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

Rafael B. de Almeida[1], José Augusto M. Nacif[2], Ana Paula C. da Silva[3], Alex B. Vieira[1]

[1]DCC Universidade Federal de Juiz de Fora – Juiz de Fora – MG
[2]Universidade Federal de Viçosa – MG [3]Universidade Federal de Minas Gerais – MG

rafael.barra@ice.ufjf.br, jnacif@ufv.br
ana.coutosilva@dcc.ufmg.br, alex.borges@ufjf.edu.br

Abstract. *In this paper, we analyze the content pollution attack and whitewashing in a P2P live streaming system. Moreover, we propose and implement a distributed reputation system to fight these attacks. Our analytical results show that pollution attacks are harmful, even in a system with a small number of malicious peers. In this case, the peers in the P2P live streaming system may experience up to 400% network overhead. The proposed reputation system quickly identifies and blocks polluters. Our PlanetLab experiments show that, during a content pollution and whitewashing attack, the new reputation mechanism drops the system network overhead to 20% of the streaming rate and the chunk miss rate to values lower than 3%.*

Resumo. *Este artigo analisa o impacto causado por ataques de poluição e whitewashing a sistemas P2P de vídeo ao vivo. Um mecanismo simples e descentralizado de reputação é proposto e implementado em um ambiente de rede real. Os modelos analíticos criados mostram que ataques de poluição são prejudiciais, mesmo em um sistema com poucos poluidores. Nesse caso, um sistema P2P de transmissão ao vivo pode sofrer uma sobrecarga de até 400% em relação à mídia original. O novo mecanismo de reputação bloqueia ataques de poluição e whitewashing rapidamente. No caso de ataque combinado de poluição e whitewashing, o mecanismo de reputação diminui a sobrecarga no sistema para 20% e a perda de chunks para menos de 3%.*

1. Introdução

Nos últimos anos, aplicações de vídeo ao vivo em arquiteturas P2P tem atraído a atenção tanto da academia quanto da indústria. Diversos sistemas comerciais populares, como SopCast[1] e PPLive[2], sustentam milhões de usuários registrados e, a cada dia, esse número cresce. De fato, eventos globais, como a posse do presidente Obama, foram transmitidos com sucesso pela Internet com a ajuda de arquiteturas P2P [3].

Apesar de diversos estudos recentes, os sistemas de transmissão ao vivo em P2P são suscetíveis a comportamentos maliciosos. De fato, a maioria dos sistemas populares tem suas mensagens (dados ou controle) transmitidas sem nenhuma proteção ou criptografia [Hei et al. 2007, Vieira et al. 2013]. Assim, durante as trocas de dados,

[1]www.sopcast.com
[2]www.pplive.com
[3]*"CNN: Inauguration P2P Stream a Success, Despite Backlash"*, The NY Times, fev. de 2009.

participantes (*peers*) maliciosos podem tirar vantagem das falhas existentes para que, de alguma forma, causem dano ao sistema P2P e a seus participantes.

Durante um ataque de poluição, *peers* maliciosos (poluidores) injetam ou forjam dados falsos no sistema P2P. Os demais participantes podem requisitar dados aos poluidores e assim, assistir um trecho de mídia falso. Além disso, os poluidores podem tentar burlar mecanismos de defesa praticando *whitewashing*, ou seja, trocando suas identidades constantemente [Feldman et al. 2006, Kudtarkar and Umamaheswari 2009]. Dada a facilidade de se obter uma nova identidade, *whitewashing* é um desafio, principalmente sob as fortes restrições temporais existentes nas aplicações de vídeo ao vivo. Mais ainda, mecanismos de defesa existentes não conseguem lidar com ataques de *whitewashing* [de Almeida et al. 2012].

Assim, no presente artigo propõe-se um modelo para avaliar os danos causados por ataques de poluição. Também é criado um sistema de reputação distribuído para que os *peers* identifiquem e isolem os poluidores, mesmo quando estes praticam *whitewashing*. Nessa linha, o modelo analítico desenvolvido mostra que poluição é, de fato, um problema. Mesmo em sistemas com um número baixo de poluidores, a sobrecarga na banda de rede, imposta pelas retransmissões de dados, chega a 400%.

Da mesma forma, os resultados experimentais conduzidos no PlanetLab 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%. Nesse cenário, os *peers* também apresentam uma alta taxa de perda no tempo de execução, o que indica que os usuários não assistem um vídeo com qualidade adequada.

O sistema de reputação proposto apresenta valores de sobrecarga abaixo de 5% quando os poluidores não realizam *whitewashing*. Quando há ataque de *whitewashing*, a sobrecarga aumenta para cerca de 20%. Em ambos os casos, a taxa de perda de tempo de execução é baixa, com valores por volta de 3% em momentos de pico. Finalmente, os resultados mostram que os *peers* com problemas temporários, identificados como poluidores, tem oportunidade de se redimir e voltar a contribuir com o sistema P2P.

2. Trabalhos Relacionados

Ataques de poluição de dados são, de fato, um problema em sistemas de transmissão ao vivo em P2P. Caso o sistema P2P não adote medidas, mesmo ataques simples levam a uma alta sobrecarga dos participantes do sistema [Dhungel et al. 2007, Vieira et al. 2008, Haizhou et al. 2011, de Almeida et al. 2012]. Nesse sentido, várias propostas foram criadas para combater esses ataques. Inicialmente, tais propostas baseiam-se na identificação do conteúdo poluído e no pedido de retransmissão desse dado [Dhungel et al. 2007, Haridasan and Renesse 2008].

