P2P *Streaming* de Alta Definição: Análise Quantitativa de Esquemas de Assinatura Digital para Autenticação de Conteúdo

Rafael Vieira Coelho, Marinho Pilla Barcellos, Ingrid Jansch-Pôrto, Luciano Gaspary

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

rvcoelho,marinho,ingrid,paschoal@inf.ufrgs.br

Abstract. *P2P live streaming applications have great potential for improvements in terms of popularity as well as quality. However, they are subject to pollution attacks, in which a malicious user tampers with audio/video of streams in order to fool other users. The authentication of blocks in a stream allows lately the detection of polluted content and can lead to the identification of malicious users. The literature includes several proposals of light digital signature schemes. This paper, unlike previous ones, analyses such schemes in the perspective of high definition P2P live streaming. For such, it quantitatively compares overheads and the security level during P2P streaming sessions, identifying research challenges to be faced.*

Resumo. *As aplicações de* live streaming *em redes P2P têm grande potencial de crescimento, em popularidade e qualidade, mas são potencialmente vulneráveis a ataques de poluição de conteúdo. Tal ataque corresponde a qualquer adulteração de áudio e/ou vídeo na mídia transmitida buscando lubridiar espectadores. A autenticação de blocos do fluxo de dados (*stream*) permite a detecção posterior de conteúdo poluído e a identificação de pares maliciosos na rede P2P. A literatura contem diversas propostas de autenticação baseadas em assinatura digital leve. O presente trabalho, como nenhum anterior, analisa tais esquemas sob a perspectiva de transmissão de alta definição em redes P2P de* live streaming*. Para isto, faz uma comparação quantitativa de sobrecargas e nível de segurança durante transmissões em alta definição, observando fenômenos e identificando desafios de pesquisa que precisarão ser vencidos.*

1. Introdução

As aplicações de *live streaming* em redes P2P visam a transmissão de áudio e vídeo em tempo real (*soft*) de uma fonte de dados a vários receptores, para exibição instantânea nos mesmos. As aplicações de transmissão em alta definição na *Internet* dominarão o mercado nos próximos cinco anos, sendo que países como, por exemplo, Japão e Holanda já apresentam infra-estrutura suficiente para sustentar tal tecnologia [Fu et al. 2009].

Recentemente, trabalhos na literatura identificaram a manifestação do problema de poluição de conteúdo em *live streaming* [Dhungel et al. 2007, Yang et al. 2008]. A criação de mecanismos que reduzam os níveis de poluição é um problema desafiador e bastante relevante aos olhos da comunidade científica. Embora a prevenção contra a poluição de conteúdo tenha avançado no compartilhamento de arquivos, tais estratégias não podem ser diretamente aplicadas em sistemas de *live streaming*. Há duas razões para tal: as restrições temporais impostas por aplicações de *live streaming* e a falta de

conhecimento prévio sobre o conteúdo a ser enviado (o conteúdo é enviado no momento de sua geração, impossibilitando assim o cálculo de todos os *hashes* do arquivo antes da transmissão).

Poluição de conteúdo é definida de diferentes formas na literatura. Segundo Borges et al. (2008), a disseminação de conteúdo poluído ocorre quando um par altera o conteúdo da mídia, degradando a qualidade percebida pelos demais pares (estritamente, a definição omite o caso de inserção ou remoção de dados). Haridasan e Renesse (2008) definem poluição como qualquer fabricação ou adulteração dos dados sendo transmitidos no sistema, portanto incluindo também a criação de dados poluídos por atacantes mas não a remoção. Por fim, Dhungel et al. (2007) defendem que esse tipo de ataque acontece apenas quando o atacante substitui *chunks* válidos por *chunks* com dados corrompidos na distribuição de dados da rede P2P, mas não consideram a alteração de dados gerados pela fonte. Neste trabalho, define-se o ataque de poluição de conteúdo em sistemas P2P de *live streaming* como qualquer modificação da mídia (áudio/vídeo) que está senda transmitida pela fonte para um grupo de receptores com o objetivo de ludibriar os usuários que estão assistindo à transmissão.

Dhungel et al. (2007) referiram-se a um ataque de poluição em sistemas de *live streaming* como algo devastador, porque *chunks* adulterados são rapidamente distribuídos pela rede. Para evitar, ou pelo menos diminuir o impacto desse ataque, é necessária a detecção de conteúdo poluído. A estratégia clássica para autenticação de dados é o uso de assinaturas digitais [Challal et al. 2004]. Em teoria, a fonte da *stream* pode assinar os dados com uma chave privada e possibilitar aos destinatários que verifiquem a autenticidade dos dados recebidos através da chave pública da fonte. Desta forma, os pares receptores podem descartar dados falsos ou corrompidos (enviados por pares maliciosos).

A crescente demanda mundial por transmissões em alta definição leva a desafios científicos que ainda não foram apropriadamente identificados, muito menos resolvidos. O presente trabalho tem como objetivo analisar os principais esquemas de assinatura digital sob a perspectiva de transmissão de alta definição em redes P2P de *live streaming*. Para isto, é feita uma comparação quantitativa de sobrecargas e nível de segurança durante transmissões em alta definição.

O restante do trabalho é organizado como segue. Na Seção 2, são apresentadas as características dos sistemas de *live streaming* em redes P2P, convencionais e de alta definição, e as alternativas para autenticar conteúdo nesses sistemas. Na Seção 3, os esquemas mais relevantes e atuais são modelados em função do nível de segurança e sobrecargas dos mesmos. A avaliação do comportamento dos esquemas modelados e os resultados sob a perspectiva de transmissões em alta definição de sistemas P2P de *live streaming* compõem a Seção 4. Por fim, a Seção 5 resume as conclusões obtidas e perspectivas de trabalhos futuros.

2. Autenticação de Conteúdo em *Live Streaming*

Esta seção resume o funcionamento de sistemas P2P de *live streaming* e então descreve os esquemas que foram propostos para autenticação de conteúdo nos mesmos. A seção encerra com considerações sobre a transmissão de *streams* de alta definição em redes P2P, tópico que, ao que sabemos, ainda não foi abordado na literatura científica.

