Análise e Contra-Ataque à Poluição e *Whitewashing* em Sistemas P2P de Vídeo ao Vivo.

Rafael Barra de Almeida[1]
Orientador: Alex B. Vieira[1]**, Coorientadora: Ana Paula C. da Silva**

[1]DCC Programa de Pós-Graduação em Ciênca da Computação
Universidade Federal de Juiz de Fora – Juiz de Fora – MG

Resumo. *Este artigo resume a dissertação de mestrado de Rafael Barra, aprovada pelo PPGCC/UFJF em janeiro de 2013.*

1. Motivação e relevância do tema de dissertação

Nos últimos anos, aplicações de vídeo ao vivo na Internet tem atraído a atenção de muitos usuários e pesquisadores na área de redes. As primeiras versões destas aplicações eram baseadas na arquitetura cliente-servidor. Atualmente, utilizam a arquitetura P2P (*peer-to-peer*) que tende a ser escalável e mais resiliente a falhas. Por não se basear em uma infraestrutura dedicada, a arquitetura P2P oferece uma rápida implementação a um baixo custo, sem requerer altos recursos centralizados [Hei et al. 2008].

Ao se utilizar a arquitetura P2P, a capacidade de *upload* de cada participante pode ser utilizada para auxiliar na disseminação do conteúdo de vídeo por toda a rede. Desse modo, a largura de banda, que seria necessária em um único ponto no modelo cliente-servidor, é dividida entre vários participantes do sistema, reduzindo consideravelmente a carga no servidor origem da transmissão.

Como exemplo da importância destas aplicações, a CNN utilizou uma plataforma P2P para auxiliar na distribuição vídeo da posse do presidente Obama em 2009. Este evento, um dos maiores na história da Internet, atendeu a 1.3 milhões de usuários simultâneos. Mais da metade deles utilizavam a estrutura P2P[1].

Espera-se que os participantes do sistema P2P compartilhem dados de maneira proporcional aos que recebem. Mais ainda, espera-se que eles não realizem nenhum de ataque contra o sistema. Entretanto, em um sistema real podem existir vários usuários que assumem um comportamento malicioso e oportunista, se aproveitando das características do sistema para provocar algum tipo de dano ao mesmo.

Um dos ataques realizados em sistemas P2P de transmissão de vídeo ao vivo é a poluição de conteúdo [Vieira et al. 2008]. Nesse ataque, participantes maliciosos alteram ou danificam o conteúdo do dado compartilhado antes de encaminhá-lo a seus parceiros. Esse ataque pode criar um falso fluxo de vídeo ou danificar o conteúdo original. Caso os participantes que recebem esses dados não percebam que estes estão poluídos, eles podem repassá-los, agravando o problema da poluição. Com isso, o conteúdo legítimo fica menos disponível, comprometendo o funcionamento do sistema. Este ataque obriga os *peers* a pedir novamente o dado de vídeo que não foi recebido com sucesso, gerando sobrecarga no sistema. Como consequência os participantes poderão sofrer com perdas

[1]www.nytimes.com/external/gigaom/2009/02/07/07gigaom-cnn-inauguration-p2p-stream-a-success-despite-bac-17849.html

de *chunks* e atraso na exibição do conteúdo de vídeo. Assim, a qualidade da exibição do vídeo nos usuários fica comprometida.

Na literatura são encontrados vários esquemas de combate a ataques específicos a sistemas P2P de vídeo ao vivo [Dhungel et al. 2007, So and Reeves 2012, Vieira et al. 2008, Vieira et al. 2009]. No entanto, essas soluções podem falhar em situações onde participantes maliciosos mudam frequentemente suas identidades, ou seja, em cenários com *whitewashing*. A facilidade de se obter uma nova identidade e a dificuldade de caracterizar um *whitewasher* fazem com que este comportamento se torne um desafio [Feldman et al. 2006].