Nessa linha, [Hu and Zhao 2010] propõem um esquema que detecta ataques de poluição em sistemas P2P de vídeo ao vivo. Eles verificam o conteúdo trocado entre os participantes e tentam detectar poluição o mais rápido possível. Para identificar os poluidores, é proposto um gerenciamento de confiança e assim, o isolamento do poluidor depende do consenso da rede P2P. Embora esse esquema auxilie na redução de sobrecarga no sistema, a convergência do consenso da rede sobre a reputação de um *peer* é demorado e assim, poluidores podem causar problemas por um longo período.

Os mecanismos de reputação propostos, geralmente, usam 2 componentes para avaliar um *peer*. Cada *peer* usa sua experiência individual e o consenso da rede [Vieira et al. 2008, Seibert et al. 2010]. Além da demora na convergência do testemunho da rede P2P, tal sistema de reputação pode sofrer com conluio e falsos testemunhos [Vieira et al. 2009, So and Reeves 2012]. O tratamento pode ser custoso, com necessidade de mecanismos de autenticação centralizado/distribuído [So and Reeves 2012].

Em alguns casos, os *peers* recém-chegados ao sistema são marcados como suspeitos [Seibert et al. 2010]: uma tentativa de reduzir os recursos disponíveis para estes *peers*. Com menos recursos disponíveis, ao se juntar ao sistema, é esperado que a troca de identidade (*whitewashing*) seja desencorajada. Uma das medidas mais comuns para tratar *whitewashing* é a utilização de uma entidade centralizada responsável por identificar, unicamente, participantes recém chegados ao sistema [Oualha and Roudier 2009]. Porém, a desvantagem dessa abordagem é a centralização de informações em um sistema de natureza distribuída, o que pode afetar a escalabilidade [Feldman et al. 2006].

O mecanismo de reputação proposto neste trabalho bane poluidores e combate a prática de *whitewashing*. O sistema proposto é simples e dispensa a necessidade de recursos centralizados. Além disso, o mecanismo proposto se diferencia dos descritos anteriormente por não prejudicar nenhum *peer* recém-chegado ao sistema. Um *peer* novato não precisa ser marcado como suspeito e ter seus recursos limitados. Finalmente, os mecanismos propostos nesse trabalho permitem a reabilitação de participantes que tenham sido classificados como atacantes, por causa de problemas temporários.

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) [Hei et al. 2008]. 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. O tamanho do *buffer* é delimitado por cada *peer*. Posições disponíveis no *buffer* são inicializadas como vazias. Os participantes do sistema mapeiam seus respectivos *buffers*, criando um mapa de chunks, para posterior sinalização de dados disponíveis e desejados.

O *buffer* B_i é implementado com o mecanismo de janela deslizante. Por simplificação, neste trabalho, considera-se que $B_i[s]$ armazena o dado menos recente, necessário para a execução do vídeo, enquanto $B_i[0]$ armazena o *chunk* que será iminentemente consumido pela aplicação. Periodicamente, os *peers* trocam entre si mapas de *chunks*. Assim, p_i possui o conhecimento dos *chunks* disponíveis nos *buffers* dos seus parceiros e novas requisições de *chunks* podem ser agendadas.

Existem duas políticas para requisição de *chunks* que se destacam: a política dependente da disponibilidade do *chunk* na rede, conhecida como *rarest first* (RF); a política dependente do tempo de exibição do *chunk*, conhecida como *earliest deadline first*, EDF. No primeiro caso (RF), busca-se disseminar rapidamente o *chunk* mais recentemente gerado pelo servidor. No segundo caso (EDF), busca-se uma visualização

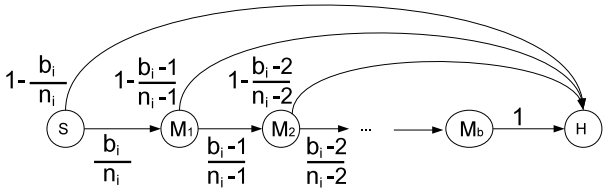


Figura 1. Requisição de um *chunk* até o sucesso.

contínua do vídeo. Neste artigo, é utilizada a política RF, dado que os *peers* tentam reproduzir o conteúdo mais recentemente gerado, preservando a característica de vídeo ao vivo. Consequentemente, a latência[4] de execução do conteúdo é baixa.

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. Sem perda de generalidade, assume-se que os poluidores estão uniformemente distribuídos entre as parcerias de todos os *peers*. Mais ainda, cada *peer* possui recursos suficientes para servir os seus parceiros. Finalmente, considera-se que os *chunks* de vídeo estão distribuídos uniformemente entre os *peers* do sistema.

Assim, a Figura 1 modela o processo de obtenção de um determinado *chunk* por um determinado peer p_i. A partir do estado inicial S, p_i escolhe um dos seus parceiros com a finalidade de requisitar dados. A escolha de um parceiro é uniforme, considerando que todos os seus parceiros possuem recursos e *chunks* equivalentes. 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* deverá ser 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.

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[5].

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

[4]Tempo médio decorrido entre a geração e a execução de um *chunk*.

