MULTICAST GROUP AUTHENTICATION

Ritesh Mukherjee, J. William Atwood

Department of Computer Science and Software Engineering, Concordia University, Montreal, Canada

E-mail: mukherj@cse.concordia.ca, bill@cse.concordia.ca

Abstract.

Multicast is an attractive mechanism for delivering data to multiple receivers over the Internet as it saves bandwidth. However the delivery of copyrighted data over the Internet requires it to be encrypted to render the data useless for eavesdroppers or illegal users. Authentication is necessary to ensure that the received packet is sent by the actual sender. Message integrity is necessary to ensure that the packet was not changed by an attacker while in transit. The network must be able to perform these tasks even if packets are dropped, rearranged, changed or injected into the stream. This paper presents an efficient scheme for multicast authentication and to check multicast message integrity when asymmetric keys are used to protect the data. The proposal is validated using SPIN, which uses PROMELA to design the validation model.

1. INTRODUCTION

IP multicast is a bandwidth saving technology for the delivery of high bandwidth multimedia content over the Internet. Multicast replaces multiple packets over a shared link addressed to individual receivers with a single packet addressed to a group. This large saving in bandwidth makes multicast the desired choice of packet delivery when a large number of receivers are involved as is the case in applications such as pay-per-view TV, online games, stocks and news feeds, etc.

For the delivery of copyrighted and confidential data it is necessary to encrypt packets. This makes it necessary to deliver a decryption key to a potentially large number of receivers. Solutions for multicast key management and distribution such as GKMP [1] [2], SKMD [3], IOLUS [4], Broadcast Encryption [5] and SIM-KM [6] [7] have been proposed.

In addition to the requirement to distribute keys securely, multicast authentication is necessary to ensure that

- the received packets actually originated from a legitimate sender.
- the received packets were not modified while in transit.
- packets sent by an adversary are not mistaken as legitimate packets.

Multicast authentication is important when receiving news feeds, stock quotes or when watching pay-per-view TV. These kinds of applications require low-cost, highly efficient packet authentication. Consider a TV station, such as ESPN broadcasting sporting events live. The TV station is broadcasting to thousands of receivers where people are logged in watching the telecast. The users would want to ensure that the broadcast is from ESPN rather than a malicious third party transmitting offensive material. Another scenario could be a different brokerage houses receiving stock quotes from the New York Stock Exchange (NYSE) for their agents and displaying it to their clients through their respective brokerage house websites. Also online newspapers could receive quotes from the stock exchange and display it on their websites for the general public. The brokerage house and newspapers around the world would like to ensure that the quotes they are receiving have not been tampered with. A host of other scenarios can be envisaged where multicast authentication is imperative such as online teaching, soft-ware updates, company broadcasts, etc. These services all have a one sender-multiple receiver model. In fact multicast has the maximum bandwidth saving ad-vantage when there are thousands of receivers and very few senders.

From a key multicast key management perspective, many of the key management schemes use the multicast protocol itself for re-keying and this requires that the key distribution packets be authenticated otherwise an adversary may sent bogus re-key packets and disrupt the service.

In this paper we present a technique for performing multicast packet authentication and a message integrity check when asymmetric keys are used to protect the data. As an example, we will show how our technique can provide group authentication when used with SIM-KM. Our method may

also be used independently with any asymmetric key distribution scheme. It uses symmetric message authentication codes (MACs) to add data source authentication to secure group communication. Section 2 describes the related work that has been done in this area. Section 3 describes the challenges associated with multicast packet authentication. Section 4 gives an overview of SIM-KM. Section 5 presents our multicast authentication and message integrity check. Section 6 analyzes the protocol against known attacks. Section 7 compares the scheme to other available schemes. Section 8 discusses the construction of a validation model using PROMELA and the validation of this model using SPIN. Section 9 contains the concluding remarks.

2. RELATED WORK

There have been a few approaches to multicast authentication. In this section we present an overview of some of the approaches.

One approach is to use MACs where the group members share a secret key and a MAC is included in every packet. In this scheme any group member can spoof packets. To avoid this each receiver can be given a secret key and the sender can have all such keys. The sender now has to add a MAC for each receiver. This scheme is prone to collusion attacks and the size of each packet increases with the number of receivers resulting in enormous communication overhead. The Multiple MACs scheme [8] is not scalable and suffers from large communication overhead. Timed Efficient Stream Loss-tolerant Authentication (TESLA) [9] has low communication overhead but requires time synchronization between senders and receivers, which is difficult to maintain in large groups.

