AMLCC : Adaptive Multi-Layer Connected Chains mechanism for multicast sender authentication of media-streaming

Hamdi Eltaief
Research Unit PRINCE
ISITCom of Hammam Sousse
Tunisia
Hamdi.Eltaief@issatso.rnu.tn

Habib Youssef
Research Unit PRINCE
ISITCom of Hammam Sousse
Tunisia
Habib.Youssef@fsm.rnu.tn

One of the main issues in securing multicast communication is the source authentication service. In this work we address the multicast stream authentication problem when the communication channel is under the control of an opponent who can drop, reorder or inject data. In such a network model, packet overhead, computing efficiency and robustness against packet loss are important parameters to be taken into account when designing a multicast source authentication mechanism. The main contribution of this paper consists of a multicast source authentication mechanism based on an adaptive hash chaining structure. Our mechanism tolerates packet loss and guarantees non-repudiation of multicast origin. It adapts the redundancy chaining degree (the amount of authentication information) depending on the actual packet loss ratio in the network. Compared to other mechanisms fitting in the same category, NS-2 simulations show that the adaptation of the redundancy degree allows to save bandwidth, allows to increase the robustness to packet loss and generate an authentication delay required by the used application.

Multicast source authentication, Non-repudiation, Packet loss resistance, Authentication probability

1. INTRODUCTION

The increase of bandwidth in today's networks encourages the deployment of group communication applications, such as the distribution of stock quotes, Pay Per View services, video-streaming, video-conferencing, TV and radio broadcasts, etc. Unfortunately, a malicious user may drop, delay, or modify intercepted communication packets or inject their own packets into the data stream. Security measures offer different benefits, such as authentication, confidentiality, integrity, and non-repudiation. Among the security requirements of several of these multicast applications is the authentication. We distinguish between two types of authentication in group communication Hardjono and Tsudik (2000), group authentication and data source authentication. Group authentication aims at assuring that the received multicast messages by group members originate from a valid group member (regardless of its identity). Data source authentication aims at assuring that the received multicast messages originate from a source having a specific identity. In order to assure group authentication, generally group members use a

shared key. This key is commonly called group key. Applying a MAC to a message with the group key assures that the message originates from a valid group member, since only valid group members are supposed to know the group key. Hence, the group authentication problem is reduced to the group key management and essentially to its scalability to large groups Rafaeli and Hutchison (2003); Hardjono and Tsudik (2000); Judge and Ammar (2003); Challal and al. (2004). In contrast, multicast data source authentication is more complicated because the group key which is known by all group members cannot be used to identify a specific sender. Many schemes have been proposed to assure data source authentication of a multicast flow with non-repudiation of the origin relying on signature amortization scheme, which uses hash-chaining techniques. Among the first and most widely used such schemes are Efficient Multi-Chained Stream Signature (EMSS) Perrig and al. (2000) and Augmented Chain (AC) Golle and Modadugu (2001). They append the hash of a packet to several other packets. The signature and its amortization induce some extra-information called the authentication

information. Besides, most of multicast media-streaming applications do not use reliable transport layer. Hence, some packets may be lost in course of transmission. Therefore, the proposed solutions introduce redundancy in the authentication information, in a way that even if some packets are lost, the required authentication information can be recovered in order to verify received packets authenticity. In this case, the bandwidth overhead, induced by the redundant authentication information, increases. Proposed solutions deal with how to trade bandwidth for tolerance to packet loss. Typically, existing solution fix the hash-chaining degree for a given value of the packet loss ratio (PLR). However, for modern networks, the network state is highly dynamic, leading to a considerable variation of the PLR. In this paper, we propose an adaptive multicast source authentication scheme called Adaptive Multi-Layer Connected Chains (AMLCC). It assures non-repudiation and tolerates packet loss. In contrast to other schemes Eltaief and Youssef (2009); Abuein and Shibusawa (2005); Gennaro and Rohatgi (2001); Perrig and al. (2000); Golle and Modadugu (2001) based on static hash-chaining, our scheme adapts the redundancy chaining degree (the amount of authentication information) depending on the actual packet loss ratio in the network. Indeed as we shall see in the experimental section packet loss varies considerably during a multicast session Yajnik and al. (1999). Hence, adapting the hash-chaining degree to the value of PLR may improve considerably the performance of chaining scheme with respect to communication, computation and delay overhead. This paper is organized as follows. Section 2 reviews related work that uses hash-chaining techniques to amortize signatures over a sequence of packets of the stream. Section 3 presents our scheme AMLCC. In section 4 we evaluate the performance of AMLCC and compare it with other schemes using NS-2 simulations. We conclude in section 5.