[5]Como $0 \leq b_i < n_i$, p_i garantidamente consegue um *chunk* não poluído.

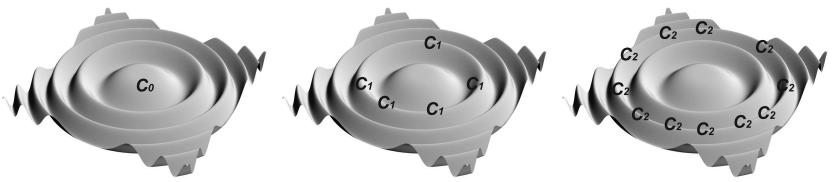
(estado H - Fig.1). Tal sobrecarga *média*, $E[l]$ pode ser calculada como segue:

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

Até o momento, o modelo proposto considera que os *chunks* de vídeo estão distribuídos homogeneamente entre os *peers* do sistema P2P. No entanto, o impacto desse ataque é fortemente influenciado pela estratégia de seleção de *chunks* que um *peer* adota. Por exemplo, ao requisitar um *chunk* mais raro, p_i terá menos chance de encontrá-lo entre os seus parceiros. 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.

Mais precisamente, 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*. Quando o servidor cria um novo *chunk*, este desloca uma posição do *buffer*. O *chunk* que estava previamente na posição 0, vai para posição 1. Neste momento, alguns parceiros do servidor podem possuir este *chunk*, e então, este não é mais considerado o mais raro.

A Figura 2 apresenta o processo de difusão de *chunk* raro, sem a presença de poluidores. No instante inicial de observação, mostrado na Figura 2(a), somente o servidor tem o *chunk*. No próximo instante, Figura 2(b), alguns dos parceiros do servidor recebem esse *chunk*. Finalmente, a Figura 2(c) mostra quando esse *chunk* alcança o terceiro salto a partir do servidor. Nesse momento, o *chunk* está difundido entre diversos participantes do sistema, e *peers* que estão 4 saltos distantes do servidor, podem requisitá-lo aos seus parceiros. Até que o *chunk* alcance pelo menos um dos seus parceiros, p_i não poderá recebê-lo. No entanto, se o sistema P2P está sob ataque de poluição, poluidores podem anunciar dados falsos e assim, p_i poderá requisitá-lo. Até que o sistema alcance um estado em que um *chunk* específico esteja difundido, os *peers* estarão sob um severo ataque de poluição. Notadamente, o período que um *peer* fica sob ataque intenso dos poluidores é proporcional à distância média entre o servidor e os *peers* do sistema.



(a) (b) (c)

Figura 2. Difusão de *chunk* raro em sistemas sem ataque de poluição. (a) Servidor produzindo um *chunk*. (b) Alguns parceiros dos servidores recebendo o *chunk*. (c) *Peers* de terceiro salto recebendo *chunk*.

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*.

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

O modelo de sobrecarga proposto foi analisado usando a ferramenta de verificação de modelo probabilístico PRISM [Kwiatkowska et al. 2009]. Sessões SopCast de vídeo ao vivo foram coletadas para parametrizar o modelo proposto. Todos os parâmetros usados foram coletados durante transmissões de eventos reais ao vivo [Vieira et al. 2012]. A Tabela 1 apresenta os valores médios utilizados na análise. Por exemplo, foram avaliadas situações nas quais cada *peer* apresentava cerca de 50 parceiros e a distância média da rede é pequena (menor que 2 saltos em média). Tal cenário se enquadra bem na maioria das redes P2P dos canais da aplicação popular SopCast.

Parâmetro	Descrição	Valor
m	Total de *peers* no sistema	1000
n_i	Número médio de parceiros	$50 \sim 100$
b_i	Número médio de parceiros poluidores	$1 \sim 10$
y	Proporção de parceiros do tipo *bom peer*	$0{,}5 \sim 1$
ρ	Distância média servidor/*peer*	1,677

Tabela 1. Parâmetros usados na avaliação do 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%.

Em resumo, os resultados mostram que mesmo em um cenário no qual os *peers* tem um grande número de parceiros e que quase todos podem fornecer *chunks* sem poluição, os ataques de poluição impactam negativamente no funcionamento do sistema. Os participantes gastam até 3 vezes mais a largura de banda necessária para a obtenção de um *chunk*. Desta forma, medidas devem ser realizadas para amenizar o dano causado por ataques de poluição em um sistema P2P de vídeo ao vivo.

4. Mecanismo de Defesa Baseado em Reputação

Neste artigo, propõe-se um modelo de reputação local em que cada *peer* monitora a troca de dados com seus parceiros. Dessa forma, os participantes do sistema são capazes de associar um valor de reputação a cada um de seus parceiros. O objetivo é permitir que cada *peer* identifique e isole parceiros que disseminem conteúdo poluído.

Em uma abordagem de reputação distribuída tradicional, cada *peer* associa uma nota a cada parceiro. Essa nota é computada utilizando a experiência individual entre o *peer* e seu respectivo parceiro, e o depoimento da rede sobre este

parceiro [Vieira et al. 2008]. Entretanto, ainda não está claro se o uso do depoimento da rede resulta em um custo-benefício satisfatório para aplicações P2P de vídeo ao vivo.

