Challenging the Feasibility of Authentication Mechanisms for P2P Live Streaming

Rafael V. Coelho, Jonata T. Pastro, Rodolfo S. Antunes
Marinho P. Barcellos, Ingrid Jansch-Porto, Luciano P. Gaspary
Federal University of Rio Grande do Sul – UFRGS
Porto Alegre, RS – Brazil
{rvcoelho,jtpastro,rsantunes,marinho,ingrid,paschoal}@inf.ufrgs.br

ABSTRACT

Peer-to-peer live streaming applications are becoming increasingly popular, and already reach millions of users worldwide. There are, however, many aspects in which these applications must evolve, among which video quality and security deserve particular attention. While the former is about to become a reality given recent advances in codecs and link bandwidth, the latter still presents great challenges. The decentralized nature of P2P live streaming systems makes them vulnerable to attacks, among which pollution is arguably the most harmful. Multimedia stream authentication allows the detection of polluted content and can help identifying malicious users, but its costs may be prohibitive. This paper presents an analytical evaluation of recent authentication mechanisms in light of their application in P2P live streaming. Its key contribution is to compare, unlike previous work, overheads and security of various signature amortization and light digital signature mechanisms for P2P live streaming of standard and high definition video. We demonstrate, contrary to previous work, that current authentication mechanisms do not scale in therms of quality, and are feasible in current hardware only with a favorable set of parameters. Our results justify the investigation of more efficient and adaptable approaches to the next generation of P2P streaming with secure, high definition video.

Categories and Subject Descriptors

C.2 [**General**]: Security and Protection

General Terms

Algorithms, Security, Experimentation

Keywords

Streaming, Security, Evaluation

1. INTRODUCTION

In peer-to-peer live streaming, a source peer distributes over the Internet one or more live video streams to a large population of users (receiving peers), who help with the dissemination process. Studies from Cisco [4] show a constant growth in the popularity of these applications, with several millions of users worldwide already. Current technology considers only standard quality content, but streaming applications that operate in high definition (HD) are expected to dominate the market in the next five years [6].

The popularity of P2P live streaming also increases the motivation for attacks such as *content pollution* [5, 30]. This kind of threat may be defined as any modification to the media (audio/video) during its hop-by-hop forwarding from the source to destination peers. This might be done with the intent of (while avoiding detection) deceiving users watching the transmission, or performing a Denial-of-Service attack. The creation of corresponding defense mechanisms is a major research challenge. Although the prevention against content pollution has seen important advances in the context of file sharing [1, 27], unfortunately the same solutions cannot be employed in live streaming. This is so because of the typical characteristics of live streaming applications, e.g. temporal restrictions, and on-the-fly generation and exhibition of contents; therefore the defense mechanisms must be feasible and efficient for this dynamic data context.

A pollution attack can be devastating to P2P live streaming applications [5]: assuming the content to be organized in *chunks*, these spread rapidly through the network, regardless whether they are fake or legitim. To avoid or reduce the impact of a pollution attack, contaminated chunks need to be detected before they spread. The classic strategy for data authentication is digital signatures. In theory, the source can sign every chunk with a private key and allow receivers to verify chunk authenticity with the corresponding public key, obtained externally from a trusted source. Hence, receiving peers can discard contaminated chunks sent by malicious peers. If fake chunks can be immediately discovered, then it becomes possible to identify (and isolate) malicious peers.

This paper presents an analytical evaluation of recent authentication mechanisms in light of their application in P2P live streaming. Its key contribution is to compare, unlike previous work, overheads and security of both signature amortization and light digital signature mechanisms not only for standard quality, but also for high definition content.We hypothesize that state-of-the-art authentication mechanisms are unable to provide reasonable security guarantees considering existing off-the-shelf hardware. So, in our analytical study we compare these mechanisms and establish their se-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
LANC 2011 October 12-13, 2011 Quito, Ecuador
Copyright © 2011 ACM ISBN 978-1-4503-0886-1/11/10 ...$10.00.

curity levels and at which computational and communication costs.

The remainder of this paper is organized as follows. Section 2 presents other studies related to pollution and authentication mechanisms for live streaming systems. Section 3 presents a detailed overview of the main authentication strategies that are analyzed in our study. Section 4 presents the developed analytical model, which is the basis of the comprehensive analysis described in Section 5. Finally, in Section 6, we draw some final remarks about the study and directions of future work.

2. BACKGROUND & RELATED WORK

P2P live streaming systems distribute content that is generated and encoded *on-the-fly*, creating a sequence of chunks. To ensure that the quality of experience of the application is not degraded, chunks must be obtained and processed by receiving peers regularly, at the transmission rate. This way every chunk is available to the player before its playback deadline. A general review on P2P streaming systems, explaining about topology types and chunk distribution models, can be found in [17] and [20].

Pollution is a well-known threat to content distribution peer-to-peer systems in general. With the growing popularity of streaming applications, recent studies are focused on understanding the dynamics and impact of pollution in P2P live streaming systems and possible countermeasures for their protection. Dhungel et al. [5] study content pollution in P2P live streaming systems. Experiments conducted with PPLive [12], demonstrate that a pollution attack can reduce the user population to 14% of the original number observed during normal operation. The study also analyzed four different countermeasures to protect P2P live streaming against pollution, namely: blacklisting, traffic encryption, hash verification, and chunk signing. The authors' analysis indicates that chunk signing offers the best protection against pollution.