2. Objetivos e Contribuições

Dado esse contexto, o objetivo principal desse trabalho é analisar o impacto causado por ataques de poluição combinados com *whitewashing* em sistemas P2P de transmissão de vídeo ao vivo. Além disso, é proposto um mecanismo de reputação distribuído para minimizar os efeitos de tal prática. Para realizar esse trabalho, foi desenvolvido um protótipo de aplicação de transmissão de vídeo ao vivo em P2P, capaz de simular ataques de poluição e *whitewashing*, assim como combater esses tipos de ataques.

Este protótipo foi avaliado no PlanetLab (www.planet-lab.org) e os resultados experimentais mostram que identificar os dados poluídos e pedir retransmissão é ineficiente. Nesse caso, como os *peers* pedem retransmissão, a sobrecarga média na banda de rede chega a 230% em momentos de pico. Nesse cenário, os *peers* também experimentam uma alta taxa de perda no tempo de execução, o que indica que os usuários não assistem um vídeo com qualidade adequada. O primeiro mecanismo de reputação implementado (proposto originalmente por [Vieira et al. 2009]), apresenta valores de sobrecarga abaixo de 5% quando os poluidores não realizam *whitewashing*. Porém, quando há ataque de *whitewashing*, a sobrecarga no sistema alcança 112% e a taxa de perda chega a 50%. Assim, nessa dissertação, foi proposto um segundo mecanismo de reputação que é capaz de reduzir a sobrecarga para cerca de 20% e a taxa de perda de pedaços de vídeo é baixa, com valores em momento de pico por volta de 3%.

Além das contribuições teórico-práticas descritas acima, ressaltamos que esse trabalho de mestrado foi o primeiro defendido no Programa de Pós-Graduação em Ciência da Computação da UFJF. Mais ainda, destacamos as seguintes publicações:

(1) Alex B. Vieira; Rafael Barra; Jussara Almeida; Sérgio Campos. *"SimplyRep: A Simple and Effective Reputation System to Fight Pollution in P2P Live Streaming."*. Elsevier Computer Networks, Volume 57, Issue 4, Páginas 1019-1036, Março de 2013.
(2) Rafael Barra de Almeida; José Augusto Miranda Nacif; Ana Paula Couto da Silva; Alex Borges Vieira. *"Pollution and Whitewashing Attacks in a P2P Live Streaming System: Analysis and Counter-Attack"*. IEEE ICC 2013.
(3) Rafael Barra de Almeida; José Augusto Miranda Nacif; Ana Paula Couto da Silva; Alex Borges Vieira. *"Análise e Contra-Ataque à Poluição e Whitewashing em Sistemas P2P de Vídeo ao Vivo"*. SBRC 2013
(4) Rafael Barra de Almeida ; Ana Paula Couto da Silva ; Alex B. Vieira. *"Análise do Impacto de Ataques de Poluição Combinado com Whitewashing em Sistemas P2P de Live Streaming."*. XIII Workshop de Testes e Tolerância a Falhas (WTF 2012), SBRC 2012.

3. Impacto do Ataque de Poluição a Sistemas P2P de Transmissão ao Vivo

Os sistemas de transmissão ao vivo em P2P mais populares são baseados em malha com pedidos explícitos por dados (mesh-pull). Esses sistemas apresentam um total de m participantes que colaboram entre si para realizar a transmissão do conteúdo. Um *peer* especial (*servidor*) codifica o vídeo e inicia a transmissão. Os dados a serem transmitidos são particionados (*chunks*) e identificados de forma única. Cada *peer* p_i possui uma lista com n_i parceiros. Além desta lista, p_i mantém um *buffer* B_i para armazenar *chunks* de vídeo antes de serem executados/compartilhados.