2.1. P2P *Live Streaming*

Neste tipo de sistema, na perspectiva usual, o conteúdo é gerado *"on-the-fly"*, codificado e transmitido aos receptores. O conteúdo gerado é dividido em *chunks* sequenciais e

identificados, de forma que cada receptor saiba que *chunks* possui e quais necessita. As topologias mais comuns para a construção do *overlay* são: em árvore ou em malha, ou combinações dessas. Tipicamente, a fonte gera *chunks* e os envia aos seus vizinhos, que redistribuem aos seus próprios vizinhos. A fonte/um par envia os *chunks* simplesmente (*push*) ou anuncia a disponibilidade de novos *chunks* e aceita solicitações de transmissão (*pull*).

Caso não seja empregada alguma forma de autenticação dos *chunks*, pares maliciosos podem adulterar os *chunks* recebidos antes de repassá-los aos demais pares da rede P2P. Existem três estratégias básicas para combater isso, tratadas a seguir. Primeiro, algoritmos clássicos (*e.g.* DSA) proporcionam uma forma segura de autenticação de dados através do uso complexidade matemática, porém as sobrecargas impostas por tais esquemas não permitem seu uso direto em sistemas de *live streaming* devido às restrições temporais.

A segunda estratégia, que consiste em esquemas de amortização, gera uma única assinatura para um bloco de *chunks*, assim espalhando as sobrecargas no tempo. A terceira estratégia visa reduzir o custo computacional através da utilização de funções unidirecionais de *hash* para a criação e verificação de assinaturas. Neste trabalho são consideradas as duas últimas estratégias.

2.2. Esquemas de Amortização de Assinatura

O esquema mais simples de amortização é aquele no qual deve ser aplicada uma função de *hash* sobre cada *chunk* que será enviado. O resultado desse *hash* deve ser codificado com a chave privada da fonte, gerando assim o respectivo *digest*. O *digest* deve ser enviado juntamente com o *chunk* correspondente. Desta forma, os receptores podem verificar a autenticidade dos dados que estão recebendo através da comparação entre o *hash* sobre os dados e a decodificação do *digest*. A coincidência entre eles indica que, os dados recebidos são válidos. Caso contrário, os dados não devem ser repassados ou reproduzidos, pois foram comprometidos ou estão corrompidos.

Em busca de eficiência, surgiram diversos trabalhos, incluindo os baseados em árvore [Wong and Lam 1999], em grafos [Miner and Staddon 2001] e em FECs para resistir à perda de pacotes (SAIDA) [Park et al. 2003]. Uma das propostas mais recentes de amortização é o esquema Linear Digests [Haridasan and van Renesse 2008]; resultados preliminares apresentados pelos autores dão indícios de eficiência. Por essa razão, foi um dos esquemas adotados para avaliação no presente trabalho.

O Linear Digests é utilizado no SecureStream [Haridasan and van Renesse 2008], uma aplicação de *multicast* na camada de aplicação para transmissão de *live streaming*, e consiste no seguinte mecanismo: o agrupamento de *chunks* em blocos de tamanho fixo, o cômputo de um *hash* para cada *chunk* do bloco, a geração de uma assinatura da concatenação dos *hashes* do bloco com a chave privada da fonte, seguida do envio da mensagem contendo essa assinatura previamente à distribuição dos *chunks* pertencentes ao bloco. A partir do momento que um par recebe a mensagem contendo a assinatura dos *hashes* no bloco, ele pode verificar a autenticidade de qualquer *chunk* do bloco correspondente.

2.3. Esquemas de Assinatura Baseados em Funções Unidirecionais de *Hash*

Um conjunto de estratégias usadas para autenticação de conteúdo em P2P *live streaming* emprega assinatura digital baseada em função unidirecional de *hash*. Os esquemas mais relevantes são resumidos a seguir. A segurança dos mesmos baseia-se no fato de que as

funções de *hash*, utilizadas para criar as assinaturas e a chave pública, são unidirecionais e não podem ser revertidas por possíveis atacantes.

BIBA é um protocolo de autenticação de *broadcast* que utiliza um esquema de assinatura digital de mesmo nome para fazer a verificação da origem de uma seqüência de *chunks* [Perrig 2001]. Ele se baseia na tentativa de encontrar colisões para criar assinaturas. Como tentativa de aperfeiçoamento, em L. Reyzin e N. Reyzin (2002) é apresentado um esquema de assinatura chamado HORS, que busca diminuir o tempo de assinatura, tamanho das chaves e das assinaturas do BIBA.

PARM [Lin et al. 2006] é um esquema de autenticação de conteúdo baseado em funções unidirecionais de *hash* para aplicações *multicast*. Atua em conjunto com o SAIDA [Park et al. 2003], um sistema de amortização de assinatura projetado para uso em ambientes sujeitos a altas perdas de pacotes através de esquemas de codificação. No esquema de assinatura PARM, a fonte de dados deve criar um vetor com valores randômicos. Previamente à transmissão, são feitas operações sucessivas de *hash*, formando assim uma **matriz de evidências**. A última linha de valores corresponde à chave pública e deve ser disponibilizada de forma segura aos receptores.

Para criar a assinatura de um *chunk*, a fonte faz o *hash* do mesmo. O resultado desse *hash* é divido em segmentos de *bits* que são interpretados como índices das colunas das evidências. Já os índices das linhas são obtidos através de uma **tabela de uso** na qual são mantidos contadores (iniciados em zero, indicando a última linha) de uso dos índices das linhas. Esse último índice indica a quantidade de operações de *hash* que devem ser feitas nas evidências para comparar com os valores do vetor que forma a chave pública (os índices das colunas indicam os índices dos vetores na chave pública).

Cada receptor mantém também uma tabela de uso para controle dos *chunks* recebidos. Mensagens devem ser entregues aos receptores em ordem. Para que os receptores possam verificar a autenticidade dos *chunks*, todas as evidências devem ser validadas através da tabela de uso. Além disso, quando o número de elementos usados na primeira linha da matriz excede certo limiar, é necessário criar novos números randômicos para a primeira linha de valores da matriz de evidências e novas cadeias de *hash* das colunas em questão. O vetor de chave pública parcial deve ser assinado digitalmente por um esquema clássico, codificado com códigos de correção de erros e enviado aos receptores para atualização local.

Lin et al. (2008) propuseram diversas alterações no esquema de assinatura PARM como tentativa de melhoria, resultando em uma nova versão (doravante denominada PARM 2). Primeiro, os receptores não utilizam tabela de uso, pois a fonte concatena os valores da tabela de uso (número de *hashes* para chegar à chave pública) e um número de seqüência junto ao *chunk* que também é verificado pelos receptores. Segundo, os receptores utilizam w *buffers* contendo evidências recebidas por último para tentar diminuir o número de operações de *hash* e com isso o tempo de verificação da assinatura. Terceiro, a fonte inicia o processo de renovação das colunas de *hashes* nas quais a metade de suas evidências tenha sido enviada.