2. RELATED WORK

Recent source authentication schemes rely mainly on using MACs and hashes combined with digital signatures. MAC-based approaches Perrig and al. (2005); Bergadano and al. (2000); Canetti and al. (1999) are generally used when only source authentication (without non-repudiation) is required. Whereas, hash / digital signature based approaches Gennaro and Rohatgi (2001); Abuein and Shibusawa (2005); Perrig and al. (2000); Park and al. (2002); Eltaief and Youssef (2009); Lysyanskaya and al. (2004); Chan (2003); Christophe and Huaxiong (2007) are generally used when non-repudiation is required beyond source authentication. Since our mechanism uses a hash-based technique to

sign multicast streams, we will discuss particularly some mechanisms within this approach in the following paragraphs. We present in this section approaches fitting into the same category of our mechanism. These approaches are Efficient Multi-Chained Stream Signature (EMSS) Perrig and al. (2000), Hybrid Hash-chaining scheme for Adaptive multicast source authentication of media-streaming (H2A) Challal and al. (2005) and an Adaptive source authentication protocol for Multicast streams (A2Cast) Challal and al. (2004).

EMSS is a solution proposed by Perrig and al. (2000), where they introduced the notion of redundant and random hash-chaining, which means that each packet of the stream is hash-linked to several target packets. Thus, even if some packets are lost, a received packet is verifiable if it remains a hash-link path that relates the packet to a signature packet. For a given packet, EMSS chooses target packets randomly. Hence, EMSS provides probabilistic guarantees that a hash-link path remains between the packet and a signature packet, given a certain network packet loss ratio. EMSS operates as follows. When a packet is presented to be sent, the source embeds some hashes of other packets in this packet and computes the overall hash code. This hash code is buffered to be included later in d target packets chosen randomly by the sender (where d is the redundancy degree). In order for the sender to continuously assure the authentication of the stream, the source sends periodic signature packets. To verify the authenticity of received packets, a receiver buffers received packets and waits for their corresponding signature packet. The signature packet carries the hashes that allow the verification of few packets. These latter packets carry, in turn, the hashes that allow the verification of other packets, and so on until the authenticity of all received packets is verified. The main drawback of this scheme is that receivers experience latencies before verifying received packets, since they must wait for the signature packet corresponding to the received packets.

A2Cast is a solution proposed by Challal and al. (2004) to reduce the computation overhead they amortize one signature over multiple packets using Random hash-chainig structure like EMSS Perrig and al. (2000). H2A is also a solution proposed by Challal and al. (2005). In order to tolerate packet loss, they make redundant an hybrid hash-chaining. Even if some packets are lost, there is a probability that it remains hash-link paths between received packets and the signature packet. If a hash-link path exists between a received packet and the signature packet, then the authenticity of the received packet is verifiable Gennaro and Rohatgi (2001); Perrig and al. (2000)). When a packet is

Table 1: NOTATION

Symbol	Representation
nmes	the number of messages in the stream.
nline	the number of chains in a layer (line number).
ncol	the number of packets in the chains of the same layer (column number).
nhline	the number of packets ($P_{j'}^{i'}$) that contain the hash of P_j^i where ($i' \neq i$).
nhlineC	the current number of packets ($P_{j'}^{i'}$) that contain the hash of P_j^i where ($i' \neq i$).
nhlineP	the precedent number of packets ($P_{j'}^{i'}$) that contain the hash of P_j^i where ($i' \neq i$).
nsig	the number of signature packets.
nlay	the number of layers.
indline	the index of line in the layer ($0 \leq \text{indline} < \text{nline}$).
indcol	the index of column in the layer ($1 \leq \text{indcol} \leq \text{ncol}$).
BLS	the burst loss size.

presented to be sent at the sender, it is hash-linked to k subsequent packets following two steps. First, in this step the hash of the current packet is embedded into the next packet systematically. In the second step the hash of the current packet is embedded within $k-1$ subsequent packets chosen randomly. According these two mechanisms and in order for the sender to continuously verify the signature of the stream, the sender sends periodic signature packets. In a bursty loss model, with an average bursts' length b , these mechanisms includes systematically the hashes of the b packets that precede a signature packet in the signature packet. The authors assume that there exists a certain mean for receivers to communicate the quality of reception in term of packet loss ratio. So, after each period of time, the source analysis the received quality of reception reports to determine the actual packet loss ratio. Then, the source adjusts the redundancy degree accordingly in order to maintain the desired verification ratio.