No cenário de um ataque de poluição, o sistema P2P possui b *peers* maliciosos, chamados de poluidores. Cada *peer* não poluidor, denominado *peer bom* possui b_i parceiros poluidores, com $0 \leq b_i < n_i$. Ao receber um *chunk* poluído, o *peer bom* p_i deve descartá-lo e pedi-lo novamente a um outro parceiro. Os *peers* possuem uma maneira eficiente para determinar a integridade de um *chunk* (e.g. [Haridasan and Renesse 2008]). O *chunk* danificado será requisitado por p_i até que o mesmo possa ser consumido.

A Figura 1 modela o processo de obtenção de um *chunk* por um peer p_i. A partir do estado inicial S, p_i escolhe um dos seus parceiros para requisitar dados. Caso p_i escolha um *parceiro bom*, o *chunk* é obtido com sucesso (*data hit*), e assim, o processo de busca pelo *chunk* se encerra (estado H). Caso contrário, se o peer p_i escolhe um poluidor, o *chunk* obtido deverá ser descartado. O processo de escolha para requisição do *chunk* será repetido. Nesse caso, p_i não requisita o mesmo *chunk* para parceiros a quem ele já requisitou. Esta dinâmica se repete até que o dado seja obtido sem poluição.

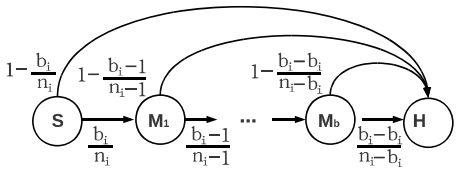

Fig. 1. Requisição de um *chunk* até o sucesso.

Dado que p_i possui b_i parceiros poluidores, entre n_i parceiros, a probabilidade de se obter, na primeira tentativa, o *chunk* não poluído é igual a $1 - (b_i/n_i)$. Caso p_i receba um dado poluído, este remove o poluidor da sua lista de parceiros candidatos e repete o processo de requisição. A probabilidade de sucesso, na segunda tentativa é igual a $1 - \left({}^{b_i - 1}/_{n_i - 1}\right)$. No pior caso, p_i irá pedir o *chunk* h a todos os parceiros poluidores, antes de conseguir escolher um *peer bom*. A probabilidade de obter o *chunk* não poluído, após p_i requisitar o mesmo a todos os poluidores, é igual a $1 - \left({}^{b_i - b_i}/_{n_i - b_i}\right)$. Essa probabilidade é igual a 1, dado que todos os parceiros poluidores são descartados da lista de candidatos.

As retransmissões realizadas enquanto um *peer* p_i não obtém um *chunk* sem poluição impõe ao sistema P2P uma sobrecarga importante. A sobrecarga l, é definida como o número de *chunks* poluídos recebidos por p_i, até que p_i consiga um *chunk* bom (estado H - Fig.1). Tal sobrecarga *média*, $E[l]$ pode ser calculada como segue:

$$
\begin{aligned}
E[l] &= \left[1 - \left(\frac{b_i}{n_i}\right)\right] * 1 \\
&+ \left[1 - \left(\frac{b_i - 1}{n_i - 1}\right)\right] * \frac{b_i}{n_i} * 2 \\
&\cdots \\
E[l] &= 1 - \left(\frac{b_i}{n_i}\right) + \sum_{s=2}^{b_i+1} 1 - \left(\frac{b_i - (s-1)}{n_i - (s-1)}\right) * s * \prod_{j=0}^{s-2} \frac{b_i - j}{n_i - j} \qquad (1)
\end{aligned}
$$

Ao requisitar um *chunk* mais raro, p_i terá menos chance de encontrá-lo entre os seus parceiros. Em um sistema sem poluidores, o *chunk* mais raro é aquele criado mais recentemente pelo servidor. O servidor anuncia seu mapa de *chunks* e somente seus parceiros tem a possibilidade de conseguir o *chunk* da posição 0 do *buffer*. Como poluidores podem anunciar um mapa de *chunks* falso, p_i é forçado a escolher um parceiro que, na realidade, não possui o *chunk* raro verdadeiro.

