Computers & Security (2005) **24**, 57–68

ELSEVIER

Computers & Security

www.elsevier.com/locate/cose

H_2A: Hybrid Hash-chaining scheme for Adaptive multicast source authentication of media-streaming

Yacine Challal*, Abdelmadjid Bouabdallah, Hatem Bettahar

Compiegne University of Technology, Computer Science Department, Heudiasyc Laboratory, Royallieu Research Center, PO Box 20529, 60205 Compiegne, France

Received 3 April 2004; revised 15 June 2004; accepted 18 June 2004
Available online 28 January 2005

KEYWORDS
Data source authentication;
Non-repudiation;
Adaptive hash-chaining;
Multicast flows;
Multicast security

Abstract Many applications, such as broadcasting stock quotes and video-conferencing require data source authentication of the received multicast traffic. Multicast data source authentication must take into consideration the scalability and the efficiency of the underlying cryptographic schemes and mechanisms, because multicast groups can be very large and the exchanged data are likely to be important in volume (streaming). Besides, multicast data source authentication must be robust enough against packet loss because most of multicast multimedia applications do not use reliable packet delivery.

In this paper, we propose a hybrid hash-chaining scheme in conjunction with an adaptive and efficient data source authentication protocol which tolerates packet loss and guarantees non-repudiation of media-streaming origin. We have simulated our protocol using NS-2, and the simulation results show that the protocol has remarkable features and efficiency compared to other recent data source authentication protocols.

Introduction

The increase of bandwidth in today's networks encourages the deployment of multi-party applications, such as video-conferencing, TV over Internet, e-learning and video-on-demand. Broadcasting information to a group of participants can be achieved using multiple point-to-point transmissions (unicast). This solution is not efficient because of information duplication which induces a high bandwidth consumption. The alternative approach is multicasting (Deering, 1988) which is an efficient communication mechanism for group-oriented applications. IP multicast saves

* Corresponding author. Tel.: + 33 3 44 23 44 23.
E-mail addresses: ychallal@hds.utc.fr (Y. Challal), bouabdal@hds.utc.fr (A. Bouabdallah), hbettaha@hds.utc.fr (H. Bettahar).

0167-4048/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cose.2004.06.012

bandwidth by sending the source traffic on a multicast tree that spans all the members of the group. The lack of security obstructs the large scale deployment of multicast communication applications (Judge and Ammar, 2003): *data integrity, secrecy, authentication and access control*. Therefore, securing the multicast communication model is a strategic requirement for effective deployment of large scale business multi-party applications (TV over Internet, Video-on-Demand (VoD), video-conferencing, interactive group games,...). One of the main issues in securing multicast communication is the *authentication service*; a keystone of every secure architecture. Even though several authentication mechanisms have existed so far, data source authentication in multi-party communications remains a challenging problem in terms of scalability, efficiency and performance. Indeed, hashes (Kaliski, 1992; Rivest, 1992; Eastlake and Jones, 2001), MACs (Krawczyk et al., 1997), and digital signatures (Rivest et al., 1978; Federal Information Processing Standards Publication, 1994) are the cryptographic answers to integrity, authentication, and non-repudiation in data transmission. However, these mechanisms have been designed typically for point-to-point transmissions, and using them in multicasting yields inefficient and non-adequate solutions. This non-suitability of existing authentication mechanisms is mainly due to the number of group members which may be high in multi-party applications, and to the type of transmitted data which consist generally in continuous streaming of multicast messages with real-time transmission requirement. We distinguish between two types of authentication in group communication (Hardjono and Tsudik, 2000):

- *Group authentication*: aims to assure that the received multicast messages by group members originate from a valid group member (no matter its identity).
- *Data source authentication*: aims to assure that the received multicast messages by group members 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 et 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 protocols 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. 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.

In this paper, we propose a new adaptive and efficient protocol called *H_2A* which authenticates the source of a multicast flow, assures non-repudiation and tolerates packet loss. In contrast to other protocols (Gennaro and Rohatgi, 2001; Golle and Modadugu, 2001; Perrig et al., 2000; Miner and Staddon, 2001) based on static hash-chaining, with our protocol we propose a new *hybrid and adaptive hash-chaining* technique which adapts the *redundancy chaining degree* (the amount of authentication information) depending on the actual *packet loss ratio* in the network. Besides, this new hash-chaining technique combines deterministic hash-chaining with random hash-chaining, in contrast to existing protocols that use either deterministic (Golle and Modadugu, 2001; Miner and Staddon, 2001) or random hash-chaining (Perrig et al., 2000). The carried out simulations using NS-2 show that the adaptation of the *redundancy degree* allows to save bandwidth, and the combination of the random with deterministic hash-chaining allows to increase the robustness to packet loss.

In the following section, we present an overview of multicast data source authentication approaches, then we focus on related works that use hash-chaining techniques to amortize signatures over a sequence of packets of the stream. In the subsequent section, we describe our protocol *H_2A*, then we evaluate and compare it with other protocols using NS-2 simulations.

Multicast data source authentication

Definition 1. Data origin authentication service is a security service that verifies the identity of a system entity that is claimed to be the original source of received data (Shirey, 2000).

In what follows we use the terminology *data source authentication* as a synonym to *data origin authentication*. We distinguish between two levels of multicast data source authentication (Challal et al., 2004) as given below.