Another approach is to use digital signatures. As this is a computationally intensive operation schemes work with fast signature techniques and amortize a signature operation over several packets [10] but this scheme does not tolerate packet loss. BiBa [11] is a fast individual packet authentication signature scheme but re-quires time synchronization between sender and receivers, which limits the authentication rate and also suffers from communication overhead. A Merkle hash tree [12] can be used for authentication and the scheme is tolerant to packet loss but has enormous communication overhead. Erasure codes [13] can also be used for authentication. The scheme performs encoding twice to reduce communication overhead but the scheme fails if a single packet is injected. Graph-based

authentication schemes amortize a signature over a hash chain in such a way so as to tolerate packet loss. Hashes can be inserted in strategic locations to make it resistant to a burst loss [14]. In general, graph-based schemes offer probabilistic security guarantee. They do not consider adversarial packet losses caused by attackers. RSA Digital Signature Algorithm can be used for authentication [15]. This scheme is prone to replay attacks and requires use of the RSA signature algorithm which is very costly in terms of processing time.

Each of the schemes for multicast authentication present today suffers from one at least of these drawbacks:

- They are computationally very expensive hence they cannot be used with a wide variety of devices. With the widespread use of PDAs, wireless devices, Internet enabled mobile phones and other low power devices with limited re-sources it becomes infeasible to use computationally intensive techniques for the kinds of applications multicast is targeting such as stock updates.
- They introduce a large overhead communication overhead. With the ever in-creasing use of audio and video streams bandwidth is limited. Wireless net-works also have limited bandwidth. It is unacceptable to clog the network with enormous communication overhead.
- They require time synchronization between the sender and the receivers. With the receivers located at different locations for applications such as online stock quotes, pay-per-view TV, etc., it becomes difficult to maintain time synchronization between the sender and the receivers. Wireless devices are mobile and their distances changes, which changes the network propagation delay frequently. This requires constant resynchronization of the sender and the receivers making it impossible to maintain the service efficiently.
- They are prone to collusion attacks. While senders are few (in most cases it is only a single sender group) and are highly trusted, the receivers cannot be trusted not to form collusions and share keys. Hence it is unreasonable to accept a solution where receivers can spoof one another.

None of these schemes uses the fact that multicast data for commercial purposes will be protected such as pay-per-view TV, etc., so that people without ac-cess cannot see the multicast traffic. This knowledge can be used to add a simple robust multicast packet authentication scheme to any group communication scheme using asymmetric keys.

3. MULTICAST AUTHENTICATION

Multicast authentication is a challenging problem because of the large number of participants involved in the communication and the need to authenticate a large number of packets any of which may be lost. The simplest solution to authentication is if each packet is signed by the sender. The receiver could verify the signature and discard packets whose signatures were not verified. However this solution is unacceptable because of the repeated use by the sender of a computationally ex-pensive sign primitive for each packet and the communication overhead caused by the addition of a signature to each packet. Another solution to authentication is to use hash based message authentication codes (HMACs) as in unicast transmission. This does not work well in a multicast setting if small overhead is required and time synchronization between sender and receivers is difficult to maintain. The use of simple symmetric message authentication codes (MACs) is unacceptable as it would enable any of the receivers to impersonate the sender. Because the stream of multicast packets is implemented using UDP and the loss of packets is accept-able in many applications such as pay-per-view TV, the authentication scheme must tolerate packet loss. Also participants may join at any point in time and they should be able to begin authenticating packets starting from any packet. It is possible to compute signatures in advance and to buffer them for content that is al-ready available, such as applications showing movies over the Internet. However real time multicast applications such as stock and news feeds cannot use signatures that take a long time to compute.

All of the schemes discussed in Section 2 can only perform group authentication and not individual sender authentication. Therefore a sender in a multicast group may be able to masquerade as another sender. However as senders are few in number for large multicast scenarios (in most cases the group is a single sender group) and are trusted, they are not likely to be masquerading as other senders. Receivers are large in number and are not trusted and they may disrupt service by masquerading as a sender. Any multicast authentication scheme must prevent receivers from posing as a legitimate sender. However, it is acceptable to assume that senders will not be posing as one another.