Um *peer* em um sistema de transmissão ao vivo em P2P, tipicamente, apresenta muitas interações com seus parceiros, em curtos intervalos de tempo. Assim, o depoimento da rede sobre um determinado *peer* pode convergir muito lentamente, se comparado à taxa de interações entre 2 *peers*. Mais ainda, caso um determinado p_i queira calcular a reputação de p_j e não tenha muitos parceiros que também sejam parceiros de p_j, o depoimento da rede em relação a p_j pode não ser confiável.

Para evitar esses possíveis problemas, neste artigo é proposto um mecanismo de reputação descentralizado, baseado somente na experiência individual que cada *peer* tem em relação a seus parceiros. Este mecanismo será chamado *mecanismo de reputação simples*. Por esse novo mecanismo, cada *peer* p_i periodicamente calcula a reputação de cada parceiro p_j ($R_i[p_j]$). Mais precisamente, de acordo com a Equação 3, durante cada intervalo de tempo, p_i requisita *r chunks* a p_j. O parceiro p_j pode prover *n* respostas ruins para p_i (onde $0 \leq n \leq r$). Uma resposta é definida como *ruim* quando p_i é forçado a pedir o *chunk* novamente a outro parceiro. A razão n/r representa a qualidade da experiência de p_i em relação a p_j. Se essa razão n/r tem valor acima de um limite T_i^{max}, p_i diminui a reputação local de p_j. Caso contrário, o valor da reputação local de p_j é aumentada.

$$R_i[p_j] = \begin{cases} max(0, R_i[p_j] - \alpha_{p_i} * (1 + n/r)^{y_i}) & \text{Se } n/r > T_i^{max} \\ min(1, R_i[p_j] + \alpha_{g_i} * (1 - n/r)) & \text{Caso contrário} \end{cases} \quad (3)$$

Os parâmetros α_{p_i} e α_{g_i} são, respectivamente, fatores de penalidade e gratificação. Com o objetivo de rapidamente identificar e penalizar *peers* poluidores, considera-se $\alpha_{p_i} \geq \alpha_{g_i}$. Todos os *peers* recém-chegados no sistema recebem um valor inicial de reputação. Cada *peer* p_i tem um limite de reputação mínima R_i^{min}, com $0 \leq R_i^{min} \leq 1$. Caso $R_i[p_j]$ seja menor que R_i^{min}, p_i remove p_j de sua lista de parceiros.

Nesse esquema de reputação, uma vez que p_i considera um parceiro p_j como poluidor, p_j não terá mais oportunidade de trocar dados com p_i. No modelo de reputação clássico, o depoimento da rede poderia ajudar p_j a se redimir e voltar a trocar dados com seu parceiro p_i. Nesse sentido, para permitir reabilitação, é proposto um mecanismo que dinamicamente altera o limite mínimo de reputação R_i^{min}. O princípio para alterar R_i^{min} é fazer com que cada *peer* p_i reaja às condições da rede, percebidas através de suas medições locais. Por exemplo, caso p_i infira que a rede está sob ataque, este aumenta o valor de R_i^{min}, penalizando mais rapidamente os prováveis poluidores. Caso contrário, este diminui o valor de R_i^{min} para permitir a reabilitação das parcerias punidas.

Para mudança do valor de R_i^{min}, dois estados são considerados: (1) *calmaria* e; (2) *tempestade*. Na visão de p_i, o sistema está em *calmaria* se percebe um nível de poluição abaixo de um limite pré-definido. Caso contrário, na visão de p_i, o sistema se encontra no estado de *tempestade*. A cada intervalo de tempo, p_i verifica o estado do sistema e atualiza R_i^{min} de acordo com a Equação 4. Se o sistema está em *calmaria*, p_i diminui seu limite local R_i^{min} no fator de γ_{g_i}; caso contrário, R_i^{min} é aumentado no fator γ_{p_i}. Para que o sistema identifique rapidamente participantes poluidores, $\gamma_{p_i} > \gamma_{g_i}$. Os limites RT_i^{min} e RT_i^{max} são estabelecidos de tal maneira que $0 \leq RT_i^{min} \leq R_i^{min} \leq RT_i^{max} \leq 1$.

$$R_i^{min} = \begin{cases} max(RT_i^{max}, R_i^{min} + \gamma_{p_i}) & \text{Se o sistema está no } \textit{estado de tempestade} \\ min(RT_i^{min}, R_i^{min} - \gamma_{g_i}) & \text{Se o sistema está no } \textit{estado de calmaria} \end{cases} \quad (4)$$

O estado do sistema é definido na perspectiva local de cada *peer*, baseado somente nas experiências obtidas em relação a seus parceiros: caso p_i receba uma quantidade de dados poluídos de seus parceiros, a sua visão local é de que o sistema está sofrendo ataque de poluição. Além disso, as mudanças em R_i^{min}, de acordo com a Eq. 4, são realizadas independentemente das mudanças na reputação local de cada parceiro, definida na Eq. 3

Os poluidores podem trocar suas identidades frequentemente em conjunto com o ataque de poluição. Tais trocas de identidade, conhecida como *whitewashing*, tem por objetivo enganar o sistema de reputação. Ataques combinados podem causar grandes danos ao sistema P2P, e assim, propõe-se um *mecanismo de reputação modificado*.