A formal analysis about content pollution and its implications to P2P streaming is presented in [30]. Authors devise a probabilistic model to capture the progress of polluted content dissemination, based on data collected using AnySee [14]. They observe an exponential growth in contaminated peers, and conclude that even 1% of polluters is enough to compromise a network in a few minutes. Their results indicate that unidirectional hash functions are the most efficient strategy against polluters, especially when employed early in the streaming session.

Authentication schemes allow receivers to spot contaminated chunks and discard them. Comparisons among mechanisms have been limited in nature. For example, authors of [21] compare ALPS with light digital signature schemes like BiBa [25] and traditional signature schemes like RSA, but not with more recent proposals such as PARM2 [15]. Our study presents analytical comparisons among the most recent and relevant proposals of authentication mechanisms applicable to P2P live streaming systems.

A comparative study of *multicast* authentication mechanisms is presented in [9]. Analytic formulas and simulation results are used to evaluate metrics of computational cost, communication overhead, tolerance to packet losses, receiver buffer size, and delay imposed by the authentication process. That study, however, is focused on IP multicast and only evaluates authentication mechanisms based on signature amortization. A contribution of our work is to extend the comparison to include the three most relevant light digital signature mechanisms: ALPS [21], PARM [16] and PARM2 [15].

Finally, existing studies on P2P streaming authentication have dealt with the distribution of standard quality content, that is, videos with resolutions of up to 640x360, over high definition television (HDTV), characterized by videos of up to 1920x1080 (and 5.1 audio) [26]. Typically H.264 codec is used to generate high definition content under a high profile configuration. Note that the recent standardization of SVC (the scalable extension of H.264/AVC) allows three types of scalability and the transmission of streams with layering to support receivers with different capacities. The standard H.264, SVC and other encoding issues are outside the scope of our analysis, because encoding is transparent for P2P live streaming systems, which only deals with sequential chunks. Regardless of the encoding employed, in high definition scenarios, the volume of data to be streamed will be much larger. This is likely to increase the cost of using an authentication mechanism to protect a stream, making it potentially unaffordable.

Our study focus on the overheads generated by authentication systems used to guarantee that no chunk tampering will go unnoticed. There are mechanisms other than chunk authentication to allow polluted content to be identified. Studies such as [13] propose the use of probabilistic analysis to allow peers to identify potential polluters and remove them from their neighbor set. Another example is MIS [28], in which receivers report evidences to a *stream manager*, allowing it to take action against potentially malicious peers. An authentication mechanism is still required to allow a peer to identify polluted chunks, hence these and other similar initiatives are complementary to our work.

3. AUTHENTICATION STRATEGIES

A naive strategy to implement authentication in streaming systems consists in using traditional cryptographic algorithms, such as RSA. They provide secure authentication based on mathematical complexity. Each chunk is signed by the source before transmission, and the receiver can verify its authenticity by applying the same algorithm over the received data and checking the signatures. The high processing overhead imposed by signing and verifying each chunk with such algorithms, however, prevents their direct application in live streaming due to temporal restrictions.

3.1 Signature Amortization

One solution to lessen the overhead generated by signature algorithms is the employment of amortization mechanisms. These mechanisms rely on the fact that is cheaper to sign a large data block than to individually sign smaller blocks. This way, the signature cost can be diluted: the sum of the costs of signing a block of chunks is smaller than signing individual chunks. Hash chaining [7] is one of the early and simplest mechanisms of signature amortization. The chunk carries the hash of its successor in the stream. The hash of the first chunk of a block is sent in a separated message, which is signed by the source to guarantee the authenticity of the chain. The mechanism has a major drawback: a single chunk loss may compromise the authentication of several chunks in the block.

An option to tolerate packet losses on amortization strate-

gies is to employ a directed acyclic graph (DAG) to control dispersion of authentication information among chunks. In this case, a graph is generated with chunks as vertices, and different algorithms may be used to generate edges that will dictate which chunks should receive authentication information of others. Zhang et al. [31] propose the application of butterfly graphs in such setup. Chunks of a block are organized in columns, and each chunk is linked to two others from the previous column in the structure. Chunks from the first column are all linked to a single signature packet, which will be distributed separately from chunks.

Another strategy to amortize signature costs is the use of Merkle hash trees, as suggested by Wong and Lam [29]. In this scheme, a Merkle hash tree is built from the hashes of each chunk in the block, and its root is signed to guarantee authenticity of the information. Each chunk, then, is augmented with the necessary information to allow its individual verification by the client, including hashes of intermediary levels of the tree in the path between the chunk and the root. This method allows individual verification of chunks and is not affected by packet losses, but its shortcoming is the communication overhead generated by all the information that must be added to each chunk.

To mitigate the impact of packet loss, Park et al. [23] propose the use of an information dispersal algorithm (IDA) to introduce redundancy in the authentication data distributed with chunks. This mechanism, named SAIDA, generates a concatenation of the block hashes, which is signed by the stream source to guarantee its authenticity. The hash concatenation and the signature compose the authentication that is processed by IDA, generating a number of redundant information pieces equal to the number of chunks in the block. Each piece is then added to one of the chunks before they are transmitted. Receivers do not need to receive all the chunks of the block in order to reconstruct the authentication information. Instead, a fraction of the block is enough. eSAIDA [24] is an optimization to this basic algorithm, where hashes are applied to pairs of chunks in order to reduce authentication overhead. Lysyanskaya et al. [19] present an authentication strategy that is very similar to SAIDA, but employs Reed-Solomon codes instead of IDA. This choice allows the system to better tolerate packet losses in exchange of a higher authentication overhead. An optimization to SAIDA, called cSAIDA, was later proposed by Pannetrat and Molva [22], taking advantage of IDA with systematic coding. That is, when the IDA is applied over the concatenation of hashes a parity set will be added and allow the original data to be reconstructed even on the occurrence of losses.