Em um cenário mais realista, somente uma porção dos parceiros bons de um *peer* podem responder por uma requisição. Nesse sentido, o modelo proposto terá o número de participantes bons alterado para um número menor, definido por $w = y * (n_i - b_i)$; onde y é a proporção de parceiros que podem servir o *chunk*. Assim, a Eq. 2 apresenta a nova sobrecarga média do sistema, $E[l_n]$. Esta sobrecarga aumenta com o aumento do raio médio ρ da rede, dado que o modelo considera a política de requisição *rarest first*.

$$E[l_n] = \rho + 1 - \left(\frac{b_i}{w}\right) + \sum_{s=2}^{b_i+1} 1 - \left(\frac{b_i - (s-1)}{w - (s-1)}\right) * s * \prod_{j=0}^{s-2} \frac{b_i - j}{w - j}, \tag{2}$$

O modelo de sobrecarga proposto foi analisado usando a ferramenta de verificação de modelo probabilístico PRISM (www.prismmodelchecker.org). Sessões SopCast de vídeo ao vivo foram coletadas para parametrizar o modelo proposto [Vieira et al. 2012]. Por limitações de espaço, não detalhamos os parâmetros utilizados para alimentar o modelo.

No cenário no qual os participantes têm 100 parceiros e 1 poluidor entre eles, a sobrecarga média é alta. Mesmo que todos os parceiros possam servir a uma requisição ($y = 1$), foi encontrado cerca de 167% de sobrecarga de dados devido a retransmissões impostas pela poluição. Quando o número de poluidores é aumentado para 10, a sobrecarga média aumenta para 177%. O valor da sobrecarga aumenta se o número de parceiros que um *peer* possui diminui. Quando são considerados 50 parceiros para cada *peer*, a sobrecarga aumenta para cerca de 170%. Se somente 50% desses parceiros podem atender às requisições realizadas, a sobrecarga média aumenta para 330%.

4. Mecanismo de Defesa baseado em Reputação

Este trabalho implementa em um ambiente real um sistema de reputação distribuído (originalmente proposto em [Vieira et al. 2009]). Neste trabalho, esse mecanismo é chamado *mecanismo de reputação simples*. Por essa proposta, cada *peer* p_i periodicamente calcula a reputação de cada parceiro p_j ($R_i[p_j]$) de maneira individual, de acordo com a Eq. 3. Mais precisamente, um *peer* p_i requisita r_{ij} *chunks* a p_j durante cada intervalo de tempo. O parceiro p_j pode prover n_{ji} respostas ruins para p_i. Uma resposta é definida como *ruim* quando p_i é forçado a pedir o *chunk* novamente a outro parceiro. A razão n_{ji}/r_{ij} representa a qualidade da experiência de p_i em relação a p_j. Se a razão n_{ji}/r_{ij} tem valor acima de um limite máximo T_i^{max} definido em cada *peer*, ou seja, se o grau de poluição percebido por p_i nas trocas de dados com p_j for maior do que o aceitável, p_i diminui a reputação de p_j. Caso contrário, a reputação de p_j é aumentada. Caso o valor da reputação de p_j com p_i ($R_i[p_j]$) fique abaixo de um determinado limite R_{min}, p_i interrompe as trocas de dados com p_j e o remove de sua lista de parceiros.

$$R_i[p_j] = \begin{cases} max(0, R_i[p_j] - \alpha_{p_i} * (1 + n_{ji}/r_{ij})^{y_i}) & \text{se } n_{ji}/r_{ij} > \mathrm{T}_i^{max} \\ min(1, R_i[p_j] + \alpha_{g_i} * (1 - n_{ji}/r_{ij})) & \text{caso contrário,} \end{cases} \tag{3}$$

Considerando a Eq. 3, o *mecanismo de reputação simples* não permite a reabilitação de participantes classificados como poluidores. Caso p_j passe por problemas

temporários e seja considerado poluidor, p_j não terá mais oportunidade de trocar dados com p_i, ainda que p_j seja um excelente parceiro para os demais participantes do sistema.

