Efficient Multicast Stream Authentication Using Erasure Codes

JUNG MIN PARK
Purdue University
and
EDWIN K. P. CHONG and HOWARD JAY SIEGEL
Colorado State University

We describe a novel method for authenticating multicast packets that is robust against packet loss. Our focus is to minimize the size of the communication overhead required to authenticate the packets. Our approach is to encode the hash values and the signatures with Rabin's Information Dispersal Algorithm (IDA) to construct an authentication scheme that amortizes a single signature operation over multiple packets. This strategy is especially efficient in terms of space overhead, because just the essential elements needed for authentication (i.e., one hash per packet and one signature per group of packets) are used in conjunction with an erasure code that is space optimal. Using asymptotic techniques, we derive the authentication probability of our scheme using two different bursty loss models. A lower bound of the authentication probability is also derived for one of the loss models. To evaluate the performance of our scheme, we compare our technique with four other previously proposed schemes using empirical results.

Categories and Subject Descriptors: K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*authentication*

General Terms: Algorithms, Security

Additional Key Words and Phrases: Digital signatures, multicast, Information Dispersal Algorithm (IDA), erasure codes

1. INTRODUCTION

Fueled by the explosive growth of the Internet and growing demand for novel types of group communications, multicast has received a lot of attention in recent years. In multicast, a single copy of packets is sent by the sender and routed

A preliminary version of portions of this material was presented in Park et al. [2002].
This research was supported in part by the Center for Education and Research in Information Assurance and Security (CERIAS), by the Colorado State University George T. Abell Endowment, and by NSF under grants 0098089-ECS and 0099137-ANI.
Authors' addresses: J. Park, School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907; email: parkjm@ecn.purdue.edu; E. K. P. Chong and H. J. Siegel, Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523; email: {echong,hj}@colostate.edu.

to every receiver within the multicast group via multicast-enabled routers. For a wide range of applications, multicast is an efficient and natural way of communicating information. Some examples include information broadcasts (e.g., news feeds, weather updates, and stock quotes), multiparty videoconferencing, and software updates. For successful implementation, many of these applications will require varying degrees of security requirements (i.e., confidentiality and authentication).

Confidentiality for multicast transmissions can be provided using techniques that utilize symmetric (secret) key cryptography. Confidentiality would be provided by encrypting the message with the secret key being shared by the sender and the receivers of the multicast group before transmission. Consequently, off-the-shelf solutions such as the Advanced Encryption Standard (AES) can be readily employed for this purpose. For confidentiality, the main concern is the complexity involved in key management (e.g., key distribution, revocation, and group updates).

The solution to the authentication problem for unicast transmission is well known. For example, efficient hash-based message authentication codes (MAC) known as HMACs can be employed. However, this solution is inadequate for the multicast setting. The difficulty of the problem lies in the fact that preventing the forgery of packets by a colluding group of receivers precludes the use of any MAC-based scheme, if small communication (space) overhead is required, and time synchronization between the sender and the receiver is difficult to maintain. The rationale for this argument is given in Section 2.1. Without using symmetric key cryptosystems, the most obvious way of providing authentication is to sign each individual packet using the sender's digital signature. However, the computation overhead of current signature schemes is too high to make this practical. According to Wong and Lam [1998], a Pentium II 300 MHz machine devoting all of its processor time can only generate 80 512-bit RSA signatures per second. Clearly, this approach is not practical.

Packet loss is another important issue that needs to be considered. While this may not be a problem for applications using reliable transport protocols (e.g., TCP/IP), it is a serious issue for multicast applications that use UDP over IP-Multicast. Because UDP only provides best-effort service, packet loss can be high. Therefore, while the content being broadcast might be able to bear packet losses, the authenticity of a nonnegligible percentage of the received packets might not be verifiable by the receiver, if the authentication scheme is not robust against loss.

Techniques for reliable multicast exist, such as Scalable Reliable Multicast (SRM) [Floyd et al. 1997] and Reliable Adaptive Multicast Protocol (RAMP) [Koifman and Zabele 1996], but they are complex and not yet standardized. Within the Internet Engineering Task Force (IETF), the Reliable Multicast Transport (RMT) working group is chartered to standardize reliable one-to-many transport protocols for bulk data. Because it is difficult for a single protocol to meet the requirements of all multicast applications, the RMT working group is standardizing three protocol instantiations, one from each of the following three categories: (i) a NACK-based protocol, (ii) a tree-based ACK protocol, and (iii) an asynchronous layered coding protocol that uses forward error

correction (FEC). The Digital Fountain model [Byers et al. 1998] is a scheme that belongs to the third category [Luby et al. 2002], and provides reliable one-to-many transport of bulk data using Tornado codes [Luby et al. 1997]. The standardization of the reliable multicast protocols by the RMT working group is not yet complete. Moreover, dissemination of information via broadcast (opposed to multicast) protocols incurs loss, and hence packet loss remains as an important issue in the authentication of packet streams.

Our approach to multicast message authentication is based on the technique of signature amortization, that is, amortizing a single signing operation over multiple packets. Our technique is especially efficient in space overhead, because just the essential elements needed for authentication (i.e., one hash per packet and one signature per block of packets) are used in conjunction with an erasure code that is space optimal. Note that our approach is similar to the RMT working group's third category of protocol instantiation with the difference that we are using FEC techniques to encode authentication information and not the data itself.

We also show how our approach can readily be extended to combat denial-of-service (DoS) attacks, where the attacker inserts bogus packets into the message stream. This problem is raised in Luby et al. [2002] within the context of reliable multicast protocols. One solution is to implement additional functionality into the multicast router or the receiver's computer so that the source IP address can be screened against a list of authentic transmitter addresses. This solution can be easily defeated, however, if the attacker has the capability to spoof source addresses. The techniques discussed in Section 6 avoid this weakness.

In the next section, we briefly discuss related work and give an overview of erasure codes. The rationale for our approach, along with the detailed authentication/verification procedure, is given in Section 3. In Section 4, we derive the asymptotic authentication probability of our scheme using two different bursty loss models. A lower bound of the authentication probability is also derived for one of the loss models. In Section 5, we evaluate the performance of our technique, comparing it with four other previously proposed schemes. Techniques for thwarting DoS attacks are given in Section 6. Finally, concluding remarks are given in Section 7.

2. TECHNICAL BACKGROUND

2.1 Related Work

Multicast authentication schemes can be divided into two major classes—unconditionally secure authentication and computationally secure authentication. Pioneering research on unconditionally secure authentication was done by Simmons [1984] and later extended to the multicast setting by Desmedt et al. [1992]. As the name suggests, unconditionally secure authentication provides very strong security guarantees, but is less practical than computationally secure techniques. Our focus is on computationally secure methods.

In computationally secure multicast authentication, two approaches can be taken. One approach is to use Message Authentication Codes (MAC), and the

other is to utilize digital signatures. Schemes based on MACs do not require any computationally intensive computations. The *asymmetric MAC* scheme proposed by Canetti et al. [1999] and the *Timed Efficient Stream Loss-tolerant Authentication* (*TESLA*) proposed by Perrig et al. [2000] fall into this category. The asymmetric MAC scheme is vulnerable to collusion attacks and is not scalable, because the size of the authentication information increases as the number of receivers increases. Using TESLA, packet streams can be authenticated with a small communication overhead, but this method requires time synchronization between the sender and the receivers, which might not be feasible in large multicast groups. In their recent work, Boneh et al. [2001] showed that one cannot build an efficient (in terms of communication overhead) collusion resistant multicast authentication scheme solely based on MACs. Although TESLA is resistant to collusion attacks and its communication overhead is small, the result of Boneh et al. is still valid, because TESLA uses a digital signature (e.g., RSA signature) to "bootstrap" the first packet of an authentication chain.

If MAC-based techniques are not employed, the obvious alternative is to use digital signatures. The biggest challenge in using digital signatures for authentication is the computationally intensive nature of the asymmetric-key-based signatures. For this reason, previous authentication schemes approached this problem in two directions—designing faster signature techniques and amortizing a signature operation over multiple packets.

In general, making the signature scheme faster comes at the cost of increased space overhead. Rohatgi [1999] proposes using a combination of k-time signatures and certificates for the k-time public keys (created with a regular signature scheme) to authenticate packets. Despite its improvement over the one-time signature scheme, this method still requires a space overhead on the order of several hundred bytes per packet.

Perrig [2001] proposes a one-time signature scheme called BiBa, which has two important advantages over regular signature schemes—its signature generation and verification times are significantly faster. This implies that each packet can be individually signed (via a BiBa signature) at a rate that is fast enough to match the packet transmission rate. By individually signing each packet, there is no need to buffer packets for authentication or verification, and thus reducing delay. Hence, for applications such as the broadcast of stock quotes, which require the authentication of real-time data, the broadcast authentication protocol based on the BiBa signature scheme might be appropriate. However, BiBa also has limitations. Its security is dependent on the time synchronization maintained between the sender and the receivers, and the maximum allowable time synchronization error between the sender and the receivers, which we denote as τ, limits the authentication rate. Because τ has to be set at a feasible value, there is a practical limitation on the authentication rate. For applications that require the sender to authenticate packets that have relatively fast transmission rates (e.g., video streams), this can be a problem. Another point to consider is the size of the communication overhead created by the BiBa authentication protocol. With reasonable parameter values, each BiBa signature size is 100–200 bytes, and a new set of public keys, whose size is on the order of 10 KB, needs to be transmitted at regular intervals.

Many schemes based on signatures attempt to reduce the computation and communication overhead by amortizing a single signature operation over multiple packets. For example, Wong and Lam [1998] employ Merkle's *authentication trees* [Merkle 1989] to reduce the size of the authentication information and sign multicast packets. However, because each packet is individually verifiable, every packet needs to contain a signature with the authentication information, which requires a large communication (space) overhead.