O último esquema considerado é o ALPS, um mecanismo de assinatura digital que foi desenvolvido para aplicações de *live streaming* em redes P2P [Meier and Wattenhofer 2008]. Propõe superar limitações (atrasos, tamanho da assinatura e complexidade computacional) impostas por abordagens baseadas em esquemas clássicos de assinatura digital (*e.g.* DSA) e esquemas de assinaturas propostos anteriormente.

ALPS cria uma matriz de evidências (o termo não é utilizado pelos autores, mas é empregado no presente trabalho para uniformização), similarmente ao PARM. No entanto, divide a matriz de evidências em sub-matrizes (**blocos**). Portanto inicia a seleção de evidências pelo último bloco e sobe na matriz até o início da mesma, ou seja, até o primeiro bloco criado. Para a seleção das evidências, são feitos *hashes* sucessivos no *chunk* em questão e num contador de segurança.

Para verificar a autenticidade do *chunk*, um receptor efetua o *hash* do *chunk* e do contador recebidos para calcular os índices das colunas, indicando assim os índices do vetor de chave pública para validação. Em seguida, faz o *hash* do *chunk*, do contador e dos índices de coluna para calcular os índices das linhas. É necessário seguir a cadeia de *hash* das evidências recebidas até encontrar uma evidência pré-armazenada em *buffer* ou da chave pública. Ao ser encontrado um valor idêntico ao computado, compara-se o número de passos do cálculo com o índice da linha na qual está o valor. Além disso, o ALPS utiliza renovação completa de chaves: caso o número de blocos tenha terminado e a fonte ainda tenha dados para enviar, ela deve recriar a matriz de evidências.

2.4. *Streaming* P2P em Alta Definição

A televisão de alta definição (*High Definition Television - HDTV*) é caracterizada pela transmissão de vídeos com alta resolução (1280x720 ou 1920x1080 *pixels*) [Poynton 2003]. O padrão H.264 é o mais indicado para transmissões em alta definição (perfil *High*). Ele dobrou a taxa de compressão se comparado ao MPEG-2, mas a complexidade computacional do H.264 é maior que a do MPEG-2. Maiores informações sobre H.264 podem ser encontradas em Topiwala et al. (2009).

Até hoje, pelo que sabemos, nenhum trabalho analisou a questão de transmissões de alta definição em redes P2P de *live streaming*. Mais especificamente, é inédita a investigação sobre formas de autenticar o conteúdo dessas transmissões que não sejam demasiadamente onerosas e que proporcionem um nível de segurança aceitável. Tendo como base essa lacuna existente na literatura, o presente trabalho busca avaliar o nível de segurança e as sobrecargas dos esquemas de assinatura digital e de amortização de assinatura mais relevantes ao contexto de transmissões de alta definição em redes P2P.

Embora Hefeeda e Mokhtarian (2010) tenham feito uma análise comparativa entre esquemas de amortização de assinatura digital, utilizando como métricas: custo computacional, sobrecarga de comunicação, tamanho do *buffer* nos receptores, atraso e tolerância à perda de pacotes, o estudo está restrito a transmissões convencionais. Além disso, eles se detêem em esquemas de amortização de assinatura enquanto que o presente trabalho, além dos esquemas de amortização, inclui os baseados em funções unidirecionais de *hash*.

3. Modelagem dos Esquemas de Assinatura

Nesta seção, é proposto um conjunto de equações que expressam a eficiência de cada um dos esquemas de assinatura e de amortização relevantes ao contexto de transmissão de *streaming* em alta definição. Os esquemas analisados são **Linear Digests** (do SecureStream), **PARM**, **PARM 2** e **ALPS**. As métricas consideradas são: o nível de segurança oferecido durante a transmissão (o esquema Linear Digests não é considerado na seção 1 pois o nível de segurança dele depende fortemente do esquema de assinatura empregado, que fica em aberto na publicação), as sobrecargas computacionais e de armazenamento impostas aos receptores (possivelmente subprovisionados) e a sobrecarga de comunicação.

As equações doravante utilizadas foram propostas por: 1 e 8, por Lin et al. (2006); 6, 10 e 13, por Meier e Wattenhofer (2008); e 9, 12 e 16 por Lin et al. (2008). As demais equações foram desenvolvidas por nós para este trabalho.

3.1. Nível de Segurança

O nível de segurança expressa o inverso da probabilidade de um par malicioso conseguir produzir uma assinatura válida para um *chunk* poluído (ou seja, um *chunk* não criado pela fonte). Este valor tende a diminuir à medida que assinaturas são recebidas da fonte (Equações 1, 3 e 5). Também é interessante verificar a quantidade máxima de assinaturas proporcionados pelos esquemas pela visão da fonte de dados (Equações 2, 4, 6). Na Tabela 1, são apresentadas as equações de estimativa de segurança dos esquemas mais recentes e a quantidade total de possíveis assinaturas que podem ser criados pelos mesmos. Os símbolos utilizados nas equações são: número de evidências (p); número de colunas (k); número de linhas (q); tamanho do bloco (b); probabilidade de falha na assinatura (P_f); limiar da renovação (t); número total de evidências reveladas (R).

Tabela 1. Segurança nos Esquemas de Assinatura Digital

	Nível de Segurança		**Número de Possíveis Assinaturas**	
PARM	$(q \times k/R)^p, R/q \leq t$	(1)	$q^p \times (k+p-1)!/(p! \times (k-1)!)$	(2)
PARM2	$(q \times k/R)^p, (2 \times R)/k \leq t$	(3)	$q^p \times (k+p-1)!/(p! \times (k-1)!)$	(4)
ALPS	$(k \times b/R)^p$	(5)	$b^p \times (1-Pf) \times k!/(p! \times (k-p)!)$	(6)

As Equações 1, 3 e 5 referem-se ao nível de segurança baseado no inverso da probabilidade de um poluidor conseguir criar uma assinatura válida, tendo como base as evidências reveladas por assinaturas prévias. Nesse caso, é necessário dividir o número de possíveis evidências pelo número de evidências reveladas aos receptores e elevar esse resultado ao número de evidências por assinatura.

