitirá a interpretação e manipulação dos dados contidos nos pacotes de controle (RTCP) gerados pelo protocolo RTP [18]. Para estruturar a programação será utilizado o padrão de design Model-View-Controller (MVC) que permite aliar um desenvolvimento ágil com um baixo acoplamento, isolando a lógica e a interface com o usuário que se comunicam através do controlador [19]. O ciclo de desenvolvimento terá três iterações, sendo que primeiro será feita a lógica, em seguida o design, para então serem interligados pelo controlador e o SmartBand ser testado. O controlador será desenvolvido em paralelo com os outros itens.

3.4 Validação

O SmartBand somente terá êxito total se puder ser comprovado que os dados gerados por ele estão corretos. Esta etapa prevê a validação do funcionamento e integridade da informação gerada pela aplicação. Para atingir estes objetivos serão utilizadas duas ferramentas - já discutidas anteriormente - que farão a simulação (NS-2) e teste em um ambiente real (Asterisk).

Num primeiro momento será executado o simulador de redes com uma configuração semelhante a que é incluída nos exemplos da extensão RTPUP para poder gerar tráfego, armazená-lo e passar offline pelo SmartBand para verificar seu funcionamento. Quando o sistema passar neste teste será feita a validação através de um sistema real em funcionamento.

Após o funcionamento com os dados gerados pelo simulador o SmartBand será colocado numa rede real rodando a suíte Trixbox, que contém o programa Asterisk, para poder verificar seu funcionamento num ambiente de produção. Este sistema estará configurado num servidor, e dois ou mais hosts estarão configurados com softphones para estabelecerem uma comunicação *live streaming*. A aplicação SmartBand entrará em funcionamento e um dos hosts realizará uma chamada para outro host, que o atenderá, iniciando uma videoconferência. Ao longo desta sessão, o sistema monitorará a comunicação e os pacotes RTCP para gerar as estatísticas previstas na etapa anterior. Se o SmartBand passar neste teste, estará validado.

4 RESULTADOS PARCIAIS

Como o projeto teve uma mudança brusca no escopo, as revisões bibliográficas e metodologia tiveram que ser modificadas por completo. Isto ocorreu após uma minuciosa análise de viabilidade do projeto e foi percebido que como estava estruturado agregaria pouco e, ainda assim seria inviável num ambiente em produção devido às reformulações que teriam de ocorrer.

Atualmente, a nova metodologia foi finalizada e está sólida o suficiente para se analisar e perceber que o resultado final é tangível. Além disso, em relação a produção realizada até então, foram instaladas duas máquinas virtuais, ambas com ambiente Linux, utilizando o software VMWare. Uma das máquinas virtuais será utilizada para executar o simulador de redes NS-2 e outra para a suíte Trixbox, e ambas estão funcionais. Também, com a estruturação completa da metodologia, foram detectadas as ferramentas necessárias para a criação, execução e conclusão do projeto.

5 CONCLUSÃO

Nesta produção foram abordados os amplos aspectos teóricos e metodológicos sobre a criação do sistema de auditoria para transmissões *live streaming* SmartBand. Para isso, foram descritas cada uma das etapas necessárias de estudo e desenvolvimento envolvidas

neste processo. O projeto SmartBand teve seu escopo alterado após uma análise de viabilidade que deixou claro que não seria possível atingir o resultado do projeto da maneira que era proposta. Com o escopo alterado, as revisões bibliográficas e metodologias tiveram que ser reformuladas por completo, no entanto esta reformulação trouxe mais clareza e objetividade ao projeto. A próxima etapa a ser realizada é execução da metodologia proposta. Como todas as etapas para conclusão do projeto foram planejadas e, cuidadosamente pensadas a possibilidade de falha de planejamento foi minimizada, propiciando a este projeto uma metodologia estruturada.

REFERÊNCIAS

- [1] Velos, E.; Almeida, V.; Meira, W.; Bestravos, A. e Jin, S. A Hierarchical Characterization of a Live Streaming Media Workload. IEEE IMW'02, p. 117-130. 2002.
- [2] Fischer, G. S. Um ambiente virtual para multimídia de ensino na Web, com transmissão ao vivo e interatividade. Porto Alegre: PPGC da UFRGS, 2000.
- [3] Firmo, R.M. Educação & Tecnologia, Belo Horizonte, v.8, n.2, p.26-34, dez. 2003.
- [4] Coventry, L. Video conferencing in higher education. Edinburgh: Institute for Computer Based Learning, Heriot Watt University. 1998.
- [5] Apostolopoulos, J.; Tan, W.; Wee, S. Video Streaming: Concepts, Algorithms, and Systems. HP Laboratories Palo Alto. 2002. p. 11.
- [6] Topic, M. Streaming Media Demystified, McGraw-Hill, New-York, 2002.
- [7] Youtube Inc. Company History, disponível em <http://www.youtube.com/t/about>. Acesso em: 29 de junho, 2009.
- [8] Skype Technologies. Hello. Were Skype..., disponível em <http://about.skype.com/>. Acesso em: 29 de junho, 2009.
- [9] Schulzrinne, H.; Casner, S.; Frederick, R.; Jacobson, V. RTP: A Transport Protocol for Real-Time Applications. RFC 3550, jul. 2003.
- [10] Schulzrinne, H.; Casner, S.; Frederick, R.; Jacobson, V. RTP: A Transport Protocol for Real-Time Applications. RFC 1889 - obsolete, jan. 1996.
- [11] Meggelen, J.; Smith, J.; Madsen, L. Asterisk: The future of telephony, O'Reilly Books, USA, 2005. p. 212.
- [12] University of Souther California. The Network Simulator - ns-2, disponível em <http://www.isi.edu/nsnam/ns/>. Acesso em: 24 de junho, 2009.
- [13] Bouras, C.; Gkamas, A.; Kioumourtzis, G. Extending the Functionality of RTP/RTCP Implementation in Network Simulator (NS-2) to support TCP friendly congestion control. University of Patras. 2008.
- [14] Wireshark Foundation. Wireshark - Network protocol analyzer, disponível em <http://www.wireshark.org/>. Acesso em: 25 de junho, 2009.
- [15] Digium, Inc. Asterisk, disponível em <http://www.asterisk.org/>. Acesso em: 25 de junho, 2009.
- [16] Fonality. Trixbox, disponível em <http://www.trixbox.com/>. Acesso em 25 de junho, 2009.
- [17] Dempster, B.; Garrison, K. TrixBos Made Easy, Packt Publishing, Birmingham, 2006. p. 160.
- [18] University of Columbia. Java implementation of RTP protocol, disponível em http://www.cs.columbia.edu/hgs/teaching/ais/1998/projects/java_rtp/report.html. Acesso em: 26 de junho, 2009.
- [19] Sun Microsystems. Java BluePrints - Model-View-Controller, disponível em <http://java.sun.com/blueprints/patterns/MVC-detailed.html>. Acesso em: 27 de junho, 2009.