Nesse sentido, propõe-se um *mecanismo de reputação modificado* no qual os participantes recém-chegados ao sistema recebem um baixo valor de reputação inicial. O valor estabelecido para o valor inicial é um valor próximo ao valor limite, para que troca de *chunks* aconteça ($R_i[p_j] \tilde{=} R_i^{min}$). Para qualquer tentativa de poluição, $R_i[p_j]$ ficará abaixo de R_i^{min} e então p_j será removido de sua lista de parceiros de p_i. Vale ressaltar que apesar da diminuição no valor de reputação na visão de p_i, *peers* recém-chegados podem trocar dados com os demais participantes do sistema. Para evitar punições de *peers bons* que tiveram problemas momentâneos (falso positivos), é proposto que os *peers* do sistema 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.

5. Metodologia e Ambiente de Experimentação

Os mecanismos de reputação propostos (*simples* e *modificado*) foram avaliados em um ambiente real, configurado no PlanetLab. Uma aplicação P2P de vídeo ao vivo baseada em malha e com pedidos explícitos por dados foi implementada. Nessa aplicação, foram incorporados os mecanismos de combate a ataques de poluição e *whitewashing*.

O servidor foi instalado em uma máquina dedicada na rede do campus, transmitindo 30 minutos de vídeo a uma taxa de 120 kbps. 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 (10% do total). Não foram impostas quaisquer restrições adicionais, tanto para o servidor quanto para os *peers*. Dessa forma, processamento e largura de banda são restringidas somente pela capacidade de cada máquina e pelas conexões reais existentes entre elas.

Durante os experimentos, todos os *peers* permanecem ativos até o fim da transmissão. Cada *peer* se conecta no máximo a 18 parceiros. Esse número foi definido após observação de comportamento de clientes no SopCast. Se um de seus parceiros falha ou deixa o sistema, um *peer* deve requisitar novos parceiros candidatos ao *bootstrap* do sistema. Além disso, a duração das parcerias foi definido como o comportamento encontrado no SopCast [Vieira et al. 2012].

Cada *chunk* enviado pelo poluidor tem em seu cabeçalho uma marcação para indicar que seu conteúdo é inválido (código *hash* dos dados da mídia). Ressalta-se porém que, qualquer mecanismo existente de verificação de dados (e.g., assinatura *hash* em cadeia [Dhungel et al. 2007, Haridasan and Renesse 2008]) pode ser usado para identificar poluição. A saber, a latência adicional para assinar os dados de uma mídia ao vivo em P2P é inferior a 2s no pior caso. Uma assinatura completa dos *chunks*,

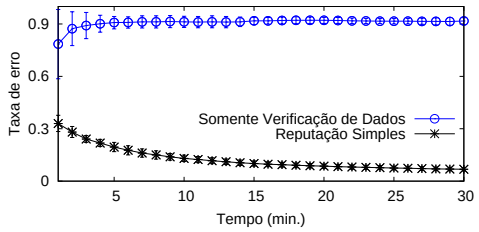


Figura 3. Taxa de erro de chunk.

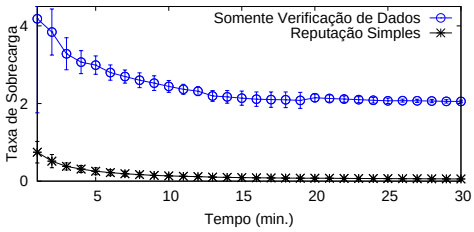


Figura 4. Sobrecarga no sistema.

pode gerar uma sobrecarga de até 30%. Porém, outras técnicas (como assinatura encadeada) apresentam uma sobrecarga menor, por volta de 5% em comparação ao fluxo de dados original [Dhungel et al. 2007, Haridasan and Renesse 2008]. Nos resultados apresentados foram desconsideradas as sobrecargas introduzidas por essas técnicas de verificação de dados. Portanto, os resultados focam na sobrecarga e atraso impostos somente pela retransmissão de *chunks* poluídos e dependem somente do tipo de mecanismo de defesa adotado.

Nos experimentos conduzidos, foram considerados 2 diferentes cenários. No primeiro, os poluidores atacam o sistema durante todo o experimento. No segundo, os poluidores atacam o sistema, porém saem e entram no sistema a cada 3 minutos aproximadamente, assumindo assim, uma nova identidade (ataque de *whitewashing*). Em ambos os cenários, os *peers* poluidores anunciam um mapa de *chunks* completo, forjando possuir todos os dados possíveis.

Não foi considerado *churn* (entrada e saída do sistema) dos *peers*. A dinâmica dos *peers* pode impactar na taxa de erro e no atraso percebido pelo usuário, porém, *churn* não afeta a eficiência dos mecanismos de reputação simplificado/modificado. Tais mecanismos não dependem da convergência da rede sobre o testemunho de um participante. Nos experimentos conduzidos, foi avaliada a dinâmica sobre o ponto de vista da troca de identidade (entrada e saída) somente dos pares maliciosos, o que de fato, pode impactar o desempenho dos sistemas de reputação.

6. Resultados Experimentais e Discussão

Nesta seção são apresentados os resultados obtidos através de experimentos realizados no PlanetLab. As métricas apresentadas são valores médios da realização de 5 experimentos, com 30 minutos de duração cada. A eficiência do sistema P2P sob ataque foi avaliado a partir de 3 métricas: a) taxa de *chunks* poluídos recebidos ou *taxa de erro*; b) a sobrecarga no sistema e; c) taxa de *chunks* perdidos ou taxa de perda. Essas métricas permitem capturar o dano causado ao sistema e também inferir a qualidade de experiência dos usuários.