A first level guarantees *only* data source authentication of the multicast data origin. In this case, a sender needs to use an *asymmetric* mechanism which allows receivers to verify multicast messages authenticity without being able to generate valid authenticators for messages on behalf of the sender. Some solutions (Desmedt et al., 1992; Safavi-Naini and Wang, 1999; Hiroshi et al., 1996; Obana and Kurosawa, 2001; Canetti et al., 1999; Boneh et al., 2001) propose to introduce *asymmetry* in the *key material* used to authenticate messages. In other words, the sender knows the *entire* key material required to authenticate messages, and receivers know only a partial view of the key material, that allows them to verify received messages' authenticity *without being able to generate valid authenticators*. Other solutions (Bergadano et al., 2002; Perrig et al., 2001; Perrig, 2001) suggest to use *time* as source of *asymmetry*. In other words, receivers are synchronized with the sender's clock and are instructed when to accept a specific key as being used to authenticate received messages. In this case, a fraudulent cannot use a received (or eavesdropped) sender's key to forge messages on behalf of the sender.

A second level guarantees *non-repudiation in addition to data source authentication*. In this case, the multicast stream should be signed. Current digital signature mechanisms are *very computationally expensive*. Therefore, it is not practical to sign each packet of the multicast stream. Proposed solutions rely on the concept of *amortizing a single digital signature over multiple packets*. Some protocols (Perrig et al., 2000; Golle and Modadugu, 2001; Miner and Staddon, 2001) amortize a single signature over many packets by chaining these packets using some *hash-chaining* techniques. *Hash-chaining* consists in making each packet carrying hashes that allow the verification of few packets. In turn, these few packets will carry the authentication information of some other packets, and so on.... The overall hash-chaining process culminates into a special packet called *signature packet* which is signed. This signature will then propagate throughout the hash-chain to assure non-repudiation of the chained packets. A second approach (Wong and Lam, 1999; Pannetrat and Molva, 2003; Park et al., 2003) consists in signing only a small piece of authentication information (namely hashes of block packets). The resulting authentication information (the signature as well as the original authentication information) is *processed* and *dispersed* among the block packets to be signed. The *processing* is made in a way that even if some packets (that does not exceed a certain threshold) are lost, the received packets can recover the whole authentication information which is required to verify received packets.

In what follows, we detail some protocols that use *signature amortization* relying on hash-chaining techniques, in order to ease the presentation of our protocol that belongs to the same category.

Simple off-line hash-chaining

The main idea of the solution proposed by Gennaro and Rohatgi (1997, 2001) is to divide the stream into blocks and embed in the current block a hash of the following block (which in turn includes the hash of the following one and so on...) (see Fig. 1). This way the signer needs to sign only the first block and then the properties of this single signature will *propagate* to the rest of the stream through the hash-chaining. We note that in order to construct this chain, the sender needs to know the entire stream in advance (off-line). With this solution, the authentication information is reduced to one hash per block and the sender signs only the hash of the first block. However, this solution is not fault tolerant: if a block is lost, the authentication chain is broken and hence all subsequent blocks can no longer be authenticated.

Random hash-chaining

Perrig et al. (2000) proposed the *Efficient Multi-chained Stream Signature* protocol (EMSS). This protocol introduced the notion of *redundant*

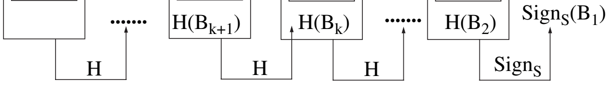

Figure 1 Simple off-line hash-chaining (example).

hash-chaining which means that each packet's hash of the stream is embedded in several *subsequent* packets. Then a final packet (which is called the signature packet) containing several hashes of previous packets is signed. Therefore, each packet has many hash-chains to the signature packet. Thus, even if some packets are lost, a received packet is verifiable if it remains a hash-chain that relates the packet to the signature packet. For a given packet, EMSS embeds its hash into k subsequent randomly chosen packets (k is called the redundancy degree). Hence, EMSS provides more or less probabilistic guarantees that there remains a hash-chain between the packet and a signature packet, given a certain rate of packet loss in the network. The robustness of the protocol to packet loss is proportional to the *redundancy degree* k. In order for the sender to continuously assure the authentication of the stream, the sender sends periodic signature packets. To verify 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 to verify other packets, and so on until the authenticity of all received packets is verified.

Deterministic hash-chaining

Golle and Modadugu (2001) have proposed to use a similar strategy to EMSS, but packets that will carry the hash of a given packet are chosen in a *deterministic* way rather than randomly. The proposed *deterministic topologies* of packet hash-chains (called *augmented chains*) are designed to be optimized to resist a *burst loss*. The goal of the proposed schemes is to maximize the size of the longest single burst of loss that the authentication scheme can withstand (once few packets have been received after a burst, the scheme recovers and is ready to maintain authentication even if further loss occurs).

Miner and Staddon (2001) proposed a similar authentication scheme (called *Piggybacking*), based on hash-chaining techniques, specifically designed to resist multiple bursts. The proposed scheme deals with the case where data carried by different packets have more or less importance from the point of view of the application level. Thus, packets are organized into classes with different priorities. Then hash-chaining is made in a way that the higher the priority of a class, the more redundant the hash-chaining of packets belonging to that class, in order to resist more against bursty losses.

In what follows, we present our protocol which uses the concept of amortizing a single digital signature over multiple packets using hash-chaining in a way that reduces the bandwidth overhead and enhances the verification ratio of received packets even if some packets are lost.