3. ADAPTIVE MULTI-LAYER CONNECTED CHAINS MECHANISM FOR MULTICAST SENDER AUTHENTICATION

3.1. Terminology

If a packet P_j contains the hash of a packet P_i , we say that a hash-link connects P_i to P_j , and we call P_j a target packet of P_i . A signature packet is a sequence of packet hashes which are signed using a conventional digital signature scheme. A hash-link relates a packet P_k to a signature packet

P_{sig} , if P_{sig} contains the hash of P_k . We designate by redundancy degree the number of times that a packet hash is embedded in subsequent packets to create redundancy in chaining the packet to a signature packet. A packet P_i is verifiable, if a path remains (following the hash-links) from P_i to a signature packet P_{sigj} (even if some packets are lost). We designate by verification ratio, the number of verifiable packets by the number of received packets. This probability is equal to the probability that it remains a hash-link path that relates the packet to a signature packet.

3.2. Overview and Motivation

To achieve non-repudiation, we rely on a conventional signature scheme for example RSA Rivest and al. (1978). Unfortunately, the computation and communication overhead of current signature schemes is too high to sign every packet individually. To reduce the overhead, one signature needs to be amortized over multiple packets. The amortization is achieved using hash-chaining, which consists in signing a single packet and amortizing this single signature over multiple packets by hash-linking the current packet to another packet in the stream. But these Mechanisms Eltaief and Youssef (2009); Abuein and Shibusawa (2005); Gennaro and Rohatgi (2001); Perrig and al. (2000); Golle and Modadugu (2001) are based on static hash-chaining. They maintain the hash-chaining degree constant during the entire multicast session. Our mechanism uses a hash-chaining structure whose degree varies with the value of the PLR currently estimated. The adaptation of the redundancy degree allows to save bandwidth overhead compared to static redundancy degree Challal and al. (2004, 2005). In the following sections, we detail our hash-chain structure used in our mechanism. Then, we describe the operation of our Adaptive Multi-layer connected chains mechanism for multicast sender authentication (AMLCC).

3.3. Deterministic Hash-chain Structure

The notation used in this paper is presented in Table 1. Our deterministic Hash-chain structure is an amortization mechanism that seeks to achieve a strong resistance against packet losses while reducing the overhead. it divides the packet stream into a multi-layer structure, where each layer is a two-dimensional matrix. The hash of a packet is included into a forward chain of packets within the same layer as well as a downward chain of packets across succeeded layer.

A packet P_j^i is defined as a message M_j^i a source sends to the receivers while appending the required authentication information. P_j^i corresponds to packet number $i+(j*nline*ncol)$ in the original stream where

Figure 1: Illustration of appended $H(P_j^i)$ to $nhlineC$ packets

j indicates the layer and i the order of the packet in that layer ($P_j^i = P_{i+(j*nline*ncol)}$, $1 \leq i \leq nline*ncol$ and $0 \leq j \leq nlay-1$) and M_j^i corresponds to message number $i+(j*nline*ncol)$ in the original stream where j indicates the layer and i the order of the packet in that layer ($M_j^i = M_{i+(j*nline*ncol)}$, $1 \leq i \leq nline*ncol$ and $0 \leq j \leq nlay-1$). Our deterministic hash chain structure divides a stream of $nmes$ messages into $nlay$ layers, each containing $nline*ncol$ messages. The sender appends the hash $H(P_j^i)$ of a packet P_j^i to other specific packets to achieve robustness against packet losses. Each two consecutive layers form a block. For each block a signature packet P_{sig} is generated. P_{sig} consists of the hashes of the last BLS packets in addition to the signature of these hashes using the sender private key. This packet is sent by the sender at the end of each block. In our proposed hash-chain structure, the hash $H(P_j^i)$ of each packet P_j^i is appended to the following $nhline+1$ packets:

- $nhline$ packets in the same layer:
 $P_j^{i+1}, P_j^{i+nhline}, P_j^{i+2*nhline}, \dots, P_j^{i+(nhline-1)*nhline}$;
- one packet in the succeeded layer : P_{j+1}^i .