If individual packet verification is relaxed so that the verification of a packet is dependent on other packets, then the communication overhead can be reduced substantially. In this approach, verification of each packet is not guaranteed and instead is assured with a certain probability. The scheme proposed by Perrig et al., called *Efficient Multi-chained Stream Signature* (*EMSS*) [Perrig et al. 2000] takes this approach and uses a combination of hash functions and digital signatures to authenticate packets. EMSS is an extension of Gennaro and Rohatgi's *stream signing* technique [Gennaro and Rohatgi 1997]. The basic idea is as follows: A hash of packet P_1 is appended to packet P_2, whose hash is in turn appended to P_3. If a *signature packet*, containing the hash of the final data packet (i.e., P_3) along with a signature of the hash, is sent after P_3, then nonrepudiation is achieved for all three packets. In essence, hash values act as chains (or directed edges) between the packets so that they form a single string that can be signed by one digital signature. To reduce the verification delay, a stream of packets is divided into blocks, and the above process is repeated for every block. EMSS achieves relatively high verification rates at the cost of increased space overhead and delayed verification. Throughout the paper, we will use the term *verification rate* to denote the number of verifiable packets of the stream divided by the number of received packets of the stream.

Golle and Modadugu [2001] propose a scheme similar to EMSS called the *augmented chain* technique. They propose a systematic method of inserting hashes in strategic locations so that the chain of packets formed by the hashes will be resistant to a burst loss. The chain of packets constructed using this method is parameterized by the integer variables a and p; a chain of packets can sustain a burst loss of up to $p(a - 1)$ packets in length. To reduce the verification delay, a stream is divided into blocks of packets, and each block is constructed using the augmented chain technique with only the last packet in the block being signed.

In Miner and Staddon [2001], the authors propose a similar authentication scheme, based on hash chaining techniques, specifically designed to resist multiple bursts. In the construction of their scheme (called the *piggybacking* scheme), n packets are partitioned into r equal-sized priority classes. The signature packet is the first packet in the highest priority class. It is assumed that the packets in the highest priority class are spaced evenly throughout the block so that two consecutive packets of the highest priority class are located exactly r packets apart. By constructing the hash chains using the piggybacking scheme, packets in the ith priority class can tolerate x_i bursts of size at most $b_i = k_i r$, where k_i is a parameter dependent on the configuration of the hash chains.

In their recent work, Pannetrat and Molva [2002] propose a stream authentication scheme that is similar to our approach (initially published in

Park et al. [2002]). They propose to authenticate real-time data streams by piggybacking the current block's encoded authentication information (via an erasure code) onto the next block. Using this method, no packets need to be buffered by the sender but, of course, additional buffering is needed at the receiver. The same technique can readily be applied to our authentication scheme.

2.2 Erasure Codes

When a stream of packets is sent via the Internet, a fraction of the packets is lost during transit. A standard solution to this problem is to retransmit data that are not received. In some applications, this solution is not practical. An alternative solution is to use forward error correction (FEC) techniques. Among FEC codes, *erasure codes* are of particular interest to our application. We briefly review the basic characteristics of two well-known erasure codes—*Information Dispersal Algorithm* (*IDA*) [Rabin 1989] and *Tornado codes* [Luby et al. 1997].

IDA was originally developed as an algorithm for providing reliable storage or transmission of information in distributed systems. The basic idea of IDA is to process the source, say a file F, by introducing some amount of redundancy and splitting the result into n pieces, which are then transmitted. Reconstruction of F is possible with any combination of m pieces, where $m \leq n$. Each distributed piece is of size $|F|/m$, which clearly shows that the scheme is space optimal. The space overhead can be controlled by adjusting the parameters n and m; ratio n/m determines the space overhead incurred by the encoding process.

Unlike IDA, Tornado codes can encode and decode data in linear time. The number of segments needed for decoding is slightly larger than the number of preencoded segments, and thus they are suboptimal in terms of space overhead. Specifically, for a set of n segments, encoding with Tornado codes increases the number to $n/(1 - p(1 + \varepsilon))$, where $0 < p < 1$ and $0 < \varepsilon < 1$. If the receiver acquires more than $1 - p$ fraction of the encoded segments, then the original data segments can be reconstructed with high probability in time proportional to $n \ln(1/\varepsilon)$.

3. OUR SCHEME FOR STREAM AUTHENTICATION

3.1 Rationale for our Approach

In EMSS, there are three factors that affect the verification rate—number of signature packets, number of hashes contained in the signature packet, and number of hashes contained in the data packet. The number of hashes in the signature and data packets is always greater than one, improving the robustness to packet loss at the cost of increased overhead. This tradeoff is unavoidable, but can be done more efficiently using erasure codes. This is best illustrated using the following simple example.

The sender transmits the hash of a packet appended to k other packets for increased resistance to loss. We assume that a block consists of n packets and packet loss is independent. The probability that at least one out of the k packets will reach the destination is $1 - q^k$, where q is the loss probability. The space overhead would be $k \cdot h$, where h is the size of the hash. Using the

same overhead, one can encode the hash using IDA and append the n encoded segments to n packets. The minimum number of encoded segments needed for reconstruction of the hash is only $m = \lceil n/k \rceil$, where $\lceil x \rceil$ denotes the smallest integer not less than x. The probability that the hash can be reconstructed successfully at the receiver is given by

$$1 - \sum_{i=0}^{m-1} \binom{n}{i} (1-q)^i q^{n-i}. \tag{1}$$

Instead of using IDA, the sender could also use probabilistic codes such as Tornado codes. In this case, the minimum number of segments needed for decoding is

$$m = n(1-p) = \left\lceil \frac{n}{k} \left(1 + \frac{\varepsilon(k-1)}{\varepsilon+1} \right) \right\rceil, \tag{2}$$

where p and ε are the performance parameters of the Tornado code (see Section 2.2). The probability of successful reconstruction of the hash is given by (1) using the value of m specified by (2). It is obvious that the probability given by (1) is higher than $1 - q^k$, and the probability for IDA is higher than that for the Tornado code.

The above example suggests that using an erasure code to encode the hash values would be more efficient than simply appending duplicate hashes to the packets. As an extended version of EMSS, Perrig et al. [2000] suggest using universal hash functions or IDA to split the packet's hash value into multiple pieces before appending them onto other packets. This certainly produces a more loss-resistant scheme with the same amount of communication overhead. However, it introduces complexities—the time-consuming process of encoding and decoding must be performed for each hash. This can be a bottleneck, especially when multiple hashes are used per packet. We suggest a simple method of avoiding this problem at the cost of sender delay. Instead of encoding individual hashes, we suggest concatenating all the hashes within the block to form a single piece before encoding. This strategy requires only one encoding/decoding per block.

The main advantage of EMSS is that there is no sender delay incurred by the authentication process—a given packet can be transmitted immediately after its hash value is computed without the need to buffer other packets. This can be an advantage in situations where data are generated in real time, and immediate dissemination is crucial. However, for some multicast applications, the sender has a priori knowledge of at least a portion of the data (e.g., prerecorded news footage) and some sender delay is tolerated. In fact, most authentication schemes incur some degree of sender delay [Golle and Modadugu 2001; Miner and Staddon 2001; Wong and Lam 1998].

If a certain amount of sender delay is allowed, then a more significant problem can be addressed. It is obvious that the delivery of the signature packets is crucial for any authentication scheme. In previous work [Golle and Modadugu 2001; Miner and Staddon 2001; Perrig et al. 2000], performance results (both analytical and empirical) were based on the assumption that the signature packets are received. The authors suggest accomplishing this task

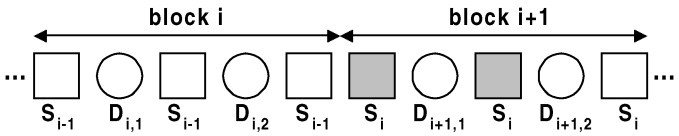

Fig. 1. Receiver delay caused by sending multiple copies of the signature packet.

by empowering the receivers to request retransmissions of the lost signature packets or sending multiple copies of the signature packet. However, the retransmission of signature packets can put considerable strain on the resources of the sender and the network, especially in large multicast networks. Yajnik et al. [1996] observe packet loss characteristics of actual multicast sessions, and show that considerable amounts of the packets would need to be retransmitted, if reliable multicast services are to be provided through retransmissions. In one particular data set, 62.6% of the packets was lost by at least one receiver. This implies that retransmission would have been necessary for 62.6% of the packets.

Sending multiple copies of the signature packet can be an alternate solution, but this also has drawbacks. Signature packets are generally large (e.g., 128 bytes, if 1024-bit RSA is used), and sending these packets several times can increase the communication overhead noticeably. Moreover, because actual losses in the Internet are highly correlated and bursty, each copy of the signature packet would have to be interspersed uniformly among the packets to ensure maximum effectiveness. If copies of the signature packets are distributed in the current block, then this would cause sender delays in schemes that utilize hash-chaining techniques with edges directed backwards (i.e., hash of a packet is appended to the packets following it)—schemes such as EMSS. The sender delay is incurred because data packets in the block cannot be transmitted before the replicas of the signature packet are interspersed among the data packets.

The obvious alternative is to distribute the copies in the next block to avoid the sender delay. However, this can cause increased delay on the receiver side—a receiver might have to buffer a maximum of $2n$ data packets before verifying a given packet, where n is the number of data packets per block. This case is illustrated as a simple example in Figure 1. In the figure, circles and squares represent data packets and signature packets, respectively. The first two signature packets in the $(i+1)$th block are assumed to be lost and are represented as darkened squares. The receiver needs to buffer $D_{i,1}$, $D_{i,2}$, $D_{i+1,1}$, and $D_{i+1,2}$ before verifying the data packets of the ith block (i.e., $D_{i,1}$ and $D_{i,2}$) using S_i (signature packet of the ith block).