Signature amortization is also employed by the SecureStream P2P streaming system [8]. Its mechanism, named Linear Digests, generates a separate authentication message for each block. This message contains the hashes of the block of chunks and a signature to guarantee its authenticity. This message is then distributed to receivers prior to transmission of the block, so they can individually verify the authenticity of chunks upon receipt. The security mechanisms embedded in SecureStream guarantee that the authentication message will eventually reach all receivers, so the authentication mechanism itself does not consider packet losses.

3.2 Light Signature

An entirely different approach to reduce the overhead of the authentication process are light signature mechanisms. They employ unidirectional hash functions for the generation and verification of signatures, reducing the computational cost in comparision with traditional signatures. Mechanisms of this class are based on an *evidence matrix*, which is generated by successive application of a hash function over a set of randomly generated numbers. These random numbers make up the first row of the matrix, each in one number per column. The remaining elements of the matrix are generated column-wise, by successively applying the hash function to the first element of each column, the random number, until the element at the bottom is calculated. The set of elements contained in the bottom row of the matrix is taken as the public key for this evidence matrix. To sign a chunk, the source applies a hash function to it, and uses the output to select from which positions of the matrix evidences will be taken. The chunk is then augmented with these evidences before transmission. Receivers can verify evidences by applying the same hash functions used by the source. That way, a receiver identifies from which position of the matrix an evidence was extracted and its validity, which is verified by consecutively applying the hash function until an element of the public key is obtained. The three mechanisms that adopt this approach, namely PARM, PARM2 and ALPS, are described next.

In PARM [16] the chunk data is used as input of a hash function to select the matrix columns from which evidences will be taken for the chunk. Evidences are "consumed" bottom-up from the public key (exclusive) until the first line of the matrix. Counters are used for each column to control the position of the unused evidences. When a certain number of columns have reached the top row, the matrix needs to be renewed. When so, the existing matrix is discarded and a new one, generated (as previously described). The resulting public key (bottom row) is then disseminated to all receivers. The chunk authenticity is verified at a receiver by first hashing the data in the chunk to determine a set of columns; then, successively applying the hash function over each evidence in the chunk until the corresponding part of the public key matches. If any of the evidences do not lead to a match, the chunk is discarded. One serious limitation with PARM is that it requires the ordered delivery of chunks to receivers, which may not be feasible if a mesh topology is employed.

PARM2 [15] is an extended version of PARM that solves the aforementioned limitation, and improves its performance and resilience to attacks. Chunks are uniquely identified by a security counter field in order to avoid replay attacks. Chunks carry usage counters for each column of the evidence matrix so that receivers do not need to keep track of the information. Finally, PARM2 allows partial renovation of the evidence matrix, where only the exhausted columns are replaced instead of the entire matrix.

ALPS [21] employs an evidence matrix but vertically divides it in a set of sub-matrices, each one with equal number of rows. To authenticate a chunk, the source applies a hash function over the chunk data. The output of the function then indicates from which positions of the sub-matrix evidences should be taken. When consumed, an evidence is marked, and the next time this position happens to be selected, the evidence is taken from the same position but in

the next sub-matrix. Evidences are consumed from sub-matrices in a bottom-up order, that is, evidences are taken first from the sub-matrix that is closest to the public key (which is the bottom row of the matrix), and when already used elements are selected, they are taken from the first sub-matrix above the one with the last used element of that position. A chunk can receive evidences from multiple sub-matrices, depending on the consumption of elements. Evidence verification follows the same principle of PARM, however buffers are used to keep for each column the most recently verified value, so that the hash chaining process does not need to reach the bottom row of the matrix to check an evidence. ALPS also requires renewal of the evidence matrix when a certain number of chains have their evidences completely consumed.

To better understand the behavior of the described authentication strategies we define an analytical model. It is used to evaluate the communication and processing overheads of state-of-the-art stream authentication mechanisms, while considering scenarios of standard and high definition content. To fairly evaluate and compare the different strategies, we define a set of parameters with as many common values as possible. We focus our efforts in the most recent amortization and light signature mechanisms: Linear Digests (henceforth referred as LDigests), Butterfly Hash Chaining (Butterfly), cSAIDA, PARM, PARM2, and ALPS.

4. ANALYTICAL MODEL

This section presents the set of equations used to evaluate various performance aspects of authentication mechanisms for live streaming systems. Some of the equations, specifically those related to cSAIDA and Butterfly signature amortization mechanisms, have been adapted from [9].

4.1 Computational Cost

Both source and receivers must be powerful enough to timely run the authentication algorithms, respecting the rate in which encoded data is generated. The computational cost of chunk verification may impose certain demands on the hardware, affecting for example mobile devices and reducing scalability. We assume that the source will be powerful enough and concentrate our analysis on the receiver side.