For each block, BLS hashes are concatenated together and signed using the sender private key.
 $P_{sig} := \text{Sig}(K, H(P_{j+1}^{i+nhline*ncol}) || H(P_{j+1}^{i+nhline*ncol-1}) || \dots || H(P_{j+1}^{i+nhline*ncol-BLS-1})) || (H(P_{j+1}^{i+nhline*ncol}) || H(P_{j+1}^{i+nhline*ncol-1}) || \dots || H(P_{j+1}^{i+nhline*ncol-BLS-1}))$.

Where $||$ denotes the concatenation operator, Sig represents the signing algorithm, and K represents the private key.

In our mechanism, we use a Hash-chain structure where the redundancy degree is adaptive with respect to the actual packet loss ratio in the network (see the next section). For each block, we use a new value of $nhline$. We consider the two parameters $nhlineC$ and $nhlineP$ that represent respectively the current and precedent number of packets ($P_{j'}^{i'}$) that contain the hash of P_j^i where ($i' \neq i$).

Figure 1 shows an example of the appended $H(P_j^i)$ to the $nhlineC+1$ packets according to the proposed hash chain structure. Where the hash packet that contain the parameter k (k in $[1 \text{ } nhlineP\text{-indCol}]$) is calculated if $nhlineP\text{-indCol} > 0$.

The receivers perform the verification steps in a reverse order as follows: First, a receiver verifies the signature packet P_{sig} , then retrieves $H(P_{j+1}^{i+nhline*ncol})$, $H(P_{j+1}^{i+nhline*ncol-1})$, ..., $H(P_{j+1}^{i+nhline*ncol-BLS-1})$. If the signature packet is secure then the hashes appended to it are considered secure too. When

Figure 2: Required redundancy degree to reach 99% of verification ratio

receiving the remaining packets of the block, the receiver computes the hash value of each packet starting from the last packet in the block, compares the computed value to the retrieved one; if both are equal then the packet is declared authentic. Otherwise, it is considered not authentic. If there is no hash carried by one of the received packets, corresponding to a received packet, this latter is considered not verifiable. This case happens when the hash-chain relating a packet to a signature packet is completely broken because of packet loss.

3.4. The Adaptive Multi-Layer Connected Chains Mechanism (AMLCC)

The redundancy degree in our mechanism is adaptive and depends on the actual packet loss ratio in the network. Indeed, using the proposed hash-chaining structure, we are interested in looking for the best redundancy degree to be used to maintain 99% of verification ratio depending on the measured packet loss ratio. Figure 2 shows the required redundancy degree when the packet loss ratio varies from 5% to 40%. Since the required redundancy degree to reach 99% of verification ratio depends proportionally on the packet loss ratio (see Figure 2), we suggest to exploit receivers feedback as mentioned in Challal and al. (2004, 2005) regarding packet loss in the network to adapt the redundancy degree and hence to use only the required amount of authentication information to reach the best verification ratio. We assume that there exists a mean for the slowest receiver to communicate to the sender the packet loss ratio in the network (for example by sending periodic RTCP Receiver Reports) Schulzrinne and al. (2003). In each bloc, and relying on the receivers feedback, the sender decides what is the best redundancy degree

Figure 3: AMLCC sequence diagram

to use in order to tolerate the actual packet loss ratio in the network. In our adaptation we use only the last reports received during the last bloc, to determine the hash-chaining degree for the following bloc. We could also chose the average value of received one during the current bloc to adapt during the succeeding bloc. So in contrast to the idea proposed in Challal and al. (2004, 2005) our mechanism doesn't have to use a synchronization between the sender and receivers.

Figure3 shows the different messages exchanged between a source and a receiver of an authentic data-stream. It illustrates also the periodic operations executed by the source and the receiver to adapt respectively the redundancy degree and to verify the authenticity of received packets.

The parameters involved in the AMLCC sequence diagram are:

- tr : The periodicity with which the quality of reception reports(QReport) are sent.
- nts : Number of packets after which a signature packet is sent.

These parameters influence the computation and communication overhead, the delay until verification, and the robustness against packet loss. In our case we want to achieve low overhead while retaining high robustness against packet loss and an authentication delay supported by the used application.