Considering these problems, the obvious alternative is to apply FEC techniques to the signature packets. We can easily make the signature packets robust against packet loss by using erasure codes and appending each encoded piece to the data packets. For our authentication scheme, we employ IDA instead of probabilistic codes such as Tornado codes. Tornado codes can encode/decode data very rapidly (i.e., linear time), but do so with a high probability only when the number of segments to encode is large. For this reason,

Tornado codes are appropriate when a large number (hundreds to thousands) of segments are being encoded [Weatherspoon 2001]. We use IDA because the encoding involves a fairly small number of segments, and IDA has the added advantage of being space optimal. It should be noted that our authentication scheme is independent of the type of erasure code, and other erasure codes (e.g., Tornado codes) that have other attractive properties can be employed.

3.2 Signature Amortization Using IDA

To reduce the computation burden of signing each packet, two approaches can be taken—designing faster signature techniques and amortizing a single signature operation over multiple packets. Our main focus is reducing the size of the authentication overhead, and therefore we took the second approach, which offers better space efficiency. We propose a scheme that employs IDA to amortize a single digital signature over a block of packets. Our scheme, appropriately named *Signature Amortization using IDA* (*SAIDA*), is designed to provide space-efficient authentication even in high packet-loss environments. The following steps describe the authentication procedure in detail:

1. Let $\|$ denote concatenation. A stream of packets is first divided into groups (or blocks). We denote a stream as $\Gamma = G_1 \| G_2 \| \ldots$, where each group G_i is a concatenated string of n packets (i.e., $G_i = P_{(i-1)n+1} \| \cdots \| P_{in}$), and each packet $P_i \in \{0,\ 1\}^\kappa$ for some constant κ. The same operations are performed on every group, so we will only focus on the first group.
2. A *packet hash* $H(P_i),\ i = 1, \ldots, n$ for each packet is computed using a hash function H.
3. The packet hashes are concatenated to form $F^1 = H(P_1) \| \cdots \| H(P_n)$ of size N (i.e., F^1 consists of N characters). Let c_i represent the ith character in F^1. In practice, c_i may be considered as an eight-bit byte, and all calculations are done in $\mathbf{Z}_{257}$ or $GF(2^8)$. One copy of F^1 is stored in a buffer while another copy is divided into blocks of length m as follows:
$$F^1 = (c_1, \ldots, c_m), (c_{m+1}, \ldots, c_{2m}), \ldots, (c_{N-m+1}, \ldots, c_N).$$
To simplify the following discussion, let $\boldsymbol{S}_i = (c_{(i-1)m+1}, \ldots, c_{im}), \quad 1 \le i \le N/m$.
4. Choose n vectors $\boldsymbol{a}_i = (a_{i1}, \ldots, a_{im}),\ 1 \le i \le n$, such that for every subset of m different vectors are linearly independent, as specified in Rabin [1989].
5. Using vectors $\boldsymbol{a}_i = (a_{i1}, \ldots, a_{im})$, $1 \le i \le n$, F^1 is processed and divided into n segments as
$$F_i^1 = (\boldsymbol{a}_i \cdot \boldsymbol{S}_1,\ \boldsymbol{a}_i \cdot \boldsymbol{S}_2, \ldots, \boldsymbol{a}_i \cdot \boldsymbol{S}_{N/m}), \quad i = 1, \ldots, n,$$
where $\boldsymbol{a}_i \cdot \boldsymbol{S}_k = a_{i1} \cdot c_{(k-1)m+1} + \cdots + a_{im} \cdot c_{km}$. It follows that $|F_i^1| = |F^1|/m$.
6. The *group hash* is computed by taking the hash of the other copy of F^1, that is, $H_G(G_1) = H(F^1) = H(H(P_1) \| \cdots \| H(P_n))$, where $H_G(G_1)$ denotes the group hash of the first group of packets.
7. The group hash is signed by a digital signature scheme using the sender's private key K_r and denoted as $\sigma(K_r,\ H_G(G_1))$. This value is IDA encoded using the same set of vectors to yield $\sigma_1(K_r, H_G(G_1)), \ldots, \sigma_n(K_r, H_G(G_1))$.

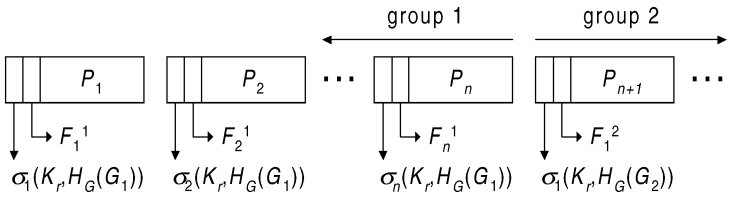

Fig. 2. Authenticated packet stream.

Although this procedure was explained as a separate step for clarity, the signature can be concatenated with F^1 before applying IDA, so that only one encoding procedure per group is necessary.

8. Each signature segment (created in the seventh step) and hash segment (created in the fifth step) are concatenated with the corresponding packet to form an authenticated packet. A group of n authenticated packets combine to form an authenticated group, which is expressed as

$$\sigma_1(K_r,\ H_G(G_1))\|F_1^1\|P_1, \ldots, \sigma_n(K_r,\ H_G(G_1))\|F_n^1\|P_n.$$

An instance of an authenticated packet stream is illustrated in Figure 2. The verification of the stream is straightforward. Assuming that at least m packets are received, the receiver can successfully reconstruct F^1 and $\sigma(K_r,\ H_G(G_1))$ from any combination of m packets as follows:

1. Assume that segments $F_1^1, \ldots, F_m^1$ are received. Using the m pieces, it is readily seen that

$$\mathbf{A} \cdot \begin{bmatrix} c_1 \\ \vdots \\ c_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{a}_1 \cdot \boldsymbol{S}_1 \\ \vdots \\ \boldsymbol{a}_m \cdot \boldsymbol{S}_1 \end{bmatrix},$$

where $\mathbf{A} = (a_{ij})_{1 \le i,j \le m}$ is an $m \times m$ matrix whose ith row is $\boldsymbol{a}_i$.

2. Because $\mathbf{A}$ is invertible, $\boldsymbol{S}_1$ can be obtained from

$$\boldsymbol{S}_1 = \begin{bmatrix} c_1 \\ \vdots \\ c_m \end{bmatrix} = \mathbf{A}^{-1} \cdot \begin{bmatrix} \boldsymbol{a}_1 \cdot \boldsymbol{S}_1 \\ \vdots \\ \boldsymbol{a}_m \cdot \boldsymbol{S}_1 \end{bmatrix}.$$

3. Using the same procedure, $\boldsymbol{S}_2, \ldots, \boldsymbol{S}_{N/m}$ can be obtained, and F^1 is reconstructed by concatenating these values.
4. The same technique is applied to reconstruct $\sigma(K_r,\ H_G(G_1))$.
5. All the packets in G_1 can be verified using F^1 and $\sigma(K_r,\ H_G(G_1))$.

For SAIDA, the trade-off between verification rate and communication overhead can be readily governed by changing the parameters n (number of encoded segments) and m (minimum number needed for decoding). Note that the space overhead (determined by n/m) only applies to the authentication information (i.e., hashes and signatures) and not to the data itself. Specifically, $(n/m)M$ represents the space overhead incurred after the IDA encoding process, where M is the size of the authentication information.

4. AUTHENTICATION PROBABILITY

4.1 Loss Models

It is well known that actual packet losses in a network are bursty rather than independent in nature. In this section, using asymptotic techniques, we derive the *authentication probability* of SAIDA using two different bursty loss models. We define the authentication probability as $\Pr\{P_i \textit{ verifiable} \mid P_i \textit{ is received}\}$, where P_i represents the ith packet of a block. This definition is adopted from Miner and Staddon [2001]. Note that this is different from the *verification rate* referred to in Section 2.1. We use this term to denote the number of verifiable packets divided by the number of received packets of a stream. In the simulation experiments of Section 5, we use verification rate as the performance measure, because its value can be directly calculated from the simulation data.

The authentication probability is directly affected by the loss behavior of the network. In Yajnik et al. [1999], it is shown that the 2-state Markov chain can accurately model bursty loss patterns in certain cases, and hence we adopt this model as one of our loss models. Throughout the paper, we denote this model as the *2-state Markov Chain* (*2-MC*) loss model.

In Miner and Staddon [2001], the authors derive an authentication probability lower bound (for the piggybacking scheme) based on a bursty loss model, which is motivated by ideas in the theory of error-correction codes. We denote this model as the *Biased Coin Toss* (*BCT*) loss model. We choose this model for our analyses in Sections 4.2 and 4.3 because it facilitates the direct comparison of the authentication probability lower bound between SAIDA and the piggybacking scheme.

For fixed n and m, finding the analytical expression for the authentication probability seems to be extremely difficult, and hence we use asymptotic techniques in our analyses. In the analyses, results from *renewal theory* are applied.

4.2 Asymptotic Authentication Probability under the 2-State Markov Chain Model

The 2-MC loss model is defined as follows:

4.2.1 *2-MC Loss Model.* The loss process is modeled as a discrete-time Markov chain with two states—0 and 1—representing no loss and loss, respectively. It is defined by the four transition probabilities (i.e., p_{00}, p_{11}, p_{01}, and p_{10}). The stationary probabilities (the long-run proportion of transitions that are into a given state) are denoted as π_0 and $\pi_1 = 1-\pi_0$. The expected burst-loss length β, and probability of loss q can be expressed using the four parameters of the Markov chain.

We represent this loss process as a discrete-time binary time series $\{S_i\}_{i=1}^{i=\infty}$ taking values in the set $\{0, 1\}$. Before deriving the authentication probability, we need the following lemmas. To express our main result (i.e., Proposition 1), it is convenient to represent the four transition probabilities in terms of the stationary probabilities and β.