No caso do ALPS (Equação 5), o nível de segurança é calculado pela multiplicação do número de colunas k pelo tamanho do bloco b, que é dividido pelo número de evidências reveladas R e elevado pelo número de evidências por assinatura p. Da mesma maneira, para o PARM e PARM 2, é multiplicado o número de colunas k pelo número de linhas q, esse resultado é dividido pelo número de evidências reveladas R e elevado pelo número de evidências por assinatura p. Uma das principais diferenças entre PARM e PARM 2 é a renovação de chaves, já que o PARM renova sua chave após serem utilizadas de forma repetida as evidências da primeira linha da matriz mais de t vezes. Já no caso do PARM 2, renovam-se as colunas de *hash* nas quais foram utilizadas mais da metade de suas evidências.

Tendo como base o número de possíveis assinaturas (Equações 2, 4 e 6), o número máximo de assinaturas é calculado pela multiplicação do número de possíveis índices para as linhas das evidências pelo número de possíveis índices das colunas. Por exemplo, para a Equação 6, o número de possíveis assinaturas do ALPS é definido pelo arranjo com repetição de b elementos escolhidos $p-1$ a $p-1$ que é multiplicado pela combinação de k colunas escolhidas p a p sem repetição. As Equações 2 e 4, por sua vez, são definidas pela multiplicação do arranjo com repetição de L elementos escolhidos p a p pela combinação de k colunas p a p com repetição.

3.2. Custo Computacional nos Receptores

No presente artigo, esta métrica representa o custo total de processamento nos receptores para verificar a autenticidade dos dados recebidos. O custo computacional do Linear Digests, para verificar a autenticidade de um *chunk* nos receptores (Equação 7) é calculado através do tempo para computar um *hash* (t_H) de um *chunk* do bloco e pelo tempo de verificação do esquema de assinatura digital utilizado pelo Linear Digests (t_V), dividido pelo número de *chunks* em um bloco (c).

$$t_H + t_V/c \tag{7}$$

Como o PARM é usado em conjunto com o esquema de amortização SAIDA, os tempos das ações de amortização e de codificação visando correção de erros antes do envio também devem ser computados e são adicionados à equação que faz o cômputo do tempo necessário ao processamento das operações de *hash* para verificar um conjunto de evidências. No PARM (Equação 8), o número de operações de *hash* é obtido através da multiplicação entre o número de linhas da matriz de evidências, o número de evidências por assinatura e o tempo de uma operação de *hash*. No PARM 2, a equação é semelhante, mas não inclui ações relacionadas ao SAIDA nem com códigos, que não são usados (Equação 9).

$$q \times p \times t_H + t_{EC} + t_S \tag{8}$$

$$q \times p \times t_H \tag{9}$$

Por fim, o tempo gasto por receptores no ALPS (Equação 10) é definido pelo número de operações de *hash* necessárias para verificar uma assinatura. Para isto, é computado o produto entre o número de linhas de um bloco da matriz de evidências (b), número de evidências por assinatura (p) e tempo de uma operação de *hash* (t_H).

$$2 \times b \times p \times t_H \tag{10}$$

3.3. Sobrecarga de Comunicação

A sobrecarga de comunicação é calculada como a quantidade adicional de *bytes* necessária ao esquema de assinatura, que deve ser enviada em cada pacote para que os receptores possam verificar a autenticidade dos dados. A sobrecarga de comunicação imposta pelo esquema de amortização de assinatura Linear Digests é diretamente proporcional ao esquema de assinatura digital utilizado pelo mesmo. Por exemplo, caso seja utilizado o esquema clássico de assinatura RSA, a sobrecarga de comunicação é definida pelo tamanho de uma assinatura (é constante e foi usado 1024 *bits* no presente trabalho), dividido pelo número de *chunks* por bloco (c). Já o tamanho da assinatura do PARM corresponde a p evidências de n *bytes* cada, como pode ser observado na Equação 11. No caso do PARM 2 (Equação 12), além da assinatura, junto ao *chunk* são enviados k valores da tabela de uso, cada um com $\lg q$ *bytes* (utilizados para obter o número de *hashes* necessários para a verificar as evidências) e o número de sequência (i *bytes*). Por fim, no ALPS (Equação 13) são enviados p evidências (assinatura) de n *bytes* e o contador de segurança utilizado nos cômputos de *hash* com d *bytes*.

$$p \times n \tag{11}$$

$$p \times n + k \times \lg q + i \tag{12}$$

$$p \times n + d \tag{13}$$

3.4. Sobrecarga de Armazenamento nos Receptores

A sobrecarga de armazenamento é calculada como a quantidade total de *bytes* que deve ser dedicada ao armazenamento de estruturas de dados no mecanismo de assinatura no receptor. No Linear Digests, é feita uma operação de *hash* por *chunk* dos blocos (Equação 14). Sendo assim, a sobrecarga de armazenamento proporcionada pelo esquema nos receptores é calculada através do número de *chunks* por bloco (c) vezes o tamanho da saída de uma operação de *hash* (h *bytes*) e do tamanho da chave pública de assinatura utilizado pelo Linear Digests (P_k *bytes*).

$$h \times c + P_k \tag{14}$$

Já no PARM (Equação 15), deve-se considerar as sobrecargas impostas pelo tamanho da chave pública (k valores de n *bytes*) e pelos valores da tabela de uso (k valores de $\lg q$ *bytes*) que precisam ser armazenados pelos receptores. Enquanto isso, no PARM 2 (Equação 16), além da chave pública e da tabela de uso, deve-se considerar o tamanho dos *buffers* (w *buffers* são compostos por k evidências de n *bytes*) e o tamanho do número de sequência (i *bytes*).

$$k \times n + k \times \lg q \tag{15}$$

$$k \times n \times w + k \times n + i \tag{16}$$

Por fim, no ALPS (Equação 17) deve-se considerar o tamanho da chave pública, o tamanho dos *buffers* e o tamanho do contador de segurança (d *bytes*) utilizados no cômputo dos *hashes*. A chave pública também corresponde a k valores de n *bytes* cada e os w *buffers* também contém k evidências de n *bytes*.

$$k \times n \times w + k \times n + d \tag{17}$$

4. Resultados

Nesta seção, os principais esquemas de assinatura digital e de amortização de assinatura para P2P *streaming* são comparados com base no conjunto de equações estabelecido na seção anterior. Para isto, são levadas em consideração a segurança e as sobrecargas de comunicação, e nos receptores, as sobrecargas de armazenamento e de custo computacional.