Nesse sentido, a fim de permitir a recuperação de participantes com a reputação simplificada, Vieira *et al.* [Vieira et al. 2009], também propõem um mecanismo que dinamicamente altera o limite mínimo de reputação R_i^{min}. Para alterar R_i^{min}, cada *peer* p_i reage às condições da rede, realizando avaliações localmente. Caso p_i infira que a rede está sob ataque, este aumenta o valor de R_i^{min}, penalizando mais rapidamente seus prováveis parceiros poluidores. Caso contrário, este diminui o valor de R_i^{min} para permitir a reabilitação das parcerias punidas anteriormente.

Combate a *Whitewashing*

Os poluidores podem, frequentemente, sair do sistema e voltar novamente com uma nova identidade. Ao realizar essa constante troca de identidade um *peer* consegue enganar o sistema de reputação (*whitewashing*). Ataques como estes podem causar grandes danos à qualidade do sistema P2P. Por nossa nova proposta, os participantes recém-chegados ao sistema recebem um baixo valor de reputação inicial, próximo ao valor da reputação mínima, ou seja $R_i[p_j] \doteq R_i^{min}$. Assim, qualquer tentativa de poluição fará com que ($R_i[p_j]$) fique abaixo da reputação mínima (R_i^{min}). Então p_j será removido da lista de parceiros de p_i. Assim, o sistema é capaz de reagir mais rapidamente a esse tipo de ataque, desencorajando os participantes poluidores a fazer *whitewashing*.

Para não prejudicar *bons peers* que deixem o sistema e voltem novamente, é proposto que todos os participantes mantenham um pequeno histórico de suas parcerias. Assim, tão logo um *peer bom*, p_j, que saiu do sistema, volte, este poderá ser pontuado com o seu valor de reputação anterior à saída.

5. Avaliação do Mecanismo de Reputação

Por limitações de espaço, vamos mostrar apenas os resultados para um cenário onde há ataques de poluição, combinado com *whitewashing*. Nós avaliamos os mecanismos de reputação em um ambiente real, configurado no PlanetLab. Foram utilizados 133 nós PlanetLab como *peers* do sistema P2P de transmissão de vídeo ao vivo. Desses *peers*, 120 são classificados como *peers bons* e 13 como poluidores. Durante os experimentos, todos os *peers* permanecem ativos até o fim da transmissão. Cada *peer* se conecta no máximo a 18 parceiros. Os poluidores atacam desde o momento em que entram no sistema até o final da transmissão. Durante o ataque, os *peers* poluidores anunciam um mapa de *chunks* completo, forjando ter todos os dados possíveis. Para maiores detalhes sobre a metodologia dos experimentos, sugerimos verificar a dissertação anexa a esse documento.

A Fig. 2 apresenta a taxa de erro em um cenário no qual poluidores também realizam *whitewashing*. A taxa de erro refere-se ao número de *chunks* que foi obtido poluído logo na primeira requisição. Os resultados mostram que, mesmo com o *mecanismo de reputação simples* (círculos vermelhos), *whitewashing* causa uma alta taxa de erro, alcançando 70%. Porém, nossa nova proposta reduz a taxa de erro à 19%. Essa alta taxa de erro se deve ao fato das constantes mudanças de identidade dos poluidores.

Ataques de *whitewashing* também causam forte impacto na sobrecarga observada. A sobrecarga, refere-se ao número de pedidos de retransmissão até se obter um *chunk* limpo. A Fig. 3 mostra que, utilizando o *mecanismo de reputação simples*, os *peers*

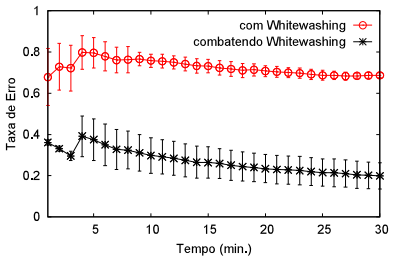

Fig. 2. Taxa de erro de *chunks*.