A Fig. 3 apresenta a taxa de erros ao se requisitar *chunks*. A taxa de erros se refere a primeira requisição por um determinado *chunk* que um *peer* faz. Caso o *chunk* esteja poluído, o *peer* tem um erro, caso contrário, ele tem um acerto. Quanto maior a taxa de erro, maior o domínio de poluidores. Nessa figura são apresentadas a taxa de erros média em um sistema P2P utilizando um mecanismo de verificação de dados e o mecanismo de reputação. Nesse cenário, os poluidores não realizam *whitewashing*.

Nota-se que a taxa de erro de *chunks* é superior a 90% em um sistema sem

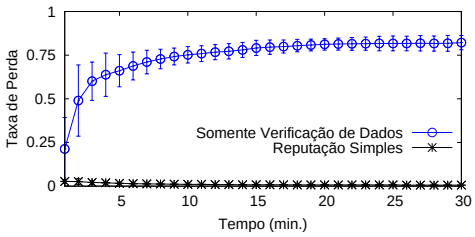


Figura 5. Taxa de perda.

reputação. Mesmo que seja verificada a integridade de um *chunk*, praticamente todos novos pedidos de dados geram uma requisição para um poluidor. Como consequência, a latência e a sobrecarga aumentam. O *mecanismo de reputação simples* proposto diminui a taxa de erro média para menos de 6%. Isso ocorre pois, ao se identificar um poluidor, o *peer* o isola rapidamente de sua lista de contatos.

A sobrecarga de retransmissão imposta por *chunks* poluídos é apresentada na Fig. 4. Observa-se que checar os dados e pedir retransmissão gera uma grande sobrecarga ao sistema P2P. Nesse caso, um *peer* apresenta em média 230% de sobrecarga por retransmissões, com pico de mais de 400%. Em outras palavras, os *peers* devem ter mais de 3 vezes a largura de banda necessária do que necessitariam em um sistema sem poluidores. O *mecanismo de reputação simples* apresenta uma sobrecarga desconsiderável. Em média, incluindo os picos de ataque, a sobrecarga é inferior a 5%.

Finalmente, a Fig. 5 mostra a taxa de perdas no tempo de execução. Uma taxa de perdas alta evidencia que o usuário está assistindo um vídeo ruim, sem a totalidade de seus quadros. Com uma janela de interesse de 20s, a taxa de perdas apresentada pela abordagem *verificar e pedir retransmissão* é elevada, acima de 75% dos *chunks* requisitados. Nesse caso, como os poluidores não são isolados do sistema, *peers* realizam requisições sucessivas a eles. Assim, não se consegue dados de um bom parceiro dentro da janela de interesse. Como o *mecanismo de reputação simples* isola rapidamente os poluidores, a taxa de perda cai para um valor desprezível, abaixo de $0,7\%$.

A Fig. 6 apresenta a taxa de erro em um cenário no qual poluidores também realizam ataques de *whitewashing*. Os resultados mostram que, mesmo com o *mecanismo de reputação simples*, *whitewashing* causa uma alta taxa de erro, alcançando quase 70%. Porém, as simples modificações realizadas reduzem a taxa de erro a 19%. Essa alta taxa de erro se deve ao fato das constantes mudanças de identidade dos poluidores.

A sobrecarga também apresenta uma piora em relação ao cenário no qual os poluidores não fazem *whitewashing*. A Fig. 7 mostra que, utilizando o *mecanismo*

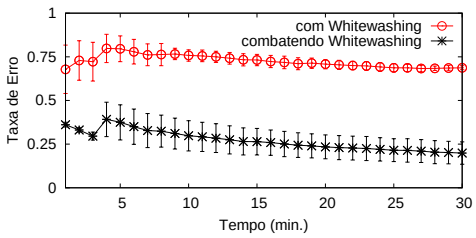


Figura 6. Taxa de erro de *chunks*.

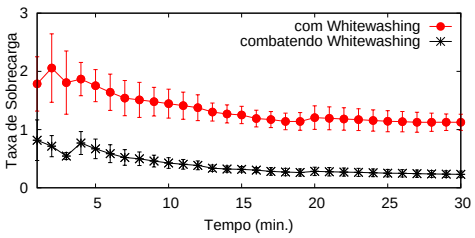


Figura 7. Sobrecarga no sistema.

de reputação simples, os *peers* experimentam 112% de sobrecarga. Ao se adequar o *mecanismo de reputação* para lidar com *whitewashing*, a sobrecarga se reduz a 20%. Novamente, esse alto valor ocorre por conta das mudanças de identidade dos poluidores.

Finalmente, de acordo com a Fig. 8, 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 alta qualidade.

Apesar dos valores altos para sobrecarga e taxa de erro, os resultados alcançados com os mecanismos de reputação propostos não podem ser subestimados. Primeiro, os esquemas de reputação propostos são simples e com implementação de baixo custo. Segundo, não são necessários custosos mecanismos de identificação de *peers*, comumente usados para evitar o ataque de *whitewashing*.

Uma das características importantes dos esquemas de reputação propostos é a recuperação de *peers bons* e que possam ter sido considerados poluidores, devido ao funcionamento incorreto da rede (falsos positivos). Para avaliar a robustez dos mecanismos propostos, um *peer* no sistema foi configurado para enviar 10% de seus *chunks*, propositalmente, como corrompidos[6], simulando assim, problemas de rede.