Para as simulações, foi utilizado um *hardware* Intel Core 2 Duo 1.66GHz com 1014M de RAM rodando Windows Vista. Assume-se como premissa que não há picos de processamento durante a sessão suficientemente fortes a ponto de afetar a viabilidade da transmissão em tempo real. São apresentados gráficos de barra para ilustrar os resultados obtidos considerando os seis cenários de P2P *streaming*.

4.1. Cenários e Parâmetros

Para realizar a análise quantitativa, foram considerados os cenários apresentados na Tabela 2. Eles representam a transmissão de *live streaming* de alta definição em redes P2P, sendo consideradas transmissões de curta e de longa duração para resoluções 720p e 1080p, além de dois cenários convencionais, para comparação. A largura de banda necessária oscila entre 5Mbps e 15 Mbps, seguindo evidências experimentais e reportadas por fabricantes de equipamentos de transmissão em alta definição [Cisco 2009]. Com a cobertura desses cenários, é possível verificar o desempenho dos esquemas de assinatura em situações de alta definição com requisitos distintos. Os parâmetros utilizados no PARM podem ser observados na Tabela 3, e foram escolhidos com base no trabalho dos

Tabela 2. Cenários empregados na análise dos esquemas de assinatura

Cenário	1. **Conv-Curta**	2. **1080-Curta**	3. **720-Curta**	4. **Conv-Longa**	5. **1080-Longa**	6. **720-Longa**
Resolução	Conv.	1080p	720p	Conv.	1080p	720p
Duração	curta	curta	curta	longa	longa	longa
Sessão (min)	10	10	10	120	120	120
Largura banda (Mbps)	0,384	16	5	0,384	16	5
Total de *chunks*	3.600	4.800	6.000	43.200	57.600	72.000
Taxa (*chunks*/s)	6	8	10	6	8	10
Tamanho *chunk* (KB)	8	256	64	8	256	64
Volume dados (MB)	28,13	1.200	375	337,5	14.400	4.500
Chunks por bloco	188	6	24	188	6	24

respectivos autores [Lin et al. 2006]. O nível de segurança é diretamente proporcional a q, o número de linhas da matriz de evidências. Além disso, aumentando o limiar t para a renovação parcial de chaves, o número de renovações diminui e o intervalo entre as renovações parciais aumenta. A renovação parcial de chaves é necessária porque o nível de segurança decresce exponencialmente a cada *chunk* assinado. Ou seja, tal se deve à queda no nível de segurança à medida que as evidências são reveladas.

Tabela 3. Parâmetros do PARM

Parâmetro	**Valor**	**Explicação**
p	16	Número de evidências por assinatura
k	512	Número de colunas da matriz de evidências
q	10	Número de linhas da matriz de evidências
t	1.000 (cenários 1, 2 e 3) 10.000 (cenários 4, 5 e 6)	Limiar para a renovação parcial da chave pública

Lin et al. (2008) não sugerem valores para configuração do PARM 2. Na impossibilidade de realizar uma análise de sensibilidade para este protocolo (e os demais), e considerando a similaridade entre PARM e PARM 2, adotou-se basicamente os mesmos parâmetros nos dois casos. A diferença reside no parâmetro t, limiar para a renovação parcial da chave pública, pois foi apontado o valor 200 por Lin et al. (2006).

Por fim, quanto ao ALPS, seus parâmetros são apresentados na Tabela 4. Os valores foram determinados tendo como base o trabalho de análise de sensibilidade feita pelos respectivos autores [Meier and Wattenhofer 2008]. Baseados na fórmula de segurança (número de possíveis assinaturas), os autores afirmam que o nível de segurança para *live streaming* em redes P2P deve se manter entre 2^{50} e 2^{60}. Sendo assim, indicam os valores necessários para alcançar um nível de segurança de 2^{56}. Os autores avaliaram o caso no qual o número de colunas da matriz de evidências (k) é igual ao número de evidências por assinatura. A partir desse cenário, eles concluiram que, para manter certo nível de segurança, é necessário existir uma compensação entre o número de evidências por assinatura (p) e o tamanho do bloco (b). Portanto, ou um número maior de evidências por assinatura ou blocos maiores devem ser usados para compensar a falta de flexibilidade na seleção dos índices das colunas.

4.2. Nível de Segurança

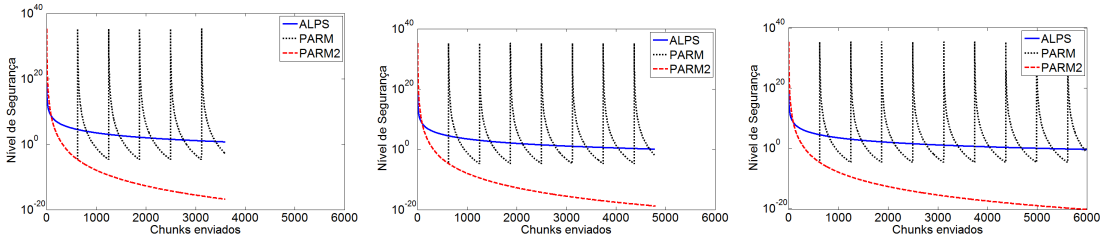
Para cada cenário proposto, foram avaliados o nível de segurança (inverso da probabilidade de um atacante conseguir criar uma assinatura válida) por *chunk* enviado para

Tabela 4. Parâmetros do ALPS

Parâmetro	Valor	Explicação
p	5	Número de evidências por assinatura
k	50	Número de colunas da matriz de evidências
b	489	Número de linhas de um bloco da matriz de evidências

cada esquema de assinatura digital baseado em funções unidirecionais de *hash* estudados: PARM, PARM 2 e ALPS.

Como pode ser observado na Figura 1, nos cenários 1, 2 e 3, houve semelhança entre os comportamentos dos diferentes esquemas. PARM e PARM 2 apresentaram, inicialmente, o maior nível de segurança. A queda gradual no nível de segurança do ALPS foi mais suave em relação ao PARM 2. Por outro lado, o esquema PARM, devido à renovação parcial de chaves, acabou voltando a ter o nível de segurança proporcionado no início da transmissão. Devido ao número menor de *chunks* transmitidos, o esquema de assinatura PARM 2 não renovou parcialmente sua chave pública, o que fez com que seu nível de segurança decaísse drasticamente após 500 *chunks* enviados.