An efficient authentication scheme must

- be able to ensure that the received packets actually originated from a legitimate sender.

- be able to ensure that the received packets were not modified while in transit.
- be able to ensure that packets sent by an adversary are not mistaken as legit-mate packets.
- work with both real time and previously available content.
- be able to ensure that receivers are not capable of posing as senders.
- be able to provide authentication from any packet onwards despite losses in the network.
- reduce the communication overhead by adding a small number of bytes per packet
- not be computationally very expensive

The authentication schemes discussed in Section 2 do not satisfy all of the above mentioned requirements hence there is a need for a multicast authentication scheme.

4. SIM-KM

SIM-KM [6] is an efficient key management scheme to distribute and change keys as required [16] for secure group communication. While we provide a description of SIM-KM to show how asymmetric keys are used for multicast data protection the proposed authentication scheme may be used with any asymmetric multicast data protection scheme to provide multicast group authentication. SIM-KM uses proxy encryptions [16] to split the multicast distribution tree into subgroups when needed. A re-key message combines the subgroups to form a single distribution tree. One node is configured as the Group Manager. It is configured with group and access control information. Other group controllers will join this group manager. Trusted intermediate group controllers may perform proxy transformations. These group controllers are on the data distribution tree with the sender as the root and the receivers as the leaves.

SIM-KM is robust and does not depend on any single controller for its operation. Proxy Functions can convert cipher text for one key into cipher text for an-other without revealing secret decryption keys or clear text messages. This allows a non-trusted third party to convert between cipher texts without access to the clear text message or to the secret component of the old key or new key.

In simple terms, the key generation algorithm generates keys for encrypting the multicast data. The senders encrypt the multicast data using the traffic encryption algorithm and the receivers decrypt the received cipher text using the traffic decryption algorithm. Whenever an intermediary entity wants to change the cipher text (for example, when a membership change occurs in the sub-tree below this intermediate node) it uses the traffic to proxy changing algorithm. The receiver then has to use the proxy decryption algorithm to recover the original clear text. SIM-KM solves the problem of key changing on each membership change by introducing proxy encryptions. However this asymmetric proxy encryption technique differs from other asymmetric key techniques. In other schemes using asymmetric keys the encryption key is freely available and the decryption key is kept a secret so any node can send a secret message to the node with the decryption key. In SIM-KM the encryption key is kept a secret and the decryption key is made avail-able to a selected group of receivers. Fig. 1 illustrates the functioning of SIM-KM.

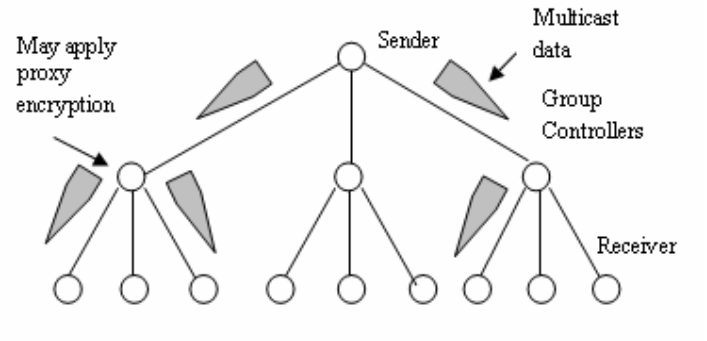

Fig. 1. Functioning of SIM-KM

SIM-KM uses asymmetric proxy functions and keys. The sender has the encryption key, the intermediate nodes have the proxy transformation key and the receivers have the decryption key. El Gamal, RSA or identity based encryption schemes may be used as proxy encryption schemes. It should be noted that SIM-KM does not use the keys as is done in a public key encryption scheme where the encryption key is made freely available to anyone who wants to send data to a particular receiver. In SIM-KM the encryption key is kept a secret and is known only to senders of the multicast data. To ensure that it is not mistaken for a public key encryption scheme we refer to the scheme as an asymmetric scheme. Here we discuss an encryption scheme based on El Gamal encryption. This example is different from [6] because for certain traffic encryption approaches (including