In signature amortization strategies, the computational cost is directly related to the cost of the algorithms used for signing and hashing. All mechanisms need to verify a digital signature, which takes t_{sign}, and perform hash operations over the block chunks, with t_{hash} per operation. Considering n_{chunk} chunks of size s_{chunk}, LDigests and Butterfly computational cost is given by Equation 1. cSAIDA must perform the same set of operations, also including the processing time for FEC-decoding. Given the efficiency of existing related algorithms such as Tornado codes [18], we assume the costs of FEC decoding to be negligible in this case. Besides verifying the hash of individual chunks, cSAIDA also verifies the hash generated by the concatenation of their hashes. Thus, we add s_{hash} (the hash size) to the data that needs to be hashed, leading to Equation 2.

$$CC_{LDigests} = CC_{Butterfly} = \frac{t_{sign}}{n_{chunk}} + t_{hash} \times s_{chunk} \quad (1)$$

$$CC_{cSAIDA} = \frac{t_{sign}}{n_{chunk}} + t_{hash} \times (s_{chunk} + s_{hash}) \quad (2)$$

Light signature mechanisms use an evidence matrix of m_{rows} lines and m_{cols} columns, from which n_{evid} different evidences are taken to be piggybacked with each distributed chunk. The last row of the matrix is distributed as a public key, so clients can verify chunk authenticity by successively applying the hash function over evidences, until an element of the public key is reached. In the specific case of PARM and PARM2, evidences are selected from different columns of the matrix (the selection of columns is based on the content) and within each column, the first unused element is selected. This process is repeated until there is no unused evidences in any column and the evidence matrix must be renewed. In equations that follow, our analysis adopts the worst-case scenario when verifying a chunk, in order to establish a safe bound on minimum hardware capacity. Otherwise, there might be periods of transmission during which many chunks might not be verifiable in time. Hence, Equation 3 reflects, for PARM and PARM2, the need to successively apply the hash to all elements of a column in order to verify an evidence. As mentioned earlier, ALPS employs the same concept of evidence matrix, but vertically partitions it in m_{sub} sub-matrices, each containing an equal number of $m_{subrows}$ rows. Evidences are selected from positions in the sub-matrix, and taken from the first sub-matrix that presents an unused evidence in that position. The verification process is similar to PARM, but in ALPS clients also store the last successfully verified evidence for each column of the matrix. The verification can then be concluded when a previously verified evidence is found, speeding up the process. This results in the cost represented by Equation 4.

$$CC_{PARM} = CC_{PARM2} = n_{evid} \times m_{rows} \times t_{hash} \quad (3)$$

$$CC_{ALPS} = 2 \times m_{subrows} \times n_{evid} \times t_{hash} \quad (4)$$

4.2 Communication Overhead

The communication overhead imposed by authentication strategies may be another limiting factor to its adoption in P2P live streaming, in particular if mobile devices are considered. This overhead represents the average number of bytes that is added to each chunk because of authentication.

The communication overhead of LDigests is inversely proportional to the number of chunks (n_{chunk}) contained in a block, and is also proportional to the size s_{sign} of the signature information to guarantee authenticity of the hashes. The Butterfly mechanism organizes chunks in columns, each containing g_{lines} rows. A chunk on an arbitrary column will be linked with two chunks from the previous column, and these chunks will carry authentication information for the first. Besides this, a separate authentication message is also created, containing signature information for all chunks in the first column of the graph. The resulting overhead of this authentication mechanism is modeled by Equation 6. cSAIDA communication overhead, on its turn, is the result of the extra information added by the FEC, which is assumed to be based on systematic coding. That is, when the FEC is applied over the concatenation of the hashes of the block's chunks, the result will be a redundant data set where the original hash concatenation is equal to the beginning of the data, and the final part is a parity set that allows the original data to be reconstructed even on the occurrence of loss. The advantage of this scheme is that only the parity data must be sent together with chunks, since the

original hashes can be obtained with the packets itself. The data coded with the FEC can be reconstructed if n_{minver} packets are received, and the first n_{minver} packets of the code are the ones equal to the original hash concatenation on cSAIDA. Adding the signature of the chunk's hashes concatenation, the resulting overhead can be modeled according to Equation 7.

$$CO_{LDigests} = s_{hash} + \frac{s_{sign}}{n_{chunk}} \quad (5)$$

$$CO_{Bfly} = \frac{n_{dupsign} \times (s_{sign} + g_{lines} \times s_{hash})}{n_{chunk}} + \frac{s_{hash} \times (2 \times n_{chunk} - g_{lines})}{n_{chunk}} \quad (6)$$

$$CO_{cSAIDA} = \frac{s_{sign} + (n_{chunk} - n_{minver}) \times s_{hash}}{n_{minver}} \quad (7)$$

The communication overhead of light signature mechanisms is mainly determined by the number n_{evid} and size s_{evid} of the evidences sent with each chunk. As PARM adds only evidences to the chunks, its overhead can be modeled as in Equation 8. Besides evidences, ALPS also includes a security counter of size $s_{counter}$ to uniquely identify each generated chunk, and so its overhead can be modeled as in Equation 10. PARM2, besides evidences and the security counter, also sends an usage table informing receivers of how many evidences of each column have already been used. The usage table contain one element for each column of the evidence matrix, resulting in m_{cols} elements, each of which will be of size $\lg m_{rows}$, resulting in the Equation 9.

$$CO_{PARM} = n_{evid} \times s_{evid} \quad (8)$$

$$CO_{PARM2} = n_{evid} \times s_{evid} + m_{cols} \times \lg m_{rows} + s_{counter} \quad (9)$$

$$CO_{ALPS} = n_{evid} \times s_{evid} + s_{counter} \quad (10)$$

4.3 Security level

The security level is determined, in the context of light signature mechanisms, by the inverse of the probability that a malicious peer will be able to create a valid signature based on evidences previously revealed (that is, security level will continuously decrease as chunks are sent, until the evidence matrix is renewed). In contrast, when amortization mechanisms are used, the security level is given by the signature algorithm employed (e.g., RSA). Hence, the amortization mechanism itself will be neutral in regards to the level of security.