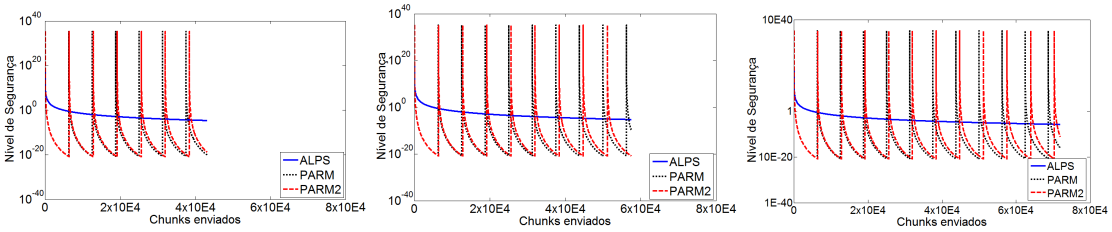


(a) Cenário 1 Conv-Curta (b) Cenário 2 1080-Curta (c) Cenário 3 720-Curta

Figura 1. Níveis de segurança dos esquemas de assinatura nos cenários 1, 2 e 3

Nos cenários 4, 5 e 6 (Figura 2), PARM e PARM 2 apresentaram comportamentos semelhantes entre si. Como pode ser observado nas Figuras 2(b) e 2(c), em todos os cenários que consideram a transmissão em alta definição de longa duração, nenhum dos esquemas de assinatura digital baseados em funções unidirecionais de *hash* estudados (PARM, PARM 2 e ALPS) conseguiu proporcionar um nível de segurança aceitável no contexto de *live streaming* em alta definição. Dentre eles, o que obteve o melhor desempenho foi o ALPS, considerando que o nível de segurança do mesmo se manteve mais estável durante a transmissão. Para ilustrar, no Cenário 5 1080-Longa, com apenas 0.5×10^4 *chunks* enviados, o nível de segurança do PARM e do PARM 2 atingiu aproximadamente 10^{-20}, enquanto o nível no ALPS ficou próximo de 1.

Para que proporcionassem um nível de segurança aceitável nos cenários de alta definição, os esquemas precisariam ser ajustados substancialmente em seus parâmetros, através de uma análise de sensibilidade. Baseados nos *tradeoffs* envolvidos, as sobrecargas dos esquemas seriam aumentadas drasticamente. No caso do PARM e PARM 2, a sobrecarga de comunicação aumentaria devido ao maior número de renovações parciais de chave. Já no caso do ALPS, após uma adaptação para o cenário de alta definição (aumento da matriz de evidências), o custo computacional nos receptores para verificar assinaturas aumentaria em grande proporção. As sobrecargas são analisadas nas subseções seguintes.

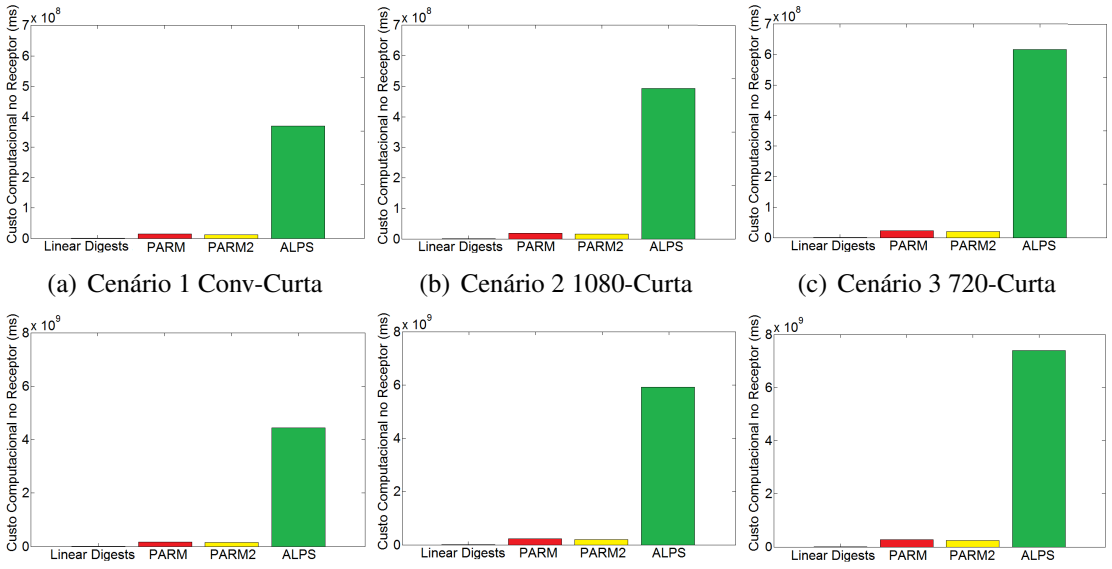


(a) Cenário 4 Conv-Longa (b) Cenário 5 1080-Longa (c) Cenário 6 720-Longa

Figura 2. Níveis de segurança dos esquemas de assinatura nos cenários 4, 5 e 6

4.3. Custo Computacional nos Receptores

A Figura 3 apresenta o custo computacional (tempo) imposto aos receptores em cada esquema estudado, por cenário. No Linear Digests, foi escolhido o esquema clássico de assinatura digital RSA. Segundo Meier e Wattenhofer (2008), o tempo de verificação do RSA é de 324μs e o tamanho da assinatura é de 128 *bytes*, em um teste feito em Java num processador 2.66GHz. O custo computacional para os receptores no Linear Digests é o menor custo entre os esquemas. Nos cenários de curta duração (cenários 1, 2 e 3), o custo computacional do Linear Digests ficou entre 7.67×10^4 ms e 1.28×10^5 ms. Enquanto isto, nos cenários de longa duração (cenários 4, 5 e 6), seu custo ficou entre 9.21×10^5 ms e 1.53×10^6 ms.