LEMMA 1. *The four transition probabilities can be expressed using* β, π_0, *and* π_1 *as follows:*

$$p_{10} = \frac{1}{\beta}, p_{01} = \frac{\pi_1}{\beta\pi_0}, p_{11} = 1 - \frac{1}{\beta}, p_{00} = 1 - \frac{1}{\beta}\left(\frac{1}{\pi_0} - 1\right).$$

PROOF. Let $\boldsymbol{B}$ be a random variable representing the length of a consecutive string of losses in steady state. Then the expected burst length is $\beta = E[\boldsymbol{B}] = (\mu_{00} - 1)/p_{01}$, where μ_{ij} denotes the expected number of transitions until the chain enters state j given that it is presently in state i. The value of μ_{00} can be written as an infinite sum of kf_{00}^k, where f_{ij}^n denotes the probability that, starting in i, the first transition into j occurs at time n. Hence,

$$\mu_{00} = \sum_{k=1}^{\infty} k f_{00}^k = p_{00} + p_{01}p_{10}\sum_{k=0}^{\infty} p_{11}^k(k+2) = 1 + p_{01}/p_{10}.$$

Substituting this value for μ_{00} in the above expression for β, we get $\beta = 1/p_{10}$. It follows that $p_{10} = 1/\beta$. Now, using the relation between stationary probabilities π_0 and π_1, we obtain

$$\pi_0 = \pi_0 p_{00} + \pi_1 p_{10} = \pi_0(1 - p_{01}) + \pi_1 p_{10}.$$

Substituting the value $1/\beta$ for p_{10} and solving for p_{01}, we obtain $p_{01} = \pi_1/(\beta\pi_0)$. The remaining transition probabilities p_{00} and p_{11} can be obtained from the relation $p_{00} = 1 - p_{01}$ and $p_{11} = 1 - p_{10}$, respectively. □

The following lemma is needed for the proof of Lemma 3.

LEMMA 2. *Let* $\boldsymbol{T}$ *denote the number of transitions between successive visits to state* 1. *Then the following holds:*

$$E[\boldsymbol{T}^2] = \frac{2\pi_0}{\pi_1}\left(\frac{1}{p_{01}}\right) + \frac{1}{\pi_1}.$$

PROOF. The 2-MC loss model is an irreducible, positive recurrent Markov chain, and hence visits to a given state constitute a *renewal process*. Hence, visits to state 1 is a (renewal) event, and a new cycle begins with each visit to state 1. By the theory of renewal reward processes, the long-run average reward per unit time is equal to the expected reward earned during a cycle divided by the expected time of a cycle. We can form a renewal reward process by imagining that a reward is given at every time instant that is equal to the number of transitions from that time onward until the next visit to state 1. Then the expected reward earned during a cycle divided by the expected time of a cycle is given by

$$\frac{E[\boldsymbol{T} + (\boldsymbol{T} - 1) + (\boldsymbol{T} - 2) + \cdots + 1]}{E[\boldsymbol{T}]} = \frac{E[\boldsymbol{T}^2 + \boldsymbol{T}]}{2E[\boldsymbol{T}]} = \frac{E[\boldsymbol{T}^2]}{2E[\boldsymbol{T}]} + \frac{1}{2}.$$

Because the long-run average reward per unit time is the same as the average number of transitions it takes to transition into state 1, it follows that

$$\frac{E[\boldsymbol{T}^2]}{2E[\boldsymbol{T}]} + \frac{1}{2} = \sum_{i=0}^{1} \pi_i \mu_{i1}. \tag{3}$$

Solving for $E[\boldsymbol{T}^2]$ and using the fact that $E[\boldsymbol{T}] = \mu_{11} = 1/\pi_1$, we obtain

$$E[\boldsymbol{T}^2] = \frac{2\pi_0\mu_{01} + 1}{\pi_1}. \tag{4}$$

By conditioning on the next state visited, we obtain $\mu_{01} = 1 + p_{00}\mu_{01}$, which can be solved to obtain $\mu_{01} = 1/p_{01}$. Substituting this result into (4) gives the desired result. □

LEMMA 3. *Define $N(n)$ as the number of visits to state 1 by time n. Then for all k*

$$Pr\left\{\frac{N(n) - \eta}{\sigma} < k\right\} \to \Phi(k) \quad as\ n \to \infty,$$

where $\eta = n\pi_1$, $\sigma^2 = \pi_1\pi_0 n(2\beta\pi_0 - 1)$, and $\Phi(k)$ is the standard normal distribution function.

PROOF. Visits to state 1 constitute a renewal event, and hence $N(n)$ is a delayed (general) renewal process. By Ross [1996, Theorem 3.3.5], the following holds:

$$Pr\left\{\frac{N(n) - \eta}{\sigma} < k\right\} \to \Phi(k) \quad as\ n \to \infty,$$

where $\eta = n/E[\boldsymbol{T}]$ and $\sigma^2 = n\mathrm{Var}(\boldsymbol{T})/(E[\boldsymbol{T}])^3$.

Using the relation $E[\boldsymbol{T}] = \mu_{11} = 1/\pi_1$, we obtain $\eta = n\pi_1$. The variance of $\boldsymbol{T}$ is given by (using Lemma 2 for $E[\boldsymbol{T}^2]$):

$$\mathrm{Var}(\boldsymbol{T}) = E[\boldsymbol{T}^2] - \frac{1}{\pi_1^2} = \frac{2\pi_0\pi_1 + p_{01}(\pi_1 - 1)}{p_{01}\pi_1^2}.$$

Using $\mathrm{Var}(\boldsymbol{T})$ (given above) and $E[\boldsymbol{T}] = 1/\pi_1$, we obtain

$$\sigma^2 = \pi_1 n(2\pi_0\pi_1 + p_{01}(\pi_1 - 1))/p_{01}.$$

Substituting p_{01} with the value derived in Lemma 1, we obtain $\sigma^2 = \pi_1\pi_0 n(2\beta\pi_0 - 1)$, and the desired result follows. □

Note that because σ^2 is nonnegative, we conclude that $\beta \geq 1/(2\pi_0)$. In the following corollary, we generalize Lemma 3 to include the case when k (on the left-hand side of the equation) is a function of n.

COROLLARY 1. *Define $F_{N(n)}(x) = \Pr\{(N(n) - \eta)/\sigma < x\}$. Let $g(n)$ denote some function of n, and define k such that $\lim_{n\to\infty} g(n) = k$. Then the following holds:*

$$F_{N(n)}(g(n)) \to \Phi(k) \quad as\ n \to \infty.$$

PROOF. Fix $\varepsilon > 0$. Then there exists an N such that for all $n \geq N$, $|g(n) - k| < \varepsilon$ (i.e., $k - \varepsilon < g(n) < k + \varepsilon$). Hence, for all $n \geq N$, $F_{N(n)}(k - \varepsilon) \leq F_{N(n)}(g(n)) \leq F_{N(n)}(k + \varepsilon)$, because $F_{N(n)}$ is a monotonically increasing function. Applying Lemma 3 to the above inequalities, we obtain

$$\Phi(k - \varepsilon) \leq \liminf_{n\to\infty} F_{N(n)}(g(n)) \leq \limsup_{n\to\infty} F_{N(n)}(g(n)) \leq \Phi(k + \varepsilon).$$

These inequalities hold for arbitrarily small ε, so by the continuity of Φ we obtain the desired result. □

Now we state our main result—derivation of the asymptotic authentication probability for SAIDA assuming the 2-MC loss model. As before, it is assumed that a block consists of n packets, and the minimum number of segments required for decoding is m. The result stated by Proposition 1 holds for $n \to \infty$. To obtain a nontrivial result, we let m increase as n becomes large, but in such a way that it grows more slowly than n. Specifically, we let $n \to \infty$ and $m = n\pi_0 - \gamma\sqrt{n}$ for some fixed constant γ.

PROPOSITION 1. *The authentication probability of the ith packet in the block is given by the following expression when the minimum number required for decoding is* $m = n\pi_0 - \gamma\sqrt{n}$*:*

$$Pr\{P_i \text{ is verifiable} \mid P_i \text{ is received}\} \to \Phi(k) \quad \text{as } n \to \infty,$$

where $k = \gamma/\sqrt{\pi_1\pi_0(2\beta\pi_0 - 1)}$.

PROOF. Define the renewal process $\{N(0),\ N(1), \ldots\}$ as follows: $N(0) = 0$, and $N(n-i)$ is the number of packet losses between P_{i+1} and P_n (the last packet in the block).