In the three light signature mechanisms, the level of security is inversely proportional to the amount of undisclosed evidences. Then, for a given time, the security level is proportional to the number of m_{rows} rows and m_{cols} columns adopted in the evidence matrix. The amount of possible evidences, i.e., the number of elements in the matrix, corresponds to the amount of choices the source has upon selecting n_{evid} evidences to create a signature. However, revealed evidences ($n_{revealed}$) increase the chance that an attacker may create a valid signature. The security level for PARM and PARM2 mechanisms can be presented through with Equation 11. ALPS, on its turn, presents a different security level due to the division of the evidence matrix in m_{sub} sub-matrices, each one with $m_{subrows}$ rows, leading to the Equation 12.

$$S_{PARM} = S_{PARM2} = \left(\frac{m_{rows} \times m_{cols}}{n_{revealed}} \right)^{n_{evid}} \quad (11)$$

$$S_{ALPS} = \left(\frac{m_{subrows} \times m_{sub} \times m_{cols}}{n_{revealed}} \right)^{n_{evid}} \quad (12)$$

The above equations are the basis for an analytical evaluation of the performance of each of the state-of-the-art authentication strategies. The next section presents this evaluation and the main results obtained.

5. EVALUATION

The goal of this section is to evaluate the performance of authentication mechanisms under different scenarios. We first describe general parameters employed in our evaluation, and then present a complete sensibility analysis of the performance and security light signature mechanisms. We then present results for amortization mechanisms, and finally some insights about the comparison of strategies.

5.1 Simulation Parameters

With respect to video quality, two main options were chosen for the evaluation: standard definition (SD), with a resolution 240 vertical lines, and high definition (HD), with 1080 vertical lines. For HD scenarios, the required network bandwidth varies from 5 Mbps up to 16 Mbps, according to experimental evidences reported by manufacturers of HDTV equipment [3]. In respect to chunk size, according to [5, 10, 11] P2P live streaming systems employ chunks between 4,774 and 7,220 bytes long; we employ chunks of 6 KBytes. Table 5.1 presents a summary of the main parameters for the two scenarios used in our evaluation, according to the values presented so far.

Table 1: Scenarios for signature mechanisms

Scenario	SD	HD
Resolution	320x240	1920x1080
Bandwidth (Kbps)	384	16384
Rate (chunks/s)	10.5	43.5

In order to evaluate the computational cost of the mechanisms, it is necessary to define the execution times for hashing and signing operations. For this evaluation, we run these algorithms on an Intel Core2 computer with a clock of 2.0 GHz and 6 GB of memory. SHA-1 is employed as the hashing algorithm, given its efficiency, the security level it provides and the availability of implementations. It generates a digest of 20 bytes and its average execution time was 5.07 µs in the specified hardware. For signing operations, we choose a 2048 bits implementation of the RSA public key algorithm, which took approximately 303 µs to verify a signature.

The selection of the key size is based on a recommendation for key management by NIST [2], according to which a key size of 112 bits is required in symmetric key-based security systems. This configuration provides a security level of 2^{112} or 5.2×10^{33}. For an asymmetric key system, such security level is achieved by a key size of 2048 bits. This value is

used in light signature mechanisms as a threshold for the minimum security: an instance of a mechanism is deemed secure if its security level does not fall below this level.

With respect to signature amortization mechanisms, we take advantage of the sensitivity analysis performed by Hefeeda and Mokhtarian [9]. They evaluated parameters such as graph construction for Butterfly and IDA parameters for cSAIDA. Such parameters are, for the Butterfly scheme, the number of duplicated signatures ($n_{dupsign}$), which is set to 2, and the number of lines in the graph (g_{lines}), set to 32. For cSAIDA, there is only one parameter, n_{minver}, which is set to 64 (half of the size of the block).

In the case of light signature mechanisms, to better comprehend their behavior, a more detailed sensitivity analysis of possible parameters was necessary. The choice of intervals for parameters matched at least one restriction of computational cost, communication overhead or security level. In the case of the number of rows of the evidence matrix (m_{rows}) and the number of ALPS sub-matrices (m_{sub}), an extremely high value will result in an unfeasible computational overhead, while an extremely low value, in an insecure system. We choose to keep m_{rows} values between 256 and 16384 and m_{sub} between 128 and 16384. The number of evidences (n_{evid}) has a similar behavior: an extremely high value will lead to an unfeasible communication overhead, while an extremely low will lead to an insecure system. In our evaluation, n_{evid} ranges from 8 to 20. The number of matrix columns (m_{cols}) and the number of rows in each ALPS submatrix ($m_{subrows}$) require a different analysis: recall that evidences are taken from different columns (and lines of submatrices) based on the result of a hash function. The size of the hash digest (s_{hash}) is divided by the number of evidences (n_{evid}), and each resulting piece will be used as the address to select positions for columns and sub-matrices. Values of m_{cols} and $m_{subrows}$ are thus calculated for each evaluated combination of parameters, according to the following equation that considers the size of the hash digest and the number of evidences:

$$2^{\left(\frac{s_{hash}}{n_{evid}}\right)}$$

The above values for parameter intervals are employed in the next subsections to evaluate performance and security of authentication mechanisms.

5.2 Computational Cost

To determine overhead of light signature mechanisms, an evaluation was conducted based on equations presented in Section 4.1. This analysis is based on a metric that considers the computational cost to authenticate chunks and the rate at which they are transmitted. The product of these two values, defined as β, represents the time fraction of the streaming session spent in authentication procedures. A value of β close to or greater than 1 indicates that the authentication cost is too high for the chunk rate, in which case the stream cannot be verified at the required speed for playback.