Foram considerados 3 valores para o limiar de calmaria (*lcalm*=$\{0{,}1;\ 0{,}2;\ 0{,}3\}$). Para $lcalm = 0{,}1$; cerca de 30% dos *peers* identificam o falso poluidor. Porém, 83% deles conseguem redimir o falso poluidor em 110 segundos. Para o caso $lcalm = 0{,}2$; 29% dos *peers* identificam o falso poluidor, sendo que 73% destes recuperam o falso positivo em 97 segundos. Por último, para o caso em que $lcalm = 0{,}3$; cerca de 27% dos *peers* identificam o falso poluidor. Nesse caso, 51% dos *peers* voltam a trocar informações com o falso poluidor em, aproximadamente, 72 segundos.

Para valores menores do parâmetro *lcalm*, uma maior percentagem de *peers* identifica os falsos poluidores, bem como os recuperam. Com valores mais baixos para esse parâmetro, os *peers* do sistema ficam mais sensíveis aos ataques de poluição. Eles constantemente vão para um estado de *tempestade* tentando se defender. Com esse comportamento, os *peers* conseguem remover rapidamente os poluidores reais, porém, eles aumentam sua taxa de falso positivo na identificação de poluidores. Mais ainda, como os *peers* podem ficar períodos maiores em estado de tempestade, o valor da reputação mínima para ser considerado um bom *peer* cresce, e assim, os falsos poluidores demoram mais tempo a serem perdoados por outros *peers*.

[6]Uniformemente distribuído entre os envios de dados.

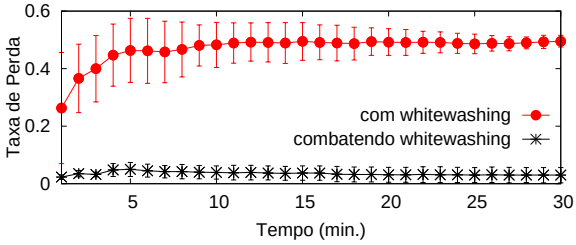


Figura 8. Taxa de Perda.

A Fig. 9 mostra o total de *chunks* recebidos pelo falso poluidor durante a experimentação. Após 10 minutos iniciais, o falso poluidor recebe entre 2900 a 3000 *chunks* a cada 30 segundos. Este valor é a taxa esperada para os demais *peers* do sistema. Assim, mesmo o falso poluidor tenha sido considerado como um poluidor de fato, ele não terá problemas ao exibir o vídeo. Este resultado está intimamente ligado à recuperação que o mecanismo implementado possibilita aos *peers* com problemas temporários.

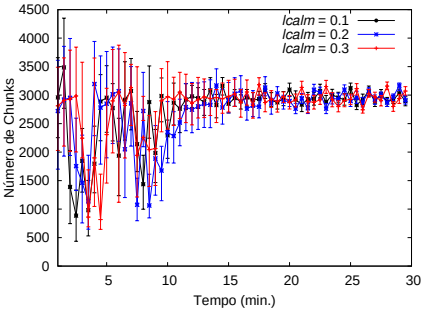


Figura 9. Número de chunks recebidos pelo falso poluidor.

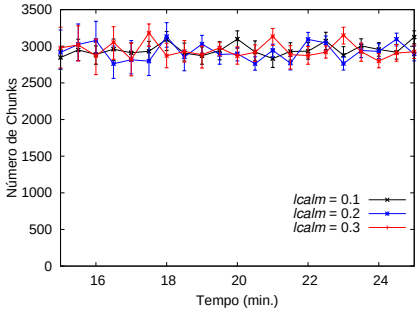


Figura 10. Número de chunks recebidos pelo falso poluidor.

A Fig. 10 mostra o número de *chunks* recebidos pelo falso poluidor, no intervalo entre 15 minutos e 25 minutos. Este período representa um intervalo estável da rede, sem o distúrbio da inicialização do ataque de poluição. Nesse intervalo, o falso poluidor recebe praticamente um fluxo de vídeo completo, 100% de *chunks*.

Para analisar a qualidade de experiência dos usuários, ao se utilizar os mecanismos de reputação propostos, considera-se a métrica atraso de exibição do vídeo. Define-se como atraso o intervalo de tempo entre criação de *chunks* pelo servidor e exibição dos mesmos pelos *peers*. A Fig. 11 apresenta o atraso médio durante uma transmissão ao vivo no sistema P2P, no qual os poluidores não realizam *whitewashing*. O atraso médio, ao se adotar somente a verificação dos dados, é superior a 2 minutos. Além do grande atraso observado, esse ambiente apresenta uma alta taxa de erros, alta taxa de perdas e grande sobrecarga. Pode-se afirmar que a qualidade de experiência dos usuários é ruim. Em contrapartida, de acordo com a Fig. 12, a adoção de um *mecanismo de reputação simples* reduz o atraso médio a apenas 6 segundos. Os usuários, além da baixa latência, apresentam baixa perda e sobrecarga imposta pelo ataque de poluição.

O atraso médio em um sistema sob ataque de poluição e *whitewashing* dos poluidores, em geral, é elevado. Ao se utilizar o *mecanismo de reputação simples*, o atraso médio é de quase 1 minuto. Com a modificação realizada, o atraso médio é reduzido a cerca de 13 segundos, valor aceitável para aplicações desta natureza.