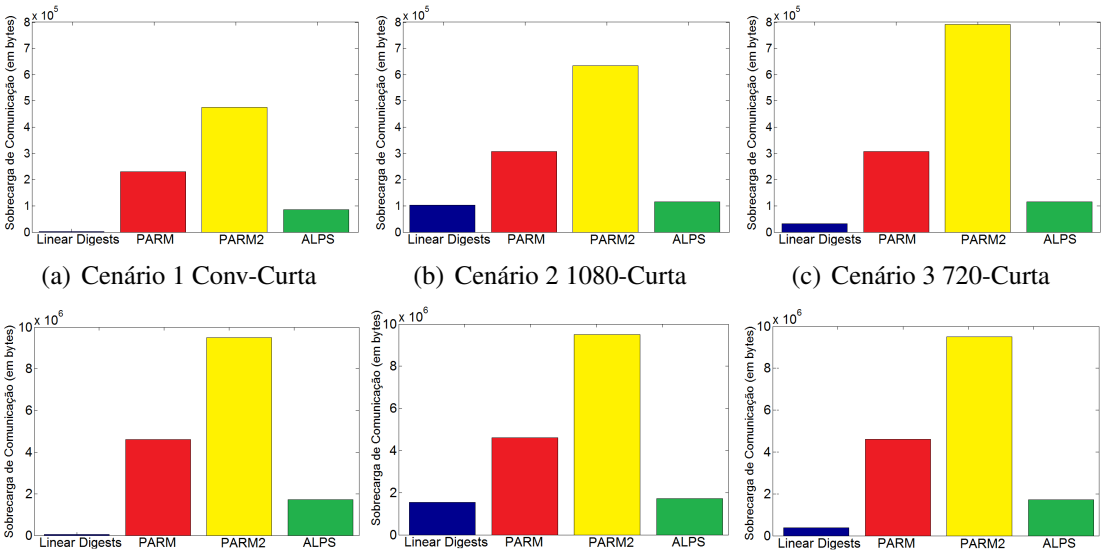


(a) Cenário 1 Conv-Curta (b) Cenário 2 1080-Curta (c) Cenário 3 720-Curta

(d) Cenário 4 Conv-Longa (e) Cenário 5 1080-Longa (f) Cenário 6 720-Longa

Figura 3. Custo computacional nos receptores para esquemas Linear Digests, PARM, PARM 2 e ALPS

A faixa de custos do PARM e PARM 2, nos cenários de curta duração, ficou aproximadamente entre 1.21×10^7 ms e 2.31×10^7 ms, como pode ser observado nas Figuras 3(a), 3(b) e 3(c). Já nos cenários de longa duração, eles ficaram entre 1.45×10^8 ms e 2.78×10^8 ms. Apesar do PARM ter um custo adicional em relação ao PARM2 devido a utilização do esquema de amortização SAIDA e de códigos de correção de erro durante a renovação parcial de chaves, ambos apresentam custos semelhantes, pois o custo incorrido pela execução de ações adicionais no PARM é proporcionalmente pequeno em



(a) Cenário 1 Conv-Curta (b) Cenário 2 1080-Curta (c) Cenário 3 720-Curta

(d) Cenário 4 Conv-Longa (e) Cenário 5 1080-Longa (f) Cenário 6 720-Longa

Figura 4. Sobrecarga de comunicação para esquemas Linear Digests, PARM, PARM 2 e ALPS

face do custo total. Segundo Hefeeda e Mokhtarian (2010), para um bloco com 50 *chunks*, o tempo gasto para utilizar o SAIDA é aproximadamente 600 ms, sendo que o custo total do PARM é de 3860 ms por *chunk*.

Por fim, o ALPS apresentou o maior custo computacional nos receptores tanto nos cenários de curta quanto nos cenários de longa duração, devido à quantidade de operações de *hash* necessárias para calcular tanto os índices das linhas quanto os índices das colunas das evidências a serem verificadas. Os custos do ALPS, nos cenários 1, 2 e 3, ficaram entre 3.69×10^8 ms e 6.16×10^8 ms e, nos cenários 4, 5 e 6, entre 4.43×10^9 ms e 7.39×10^9 ms.

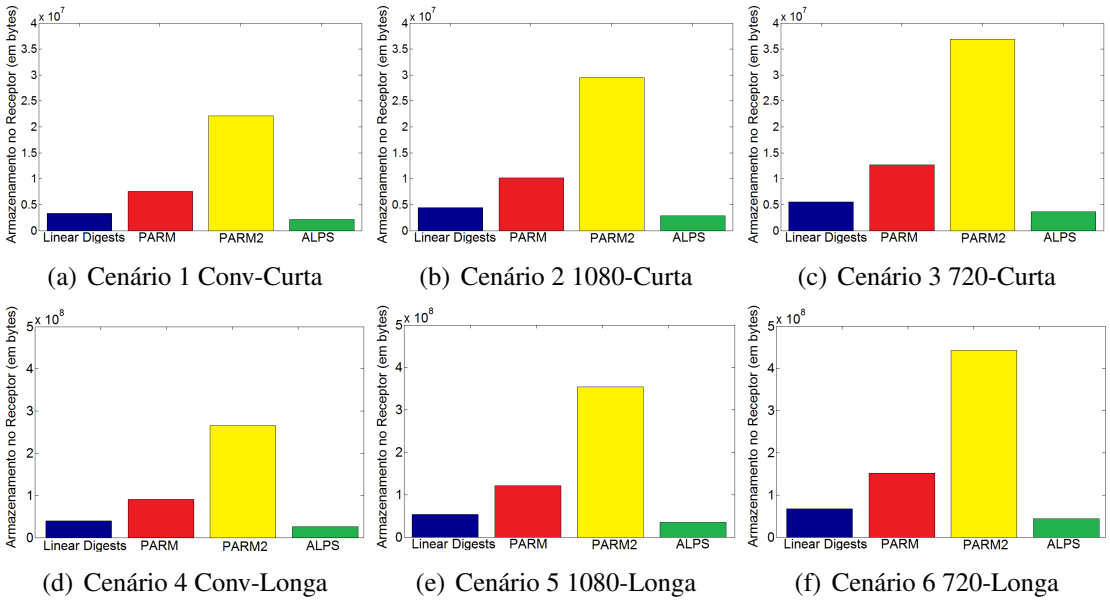
4.4. Sobrecarga de Comunicação

As sobrecargas de comunicação dos esquemas analisados por cenário são mostradas na Figura 4. A sobrecarga de comunicação apresentada pelo Linear Digests foi a mais baixa entre os esquemas pois apenas é necessário enviar uma mensagem por cada bloco de c chunks. Nos cenários de curta duração (cenários 1, 2 e 3), o Linear Digests apresentou sobrecargas entre 2.44×10^3 *bytes* e 10.23×10^4 *bytes*. Já nos cenários de longa duração, suas sobrecargas de comunicação ficaram entre 2.94×10^4 *bytes* e 1.23×10^6 *bytes*.