Figure 4 shows the algorithm at the source side. A source of a stream applies the proposed hash-chaining technique described above for each packet P_j^i before it sends it. After each nts data

```

For each packet  $P_i$  Do
  Include  $nhlineC$ ,  $nhlineP$  in the current packet;
  Include  $H(P_i) = h_i$  in the packet  $P_{i+1}$ ,  $P_{i + nline*ncol}$ ;
  Do  $k$  Times
     $k \in [1, nhlineC-1]$ ;
    Include  $H(P_i) = h_i$  in the packet  $P_{i + k * nline}$ ;
  End.
  send  $P_i$ ;
End.
After each  $nts$  packets Do
  Compute the average packet loss ratio
  from the received reports:  $apl$ ;
   $nhlineP = nhlineC$ ;
  Degree = Adapt_degree (  $apl$ ,  $dv$  );
   $nhlineC = Degree - 1$ ;
  Sign then send the signature packet  $P_{sig}$ ;
End.

```

Figure 4: Hash-chaining algorithm applied at the sender side

```

Do
  receive Packet  $P$ ;
  If (  $P$  is not a signature packet ) Then
    buffer  $P$ ;
    buffer hashes included in  $P$ ;
  Else
    verify( $P$ );
  End.
While (true)
  Upon timeout Do
    generate a quality of reception report:
    QReport including the packet loss ratio;
    send QReport;
    schedule timeout after  $tr$  seconds;
  End.

```

Figure 5: Algorithm at the receiver side

packets, the source sends a signature packet. These periodic signature packets ensure continuous non-repudiation of the stream. Besides, since the verification process depends on the reception of signature packets, the source can replicate signature packets so that their loss probability becomes very low. After each block, the sender analyzes the last received quality of reception report and adjusts the redundancy degree $nhlineC$ accordingly to maintain the desired verification ratio dv . The adapt_degree function determines the best redundancy degree (Degree) to reach the desired verification ratio (dv) given that packets may be lost in the network with an average ratio equal to apl .

When a receiver gets a signature packet P_{Sigj} , it verifies the signature of P_{Sigj} and checks the authenticity of all the packets that have a path to P_{Sigj} . After each tr (seconds), the receiver sends to the source of the stream a quality of reception

Verify (packet P)**Begin**

If (P is a signature packet) **Then**
 verify the signature of P;

If (P is valid) **Then**

P is authentic;

For each hash h_i included in P **Do**
 verify(P_i);

End.

Else

P is not authentic;

End.

Else

//verify P against its buffered hash-code h

If ($H(P) = h$) **Then**

P is authentic;

For each hash h_i included in P **Do**
 verify(P_i);

End.

Else

P is not authentic;

End.

End.

Fin

Figure 6: The verification procedure

report including the packet loss ratio during the last tr seconds. The algorithm at the receiver side is shown in Figure 5 and the verification procedure is illustrated in Figure 6.

4. SIMULATION AND PERFORMANCE EVALUATION

In this section we evaluate using NS-2 simulator the performance of AMLCC and compare it with H2A Challal and al. (2005) and A2Cast Challal and al. (2004).

The performance metrics considered are robustness and authentication overhead. Robustness is measured in terms of the authentication probability. Overhead is measured in terms of delay, computation and communication overhead. Communication overhead is measured by the average number of hashes added per packet. Computation overhead at the source corresponds to hashes and digital signatures operations that the source computes in order to authenticate all packets of the stream.

The simulation scenario considered is shown in Figure 7. In multicast applications the source sends packets to multiple destinations. However, as illustrated in Figure 7, we have reduced the number of receivers to the slowest receiver. This assume that the multicast source adapts a single rate approach where data is transmitted at the rate of the slowest receiver. Hence only the slowest receiver is represented and the link R1-R2 is considered as the bottleneck link. To get similar conditions

Figure 7: The simulation scenario

Figure 8: Two-state Markov chain to simulate bursty packet loss

to those of the Internet, we have added a VBR source (Variable Bit Rate) which sends packets to the receiver VBR using variable periods. The packets sent by the VBR source share the same link (R1, R2) as the packets of the source multicast and, therefore, the availability of the bandwidth would be variable. In an open network such as Internet, packets undergo various types of losses with variable rates. Based on the assumption that if a packet P_i is lost, the probability that the packet P_{i+1} would be lost is large. Paxson in Paxson (1999) has shown that the burst loss is likelier in Internet. A two-state Markov model is used to model the network burst loss Yajnik and al. (1999). Figure 8 shows the two-state Markov chain used in our simulations and whose transition probabilities can easily be determined using the average burst length and the packet loss ratio in the network Varela and al. (2006). The relationship between the parameters in the model and the ones we use in this paper, the packet loss rate (PLR) and the burst loss size (BLS) is as follows:

$$p = \frac{1}{BLS} * \frac{PLR}{1 - PLR}, \quad q = \frac{1}{BLS} \quad (1)$$

We consider the case of Real-time Video Broadcast used in Perrig and al. (2000) to evaluate EMSS. We analyze the performance of applying AMLCC, A2Cast and H2A to ensure the non-repudiation of the streamed data. Assume we want to distribute signed video on the Internet. The system requirements are as follows:

- The data rate of the stream is about 2 Mbps, about 512 packets of 512 bytes each are sent every second.
- Some clients experience packet drop rates up to 60%, where the average length of burst drops is 10 packets.