H_2A: Hybrid Hash-chaining scheme for Adaptive multicast data source authentication

Terminology

We define some terminology to simplify the following discussion: 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 S_l, if S_l 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 it remains a **path** (following the hash-links) from P_i to a signature packet S_j (even if some packets are lost). We designate by **verification ratio**: the number of verifiable packets by the number of received packets. The verification ratio is a good indicator of the **verification probability** which means the probability for a packet to be verifiable given that it is received: P(packet is verifiable/packet is received). This probability is equal to the probability that it remains a **hash-link path** (a hash-chain) that relates the packet to a signature packet.

Overview and motivation

To achieve non-repudiation, we rely on a conventional signature scheme for example RSA (Rivest et 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. In section 'Simple off-line hash chaining' we discussed a basic chaining scheme. In our protocol, we use a *redundant hash-chaining* scheme to tolerate packet loss. Moreover, the *redundancy degree* of our *redundant hash-chaining* scheme is *adaptive* with respect to the *actual* packet loss ratio in the network. This *adaptation* of the *redundancy degree* allows to save *bandwidth overhead* compared to *static redundancy degree*. In the following sections, we detail the hash-chaining technique used in our protocol. Then, we describe the operation of the protocol called Hybrid Hash-chaining scheme for Adaptive multicast data source authentication (H_2A).

Redundant and hybrid hash-chaining scheme

The basic idea of hash-chaining is that each packet carries the hash code of the previous packet. A final packet (the signature packet) is signed and guarantees data source authentication and non-repudiation of the chained packets (Gennaro and Rohatgi, 2001). In order to tolerate packet loss, we make *redundant hash-chaining*: instead of carrying a single hash of the previous packet, each packet carries the hashes of multiple packets, so that 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 et al., 2000).

When a packet is presented to be sent at the sender, it is *hash-linked* to *k* subsequent packets following the two steps:

(a) *Deterministic target packet:* in this step, the hash of the current packet is embedded into the next packet *systematically*. What motivates this choice of the target packet is the *bursty* nature of packet loss (Paxson, 1999). Indeed, it is easy to see that since packets are lost in a bursty way, the received packets are also received contiguously. Hence, if each packet is chained systematically to its subsequent packet, then if only one packet is verifiable then all the packets that follow it (in the same contiguous received segment) are also verifiable. In Fig. 2, packets P_{f-1} to P_{f-n} are verifiable because P_f is verifiable (it holds a path to the signature packet).

(b) *Random target packets:* in this step, the hash of the current packet is embedded within $k-1$ subsequent packets chosen *randomly*. What motivates this *random hash-chaining* is the results of Perrig et al. (2000) that show that random hash-chaining allows to reach high verification ratio in bursty packet loss model.

This combination of deterministic with random hash-chaining achieves better verification ratio compared with purely random hash-chaining. Indeed, Fig. 3 shows that *hybrid hash-chaining* resists better to packet loss: when the packet loss ratio reaches 40%, the hybrid scheme maintains a verification ratio greater than 90% while the purely random scheme drops to 60%.

Adaptive redundancy degree

In contrast to existing protocols (Perrig et al., 2000; Golle and Modadugu, 2001; Miner and Staddon, 2001), the *redundancy degree k* in our protocol is *adaptive* and depends on the *actual* packet loss ratio in the network. Indeed, using random hash-chaining, we were interested in looking for the best redundancy degree to be used to reach 99% of verification ratio depending on the packet loss ratio. Fig. 11 shows the required redundancy degree when the packet loss ratio varies from 5% to 60%. Since the required redundancy degree to reach 99% of verification ratio depends *proportionally* on the packet loss ratio (see Fig. 11), we suggest to exploit receivers' feedback 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 receivers to communicate to the sender

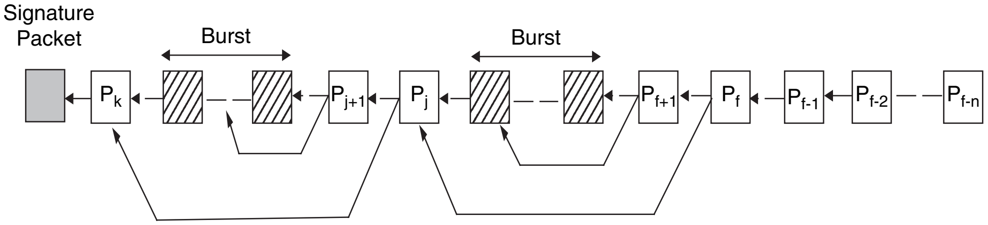

Figure 2 Hybrid hash-chaining impact on verification probability.

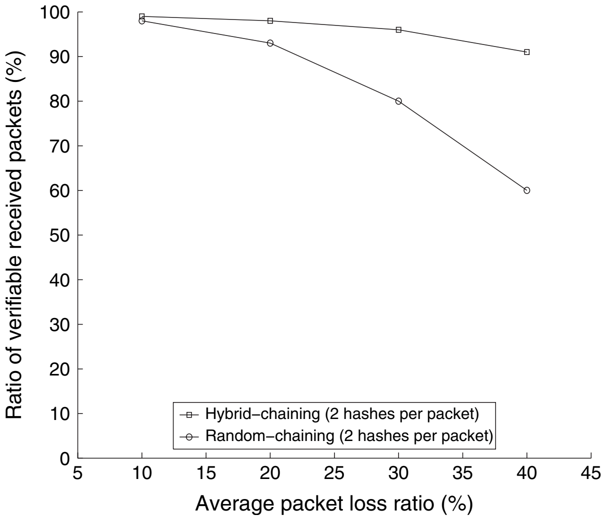

Figure 3 Robustness against packet loss: hybrid vs. only random hash-chaining.

the packet loss ratio in the network (for example by sending periodic RTCP (Schulzrinne et al., 2003) Receiver Reports). Relying on this receivers' feedback, the source decides what is the best redundancy degree to use in order to tolerate the actual packet loss ratio in the network.