We can see that $\Pr\{P_i \text{ verifiable} \mid P_i \text{ is received}\}$ is lower bounded by the probability of the number of packet losses between P_{i+1} and P_n being less than $n - m - (i-1) + 1$. This is because having at most $n - m - (i-1)$ losses after P_i guarantees that we can verify P_i regardless of what happened before P_i. Hence,

$$\Pr\{P_i \text{ verifiable} \mid P_i \text{ is received}\} \geq \Pr\{N(n-i) < n - m - (i-1) + 1\}.$$

Note that if $i \geq n - m + 2$, then the above inequality holds trivially. Now, let

$$y = \frac{\gamma\sqrt{n} + i(\pi_1 - 1) + 2}{\sqrt{\pi_1\pi_0(n-i)(2\beta\pi_0 - 1)}}.$$

Then,

$$\Pr\left\{\frac{N(n-i) - \pi_1(n-i)}{\sqrt{\pi_1\pi_0(n-i)(2\beta\pi_0 - 1)}} < y\right\} = \Pr\{N(n-i) < n - m - (i-1) + 1\}. \quad (5)$$

Now, using a similar approach, we find the upper bound. The authentication probability is upper bounded by the probability of the number of packet losses between P_{i+1} and P_n being less than $n - m + 1$. This is because the verification of P_i implies that at most $n - m$ packet losses can be tolerated after P_i. Hence,

$$\Pr\{P_i \text{ verifiable} \mid P_i \text{ is received}\} \leq \Pr\{N(n-i) < n - m + 1\}.$$

Note that if $i \geq m$, then the above inequality holds trivially. Now, let

$$z = \frac{\gamma\sqrt{n} + \pi_1 i + 1}{\sqrt{\pi_1\pi_0(n-i)(2\beta\pi_0 - 1)}}.$$

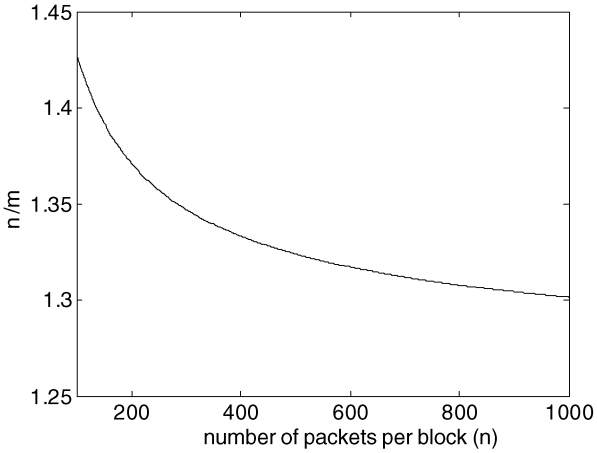

Fig. 3. Plot of n/m versus n for $\pi_0 = 0.8$ and $\gamma = 1$.

Then,

$$\Pr\left\{ \frac{N(n-i) - \pi_1(n-i)}{\sqrt{\pi_1\pi_0(n-i)(2\beta\pi_0 - 1)}} < z \right\} = \Pr\{N(n-i) < n - m + 1\}. \qquad (6)$$

Hence, the authentication probability is lower bounded by (5) and upper bounded by (6). Define

$$k = \frac{\gamma}{\sqrt{\pi_1\pi_0(2\beta\pi_0 - 1)}}.$$

Comparing the values of y, z, and k for fixed i and $n \to \infty$, it follows that $\lim_{n\to\infty} y = \lim_{n\to\infty} z = k$. Therefore, the lower bound and the upper bound for the authentication probability are asymptotically the same (for fixed i and $n \to \infty$), and the following holds by Corollary 1:

$$\Pr\{P_i \text{ verifiable} \mid P_i \text{ is received}\} \to \Phi(k) \text{ as } n \to \infty. \qquad \square$$

Proposition 1 reveals an interesting relationship between n, n/m, and the authentication probability. According to the proposition, if π_0, β, and γ are constant, then the asymptotic authentication probability is also constant. Suppose that $\gamma > 0$, which is equivalent to $\pi_0 > m/n$. This corresponds to the case when the asymptotic authentication probability is greater than 0.5. Because $m = n\pi_0 - \gamma\sqrt{n}$, increasing n (while keeping γ constant) increases m in such a way that the value of n/m decreases. This behavior can be observed in Figure 3, where n/m versus n is plotted for $\gamma = 1$ and $\pi_0 = 0.8$. The value of n/m has significance, because it determines the space overhead caused by the IDA encoding process—a decrease in n/m results in a decrease in space overhead. To elaborate, this means that for $\gamma > 0$, by increasing n, a constant (asymptotic) authentication probability can be maintained while decreasing n/m. This suggests that the authentication probability can be improved, while a constant space overhead is maintained, by increasing the number of packets per block. This hypothesis was confirmed using simulations. Figure 4 shows the change in

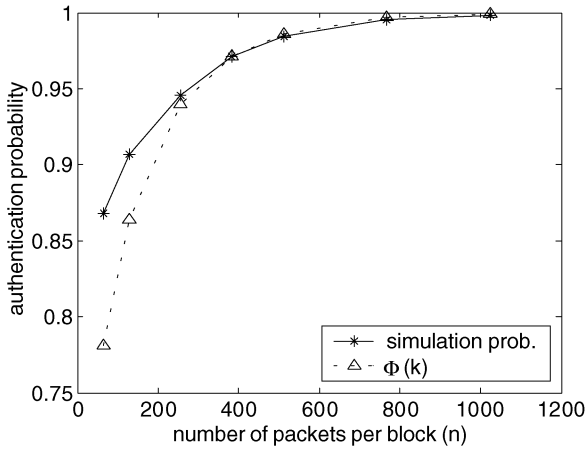

Fig. 4. Authentication probability versus n.

authentication probability as n is increased (while keeping $n/m = 1.5$, $\pi_0 = 0.8$, and $\beta = 8$ constant and $\pi_0 > m/n$). The solid curve represents the authentication probability obtained from simulations, and the dotted curve represents the value of $\Phi(k)$ given in Proposition 1. As hypothesized, the values of the two curves increase as n is increased, although n/m is kept constant.

From the above result, we can conclude that it would be advantageous to make the block size large, if relatively large verification delays (at the receiver) are tolerated. By making the block size large, the sender can decrease the space overhead incurred by the authentication process without affecting performance (i.e., authentication probability).

4.3 Asymptotic Authentication Probability under the Biased Coin Toss Model

The BCT loss model is defined as follows:

4.3.1 *BCT Loss Model.* Let $0 < q < 1$, and let $b \geq 1$ be an integer. For all i, a burst of length b packets begins with packet P_i (i.e., loss includes P_i) with probability q.

For $b = 1$, this model is equivalent to the 2-MC loss model with $p_{01} = p_{11} = q$ and $p_{10} = p_{00} = 1 - q$ (hence $\pi_1 = q$). This has the effect of removing the dependence of S_{i+1} on S_i (for all i), and hence, the loss (or no loss) of packets is determined by independent tosses of a q-biased coin. For $b > 1$, this model produces bursty loss patterns, whereas for $b = 1$, it produces independent packet losses.

We can use the techniques applied in Section 4.1 to derive the asymptotic authentication probability of SAIDA under the BCT loss model. The same techniques are also applicable in this case, because the BCT loss process can be modeled as a discrete-time Markov chain with $b+1$ states: $0, 1, \ldots, b$ (see Figure 5). In this Markov chain, state 0 represents no loss and states 1 through b denote packet loss. The transitions into state 0 are renewal events (in the derivations of Section 4.1, visits to state 1 represented renewal events). To derive

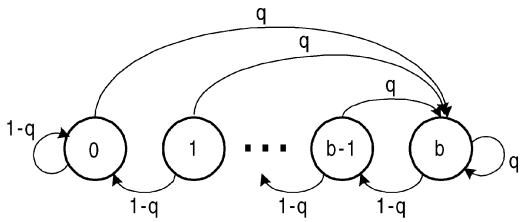

Fig. 5. BCT loss process modeled as a Markov chain.

the asymptotic authentication probability under the BCT model, we need to find the variance of $\boldsymbol{T}_0$, where $\boldsymbol{T}_0$ denotes the number of transitions between successive visits to state 0.

LEMMA 4. *The variance of* $\boldsymbol{T}_0$ *is given by:*

$$Var(\boldsymbol{T}_0) = \frac{2}{(1-q)^b} \sum_{i=0}^{b} \pi_i \mu_{i0} - \frac{(1-q)^b + 1}{(1-q)^{2b}},$$

where $\mu_{i0} = \begin{cases} \dfrac{1}{(1-q)^b}, & i = 0 \text{ or } 1 \\ \dfrac{2-q}{(1-q)^b} \sum_{i=0}^{\lfloor (i-1)/2 \rfloor} (1-q)^{2i}, & i \geq 2 \text{ and even} \\ \dfrac{1}{(1-q)^b} \left(1 + (2-q) \sum_{i=0}^{\lfloor i/2 \rfloor - 1} (1-q)^{2i+1} \right), & i \geq 3 \text{ and odd} \end{cases}$

and $\pi_i = \begin{cases} (1-q)^b, & i = 0 \\ q(1-q)^{b-i}, & 1 \leq i \leq b \end{cases}$.

Here, $\lfloor x \rfloor$ *denotes the largest integer not greater than* x.

PROOF (SKETCH). Using the same argument used in the proof of Lemma 2 (see (3)), we obtain:

$$E\left[\boldsymbol{T}_0^2\right] = \frac{2}{\pi_0} \sum_{i=0}^{b} \pi_i \mu_{i0} - \frac{1}{\pi_0}. \tag{7}$$

By using the relations among the stationary probabilities, we obtain the following:

$$\pi_i = \sum_{j=0}^{b} \pi_j P_{ji}, \quad 0 \leq i \leq b.$$

The values of π_i stated in the lemma can be obtained by using the above set of equations and the relation $\pi_0 + \pi_1 + \cdots + \pi_b = 1$.

Conditioning on the next state visited, the following set of equations can be obtained:

$$\mu_{i0} = 1 + \sum_{j=1}^{b} P_{ij} \mu_{j0}, \quad 0 \leq i \leq b.$$

Solving for the values of μ_{i0} (using the above set of equations), we obtain the values of μ_{i0} stated in the lemma.

Now, using (7) and the relation $E[\boldsymbol{T}_0] = 1/\pi_0$, the variance of $\boldsymbol{T}_0$ is given by

$$Var(\boldsymbol{T}_0) = \frac{2}{\pi_0}\sum_{i=0}^{b}\pi_i\mu_{i0} - \frac{1}{\pi_0} - \frac{1}{\pi_0^2}.$$

Using the relation $\pi_0 = (1-q)^b$ in the above equation, we obtain the desired result. □

Now we state the asymptotic authentication probability under the BCT model. The same techniques used in the proof of Proposition 1 are applied here.