Figure 9: The considered scenario of packet loss ratio variation

- The verification delay should be less than 1 second.

We considered a stream of 10,000 packets with a signature packet every 250 packets ($n_{ts}=250$) and a bursty packet loss pattern with bursts having an average length equal to 10. Packet losses occur on the link between the two routers (R1,R2) according to a Gilbert distribution with a user specified parameter PLR (packet loss ratio). Receivers send quality of reception reports including the packet loss ratio every $tr=0.2s$.

We considered the distribution of packet loss ratio over time shown in Figure 9. We aim to reduce the bandwidth overhead (redundancy degree) while increasing the verification ratio and an authentication delay supported by the used application (Maximum Delay $< 1s$).

Recall that periodically, the source analyzes quality of reception reports. Then the source adapts the redundancy degree accordingly using the *Adapt_degree* function. To develop this function, we run extensive simulations of our hash-chaining scheme, H2A and A2Cast by varying packet loss ratio from 5% to 40%, and we noted for each packet loss ratio the minimum redundancy degree which allows to reach a very high verification probability of received packets (namely 99%). Figure 2 illustrates the results. Hence, given an average loss ratio, our *Adapt_degree* function returns the minimum redundancy degree which guarantees a very high verification ratio (99%) according to the results of these simulations. In other words, the graph depicted in Figure 2 corresponds to the *adapt_degree* function used respectively by our scheme (Deterministic

Figure 10: The variation of the required redundancy degree to reach 99% of verification ratio

hash-chaining), H2A (Hybrid hash-chaining) and A2Cast (Random hash-chaining). According to AMLCC, A2Cast and H2A, the source requires only a single hash computation overhead per packet in addition to a single digital signature per block of packets. The communication overhead per packet is equal to the total number of hashes and digital signatures computed divided by the total number of packets in the stream. We considered a target

Table 2: Average number of hashes per packet overhead (hashes)

	AMLCC	A2Cast	H2A
avg_overhead	3,40	4,67	4,01

verification ratio $dv=99\%$, and we run simulations of H2A, A2Cast and AMLCC to determine the required redundancy degree by each mechanisms in order to reach the target verification ratio. The results were illuminating: first, the redundancy degree of AMLCC over time is obviously proportional to packet loss ratio (compare the shape of the graph representing the redundancy degree of AMLCC in Figure10 with the shape of the graph representing the variation of packet loss ratio in Figure 9. Besides, we found that AMLCC reaches 99% of verification ratio with only an average of 3.40 hashes per packet, whereas A2Cast and H2A require respectively 4.67 and 4.01 hashes per packet to reach the same verification ratio (see table 2). This means that AMLCC allows to save up to 1,27 hashes per packet compared to A2Cast and up to 0.61 hashes per packet compared to H2A. While sending the 10,000 packets of the stream

Figure 11: The variation efficiency depending on packet loss ratio: the numbers beside the points represent the average redundancy degree.

and if we consider a hash algorithm that produces a 20 byte hash code (such as SHA-1 Eastlake and al. (2001)), this means that AMLCC saves up to 0,254 Mbytes of authentication information compared to A2Cast and up to 0,122 Mbytes compared to H2A. In other words, AMLCC allows to save up to 27,19% of the authentication information used by A2Cast and save up to 15,21% of the authentication information used by H2A. Figure 11 illustrates the verification efficiency as a function of packet loss ratio (we consider an authentication delay supported by the used application (Maximum Delay < 1s)). It shows that AMLCC resists better to packet loss compared to H2A and A2Cast. As it can be seen, AMLCC was capable of achieving the highest authentication probability compared to H2A and A2Cast, for all values of packet loss ratio. For a packet loss ratio of 40% AMLCC was capable of achieving an authentication probability equal to 99,32% using an average communication overhead equal to 5,14 hashes, whereas H2A and A2Cast achieve an authentication probability respectively equal to 97,86% and 95,79% using the same average communication overhead used by AMLCC. With AMLCC, A2Cast and H2A the source

Table 3: Storage requirements at the source of AMLCC, A2Cast and H2A approach.