We first employ the equations from light signature mechanisms to analyze their behavior considering the parameter ranges defined in Section 5.1. Tables 2 (PARM and PARM2) and 3 (ALPS) summarize the results. These tables present, in the column β SD, the computational cost for the scenario with standard definition content, and in the column β HD, the cost for high definition content.

Table 2: β analysis for PARM and PARM2

Config	n_{evid}	m_{row}	m_{cols}	β SD	β HD
1	8	256	1048576	0.11	0.46
2	8	1024	1048576	0.46	1.83
3	8	4096	1048576	1.83	7.32
4	8	16384	1048576	7.32	29.26
5	10	256	65536	0.14	0.57
6	10	1024	65536	0.57	2.29
7	10	4096	65536	2.29	9.14
8	10	16384	65536	9.14	36.58
9	16	256	1024	0.23	0.91
10	16	1024	1024	0.91	3.66
11	16	4096	1024	3.66	14.63
12	16	16384	1024	14.63	58.53
13	20	256	256	0.29	1.14
14	20	1024	256	1.14	4.57
15	20	4096	256	4.57	18.29
16	20	16384	256	18.29	73.16

Table 3: β analysis for ALPS

Config	n_{evid}	m_{cols}	$m_{subrows}$	β SD	β HD
1	8	1048576	1048576	936.43	3745.72
2	10	65536	65536	73.16	292.63
3	16	1024	1024	1.83	7.32
4	20	256	256	0.57	2.29

Results for PARM and PARM2 show that the number of rows in the evidence matrix has a high impact on their computational overhead. In all cases with a high number of rows the value of β is well above 1. Increases in the computational cost are observed when other parameters are varied, but they do not substantially affect β. In the case of ALPS, there is a high influence from the number of rows in each sub-matrix, with the value of β rapidly reaching 1 as that variable is increased. Other parameters cause minimal impact.

For amortization mechanisms, both computational and communication overheads will be inversely proportional to the block size. This size is limited, according to [9], by restrictions such as authentication delay and chunk loss tolerance. To avoid cases in which such restrictions harm the quality of the stream, those authors argue that a block should contain 128 chunks, and thus we also adopt this value in our analysis of amortization schemes.

The remaining equations of Section 4 were employed to evaluate the performance of these mechanisms. The results indicated that all three amortization mechanisms impose equivalent computational overheads, which was 0.33 and 1.34 for β SD and β HD, respectively. This means that none of the evaluated mechanisms was able to sustain the authentication process with high definition content.

While none of the amortization mechanisms was efficient enough for the HD scenario, light signature mechanisms were capable of supporting such transmissions depending on the configuration.

5.3 Communication Overhead

While the computational cost plays an important role in the selection of an authentication mechanism for high definition live streaming, the imposed communication overhead should also be considered.

Light signature mechanisms were evaluated under different sets of parameters. The obtained results for all mechanisms are shown in Table 4, in which the communication overhead is presented as a fraction of the chunk size.

Table 4: Communication overhead of light signature mechanisms (relative to chunk size)

n_{evid}	m_{rows}	CO_{PARM}	CO_{PARM2}	CO_{ALPS}
8	256	2.70%	3.50%	2.77%
8	1024	2.68%	3.67%	2.75%
8	4096	2.67%	3.87%	2.73%
8	16384	2.67%	4.03%	2.73%
10	256	3.37%	4.35%	3.43%
10	1024	3.35%	4.57%	3.42%
10	4096	3.33%	4.78%	3.40%
10	16384	3.33%	5.02%	3.40%
16	256	5.38%	6.92%	5.45%
16	1024	5.35%	7.27%	5.42%
16	4096	5.33%	7.62%	5.40%
16	16384	5.33%	7.98%	5.40%
20	256	6.72%	8.63%	6.78%
20	1024	6.68%	9.05%	6.75%
20	4096	6.67%	9.52%	6.73%
20	16384	6.67%	9.97%	6.73%

First notice that all three light signature mechanisms produce similar communication overheads, between X and Y. Note also that the number of evidences employed is the most influential parameter to this metric. The values for all mechanisms present a linear increase according to the number of evidences of each chunk, with PARM2 having a noticeably higher cost per chunk when compared with PARM and ALPS.

With respect to amortization mechanisms, results indicate that the Butterfly and cSAIDA present similar results. Both mechanisms present an almost negligible overhead of 0.78%. LDigests produced a even lower overhead than the previous schemes: 0.35% of the chunk size.

Comparing results from amortization and light signature mechanisms, it is possible to see that amortization ones present a clear advantage in terms of communication cost, since they present an almost negligible overhead. Light signature mechanisms, on their turn, present considerable higher communication overhead, which is greatly by the size and number of evidences that are used to augment chunks.

5.4 Security Analysis

The security level of a signature amortization mechanism is proportional to that of the digital signature algorithm employed, and due to that it remains constant over time. Since the signature is applied to an entire block of chunks, the amortization process incurs a small reduction in the security. This is due to the fact that the necessary effort to break the signature must be employed for the entire block, instead of the individual chunks. This reduction, however, is negligible in practice.

According to Equations 11 and 12 of Section 4.3, there are two parameters that directly impact the security level of light signature mechanisms: the size of the evidence matrix and the number of evidences revealed with each chunk. As evidences are used, the security level gradually decreases because the attacker has access to an increasing number of matrix elements, reducing the complexity of orchestrating an attack against the authentication mechanism.