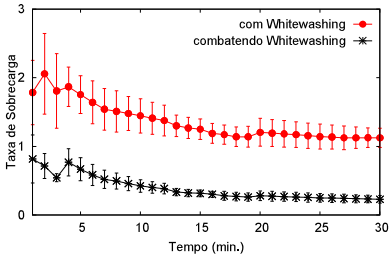

Fig. 3. Sobrecarga.

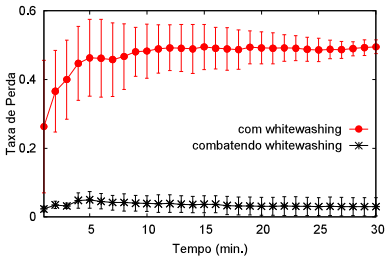

Fig. 4. Taxa de Perda.

experimentam 112% de sobrecarga média (período estável). Ao se adequar o *mecanismo de reputação* para lidar com *whitewashing*, a sobrecarga se reduz a 20%.

Finalmente, a Fig. 4 apresenta a taxa de *chunks* que chega fora do limite de tempo de execução, denominado taxa de perdas. Nesse caso, a taxa de perdas pode ser reduzida de 40%, quando há somente a reputação simples, para valores inferiores a 3%, na versão modificada para lidar com *whitewashing*. Nesse caso, mesmo sob um forte ataque, os *peers* conseguem assistir um vídeo com qualidade aceitável.

O atraso médio em um sistema sob ataque de poluição e *whitewashing* é elevado (mais de 2 min.). Ao se utilizar o *mecanismo de reputação simples*, o atraso médio cai pela metade (1 min.). Com a nova proposta, o atraso médio é reduzido para 13 segundos.

Apesar dos valores altos para sobrecarga e taxa de erro no cenário com *whitewashing*, os resultados alcançados com os mecanismos de reputação propostos não podem ser subestimados. Os esquemas de reputação propostos são fáceis e baratos de implementar. Mais ainda, não são necessários mecanismos de identificação centralizados.

Referências

Dhungel, P., Hei, X., Ross, K., and Saxena, N. (2007). The Pollution Attack in P2P Live Video Streaming: Measurement Results and Defenses. In *Proc. of ACM workshop on Peer-to-peer streaming and IP-TV*.

Feldman, M., Papadimitriou, C., Chuang, J., and Stoica, I. (2006). Free-riding and Whitewashing in Peer-to-Peer Systems. *IEEE JSAC*, 24(5):1010–1019.

Haridasan, M. and Renesse, R. V. (2008). SecureStream: An Intrusion-Tolerant Protocol for Live-Streaming Dissemination. *Computer Communications*, 31(3):563–575.

Hei, X., Liu, Y., and Ross, K. (2008). Iptv over p2p streaming networks: the mesh-pull approach. *IEEE Communications Magazine*, 46(2):86–92.

So, J. and Reeves, D. (2012). AntiLiar: Defending Against Cheating Attacks in Mesh Based Streaming. In *Proc. of IEEE Int. Conf. on Peer-to-Peer Computing*.

Vieira, A. B., Campos, S., and Almeida, J. (2008). Fighting Pollution in P2P Live Streaming Systems. In *Proc. of IEEE ICME*.

Vieira, A. B., Campos, S., and Almeida, J. (2009). Fighting Attacks in P2P Live Streaming. Simpler is Better. In *Proc. of IEEE INFOCOM Workshops*.

Vieira, A. B., Gomes, P. C., Nacif, J., Mantini, R., Almeida, J., and Campos, S. (2012). Characterizing SopCast Client Behavior. *Computer Communications*, 35:1004 – 1016.