El Gamal) collusion between the intermediate entity and receivers can permit discovery of the encryption key. However, this can be avoided by giving a part of the decryption key to the sender and requiring it to perform a partial decryption before sending the data. This ensures that there is no possibility of deriving the encryption key as shown in the following example:

Let p be a prime, and g be a generator of $Z_p = \{1, \ldots, p - 1\}$. The private key x is an integer between 1 and p – 2. Let $y = g^x \bmod p$. The public key for El Gamal encryption is the triplet (p, g, y). To encrypt a plaintext M, a random integer k relatively prime to p – 1 is selected, and the following pair is computed:

$$a \leftarrow g^k \bmod p \tag{4.1}$$

$$b \leftarrow My^k \bmod p \tag{4.2}$$

The cipher text C consists of the pair (a, b) computed above. The decryption of the cipher text C = (a, b) in the El Gamal scheme, to retrieve the plain text M is simple:

$$M \leftarrow b/a^x \bmod p \tag{4.3}$$

In the above expression, the "division" by a^x should be interpreted in the con-text of modular arithmetic, that is, M is multiplied by the inverse of a^x in Z_p. The correctness of the El Gamal encryption scheme is easy to verify. We have

$$b/a^x \bmod p = My^k(a^x)^{-1} \bmod p \tag{4.4}$$

$$= Mg^{xk}(g^{kx})^{-1} \bmod p \tag{4.5}$$

$$= M \tag{4.6}$$

In the Proxy El Gamal encryption scheme the private key is split into x_1 and x_2 such that $x = x_1 + x_2$. This split can be made when required and there can be a very large number of such possible splits resulting in different values of x_1 and x_2. The sender is given x_1. The sender transforms the data using x_1 after encryption. This is done to ensure that no collusion of intermediate nodes or receivers will succeed in obtaining the decryption key x. Hence there is no possibility of using the decryption key and other information to obtain the encryption key. This ensures that the receivers are unable to impersonate the sender. X_2 is further split into x_3 and x_4 such that $x_2 = x_3 + x_4$ when a membership change occurs and the decryption key for a part of the multicast data distribution tree has to be changed. The traf-

fic to proxy changing algorithm receives x_3 and the receiver receives x_4. The traffic to proxy changing function and the decryption function are similar to the original decryption function under x_3 and x_4 respectively. The sender performs a partial decryption given by

$$M_1 \leftarrow b/a^{x1} \bmod p \tag{4.7}$$

The traffic to proxy changing function thus performs a partial decryption given by

$$M_2 \leftarrow M_1/a^{x3} \bmod p \tag{4.8}$$

The decryption function performs the decryption by performing

$$M \leftarrow M_2/a^{x4} \bmod p \tag{4.9}$$

The correctness of the Proxy El Gamal encryption function is easy to verify. We have

$$M_2/a^{x4} \bmod p = (M_1/a^{x3} \bmod p)/a^{x4} \bmod p \tag{4.10}$$

$$= (b/a^{x1} \bmod p)/a^{x3 + x4} \bmod p \tag{4.11}$$

$$= (b/a^{x1} \bmod p)/a^{x2} \bmod p \tag{4.12}$$

$$= b/(a^{x1 + x2)} \bmod p \tag{4.13}$$

$$= b/a^{x} \bmod p \tag{4.14}$$

These proxy encryption schemes have been shown to be as secure as the original schemes [17]. SIM-KM also employs other techniques such as group controllers with varying trust levels and different responsibilities. The scheme also employs dynamic batching techniques to reduce the number of control messages. SIM-KM has been shown to have good performance when compared to other available key management schemes [18].

5. AUTHENTICATION

To provide authentication, a symmetric key is shared among all group members. This symmetric key is a unique shared secret used for authentication. Every time a sender wants to send a message to the group, it adds an index to the packet, a counter “c” and a random number “k”. The index is a number assigned by the Group Manager to a particular sender for uniquely identifying it during a multicast session. It then encrypts the

packet with the asymmetric encryption key. It then calculates a MAC (Message Authentication Code) on the cipher text using the symmetric key. It then attaches “k” and the MAC to the packet.