H_2A protocol

Fig. 4 shows the different messages exchanged between a source of an authentic data-stream and a receiver. It illustrates also the periodic operations executed by the source and the receiver to adapt the redundancy degree and to verify the authenticity of received packets, respectively. Further explanations and algorithms are given in the following paragraphs. Table 1 summarizes the parameters involved in H_2A protocol. These parameters influence the computation and communication overhead, the delay until verification, and the robustness against packet loss. We want to achieve low overhead while retaining high robustness against packet loss.

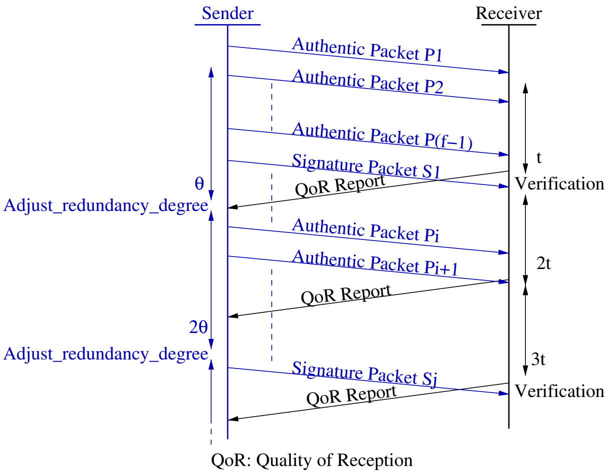

Figure 4 H_2A sequence diagram.

Table 1 H_2A parameters

(k)	The redundancy degree
(f)	Number of packets after which a signature packet is sent
(d)	The scope within which packets are chosen randomly to include the hash of a packet
(b)	The average length of bursts in a bursty packet loss pattern
(t)	The period of time after which the quality of reception reports are sent
(θ)	The period of time after which the source analyses the received quality of reception reports to update the redundancy degree (k)
(v)	The desired verification ratio of received packets

The sender side

A source of a stream applies the hash-chaining technique described above for each packet P_i before it sends it. Fig. 5 shows an example, where the redundancy degree is equal to 3. After each f data packets, the source sends a signature packet. These periodic signature packets allow to assure continuous non-repudiation of the stream. Besides, since the verification process depends on the reception of signature packets, the source replicates signature packets so that their loss probability is very low. After each period of time θ (seconds), the source analyses the received quality of reception reports and adjusts the redundancy degree k accordingly to maintain the desired verification ratio v. The algorithm at the source would then be as shown in Fig. 6. The *adjust_redundancy_degree* function determines the best redundancy degree k to reach the desired verification ratio v given that packets may be lost in the network with an average ratio equal to *avg*.

The receiver side

When a receiver receives a signature packet S_l, it verifies the signature of S_l and verifies the authenticity of all the packets that have a path to S_l. After each t (seconds), the receiver sends to the source of the stream a quality of reception report including the packet loss ratio during the last t (seconds). The algorithm at the receiver side is

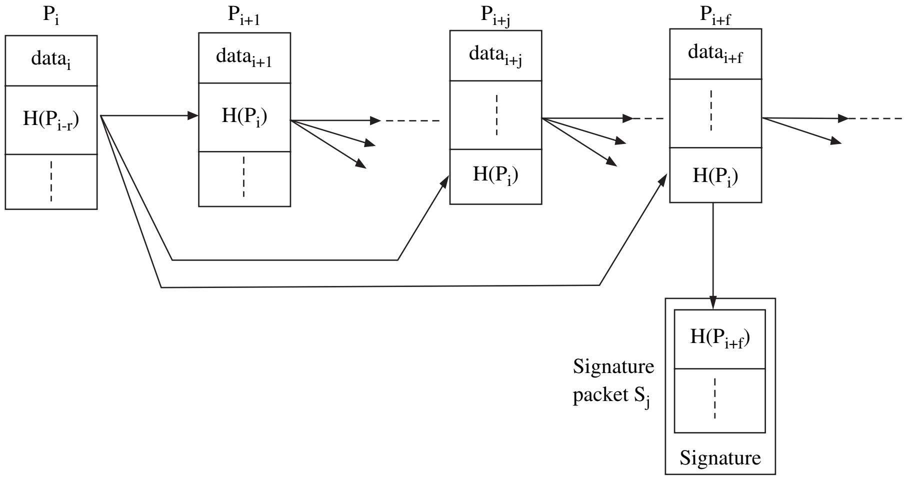

Figure 5 *H_2A* hash-chaining example.

shown in Fig. 7 and the verification procedure is illustrated in Fig. 8.

Simulations and performance evaluation

We carried out simulations using NS-2 to evaluate the performance of *H_2A* and compare it with EMSS (Perrig et al., 2000), as well as with an *H_2A* version which uses only random hash-chaining. Hereafter, we call this version Only Adaptive Protocol (OAP), and use it to illustrate the added value of the proposed hybrid hash-chaining.