Stor. Requirements (hashes)		
A2Cast	H2A	AMLCC
variable(<125)	variable(<100)	126

authenticates the packets and signs the stream on the fly. Hence, the multicast source experiences one packet delay. However, the source has to store the

hash values of some packets that are necessary to compute the hash values of succeeding packets and the signature packets. Table 3 shows the storage requirements at the source of AMLCC, A2Cast and H2A approach. We consider one signature packet every 250 packets and the same number of hashes attached to the signature packet (BLS hashes). Because of the multi-layer structure of AMLCC, source storage requirement is higher than that of A2Cast and H2A. In case of any hashes as average communication overhead A2Cast, H2A and AMLCC require to buffer respectively less than 125, 100 and 126 hashes. For example, given an effective hash size of 128 bits for MD5, AMLCC source must be able to store 2,01 KBytes of hashes. A multicast

Table 4: Maximum and average delay overhead

	A2Cast	H2A	AMLCC
Maximum Delay (s)	1,004	0,992	0,987
Average Delay (s)	0,373	0,307	0,318

receiver in the three approaches experiences a delay and needs to buffer received packets until the reception of the corresponding signature packet. The delay and the size of the buffer depend on the number of signature packets. So, there is a tradeoff between the delay at the receiver and computation overhead at the source. AMLCC allows to reduce the delay at the receiver at the price of a possible slight increase of the computation overhead at the source. Table 4 shows that AMLCC has an average

Table 5: Maximum and average buffer size at the receiver side (bytes)

	H2A	A2Cast	AMLCC
Max. buffer size	213325	232140	210580
Avg. buffer size	69257	82784	70186

delay overhead close to that obtained using H2A and less than the delay obtained using A2Cast. Although, the maximum average delay overhead for AMLCC is equal to 0,941 seconds. So, this delay is supported by the used application. Table 5 shows that AMLCC has a lower maximum and average buffer size than A2Cast. So, compared to A2Cast, AMLCC is capable of saving up to 12,598 Kbytes at the receiver memory for a stream of 10000 packets. But, AMLCC requires more storage requirement at the receiver side than H2A. Our mechanism require to buffer 0,929 Kbytes more than H2A at the receiver side for a stream of 10000 packets. As it can be seen in Table 6, to achieve 99% as authentication probability, our mechanism uses an average overhead equal to 3.40 hashes per packet, an average delay overhead equal to 0,318 seconds and an average buffer size equal to 70186 bytes, whereas H2A and A2Cast

Table 6: Evaluation table to achieves 99% of the authentication probability

	H2A	A2Cast	AMLCC
Com. Overhead	4,01	4,67	3,40
Maximum Delay	0,992	1,004	0,987
Average Delay	0,307	0,373	0,318
Max. buffer size	213325	232140	210580
Avg. buffer size	69257	82784	70186

use respectively an average overhead equal to 4,01 and 4,67 hashes, an average delay overhead equal to 0,307 and 0,373 seconds and an average buffer size equal to 69257 and 82784 bytes to achieves the same authentication probability. So, AMLCC reduces the used redundancy degree (the bandwidth overhead) while increasing the verification ratio and generates an authentication delay supported by the used application (Maximum Delay<1s). But, compared to H2A, first AMLCC requires more storage requirement at the sender and receiver side and, at the receiver side our mechanism generates slightly more delay (closed to the average delay obtained using H2A).

In conclusion, simulations show that AMLCC adapts well the required authentication information size (redundancy degree) to the actual packet loss ratio in the network and hence allows to reduce the authentication information overhead while maintaining high robustness against packet loss. Since packets cannot be verified until the correspondent signature packet is received, receivers experience some delay before the verification of received packets. In addition, our mechanism generates an authentication delay supported by the used application (Maximum Delay<1s) and an acceptable storage requirement at the receiver side. But AMLCC generate slightly more storage requirement at the sender side. Scalability is not a concern since the number of hashes attached within each packet is independent of the number of multicast group members. AMLCC computation overhead is reduced to a single hash computation per packet in addition to a periodic digital signature computation.