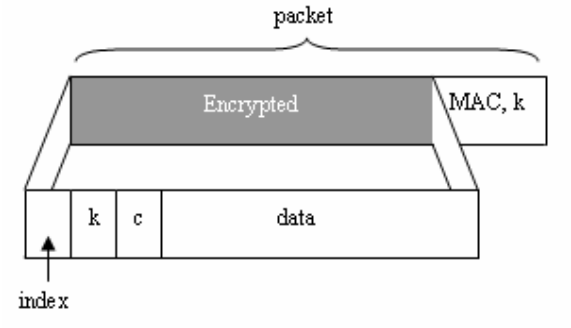

Fig. 2. Packet Structure

The packet structure is shown in fig. 2. The receiver on receiving the packet computes a MAC to ensure that the packet was not modified in transit and was not sent by someone imitating to be the sender. This however does not rule out the possibility of one of the malicious receivers masquerading as a sender. The packet is then decrypted and the value of “k” obtained after decryption is matched with the value of “k” sent in clear text with the packet. As the packet is encrypted by an asymmetric key which is unknown to the receivers there is no possibility that a malicious receiver can create a packet which when decoded produces the same value of “k” without knowing the encryption key. This effectively rules out the possibility of any receiver being able to impersonate a legitimate sender. The index identifies the particular sender providing data source authentication. The receiver stores the last value of the counter received in sequence as c1 and compares it with the value of the counter in the next packet that arrives to prevent a replay attack. The newly arrived packet must fall within the range c1 to c1 + r, where r is the size of the buffer used by the underlying protocol to re-sequence the out of order packets. If the packet already exists in the buffer then the packet is considered a duplicate and is discarded. The symmetric key is generated by the Group Manager and distributed to the group members when they join the multicast group. It may be given along with the decryption key when the members join. The scheme uses a single MAC for message authentication along with a random number, a counter and an index. This reduces the bandwidth overhead considerably and it is not as computationally intensive as digitally signing individual packets. The type of MAC and the size of the symmetric key

can be chosen depending upon the context of the application and the sensitivity of the data.

When SIM-KM is used as the key management scheme, the symmetric key is also given to the group controllers. When group controllers perform the data trans-formation they re-compute and replace the existing MAC. The random number “k” is kept as it is. The decryption key is changed by SIM-KM and is communicated to the receivers who need the new key. The symmetric key for authentication remains the same. If another key distribution scheme is being used then there is no need to re-compute and replace the MAC.

6. ATTACKS

This scheme provides message integrity as it allows the receiver to verify that the message is exactly the same as when the sender sent it. Host authentication is also achieved as it allows the sender to be uniquely identified. A number of different attacks [19] are possible against group communications. We describe some of the attacks that are relevant to message authentication and how the proposed scheme would handle these attacks.

6.1 Replay

An adversary may store messages and then send them at a later time. As these packets have been assembled by a legitimate sender, the receivers may be led to believe that they are legitimate packets even though now they are out of sequence. This attack is nullified by having a counter. If packets are replayed then the value of the counter in the packets will be out of the range defined for c and these packets can be discarded.

6.2 Message Modification

An adversary may modify messages that are sent by the sender. The symmetric MAC authentication will fail if the message is modified. As only group members have the symmetric key a non-group member cannot modify the message and attach a MAC that succeeds. A receiver may be able to masquerade as a legitimate sender by modifying the packet and attaching a MAC that succeeds but the random number “k” will not match. There is no way for any adversary or malicious receiver to make a packet

with a matching value of “k” as it does not possess the asymmetric encryption key, which is a secret known only to the sender(s).

6.3 Message Insertion

An adversary may insert messages in the stream. There is no way for an adversary to create messages which will be authenticated as legitimate packets. Inserted messages will fail authentication and will be dropped by the receivers.

6.4 Message Deletion

An adversary may delete a message from the stream. Deletion attacks are not ad-dressed by this scheme. It only deals with validation messages that are not deleted. However deleting some messages does not stop the authentication scheme from validating packets that have been delivered successfully.

6.5 Eavesdropping

This paper deals with data source authentication and not with confidentiality but the SIM-KM scheme uses proxy encryptions to secure data. It uses asymmetric keys and only receivers with legitimate decryption keys will be able to decipher the packet.