The bursty packet loss model

We used the two-state Markov chain model (Yajnik et al., 1999) to extend NS-2 with a new queuing behavior to simulate a bursty packet loss pattern. Indeed, many studies show that packet loss is correlated, which means that the probability of loss is much higher if the previous packet is lost. Paxson (1999) shows that packet loss is correlated and that the length of losses exhibits infinite variance. Borella et al. (1998) found that the average length of loss bursts is about 7 packets. Yajnik et al. (1999) show that a k-state Markov chain can model Internet packet loss patterns. For our simulation purposes, the two-state Markov chain model is sufficient, since it can correctly model simple patterns of bursty packet loss (Yajnik et al., 1999). Fig. 9 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.

Simulation parameters

In what follows, we consider a bursty packet loss pattern with bursts having an average length equal to 7 (Borella et al., 1998). Then, we considered a stream of 20,000 packets with a signature packet

```
for each packet $P_i$ do
    include $H(P_i) = h_i$ in the packet $P_{i+1}$;
    do $k - 1$ times
        generate a random number $j$ so that $j \in [i+2, i+d]$;
        include $H(P_i) = h_i$ in the packet $P_{i+j}$;
    end.
    send $P_i$;
end.

after each $f$ packets do
    sign the current packet $S_l$;
    send the signature packet $S_l$;
end.

upon timeout do
    compute the average packet loss ratio
      from the received quality of reception reports: avg;
    k=adjust_redundancy_degree(avg, v);
    schedule timeout after $\theta$ seconds;
end.
```

Figure 6 The algorithm at the source side.

```
do
    receive packet P.
    if P is not a signature packet then
        buffer P;
        buffer hashes included in P;
    else
        /* P is a signature packet */
        verify(P);
    end
while(true).

upon timeout do
    generate a quality of reception report $R$
        including the packet loss ratio;
    send $R$;
    schedule timeout after $t$ seconds;
end.
```

Figure 7 The algorithm at a receiver side.

```
verify(P)
if P is a signature packet then
    verify the signature of P;
    if the P's signature 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.
```

Figure 8 The recursive verification procedure.

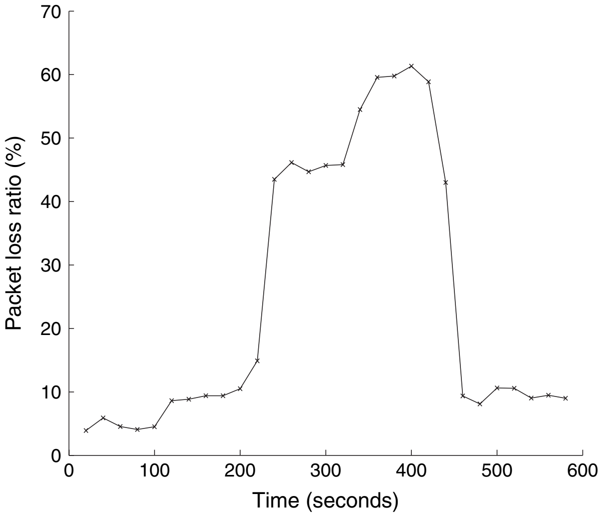

Figure 10 The considered scenario of packet loss ratio variation over time.

every 500 packets ($f = 500$), and where a packet is *hash-linked* to packets within the scope of 250 packets ($d = 250$). The value of f has been arbitrarily chosen. In reality, the value of f should be chosen depending on the application level tolerance to latencies, the computation power of communicating parties and the available bandwidth. The general rule is: if the parameter f is long, then receivers will experience important latencies before verification but will not have too much signatures to verify, and the reduced number of signatures will not consume a lot of bandwidth. We developed our simplified RTP/RTCP version over NS-2. Receivers send quality of reception reports including the packet loss ratio every $\theta = 20$ s. We considered the distribution of packet loss ratio over time shown in Fig. 10. The overall average, packet loss ratio is 26%, but over time, it varies from 5% to 60%. We aim to reduce the bandwidth overhead (redundancy degree) while increasing the verification ratio.

Adaptation of redundancy degree

Recall that periodically, the source analyses quality of reception reports. Then the source adapts the redundancy degree accordingly using the *adjust_redundancy_degree* function. To develop this function, we run extensive simulations of our hybrid hash-chaining scheme by varying packet loss ratio from 5% to 60%, 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%). Fig. 11 illustrates the results. As we can see, the hybrid scheme minimizes the redundancy degree compared to the only random scheme while maintaining the same performance in terms of verification ratio. Hence, given an average loss ratio, our *adjust_redundancy_degree* function returns the minimum redundancy degree which guarantees a very high verification ratio (99%) according to

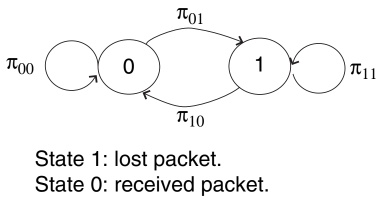

Figure 9 Two-state Markov chain to simulate bursty packet loss.

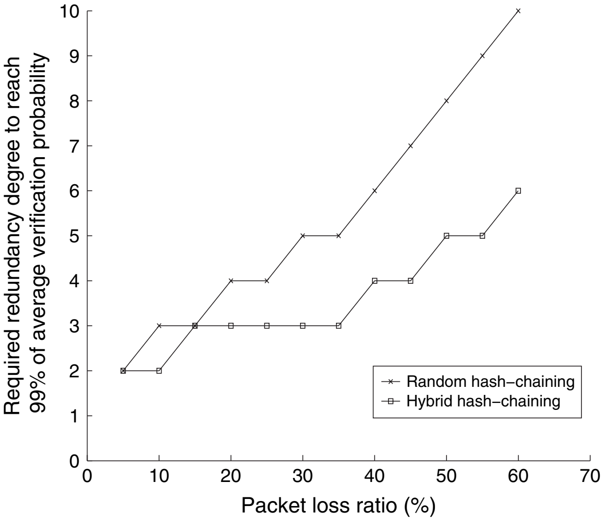

Figure 11 Required redundancy degree to reach 99% of verification ratio.

the results of these simulations. In other words, the graph (hybrid hash-chaining) depicted in Fig. 11 corresponds to the *adjust_redundancy_degree* function used by our protocol (H_2A).