5. CONCLUSION

Source authentication is an important component in the whole multicast security architecture. Besides, many applications need non-repudiation of data-streams. To achieve non-repudiation, we present an Adaptive Multi-layer connected chains scheme for multicast source authentication of media-streaming. Our mechanism uses a new multi-layer connected chains scheme to amortize a single digital signature over several packets. Performance comparisons

with others approaches using NS-2 show that our scheme resists to bursty packet loss and assures with a high probability that a received packet be verifiable. Besides, the simulations and comparisons show that our adaptive hash-chaining technique is more efficient than recent hash-chaining techniques that take into consideration the actual packet loss ratio in the network. Indeed, adapting redundancy degree to the packet loss ratio allows to save useless authentication information redundancy and hence reduces the bandwidth overhead. Furthermore, the hash-chaining technique used by our mechanism allows to maintain high robustness to packet loss and assures with a high probability that a received packet be authenticated with a lower communication and an authentication delay overhead supported by the used application.

6. REFERENCES

- Abuein, Q., Shibusawa, S. (2005) *A Graph-based New Amortization Scheme for Multicast Streams Authentication* AMO - Advanced Modeling and Optimization.
- Bergadano, F., Cavagnino, D., Crispo, B. (2000) *Individual Single Source Authentication on the MBone* IEEE International Conference on Multimedia and Expo.
- Canetti, R., Garay, J., Itkis, G., Micciancio, D., Naor, M. (1999) *Multicast Security: A taxonomy and Efficient Constructions* INFOCOM.
- Challal, Y., Bettahar, H., Bouabdallah, A.(2004) *SAKM: a scalable and adaptive key management approach for multicast communications* ACM SIGCOMM Computer Communications Review.
- Challal, Y., Bettahar, H., Bouabdallah, A. (2004) *A2Cast:an Adaptive source Authentication protocol for multiCast streams* ISCC'04, IEEE.
- Challal, Y., Bouabdallah, A., Bettahar, H. (2005)*H2A: Hybrid Hash-chaining scheme for Adaptive multicast source authentication of media-streaming* Elsevier Computers and Security Journal.
- Chan, A. (2003)*A graph-theoretical analysis of multicast authentication* Proc. of the 23rd Int. Conf. on Distributed Computing Systems.
- Christophe, T., Huaxiong, W. (2007)*Efficient multicast stream authentication for the fully adversarial network model* International Journal of Security and Networks.
- Eastlake, D., Jones, P., Frederick, R., Jacobson, V. (2001) *US secure hash algorithm 1 (SHA1)* RFC 3174.
- Eltaief, H., Youssef, H.(2009)*MLCC: A new hash-chained mechanism for multicast source authentication* International Journal Of Communication Systems.

- Gennaro, R., Rohatgi, P. (2001) *How to Sign Digital Streams* Information and Computation.
- Golle, P., Modadugu, N. (2001) *Authenticating streamed data in the presence of random packet loss* Proc. of ISOC Network and Distributed System Security Symposium.
- Hardjono, T., Tsudik, G. (2000) *IP multicast security: issues and directions* Annals of Telecom.
- Judge, P., Ammar, M. (2003) *Security issues and solutions in multicast content distribution: a survey* IEEE Network.
- Lysyanskaya, A., Tamassia, R., Triandopoulos, N. (2004) *Multicast authentication in fully adversarial networks* Proc. of IEEE Symposium on Security and Privacy.
- Park, J. M., Chong, E. K. P., Siegel, H. J. (2002) *Efficient Multicast Packet Authentication Using Signature Amortization* IEEE Symposium on Security and Privacy.
- Paxson, V. (1999) *End-to-end internet packet dynamics* IEEE/ACM Transactions on Networking.
- Perrig, A., Canetti, R., Tygar, J. D., Song, D. (2000) *Efficient authentication and signing of multicast streams over lossy channels* IEEE Symposium on Security and Privacy.
- Perrig, A., Song, D., Canetti, R., Tygar, J. D., Briscoe, B. (2005) *Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction* IETF RFC 4082.
- Rafaeli, S., Hutchison, D. (2003) *A survey of key management for secure group communication* ACM Computing Surveys.
- Rivest, R. L., Shamir, A., Adelman, L. M. (1978) *A method for obtaining digital signatures and public-key cryptosystems* Communications of the ACM.
- Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V. (2003) *RTP: a transport protocol for real-time applications* RFC 3550.
- Varela, M., Marsh, I., Grnvall, B. (2006) *A systematic study of PESQs behavior (from a networking perspective)* In Proc. Measurement of Speech and Audio Quality in Networks.
- Yajnik, M., Moon, S., Kurose, J., Towsley, D. (1999) *Measurement and modeling of the temporal dependence in packet loss* INFOCOM99.