6.6 Denial of Service

A malicious node can insert packets into the stream to slow down the receiver thus mounting a denial of service attack as the receiver will have to verify the packets. A node that is not part of the multicast session will not be able to make packets with matching MACs and thus will not be able to slow down the receiver as the MAC verification process does not involve decrypting the packet. A malicious receiver can slow down another receiver by inserting packets with matching MACs as it knows the symmetric key but the packet will still be detected as a bogus packet. A malicious sender can send packets to receivers because it possesses the encryption key as well. In this case the denial of service attack cannot be mitigated and the only solution would be to have different encryption keys for different senders. However as discussed in Section 3 senders are

trusted entities and are not likely to be disrupting the service themselves. Consider a scenario where multicast is used for conferencing where there are multiple non-trusted senders and authentication is required for such a case then the only solution is to have a separate key pair for each sender. Our scheme would work well is such as case as well. The number of key pairs required will be small as there are few senders.

7. CONCLUSION

In section 2 we have discussed how existing multicast authentication schemes fail to satisfy all the requirements for a multicast authentication scheme. Let us consider a scenario where multicast is used to deliver stock updates to a large number of receivers. Some of these receivers may be PDAs, wireless handsets with limited resource and small bandwidths hence it rules out the possibility of using computationally intensive schemes or schemes with high communication overhead. We compare our scheme with the Multiple MACs scheme, TESLA and Merkle hash schemes. We have not compared schemes that do not tolerate packet loss, are prone to message insertion attacks, etc., because these solutions are not suitable for use in the Internet.

Table 1. Overhead Comparison

Scheme	Overhead per packet	Need Synch.	Comparison
Multiple MACs Scheme	k bits where k depends on the size of the largest malicious receiver coalition	No	Suffers from collusion attack, needs to calculate k MACs before sending out every packet
TESLA	MAC + K_m where K_m is sent once every time period	Yes	Suffers from change in network propagation delay
Merkle Hash Scheme	n*(s + h log n) where n is the number of packets in the data stream, s is the signature size and h is the hash size	No	Suffers from Signature Flooding Attack, Adversarial Packet Loss Attacks, all messages need to be known in advance on sender side
Proposed Scheme	MAC + counter + 2*random number	No	Requires a single MAC computation, does not suffer from known attacks

As seen from table 1 the overhead caused by our proposal is either smaller or comparable when compared to different schemes. It does not suffer from collusion attacks or adversarial packet loss attacks, and does not require any time synchronization. It also does not require multicast data to be known in advance and works well with real time data. Given the basic traffic encryption scheme, this scheme adds only the overhead of a MAC computation. The proposed scheme works better than existing schemes when other scenarios are considered such as Pay-per-view TV, online teaching, news feeds, etc.

8. VALIDATION

We have used PROMELA (PROcess MEta LAnguage) [20] to specify the validation model and then used a tool, SPIN (Simple Promela INtepreter) [21] to validate our model. The model is designed so that it is simple but considers all the at-tacks listed in section 6. The model consists of one sender, one intermediate adversary and a receiver. The sender sends data packets. The intermediate adversary randomly modifies messages, inserts new messages, deletes messages and re-plays messages. Modified messages include messages with wrong MACs, wrong value of “k” in clear text and altered data. Inserted messages include messages that were simply created and added to the data stream as well as messages with correct MACs. Replayed messages are messages that were saved from the data stream and were added to the stream after different time periods. The probability with which the intermediate adversary introduces errors can be controlled. The receiver verifies each packet received to establish if the packet has been sent by the sender or has been inserted/modified/replayed by the adversary.

XSPIN is used to specify the high level model written in PROMELA. As a preliminary check different random simulations were performed with different SPIN options and no errors were found. The verifier was compiled using the exhaustive search option. This option causes a state space search of all possible states and message timings of the modelled processes. Then, the verifier was executed and the output confirmed that our model is free from errors and there are no assertion violations, invalid end states or unreachable states in the design.

We compared the packets modified by the intermediate adversary to those detected by the receiver as packets having errors. In each case it was

found that the packets that were randomly inserted, modified or replayed by the adversary were successfully detected by the receiver. The probability of errors introduced by the intermediate adversary was varied from 0 to 100% and it was found that the receiver was able to detect errors independent of the number of messages the adversary modified, inserted, replayed or deleted. The model was found to be free from errors, assertion violations, invalid end states or unreachable states in all cases.