7. Conclusões

Neste trabalho, é analisado o impacto de ataques de poluição de conteúdo em sistemas P2P de transmissão de vídeo ao vivo. Também é verificado o impactos de mudanças de identificação dos atacantes (conhecido como ataque de *whitewashing*). Nessa linha, os danos causados ao sistema P2P verificados sob duas perspectivas. Inicialmente, por meio de um modelo analítico proposto. Tal modelo captura a sobrecarga imposta à banda de rede dos participantes de um sistema de vídeo ao vivo em P2P, do tipo baseados em

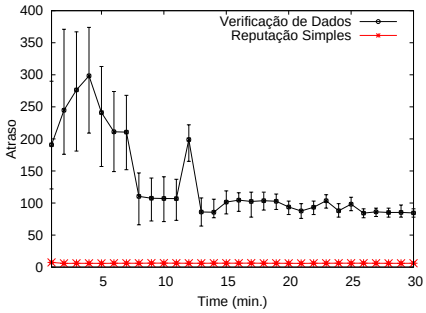


Figura 11. Atraso Médio Durante Ataque de Poluição.

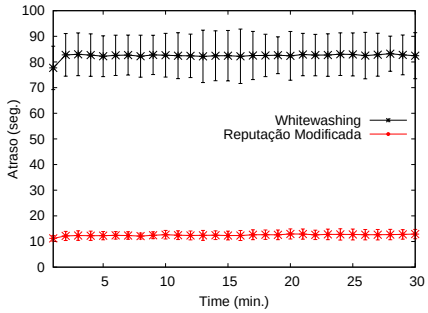


Figura 12. Atraso Médio Durante Ataque de Poluição.

malha com pedidos explícitos por dados. Em segundo lugar, por meio de experimentos conduzidos em um ambiente realista, criado no PlanetLab.

O modelo proposto evidencia que ataques de poluição são danosos a sistema P2P de transmissão ao vivo, mesmo com um número pequeno de atacantes. Os *peers* de tal sistema recebem um grande número de *chunks* poluídos e, ao realizar novas requisições, descartando o *chunk* poluído, são penalizados sobrecarregando suas respectivas larguras de banda. Por exemplo, os *peers* devem ter mais de 3 vezes a largura de banda necessária do que deveriam ter em um sistema sem poluidores.

Os resultados experimentais mostram que o sistema de reputação implementado é efetivo sob ataques de poluição. Nesse caso, a sobrecarga imposta aos *peers* do sistema cai para valores abaixo de 5%. As taxas de perda e de dados poluídos também podem ser negligenciadas. Em um cenário no qual os poluidores realizam *whitewashing*, a sobrecarga observada é de 20%. As perdas no tempo de visualização são inferiores a 3% dos *chunks*.

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.

Como trabalhos futuros, espera-se realizar verificações em novos cenários, com um número maior de participantes. Também deverão ser abordados cenários com comportamento dos participantes mais realistas, incluindo *churn*.

Referências

de Almeida, R. B., Silva, A. P. C., and Vieira, A. B. (2012). Análise do Impacto de Ataques de Poluição Combinado com Whitewashing em Sistemas P2P de Live Streaming. In *Proc. of Workshop de Testes e Tolerância a Falhas (WTF)*.

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.

Haizhou, W., Xingshu, C., and Wenxian, W. (2011). A Measurement Study of Polluting a Large-Scale P2P IPTV System. In *College of Computer Science, Sichuan University, Chengdu 610065, P. R. China*.

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

Hei, X., Liang, C., Liang, J., Liu, Y., and Ross, K. (2007). A Measurement Study of a Large-Scale P2P IPTV System. *IEEE Transactions on Multimedia*, 9(8):1672–1687.

Hei, X., Liu, Y., and Ross, K. W. (2008). IPTV Over P2P Streaming Networks: The Mesh-Pull Approach. *IEEE Communications Magazine*, 46(2):86–92.

Hu, B. and Zhao, H. (2010). Joint Pollution Detection and Attacker Identification in Peer-to-Peer Live Streaming. In *Proc. of IEEE ICASSP*.

Kudtarkar, A. and Umamaheswari, S. (2009). Avoiding White Washing in P2P Networks. In *Proc. of IEEE COMSNETS*.

Kwiatkowska, M., Norman, G., and Parker, D. (2009). PRISM: Probabilistic Model Checking for Performance and Reliability Analysis. *ACM SIGMETRICS Performance Evaluation Review*, 36(4):40–45.

Oualha, N. and Roudier, Y. (2009). A Game Theoretical Approach in Securing P2P Storage against Whitewashers. In *Proc. of 18th IEEE WETICE*.

Seibert, J., Sun, X., Nita-Rotaru, C., and Rao, S. (2010). Towards Securing Data Delivery in Peer-to-Peer Streaming. In *Proc. of IEEE COMSNETS*.

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

Vieira, A., da Silva, A. P. C., Henrique, F., Goncalves, G., and Gomes., P. (2013). Sopcast p2p live streaming: Live session traces and analysis. In *Proc. of The ACM Multimedia Systems Conference ACM MMSys 2013*.

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

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. *Elsevier Computer Communications*, 35(8):1004 – 1016.