Results

We considered a target verification ratio $v = 99\%$, and we run simulations of EMSS, H_2A and OAP to determine the required redundancy degree by each protocol in order to reach the target verification ratio. The results were illuminating: first, the redundancy degree of H_2A over time is obviously proportional to packet loss ratio (compare the shape of the graph representing the redundancy degree of H_2A in Fig. 12 with the shape of the graph representing the variation of packet loss ratio over time in Fig. 10). Besides, we found that H_2A reaches 99% of verification ratio with only an average of 3.35 hashes per packet, whereas EMSS requires 6 hashes per packet to reach the same verification ratio and OAP requires 4.5 hashes per packet (see Fig. 13).

This means that H_2A allows to save up to 2.65 hashes per packet. If we consider a hash algorithm that produces a 20 byte hash code (such as SHA-1 Eastlake and Jones, 2001), this means that H_2A saves up to 1 Mbytes of authentication information while sending the 20,000 packets of the stream. In other words, H_2A allows to save up to 44% of the authentication information used by EMSS.

Then we were interested in the impact of the packet loss ratio on the verification ratio of received packets. Fig. 14 shows that H_2A resists better to packet loss compared to EMSS and OAP.

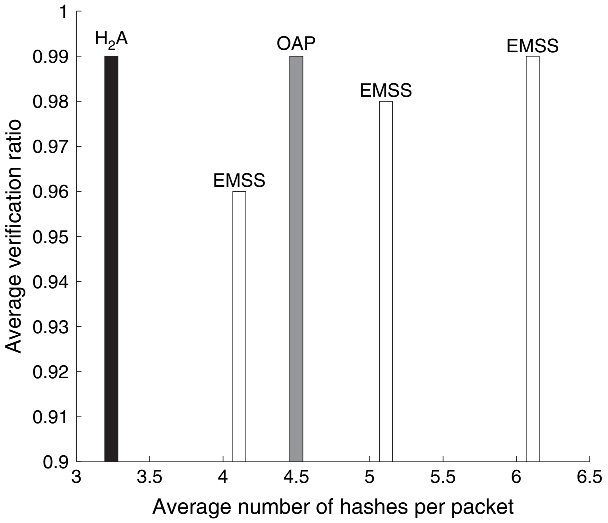

Figure 13 Verification efficiency depending on redundancy degree.

Indeed, when the packet loss ratio reaches 50%, H_2A maintains a verification ratio greater than 98% while EMSS drops to 91% and OAP to 95%. Furthermore, notice that H_2A and OAP use only 3.5 hashes per packet in the average, while EMSS uses 4 hashes per packet. We also notice, that even if EMSS has a greater verification ratio compared to OAP, when the packet loss ratio varies from 10% to 40%, the latter one (OAP) resists better to very high packet loss (50%). Indeed, this is due to the ability of OAP to adapt the *redundancy degree* in order to resist to such a high packet loss ratio.

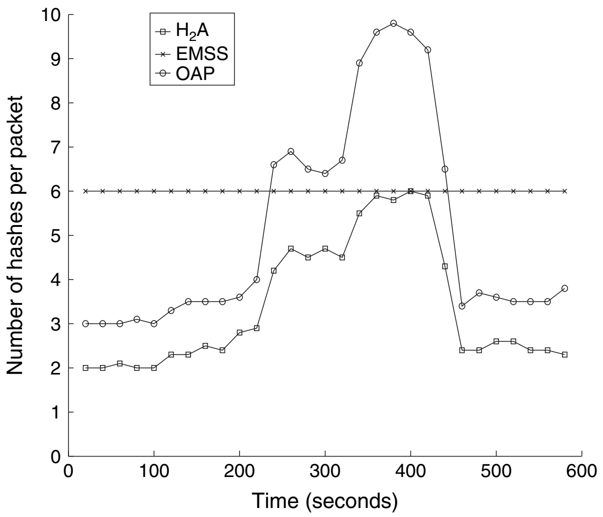

Figure 12 The variation of the required redundancy degree to reach 99% of verification ratio.

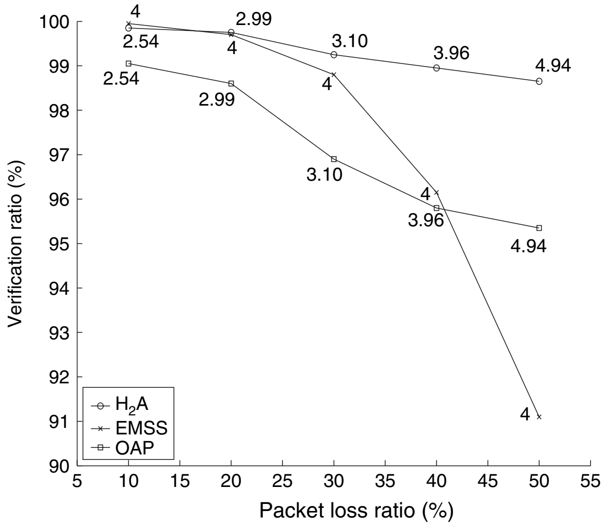

Figure 14 Verification efficiency depending on packet loss ratio: the numbers beside the points represent the average *redundancy degree*.