9. CONCLUSIONS

We have presented an efficient robust scheme for multicast group authentication. It can be used with any asymmetric multicast data security protocol for multicast group security (such as SIM-KM) to perform multicast data source authentication. The authentication scheme has no delays, requires no time synchronization, is collusion-resistant, and does not have a large packet overhead. It is simple, works in-dependent of packet losses and is not resource intensive. It is a robust, efficient and scalable solution for multicast data source authentication. The scheme is flexible and does not place any restriction on which asymmetric encryption scheme to use or on the size of the keys and the choice of MACs. The scheme was modelled using PROMELA and validated using SPIN, which showed the scheme is successful in the face of all possible known message authentication attacks.

ACKNOWLEDGEMENTS

Ritesh Mukherjee acknowledges the support of the Natural Sciences and Engineering Research Council (PGS B) scholarship.

J. W. Atwood acknowledges the support of the Natural Sciences and Engineering Research Council of Canada through its Discovery Grants Program and of Concordia University.

REFERENCES

[1] Harney H, Muckenhim C (1997) Group Key Management Protocol (GKMP) Architecture. RFC2094

[2] Harney H, Muckenhim C (1997) Group Key Management Protocol (GKMP) Specification. RFC2093
[3] Ballardie A (1996) Scalable Multicast Key Distribution (SKMD). RFC1949
[4] Mittra S (1997) Iolus: A Framework for Scalable Secure Multicast. In: ACM SIGCOMM '97, pp 277–288
[5] Fiat A, Naor M (1993) Broadcast Encryption. In: CRYPTO '93, Springer-Verlag, pp 480–491
[6] Mukherjee R, Atwood JW (2003) Proxy Encryptions for Secure Multicast Key Management. In: IEEE LCN'03, pp 377–384
[7] Mukherjee R, Atwood JW (2004) SIM-KM: Scalable Infrastructure for Multicast Key Management. In: IEEE LCN'04, pp 335–342
[8] Canetti R, Garay J, Itkis G, Micciancio D, Naor M, Pinkas B (1999) A Taxonomy and some Efficient Constructions. In: IEEE INFOCOM'97, pp 708–716
[9] Perrig A, Tygar JD, Song D, Canetti R (2000) Efficient Authentication and Signing of Multicast Streams over Lossy Channels. In: IEEE Symposium on Security and Privacy, pp 56–73
[10] Gennaro R, Rohatgi P (1997) How to Sign Digital Streams. In: CRYPTO'97, pp 180–197
[11] Perrig A (2001) The BiBa One-Time Signature and Broadcast Authentication Protocol In: 8th ACM Conference on Computer and Communications Security, pp 28–37
[12] Wong CK, Lam SS (1998) Digital Signatures for Flows and Multicasts In: 6th International Conference on Network Protocols, pp 502–513
[13] Pannetrat A, Molva R (2003) Efficient Multicast Packet Authentication. In: 10th Annual Network and Distributed System Security Symposium
[14] Golle P, Modadugu N (2001) Authenticating Streamed Data in the Presence of Random Packet Loss. In: 8th Annual Network and Distributed System Security Symposium (NDSS)
[15] Weis B (2005) The Use of RSA/SHA-1 Signatures within ESP and AH. Internet Draft, Work in Progress, draft-ietf-msec-ipsec-signatures-06.txt
[16] Wallner D, Harder E, Agee R (1997) Key Management for Multicast: Issues and Architecture. RFC2627
[17] Dodis Y, Ivan A (2003) Proxy Encryption Revisited. In: 10th Annual Network and Distributed System Security Symposium
[18] Mukherjee R, Atwood JW (2004) A Comparative Analysis of SIM-KM for Group Key Management. In: CCN 2004, pp 169–174
[19] Rescorla E, Korver B (2003) Guidelines for Writing RFC Text on Security Considerations. RFC3552
[20] Holzmann GJ (1991) Design and Validation of Computer Protocols. Prentice Hall.
[21] Holzmann GJ (1997) The Model Checker SPIN. IEEE Transactions on Software Engineering, 23, 5:279–295