PROPOSITION 2. *The authentication probability of the ith packet in the block is given by the following expression when the minimum number required for decoding is* $m = n\pi_0 + \gamma\sqrt{n}$:

$$Pr\{P_i \text{ is verifiable} \mid P_i \text{ is received}\} \to 1 - \Phi(k) \quad \text{as } n \to \infty,$$

where $k = \gamma/\sqrt{(1-q)^{3b}Var(\boldsymbol{T}_0)}$ *and* $Var(\boldsymbol{T}_0)$ *is given in Lemma* 4.

PROOF. Define the renewal process $\{M(0),\ M(1), \ldots\}$ as follows: $M(0) = 0$, and $M(n-i)$ is the number of packets received among $P_{i+1}, \ldots, P_n$. We can see that the authentication probability is lower bounded by the probability of the number of packets received among $P_{i+1}, \ldots, P_n$ not being less than $m-1$. Hence,

$$\begin{aligned}\Pr\{P_i \text{ verifiable} \mid P_i \text{ is received}\} &\geq \Pr\{M(n-i) \geq m-1\} \\ &= 1 - \Pr\{M(n-i) < m-1\}.\end{aligned}$$

Now, let

$$v = \frac{\gamma\sqrt{n} + i\pi_0 - 1}{\sqrt{((n-i)\mathrm{Var}(\boldsymbol{T}_0))/(E[\boldsymbol{T}_0])^3}}.$$

Then,

$$1 - \Pr\left\{\frac{M(n-i) - \pi_0(n-i)}{\sqrt{((n-i)\mathrm{Var}(\boldsymbol{T}_0))/(E[\boldsymbol{T}_0])^3}} < v\right\} = 1 - \Pr\{M(n-i) < m-1\}. \quad (8)$$

Similarly, the authentication probability is upper bounded by the probability of the number of packets received among $P_{i+1}, \ldots, P_n$ not being less than $m-i$. Hence,

$$\begin{aligned}\Pr\{P_i \text{ verifiable} \mid P_i \text{ is received}\} &\leq \Pr\{M(n-i) \geq m-i\} \\ &= 1 - \Pr\{M(n-i) < m-i\}.\end{aligned}$$

Now let

$$w = \frac{\gamma\sqrt{n} + i\pi_0 - i}{\sqrt{((n-i)\mathrm{Var}(\boldsymbol{T}_0))/(E[\boldsymbol{T}_0])^3}}.$$

Then,

$$1 - \Pr\left\{ \frac{M(n-i) - \pi_0(n-i)}{\sqrt{((n-i)\mathrm{Var}(\boldsymbol{T}_0))/(\boldsymbol{E}[\boldsymbol{T}_0])^3}} < w \right\} = 1 - \Pr\{M(n-i) < m-i\}. \quad (9)$$

Hence, the authentication probability is lower bounded by (8) and upper bounded by (9). Define

$$k = \frac{\gamma}{\sqrt{\mathrm{Var}(\boldsymbol{T}_0)/(\boldsymbol{E}[\boldsymbol{T}_0])^3}}.$$

Comparing the values of v, w, and k for fixed i and $n \to \infty$, it follows that $\lim_{n\to\infty} v = \lim_{n\to\infty} w = k$. Therefore, the lower bound and the upper bound are asymptotically the same (for fixed i and $n \to \infty$), and the following holds by Corollary 1:

$$\Pr\left\{P_i \text{ is verifiable} \mid P_i \text{ is received}\right\} \to 1 - \Phi(k) \quad \text{as} \quad n \to \infty.$$

Substituting the value of $1/(1-q)^b$ for $\boldsymbol{E}[\boldsymbol{T}_0]$ in k, we obtain the desired result. □

Not surprisingly, Proposition 2 reveals the same relationship between n, n/m, and the authentication probability—by increasing n, the authentication probability can be improved while a constant value of n/m is maintained (see Figure 4).

4.4 Lower Bound of the Authentication Probability

In this section, using the BCT loss model, we derive a lower bound of the authentication probability for SAIDA. According to the loss model, the maximum number of places where a burst error can occur (and still allow the guaranteed authentication of P_i) is given by $z = \lfloor (n-m)/b \rfloor$. (Recall that n and m denote the number of encoded segments and the minimum number needed for decoding, respectively.) Because the loss of a packet is determined by the flip of a q-biased coin, the probability that z or fewer coin tosses result in losses lower bounds the authentication probability. Hence, the authentication probability is bounded as follows:

$$\Pr\{P_i \text{ verifiable} \mid P_i \text{ received}\} \geq \begin{cases} \sum_{j=0}^{z} \binom{n-b}{j} q^j (1-q)^{n-b-j}, & \text{if } i > b-1 \\ \sum_{j=0}^{z} \binom{n-i}{j} q^j (1-q)^{n-i-j}, & \text{if } i \leq b-1 \end{cases}.$$

The above derivation takes into account the fact that none of the z coin tosses can occur in the $b-1$ packets immediately preceding P_i, because it is assumed that the packet was received. For $i \leq b-1$, this would be $i-1$ packets immediately preceding P_i.

In Minor and Staddon [2001], authors derive a lower bound of the authentication probability (for the piggybacking scheme) based on the BCT loss model. The lower bound is given by the following expression (see Minor and

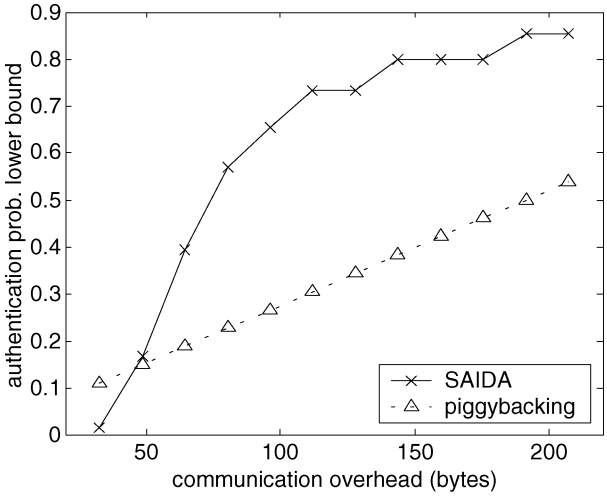

Fig. 6. Authentication lower bounds.

Staddon [2001] for derivation):

$$\Pr\{P_i \text{ verifiable} \mid P_i \text{ received}\} \times \begin{cases} \geq \sum_{j=0}^{z'} \binom{i-1-b}{j} q^j (1-q)^{i-1-b-j}, & \text{if } i > b+1+z' \\ = 1, & \text{if } i \leq b+1+z', \end{cases}$$

where $z' = \min\{x_i, \lfloor b_i/b \rfloor\}$. Parameters x_i and b_i denote the maximum number and the maximum size of the bursts that can be tolerated, respectively. Note that this lower bound was derived assuming that the signature packet was received, whereas the lower bound for SAIDA was derived without this assumption.

In Figure 6, we plot the lower bounds of the two schemes as the communication overhead (per packet) is increased. Because the lower bound changes with the position of the packet, the average lower bound (among n packets) is used. Note that the lower bound of SAIDA resembles a distorted staircase function. This is because of the floor function within the expression for the lower bound of SAIDA. The following parameters were used:

- signature size = 128 bytes, hash size = 16 bytes;
- $n = 128$, $b = 4$, $q = 0.2$, $b_i = 48$;
- parameter x_i is increased from one to twelve in increments of one;
- values for m: {67, 45, 34, 28, 23, 20, 17, 16, 14, 13, 12, 11}.

5. PERFORMANCE EVALUATION

5.1 Overhead Comparison

We compare our solution with four previously proposed schemes—authentication tree, EMSS, augmented chain, and the piggybacking scheme.

Table I. Performance Criteria

Criterion	Explanation
sender delay	delay on the sender side (in number of data packets) before the first packet in the block can be transmitted
receiver delay	delay on the receiver side (in number of data packets) before verification of the first packet in the block is possible
computation overhead	number of hashes and signatures computed by the sender per block
communication overhead	size of the authentication information required for each packet
verification rate	number of verifiable packets of the entire stream divided by the total number of received packets of the stream

Table II. Comparison of the Authentication Schemes

	Authentication Tree	EMSS	Augmented Chain	Piggybacking	SAIDA
sender delay	n	1	p	n	n
receiver delay	1	n	n	1	$[m, n]$
computation overhead	$2n-1, 1$	$n+1, 1$	$n+1, 1$	$n+1, 1$	$n+1, 1$
communication overhead	$\sigma + 1 + (h+1)\log_2 n$	variable	variable	variable	variable
verification rate	1.0	variable	variable	variable	variable

For the comparison, we only consider schemes that amortize a signing operation over multiple packets. Table II summarizes the five authentication schemes based on the performance criteria explained in Table I. Its values were obtained based on the following assumptions:

- All five schemes employ a block size of n packets.
- Communication overhead of the authentication tree was obtained for a tree of degree two and assuming a signature size of σ and a hash size of h.
- The augmented chain is parameterized by the integer variables a and p, where $p < n$.
- For SAIDA, n is the number of encoded segments, and m is the minimum number needed for decoding.