Table 2 Comparison of some *multicast data source authentication with non-repudiation* protocols

Protocol	Latency at the source	Latency at receivers	Tolerance to packet loss	Authentication information size
Simple off-line chaining	Yes	No	No	$\|d\|$
EMSS	No	Yes	The authentication probability of a packet is at least 90%	$6\|d\|$
Augmented chains $C_{a,p}$	Yes	Yes	Yes: Each block of packets tolerates a single burst of length up to $p(a-1)$, where p is the number of packets buffered by the source, and $p+a-1$ is the number of hashes buffered by the source	$2\|d\|$ in average
Piggybacking	Yes	Yes	Yes: Each prioritized packets' set S_i tolerates x_i bursts of b_i packets (x_i and b_i being input parameters)	$\|d\| \times (x_i+1)$ at least, for a packet in class S_i
H_2A	No	Yes	Yes: 99% average verification ratio	*Source driven*: depends on average packet loss ratio faced by receivers

$|d|$: Size of a digest (hash).

H_2A security and performance comparison

H_2A guarantees data source authentication and non-repudiation by relying on the existence of *hash-chains* between data packets and *signature packets*. Hence, the security of our protocol (H_2A) relies on the security of this basic technique (hash-chains), which has been proved to be secure by Gennaro and Rohatgi (2001). We have shown in previous sections that H_2A reduces the amount of *authentication information* while maintaining good performance in term of *robustness against packet loss*. However, there are some other performance criteria of H_2A that should be discussed. Table 2 compares H_2A to some data source authentication with non-repudiation protocols, described in the related works section, with respect to the following criteria:[1]

1. *The latency at the sender*: corresponds to the fact that the sender needs to buffer packets before sending them.
2. *The latency at a receiver*: corresponds to the fact that a receiver needs to buffer packets before verifying their authenticity.
3. *Tolerance to packet loss*: corresponds to the fact that the authentication process is possible even if some packets are lost.
4. *Authentication information size*: the size of the authentication information embedded to a packet.

H_2A computation overhead

H_2A improves multicast data source authentication performance by saving useless authentication information, and hence reduces the required bandwidth overhead. In what relates to computation overhead, H_2A requires only a single hash computation per packet in addition to a single digital signature computation after each period of f packets. f depends on the maximum delay that the application level can tolerate at the receiver side. Table 3 illustrates speed measurement of some famous hash functions when considering different implementations.

In conclusion, simulations show that H_2A 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

[1] With EMSS, we consider results of the special case simulated by authors.

Table 3 Speed measurement of hash functions

	Implementation	MD5	SHA-1
Pentium 4, 2.1 GHz (Dai, 2003)	Software	204.55 Mbps	72.60 Mbps
FPGA (Sklavos et al., 2003)	Hardware	2.1 Gbps	2.3 Gbps

experience some delay before verification of received packets. Scalabilty is not a concern since the number of hash embedments within each packet is independent of the number of multicast group members. H_2A computation overhead is reduced to a single hash computation per packet in addition to a periodic digital signature computation. Besides, H_2A computation efficiency can be further enhanced when considering hardware implementations of the used hash and digital signature algorithms.

Conclusion

Data source authentication is a required component in the whole multicast security architecture. Besides, many applications need non-repudiation of data-streams. To achieve non-repudiation, we proposed a new adaptive and efficient protocol called H_2A. Our protocol uses a hybrid and adaptive hash-chaining technique to amortize a single digital signature over many packets. This H_2A's hash-chaining technique allows to save bandwidth and improves the probability that a packet be verifiable even if some packets are lost.

Simulation results using NS-2 show that our protocol resists to bursty packet loss and assures with a high probability that a received packet be verifiable. Besides, the simulations and comparisons with other protocols show that our adaptive hash-chaining technique is more efficient than hash-chaining techniques that do not 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 *hybrid hash-chaining* technique allows to maintain high robustness to packet loss.

References

Bergadano F, Cavagnino D, Crispo B. Individual authentication in multiparty communications. Computers and Security 2002; 21(8):719–35.

Boneh Dan, Durfee Glenn, Franklin Matt. Lower bounds for multicast message authentication. Eurocrypt'01; LNCS (2045); 2001. p. 437–452.

Borella M, Swider D, Uludag S, Brewster G. Internet packet loss: measurement and implications for end-to-end qos. International conference on parallel processing; August 1998.

Canetti Ran, Garay Juan, Itkis Gene, Micciancio Daniele, Naor Moni, Pinkas Benny. Multicast security: a taxonomy and efficient constructions. INFOCOM; 1999.

Challal Y, Bettahar H, Bouabdallah A. A taxonomy of multicast data origin authentication: issues and solutions. IEEE Communications Surveys and Tutorials. To appear in volume 6(3), 2004.

Challal Y, Bettahar H, Bouabdallah A. SAKM: a scalable and adaptive key management approach for multicast communications. ACM SIGCOMM Computer Communications Review April 2004;34(2):55–70.

Dai Wei. Comparison of popular cryptographic algorithms, <http://www.eskimo.com/~weidai/benchmarks.html>; 2003.

Deering SE. Multicast routing in internetworks and extended LANs. ACM SIGCOMM; August 1988.