Because of the gradual decrease of the security level, the evidence matrix needs to be periodically renewed. This procedure requires the transmission and authentication of a new public key. This leads to a tradeoff between the induced communication and computational costs and maintaining the security level above a safe threshold. In the evaluation, we define that the renewal procedure should happen before one of the hash chains (one column of the evidence matrix) reaches its first row, thus avoiding the insecure case where a signature is composed of evidences from exhausted chains.

Based on previous sensibility analyses, we select parameter configurations that present acceptable computational costs in both SD and HD scenarios (that is, their β result is below 1). Results from Tables 2 and 3 show that PARM and PARM2 are the only systems that satisfy our restriction, specifically configurations of numbers 1, 5 and 9. We evaluate the security level of these configurations according to the number of authenticated chunks over time. We illustrate the obtained results in Figure 1, in which the horizontal axis represents the cumulative number of authenticated chunks after a key renewal (that is, no evidences were previously consumed) and the vertical axis, the security level of the mechanism. Our evaluation also considers the minimum security level of 2^{112} as presented in Section 5.1.

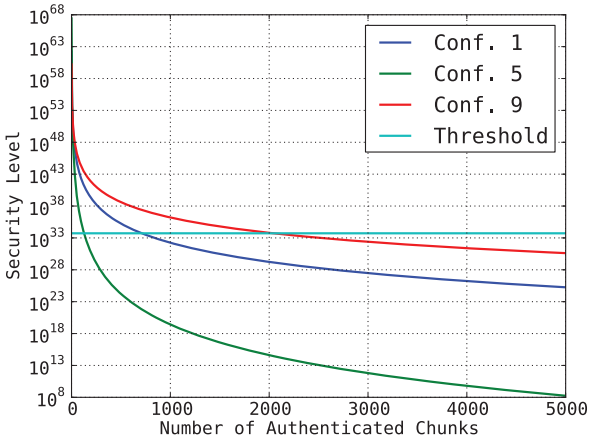

Figure 1: Security level of PARM and PARM2

Results in Figure 1 results show that none of the configurations is able to sustain a satisfactory security level between two matrix renewals, with the best result falling below the threshold after 2000 chunks are authenticated. Considering the the SD and HD scenarios presented in Table 5.1, the best result, which is from Configuration 9, will allow the transmission of 181 seconds of SD content before falling below the security threshold. In the case of HD content, this value falls to 45 seconds of transmitted content. Considering the HD result, it would be necessary to reduce the threshold for key renewal, consequently generating considerable overhead due to key renovations for each 40 seconds of content. Reflecting a trade-off between the security level and the resulting performance of the content playback, the transmission of HD content is only feasible if the security is compromised.

For amortization schemes, recall from Section 5.1 that in

our analysis we use parameters for the employed signature scheme (RSA) that will always maintain the security level above the defined threshold. As also previously presented, however, such configurations are not able to maintain an adequate level of computational overhead in high definition scenarios. In comparison, light signature schemes present the drawback of the variability in their security level, but with a favorable set of parameters will allow the reproduction of high definition content.

5.5 Comparison

With respect to the video quality, our results indicate that both amortization and light signature strategies are able to authenticate content of standard definition. However, we found that high definition content can only be authenticated by light signature mechanisms, since amortization strategies incur unaffordable computational cost. It should be noted, however, that a favorable set of parameters is required by light signature mechanisms to achieve the necessary performance.

In regard to security, the strategies are quite different. Light signature mechanisms present a varying security level over time. Our analysis showed that configurations of these mechanisms that support high definition rapidly fall below a safe security level due to the restricted size of the evidence matrix. In contrast, amortization mechanisms present a constant level of security, which depends directly on parameters of the employed hash and signing algorithms. Necessary parameters to keep their security level above the specified threshold, however, do not allow amortization mechanisms to be employed with high definition content.

State-of-the-art amortization and light signature mechanisms induce small communication overheads. Among them, amortization mechanisms are the ones with smallest overhead since they only require the distribution of small pieces of signature information with chunks. Light signature mechanisms, on their turn, induce higher overheads due to the required space for evidences.

6. FINAL REMARKS

Nowadays, P2P live streaming systems are widely available and extensively used by millions of users. Security issues of these systems, however, are only recently being explored by the research community, and thus many challenges remain open. One of these issues is the development of efficient authentication mechanisms to mitigate pollution attacks on P2P live streaming without compromising the quality of experience of the distributed content.

In this paper we presented an analytical model to evaluate performance aspects of authentication mechanisms such as communication and computational overheads, and the security level those mechanisms provide. This model served as basis for a comparison among proposals of authentication mechanisms based on signature amortization and light digital signatures strategies.

Our results show that both amortization and light signature mechanisms support the secure dissemination of standard definition content as long as there is a careful selection of configuration parameters. When high definition content is transmitted, amortization mechanisms provide appropriate security levels but at unaffordable overheads. Fortunately, there exist light signature mechanisms that can, by lowering the security level provided, authenticate HD content at the necessary rates. However, under these conditions, a pollution attack will likely be successful.

We plan to extend the analysis into a full experimental evaluation and based on obtained results present a solution that overcomes current authentication mechanisms limitations. We intend to contribute to the feasibility of P2P live streaming transmissions with high security levels, considering contents of both standard and high definition.

References

[1] M. P. Barcellos, L. P. Gaspary, W. L. da Costa Cordeiro, and R. S. Antunes. A conservative strategy to protect P2P file sharing systems from pollution attacks. *Concurrency and Computation: Practice and Experience*, 23(1):117–141, 2011.