According to Table II, the verification rate for the schemes EMSS, augmented chain, piggybacking, and SAIDA is not constant and actually depends on the communication overhead. The authentication tree technique has the favorable property of guaranteeing the verification of every received packet, but at the cost of a larger communication overhead—an overhead on the order of several hundred bytes would be required for practical block sizes. Note that the receiver delay for SAIDA is not fixed—it could be anywhere in the interval $[m, n]$. We emphasize that SAIDA's advantage over the other schemes is the ability to obtain high verification rates with minimal communication overhead (see Section 5.2). By strategy, our scheme trades off sender and receiver delay for an improvement in verification rate. Note that the technique of

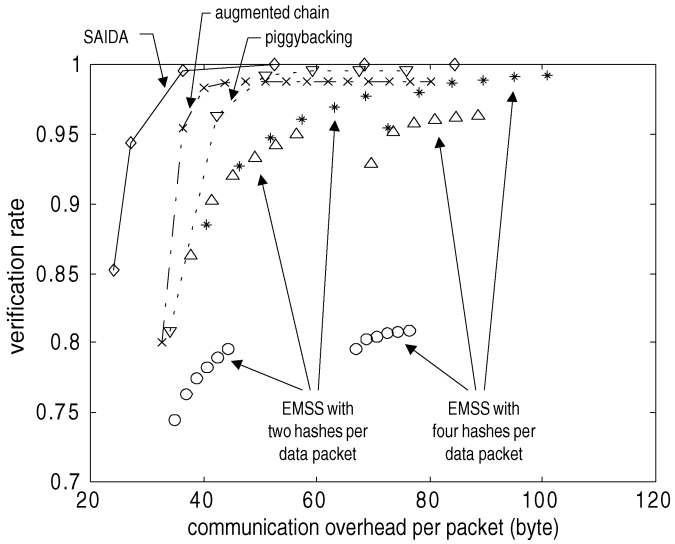

Fig. 7. Verification rate versus communication overhead.

Pannetrat et al. [2002] can be applied to SAIDA to remove the sender delay at the cost of increased receiver delay and vice versa.

5.2 Trade-off Between Verification Rate and Communication Overhead

As mentioned earlier, if the requirement on individual packet verification is relaxed, then the communication overhead can be reduced substantially. In this approach, verification of each packet is not guaranteed and instead is assured with a certain probability. EMSS, augmented chain, piggybacking, and SAIDA fall into this category, and as expected, there is a trade-off between verification rate and communication overhead for these schemes.

For the augmented chain method, the number of hash chains per packet is not a variable parameter. However, multiple copies of the signed packet can be transmitted to increase the verification rate. The same technique can be applied to the piggybacking scheme to improve performance. For simulations of the piggybacking scheme, we assume that there is only one priority class (i.e., priority class 0) with $x_0 = 2$ and $b_0 = 29$. In SAIDA, higher verification rates can be achieved by increasing the value of n/m. The tradeoff between performance and communication overhead in EMSS was already discussed in Section 3.1.

Figure 7 shows the verification rate for the four probabilistic authentication schemes—augmented chain, EMSS, piggybacking, and SAIDA. To simulate bursty loss patterns, we use the 2-MC loss model defined in Section 4.1 with a packet loss probability of 0.2. The choice of 0.2 as the loss probability was motivated by the fact that, in general, the receiver loss rate is greater for multicast compared to unicast. In Yajnik [1999], the authors, using actual network measurements, showed that the loss rates for multicast sessions are much higher (more than twice) compared to their corresponding unicast sessions.

Table III. Simulation Parameters for Figure 7

Scheme	Simulation Parameters
General Parameters	loss prob. = 0.2, avg. burst length = 8, block size = 128, size of hash = 16 bytes, size of signature = 128 bytes
EMSS	length of hash chain is uniformly distributed over [1, 127]
Augmented Chain	$p = 6$ and $a = 15$
Piggybacking	$x_0 = 2$ and $b_0 = 29$
SAIDA	$n = 128$, $m = \{90, 80, 60, 42, 32, 26\}$

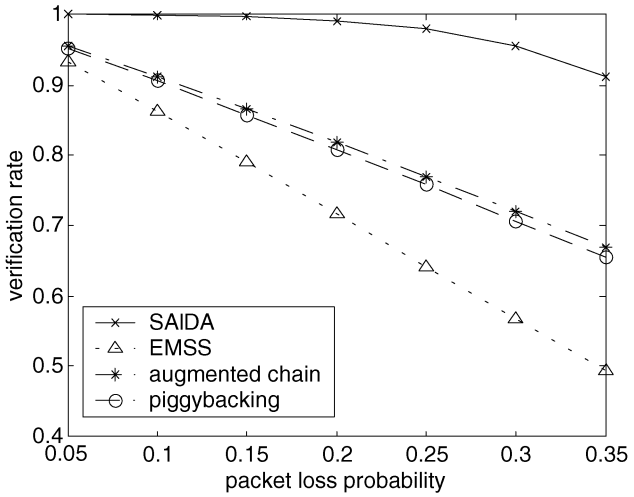

Fig. 8. Verification rate versus packet loss probability.

Some multicast sessions were observed to have loss rates exceeding 20% [Yajnik et al. 1996, 1999]. The simulation parameters are given in Table III.

For EMSS, verification rates were obtained by simulating numerous combinations of the three factors discussed in Section 3.1. The two clusters of markers represent the simulation results for EMSS—the left cluster represents EMSS implemented with two hashes appended per data packet and the right cluster represents EMSS implemented with four. Each cluster is composed of three types of markers—circle markers represent implementations with a single signature packet per block, while the triangle and asterisk markers represent implementations with two and three signature packets per block, respectively. Each type of marker was used several times to represent the different number of hashes appended in the signature packet. The number of hashes appended in the signature packet was varied from 15 to 90 in increments of fifteen.

When multicast packets (via UDP) are sent across networks with heavy congestion or route failures, packet loss can be high. Furthermore, conditions for the network can change abruptly during a relatively short time period. Even if the verification rate of the packets was satisfactory at the start of reception, it could deteriorate rapidly as the loss rate increases in the network. We performed experiments to examine the effect of increased packet loss on the robustness of the authentication scheme. Figure 8 shows the change in verification

Table IV. Simulation Parameters for Figure 8

Scheme	Simulation Parameters
General Parameters	avg. burst length = 8, block size = 128, size of hash = 16 bytes, size of signature = 128 bytes
EMSS	length of hash chain is uniformly distributed over [1, 64], number of hashes per signature packet = 5, number of hashes per data packet = 2, number of signature packets per block = 1
Augmented Chain	number of signature packets per block = 1 $p = 6$ and $a = 15$
Piggybacking	number of signature packets per block = 1 $x_0 = 2$ and $b_0 = 29$
SAIDA	$n = 128, m = 65$

rate as the space overhead is kept constant (space overhead for the four schemes is the same), and the packet loss probability is increased. The authentication overhead per packet is fixed at 34 bytes. Again, the 2-MC loss model is used. The simulation parameters are given in Table IV.

6. MAKING SAIDA ROBUST AGINST DENIAL-OF-SERVICE ATTACKS

Consider the following DoS attack: an attacker inserts bogus packets in the packet stream to interfere with the reconstruction process of SAIDA. Recall that IDA is an erasure code that is robust against missing segments of information, and is not robust against modified segments (either intentional or unintentional). If one or more of the IDA encoded segments (which are used in the reconstruction) are modified during transit, the receiver has no way of detecting this, and the reconstruction of the original message fails. Although UDP, which is used by multicast applications, provides a simple way of detecting errors within a packet via the UDP checksum [Comer 1995 p. 182], this does not protect against intentional alteration of the packet. By inserting bogus packets (with valid checksums) within the packet stream, an attacker can successfully interrupt multicast application services. Rabin [1989] proposes a solution to combat this type of an attack by cryptographically *fingerprinting* the IDA encoded segments—this enables the receiver to verify which encoded segments were modified and which were not. Although this solution is effective against attackers that are outside the multicast group, it does not prevent attacks carried out by malicious users who are members of the multicast group (see Rabin [1989] for details).

In Krawczyk [1993], an improved solution, which uses a new cryptographic tool called *distributed fingerprints*, is proposed that does not suffer from the drawback mentioned above. Krawczyk's approach is to use IDA in combination with an error-correcting code, which is robust against information modification as well as loss. As expected, error-correcting codes add more redundancy in the encoding process compared to erasure codes. The amount of redundancy is commonly measured by a parameter known as the *blowup factor*, which is defined as the ratio between the total size of the encoded information and the size of the original information. For IDA, the blowup factor is $n/(n-t)$, where t is the maximum allowable number of missing encoded segments, and n is the

total number of encoded segments. According to Lemma 1 given in Krawczyk [1993], an error-correcting code that can recover the original information in the presence of up to α altered segments and t missing segments has a blowup factor lower bounded by $n/(n - t - 2\alpha)$. Codes such as Reed–Solomon codes achieve this bound.

In terms of computation complexity, error-correcting codes, using straightforward implementations, can encode and decode in time $O(n^2)$. Using more sophisticated techniques, encoding and decoding times of $O(n \log n)$ and $O(n \log^2 n)$ are possible, respectively [Blahut 1984]. For values of n appropriate to our problem, it is generally viewed that encoding and decoding can be done in real time.

Using Krawczyk's approach, we can modify SAIDA so that it is robust against DoS attacks of the type discussed above. The following steps describe the modified authentication procedure:

1. Apply Steps 1 through 7 (of Section 3.2) to create IDA encoded segments of F^1 and $\sigma(K_r,\ H_G(G_1))$.
2. Concatenate σ_i and F_i^1 to form $\sigma_i \parallel F_i^1$ for $i = 1, \ldots, n$. Here, σ_i denotes $\sigma_i(K_r,\ H_G(G_1))$.
3. For $\sigma_i \parallel F_i^1,\ i = 1, \ldots, n$, compute its *fingerprint* $H(\sigma_i \parallel F_i^1)$ using a collision-resistant hash function H, and then concatenate these values to form a single string $\Omega = H(\sigma_1 \parallel F_1^1) \parallel \cdots \parallel H(\sigma_n \parallel F_n^1)$.
4. Encode Ω with an error-correcting code (e.g., Reed–Solomon code) to obtain $\omega_1, \ldots, \omega_n$.
5. Concatenate each signature segment σ_i, hash segment F_i^1, and ω_i with the corresponding packet to form an authenticated packet. That is, create the following:

$$\sigma_i(K_r,\ H_G(G_1)) \parallel F_i^1 \parallel \omega_i \parallel P_i, \quad \text{for } i = 1, \ldots, n.$$

   The resulting authenticated packets are transmitted.