Desmedt Yvo, Frankel Yair, Yung Moti. Multi-receiver/multi-sender network security: efficient authenticated multicast/feedback. IEEE INFOCOM'92; 1992. p. 2045–2054.

Eastlake D, Jones P. US secure hash algorithm 1 (SHA1); September 2001. RFC 3174.

Federal Information Processing Standards Publication. Digital signature standard (DSS); May 1994. FIPS PUB 186.

Fujii Hiroshi, Kachen Wattanawong, Kurosawa Kaoru. Combinatorial bounds and design of broadcast authentication. IEICE Transactions 1996;E79-A(4):502–6.

Gennaro Rosario, Rohatgi Pankaj. How to sign digital streams. Advances in cryptology, CRYPTO'97; 1997.

Gennaro Rosario, Rohatgi Pankaj. How to sign digital streams. Information and Computation February 2001;165(1):100–16.

Golle Philippe, Modadugu Nagendra. Authenticating streamed data in the presence of random packet loss. NDSS'01: the network and distributed system security symposium; 2001.

Hardjono Thomas, Tsudik Gene. IP multicast security: issues and directions. Annales de Telecom 2000.

Judge Paul, Ammar Mostafa. Security issues and solutions in multicast content distribution: a survey. IEEE Network January/February 2003:30–6.

Kaliski B. The MD2 message-digest algorithm; April 1992. RFC 1319.

Krawczyk H, Bellare M, Canetti R. HMAC: keyed-hashing for message authentication; February 1997. RFC 2104.

Miner Sara, Staddon Jessica. Graph-based authentication of digital streams. IEEE Symposium on Security and Privacy 2001.

Obana S, Kurosawa K. Bounds and combinatorial structure of (k, n) multi-receiver A-codes. Designs, Codes and Cryptography 2001;22(1):47–63.

Pannetrat A, Molva R. Efficient multicast packet authentication. 10th Annual network and distributed system security symposium; February 2003.

Park Jung Min, Chong Edwin KP, Siegel Howard Jay. Efficient multicast stream authentication using erasure codes. ACM Transactions on Information and System Security May 2003; 6(2):258–85.

Paxson Vern. End-to-end internet packet dynamics. IEEE/ACM Transactions on Networking June 1999;7(3):277—92.

Perrig A, Canetti R, Tygar JD, Song D. Efficient authentication and signing of multicast streams over lossy channels. IEEE Symposium on Security and Privacy 2000.

Perrig Adrian. The BiBa one-time signature and broadcast authentication protocol. The eighth ACM conference on computer and communications security; November 2001.

Perrig Adrian, Canetti Ran, Song Dawn, Tygar JD. Efficient and secure source authentication for multicast. Eighth annual internet society symposium on network and distributed system security; 2001.

Rafaeli Sandro, Hutchison David. A survey of key management for secure group communication. ACM Computing Surveys September 2003;35(3):309—29.

Rivest R. The MD5 message-digest algorithm April 1992. RFC 1321.

Rivest Ronald L, Shamir Adi, Adelman Leonard M. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 1978;21(2):120—6.

Safavi-Naini R, Wang H. Multireceiver authentication codes: models, bounds, constructions, and extensions. Information and Computation 1999;151:148—72.

Schulzrinne H, Casner S, Frederick R, Jacobson V. RTP: a transport protocol for real-time applications; July 2003. RFC 3550.

Shirey R. Internet security glossary; May 2000. RFC 2828.

Sklavos N, Alexopoulos E, Koufopavlou O. Networking data integrity: high speed architectures and hardware implementations. The International Arab Journal of Information Technology 2003;1(0):54—9.

Wong Chung Kei, Lam Simon S. Digital signatures for flows and multicasts. IEEE/ACM Transactions on Networking 7(4); August 1999.

Yajnik Maya, Moon Sue, Kurose Jim, Towsley Don.Measurement and modeling of the temporal dependence in packet loss. INFOCOM'99; March 1999. p. 345—52.

Yacine Challal is a Ph.D. student at the Department of Computer Engineering at the Compiegne University of Technology (UTC-France). He is member of the Networking Group at Heudiasyc Laboratory. He received his Master's degree in computer science (2002) at the Compiegne University of Technology, and the Engineering degree (2001) from the National Computer Science Institute (INI-Algiers-Algeria). He works with Professor A. Bouabdallah (UTC) on multicast security. His current research interests are group communication security, multicast routing, multimedia and QoS.

Abdelmadjid Bouabdallah received the Engineer Diploma in computer science from University Of Technology Of Algiers (USTHB) in 1986, and received the Master's (DEA) degree and Ph.D. from University of Paris-sud Orsay (France), respectively, in 1988 and 1991. From 1992 to 1996, he was Assistant Professor at University of Evry-Val-d'Essonne, France. Since 1996, he is Professor in the Department of Computer Engineering at University of Technology of Compiegne (UTC) where he is leader of Networking and Optimization Research Group. His research interest includes Internet Qos and security, unicast/multicast communication, and fault tolerance in wired/wireless networks and distributed systems.

Hatem Bettahar received the M.S. degree and Ph.D. degree in computer science for work on Multicast routing and Quality of Service in IP networks from the University of Technology of Compiegne (UTC), France in 1998 and 2001, respectively. Since 2001 he is Assistant Professor in the Department of Computer Engineering at the UTC. He is member of the networking and optimization research group within the Heudiasyc UMR-CNRS-6599 Laboratory. His research Interest includes Internet QoS routing, multicast communication, multicast security and mobile IP.