O ALPS, por sua vez, obteve a segunda menor sobrecarga. Nos cenários 1, 2 e 3, as sobrecargas de comunicação ficaram entre 8.64×10^4 *bytes* e 1.15×10^5 *bytes*, e nos cenários 4, 5 e 6, entre 1.03×10^6 *bytes* e 1.73×10^6 *bytes*. Como pode ser observado na Figura 4, o PARM 2 apresenta a maior sobrecarga de comunicação entre os esquemas analisados, devido ao envio da tabela de uso junto com a mensagem.

4.5. Sobrecarga de Armazenamento nos Receptores

Como pode ser observado na Figura 5, o ALPS apresentou a menor sobrecarga de armazenamento nos receptores. Nos cenários 1, 2 e 3 (curta duração), o ALPS apresentou sobrecarga de armazenamento entre 2.17×10^6 *bytes* e 3.62×10^6 *bytes*. Como pode ser



(a) Cenário 1 Conv-Curta (b) Cenário 2 1080-Curta (c) Cenário 3 720-Curta

(d) Cenário 4 Conv-Longa (e) Cenário 5 1080-Longa (f) Cenário 6 720-Longa

Figura 5. Sobrecarga de armazenamento nos receptores para esquemas Linear Digests, PARM, PARM 2 e ALPS

observado nas Figuras 5(d), 5(e) e 5(f), o ALPS apresentou a faixa de sobrecarga entre 2.61×10^7 *bytes* e 4.34×10^7 *bytes* nos cenários de longa duração.

O maior valor de sobrecarga é proporcionado pelo PARM 2, devido aos *buffers* e números de sequência armazenados. A maior sobrecarga do PARM 2 nos cenários 1, 2 e 3 é de 3.68×10^7 *bytes* e a menor de 2.21×10^7 *bytes*. Nos cenários 4, 5 e 6, o PARM 2 apresentou sobrecargas entre 2.65×10^8 *bytes* e 4.42×10^8 *bytes*. Por fim, para utilizar o PARM, os pares receptores precisam armazenar entre 7.6×10^6 *bytes* e 1.26×10^7 *bytes* nos cenários de curta duração e entre 9.12×10^7 *bytes* e 1.52×10^8 *bytes* nos cenários de longa duração, sendo que cada tabela de uso corresponde a 512 *bytes*.

5. Conclusões

Transmissões em alta definição são uma tendência do mercado, porém ainda não foram exploradas no contexto de *live streaming* em redes P2P. Neste trabalho, foram analisados os principais esquemas de assinatura digital sob essa perspectiva.

A partir da análise quantitativa realizada, foi possível concluir que os esquemas atuais são incapazes de lidar com as demandas de um sistema P2P de *live streaming* para alta definição no que concerne autenticação de conteúdo. Quando esses sistemas estiverem em uso, serão vulneráveis a ataques de poluição de conteúdo. Para atingir um nível de segurança aceitável nos cenários de alta definição, seria necessário ajustar os parâmetros levando a um aumento substancial das sobrecargas de processamento, comunicação e armazenamento. Em particular, no caso do PARM e PARM 2, o maior impacto seria na sobrecarga de comunicação, devido ao maior número de renovações. No caso do ALPS, seria necessário aumentar substancialmente a matriz de evidências, o que resultaria em um alto custo computacional nos receptores devido às operações de *hash* para verificar assinaturas.

Como trabalhos futuros, identificamos três ações. A primeira, uma extensa análise de sensibilidade com cada um dos esquemas, valendo-se do arcabouço de estudo e

equações apresentado neste trabalho, e varrendo o espaço de parâmetros, possivelmente interagindo com os autores de cada esquema. Segundo, a implementação e avaliação experimental de cada esquema, permitindo uma avaliação mais precisa (embora os desafios e dificuldades associados às implementações sejam óbvios). A terceira e mais importante ação, um desafio proposto à comunidade de pesquisa, é a investigação de novos algoritmos de assinatura digital eficiente para *live streaming* de alta definição que proporcionem um nível de segurança aceitável.

Referências

Challal, Y., Bettahar, H., and Bouabdallah, A. (2004). A taxonomy of multicast data origin authentication: Issues and solutions. *IEEE Communications Surveys and Tutorials*, 6(1-4):34–57.

Cisco (2009). Cisco digital media systems solution overview. `http://www.cisco.com`.

Dhungel, P., Hei, X., Ross, K. W., and Saxena, N. (2007). The pollution attack in p2p live video streaming: Measurement results and defenses. In *Workshop on Peer-to-Peer Streaming and IP-TV, P2P-TV '07*, pages 323–328. ACM.

Fu, W., Jain, S., and Vicente, M. R. (2009). Global broadband quality study shows progress, highlights broadband quality gap. `http://www.cisco.com`.

Haridasan, M. and van Renesse, R. (2008). SecureStream: An intrusion-tolerant protocol for live streaming dissemination. *Computer Communications*, 31(3):563–575.

Lin, Y.-J., Shieh, S., and Lin, W. W. (2006). Lightweight, pollution-attack resistant multicast authentication scheme. In *ACM Symposium on Information, Computer and Communications Security, ASIACCS '06*, pages 148–156. ACM.

Meier, R. and Wattenhofer, R. (2008). Alps: Authenticating live peer-to-peer live streams. In *SRDS '08: Proceedings of the 2008 Symposium on Reliable Distributed Systems*, pages 45–52. IEEE Computer Society.

Miner, S. and Staddon, J. (2001). Graph-based authentication of digital streams. In *SP '01: Proceedings of the 2001 IEEE Symposium on Security and Privacy*, page 232. IEEE Computer Society.

Park, J. M., Chong, E. K. P., and Siegel, H. J. (2003). Efficient multicast stream authentication using erasure codes. *ACM Transactions on Information and System Security*, 6(2):258–285.

Perrig, A. (2001). The biba one-time signature and broadcast authentication protocol. In *CCS '01: Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 28–37. ACM.

Poynton, C. (2003). *Digital Video and HDTV Algorithms and Interfaces*. Morgan Kaufmann Publishers Inc.

Topiwala, P., Raju, A., and Singh, D. (2009). A real-time H.264 high 4:4:4 codec. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conf. Series*, vol. 7443.

Wong, C. K. and Lam, S. S. (1999). Digital signatures for flows and multicasts. *IEEE/ACM Transactions on Networking*, 7(4):502–513.

Yang, S., Jin, H., Li, B., Liao, X., Yao, H., and Tu, X. (2008). The content pollution in peer-to-peer live streaming systems: Analysis and implications. In *International Conference on Parallel Processing*, pages 652–659. IEEE Computer Society.