The verification procedure is as follows. We assume that t (number of missing segments) and α (number of altered segments) are small enough so that IDA and the error-correcting code are able to reconstruct the original information.

1. Using the decoding algorithm of the error-correcting code, reconstruct Ω. Let the string $H(\sigma_1 \parallel F_1^1) \parallel \cdots \parallel H(\sigma_n \parallel F_n^1)$ represent the reconstructed Ω.
2. For each authenticated packet that is received, extract its signature segment and hash segment, concatenate the two values, and hash it, which we denote as $H(\tilde{\sigma}_i \parallel \tilde{F}_i^1)$. Compare this value with the corresponding part of Ω (i.e., $H(\sigma_i \parallel F_i^1)$). If $H(\sigma_i \parallel F_i^1) = H(\tilde{\sigma}_i \parallel \tilde{F}_i^1)$, then the signature segment $\tilde{\sigma}_i$ and hash segment $\tilde{F}_i^1$ are considered to be legitimate (i.e., unmodified). Otherwise, the signature and hash segments are considered to be modified.
3. Employing the decoding algorithm of IDA (see Section 3.2), reconstruct F^1 and $\sigma(K_r,\ H_G(G_1))$ using only legitimate segments.

In Krawczyk [1993], it was shown that the distributed fingerprint scheme is asymptotically (in the size of the original message) space optimal. If the

distributed fingerprint scheme (with Reed–Solomon encoding) is applied to SAIDA, the resulting authentication space (or communication) overhead per block is given by the following:

$$\frac{n(nh+s)}{n-t-\alpha}+\frac{n^2h}{n-t-2\alpha},$$

where h denotes the size of the hash and s denotes the size of the signature. Here, we assumed that the same hash function is used to compute the packet hashes and the fingerprints. The resulting authentication scheme can reconstruct the original authentication information in the presence of up to α modified packets and t lost packets.

7. CONCLUSIONS

Through our results, we showed that SAIDA is an efficient method for multicast stream authentication that is highly robust against packet loss. For the same amount of communication overhead, it achieved the highest verification rate among all the probabilistic schemes that were examined. Table II suggests that there is no single scheme that is superior in all aspects. Depending on the delay, computation, and communication-overhead requirements, different schemes are appropriate for different applications. We expect that our scheme's high tolerance for packet loss and low communication-overhead requirement will make it useful in many multicast applications. As already mentioned, SAIDA might not be appropriate in situations where the data to be sent are generated in real time, and immediate broadcast of it is crucial. Our scheme will be most useful in cases where the sender has a priori knowledge of at least a portion of the data to be broadcast (e.g., broadcast of prerecorded material).

ACKNOWLEDGMENTS

The authors gratefully acknowledge Jeff Herdtner and Gang Wu for their insightful comments. The authors also thank the anonymous referees for their suggestions that helped to improve the paper. A preliminary version of portions of this material was presented in Park et al. [2002].

REFERENCES

BLAHUT, R. 1984. *Theory and Practice of Error Control Codes*. Addison-Wesley, Reading, MA.

BONEH, D., DURFEE, G., AND FRANKLIN, M. 2001. Lower bounds for multicast message authentication. In *Proceedings of the Conference on Advances in Cryptology* (EUROCRYPT '01, Innsbruck, Austria, May), B. Pfitzmann, Ed. Springer-Verlag, New York, NY, 437–452.

BYERS, J. W., LUBY, M., MITZENMACHER, M., AND REGE, A. 1998. A digital fountain approach to reliable distribution of bulk data. In *Proceedings of the Conference of the ACM's Special Interest Group on Data Communication* (SIGCOMM '98, Vancouver, BC, Sept.), ACM Press, New York, NY, 56–67.

CANETTI, R., GARAY, J., ITKIS, G., MICCIANCIO, D., NAOR, M., AND PINKAS, B. 1999. Multicast security: a taxonomy and some efficient constructions. In *Proceedings of the IEEE Conference on Computer Communications* (INFOCOM '99, New York, NY, Mar.), IEEE Press, Piscataway, NJ, 708–716.

COMER, D. E. 1995. *Internetworking with TCP/IP Vol. I*: *Principles, Protocols, and Architecture*, 3rd ed. Prentice Hall, Englewood Cliffs, NJ.

DESMEDT, Y., FRANKEL, Y., AND YUNG, M. 1992. Multi-receiver/multi-sender network security: efficient authenticated multicast/feedback. In *Proceedings of the IEEE Conference on Computer Communications* (INFOCOM '92, Florence, Italy, May), IEEE Press, Piscataway, NJ, 2045–2054.

FLOYD, S., JACOBSON, V., LIU, C., MCCANNE, S., AND ZHANG, L. 1997. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Trans. Networking 5*, 6, 784–803.

GENNARO, R. AND ROHATGI, P. 1997. How to sign digital streams. In *Proceedings of the Conference on Advances in Cryptology* (CRYPTO '97, Santa Barbara, CA, Aug.), B. S. Kaliski Jr., Ed. Springer-Verlag, New York, NY, 180–197.

GOLLE, P. AND MODADUGU, N. 2001. Authenticating streamed data in the presence of random packet loss. In *Proceedings of the Network and Distributed System Security Symposium* (NDSS '01, San Diego, CA, Feb.), Internet Society, 13–22.

KOIFMAN, A. AND ZABELE, S. 1996. RAMP: a reliable adaptive multicast protocol. In *Proceedings of the IEEE Conference on Computer Communications* (INFOCOM '96, San Francisco, CA, Mar.), IEEE Press, Piscataway, NJ, 1442–1451.

KRAWCZYK, H. 1993. Distributed fingerprints and secure information dispersal. In *Proceedings of the ACM Symposium on Principles of Distributed Computing* (PODC '93, Ithaca, NY, Aug.), ACM Press, New York, NY, 207–218.

LUBY, M., MITZENMACHER, M., SHOKROLLAHI, M., SPIELMAN, D., AND STEMANN, V. 1997. Practical loss-resilient codes. In *Proceedings of the ACM Symposium on Theory of Computing* (STOC '97, El Paso, TX, May), ACM Press, New York, NY, 150–159.

LUBY, M., VICISANO, L., GEMMELL, J., RIZZO, L., HANDLEY, M., AND CROWCROFT, J. 2002. The use of forward error correction in reliable multicast. *IETF Internet Draft draft-ietf-rmt-info-fec-03.txt*, available at http://www.ietf.org/internet-drafts/draft-ietf-rmt-info-fec-03.txt.

MERKLE, R. 1989. A certified digital signature. In *Proceedings of the Conference on Advances in Cryptology* (CRYPTO '89, Santa Barbara, CA, Aug.), S. Goldwasser, Ed. Springer-Verlag, New York, NY, 218–238.

MINER, S. AND STADDON, J. 2001. Graph-based authentication of digital streams. In *Proceedings of the IEEE Symposium on Research in Security and Privacy* (Oakland, CA, May), IEEE Computer Society Press, 232–246.

PANNETRAT, A. AND MOLVA, R. 2002. Authenticating real time packet streams and multicasts. In *Proceedings of the 7th International Symposium on Computers and Communications* (ISCC '02, Taormina, Italy, July), IEEE Computer Society Press, 490–495.

PARK, J., CHONG, E. K. P., AND SIEGEL, H. J. 2002. Efficient multicast packet authentication using signature amortization. In *Proceedings of the IEEE Symposium on Research in Security and Privacy* (Oakland, CA, May), IEEE Computer Society Press, 227–240.

PERRIG, A. 2001. The BiBa one-time signature and broadcast authentication protocol. In *Proceedings of the 8th ACM Conference on Computer and Communications Security* (CCS '01, Philadelphia, PA, Nov.), ACM Press, New York, NY, 28–37.

PERRIG, A., CANETTI, R., TYGAR, J. D., AND SONG, D. 2000. Efficient authentication and signing of multicast streams over lossy channels. In *Proceedings of the IEEE Symposium on Research in Security and Privacy* (Oakland, CA, May), IEEE Computer Society Press, 56–73.

RABIN, M. 1989. Efficient dispersal of information for security, load balancing, and fault tolerance. *J. ACM 36*, 2, 335–348.

ROHATGI, P. 1999. A compact and fast hybrid signature scheme for multicast packet authentication. In *Proceedings of the 6th ACM Conference on Computer and Communications Security* (CCS '99, Singapore, Nov.), ACM Press, New York, NY, 93–100.

ROSS, S. 1996. *Stochastic Processes*, 2nd ed. John Wiley and Sons, Inc., New York, NY.

SIMMONS, G. J. 1984. Authentication theory/coding theory. In *Proceedings of the Conference on Advances in Cryptology* (CRYPTO '84, Santa Barbara, CA, Aug.), G. R. Blakley and D. Chaum, Eds. Springer-Verlag, New York, NY, 411–431.

WEATHERSPOON, H., WELLS, C., EATON, P., ZAHO, B., AND KUBIATOWICZ, J. 2001. Silverback: A Global-Scale Archival System. Technical Report UCB/CSD-01-1139, Computer Science Division, University of California, Berkeley, CA.

WONG, C. AND LAM, S. 1998. Digital Signatures for Flows and Multicasts. Technical Report TR-98-15, Department of Computer Sciences, University of Texas at Austin, TX.

YAJNIK, M., KUROSE, J., AND TOWSLEY, D. 1996. Packet loss correlation in the Mbone multicast network. In *Proceedings of the IEEE Global Internet Conference* (London, England, Nov.), IEEE Press, Piscataway, NJ.

YAJNIK, M., MOON, S., KUROSE, J., AND TOWSLEY, D. 1999. Measurement and modeling of the temporal dependence in packet loss. In *Proceedings of the IEEE Conference on Computer Communications* (INFOCOM '99, New York, NY, Mar.), IEEE Press, Piscataway, NJ, 345–352.

Received June 2002; revised December 2002; accepted December 2002