[2] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid. Draft: Recommendation for key management: Part 1: General. Available at <http://csrc.nist.gov/publications/drafts/800-57/Draft_SP800-57-Part1-Rev3_May2011.pdf>., 2011.

[3] Cisco. Cisco digital media systems solution overview. Available at http://www.cisco.com/en/US/docs/solutions/Enterprise/Video/digmedsys.html, 2009.

[4] Cisco. Cisco visual networking index: Usage study, 2010. Available at <http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/Cisco_VNI_Usage_WP.pdf>.

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

[6] W. Fu, S. Jain, and M. R. Vicente. Global broadband quality study shows progress, highlights broadband quality gap. Available at http://www.cisco.com/web/MT/news/09/news_021009a.html, 2009.

[7] R. Gennaro and P. Rohatgi. How to sign digital streams. In *Advances in Cryptology: 17th Annual International Cryptology Conference*, volume 1294 of *Lecture Notes in Computer Science*, pages 180–197. Springer Berlin, 1997.

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

[9] M. Hefeeda and K. Mokhtarian. Authentication schemes for multimedia streams: Quantitative analysis and comparison. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 6(1): 1–24, 2010.

[10] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross. A measurement study of a large-scale P2P IPTV system. *IEEE Transactions on Multimedia*, 9(8):1672–1687, 2007.

[11] X. Hei, Y. Liu, and K. W. Ross. IPTV over P2P streaming networks: the mesh-pull approach. *IEEE Communications Magazine*, 46(2):86–92, 2008.

[12] Y. Huang, T. Z. J. Fu, D. ming Chiu, J. C. S. Lui, and C. Huang. Challenges, design and analysis of a large-scale P2P-VoD system. In *SIGCOMM '08: ACM Conference on Data Communication*, pages 375–388, 2008.

[13] Y. Li and J. C. S. Lui. Stochastic analysis of a randomized detection algorithm for pollution attack in P2P live streaming systems. *Elsevier Performance Evaluation*, 67:1273–1288, 2010.

[14] X. Liao, H. Jin, Y. Liu, L. M. Ni, and D. Deng. AnySee: Peer-to-peer live streaming. In *INFOCOM '06: IEEE International Conference on Computer Communications*, pages 1–10, 2006.

[15] W. W. Lin, S. Shieh, and J.-C. Lin. A pollution attack resistant multicast authentication scheme tolerant to packet loss. In *SSIRI '08: IEEE International Conference on Secure System Integration and Reliability Improvement*, pages 8–15, 2008.

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

[17] Y. Liu, Y. Guo, and C. Liang. A survey on peer-to-peer video streaming systems. *Springer Peer-to-Peer Networking and Applications*, 1(1):18–28, 2008.

[18] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann. Practical loss-resilient codes. In *STOC '97: ACM Annual Symposium on the Theory of Computing*, pages 150–159, 1997.

[19] A. Lysyanskaya, R. Tamassia, and N. Triandopoulos. Authenticated error-correcting codes with applications to multicast authentication. *ACM Transactions on Information and System Security*, 13(2):1–34, 2010.

[20] N. Magharei, R. Rejaie, and Y. Guo. Mesh or multiple-tree: A comparative study of live p2p streaming approaches. In *INFOCOM '07: IEEE International Conference on Computer Communications*, pages 1424–1432, 2007.

[21] R. Meier and R. Wattenhofer. ALPS: Authenticating live peer-to-peer streams. In *SRDS '08: IEEE Symposium on Reliable Distributed Systems*, pages 45–52, 2008.

[22] A. Pannetrat and R. Molva. Efficient multicast packet authentication. In *NDSS '03: ISOC Network and Distributed System Security Symposium*, pages 1–12, 2003.

[23] J. M. Park, E. K. P. Chong, and H. J. Siegel. Efficient multicast packet authentication using signature amortization. In *SP '02: IEEE Symposium on Security and Privacy*, pages 227–240, 2002.

[24] Y. Park and Y. Cho. The eSAIDA stream authentication scheme. In *ICCSA '04: Computational Science and Its Applications*, volume 3046 of *Lecture Notes in Computer Science*, pages 799–807. Springer, 2004.

[25] A. Perrig. The BiBa one-time signature and broadcast authentication protocol. In *CCS '01: ACM Conference on Computer and Communications Security*, pages 28–37, 2001.

[26] C. Poynton. *Digital Video and HDTV Algorithms and Interfaces*. Morgan Kaufmann Publishing, 2003.

[27] F. R. Santos, W. L. da Costa Cordeiro, L. P. Gaspary, and M. P. Barcellos. Choking polluters in BitTorrent file sharing communities. In *NOMS '10: IEEE Network Operations and Management Symposium*, pages 559–566, 2010.

[28] Q. Wang, L. Vu, K. Nahrstedt, and H. Khurana. MIS: Malicious nodes identification scheme in network-coding-based peer-to-peer streaming. In *INFOCOM '10: IEEE International Conference on Computer Communications*, pages 1–5, 2010.

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

[30] S. Yang, H. Jin, B. Li, X. Liao, H. Yao, and X. Tu. The content pollution in peer-to-peer live streaming systems: Analysis and implications. In *ICPP '08: IEEE International Conference on Parallel Processing*, pages 652–659, 2008.

[31] Z. Zhang, Q. Sun, and W.-C. Wong. A proposal of butterfly-graph based stream authentication over lossy networks. In *ICME '05: IEEE International Conference on Multimedia and Expo*, pages 1–4, 2005.