A source authentication and data confidentiality scheme for multicast networks

Abderrahim Benmachiche,[1] Salami Ali,[2] Azeddine Messikh[1]

[1]Department of Computer Science, Faculty of Information and Communication Technology, IIUM, Kuala Lumpur, Malaysia.
[2]Department of Computer Science, Faculty of Computer Science and Information Technology, Shaqra University, Arabia Saudi.

Abstract**—** Source authentication and data confidentiality are needed in many multicast applications. Most research attempted to solve either the source authentication or the confidentiality of data. This paper proposes a scheme that can treat the two problems together. Our scheme uses the TESLA protocol to ensure the source authentication and XOR encryption to ensure the confidentiality of the transmitted information so that the scheme can be applied to low bandwidth applications without requiring high computation devices. Furthermore, it tolerates packet-loss. These advantages render the proposed scheme secure in terms of authentication and confidentiality and it does not cost much.

Keywords— Authentication, confidentiality, multicast applications.

I. Introduction

The number of Internet users has increased dramatically in recent years. This motivates researchers to introduce new network services, such as multicast communication to manage the huge number of users. Multicast is a reliable communication mechanism for group-oriented applications such as Internet video transmissions, news feeds, stock quotes, software updates, live multiparty conferencing, on-line video games and shared whiteboards [1,2,3]. However, these services are plagued by numerous security issues concerning the secrecy of the transmitted data (confidentiality) and the credibility of the group members (authentication).

Most research on the issue has either addressed the confidentiality or authentication problem while some applications require both confidentiality and authentication, such as pay-per-view and video conferencing.

This paper offers a solution for source authentication and data confidentiality problems on the multicast network based on shared key mechanism and TESLA protocol [4,5], where all the recipients have a common shared key to decrypt the enciphered messages and get a key periodically from the source to authenticate the received messages.

II. Related work

In this section, we present previous works in multicast security and introduce the advantages and drawbacks for each of these protocols.

A. *One- way Chains*

The one-way chain generates a chain of secret keys. To create a chain of length n, the sender picks a random key K_k which is the first element of the chain. Next, he starts producing the rest elements of the chain by applying the one-way chain function H. Every time he applies the hash function H he obtains a new key as depicted in figure 1. The only condition for this method is the first transmitted key K_0 should arrive to the recipients in a secure way. Then, the receivers can check the credibility of each key K_i from the chain by using the following formula $H^i(K_i) = K_0$. If any key K_f belonging to the chain is lost, it can be recovered by using the last received key K_l as follow $K_f = H^{l-f}(K_l)$. Many researchers used this method to ensure the authentication of sent messages [6, 7] and some treated this problem in multicast networks.

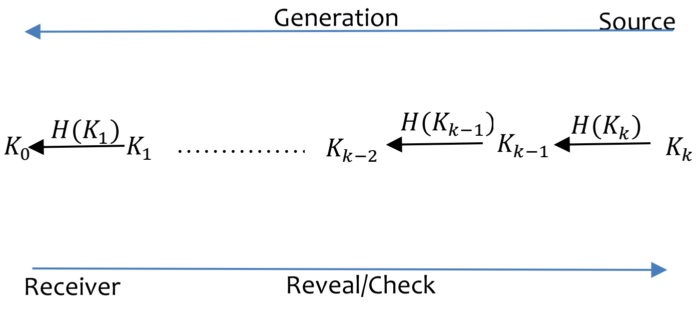

Fig. 1 The process of generating a One-way chain

B. *Bergadano et al. protocol*

Bergando et al. (2000) sought to authenticate the packets by using an MAC function (message authentication code) and the keys produced by the one-way chain function [8]. The characteristics of the one-way chain function allow the receivers to recover the lost keys and to check the validity of the received keys. It also ensures that no one from the multicast group can forge packets and send them to the other multicast members on behalf of the source; the source keeps the key K corresponding to the packet P secret until it guarantees that the packet P arrived to all multicast members.

The sender performs two steps. First, he has to broadcast the message $\mathrm{P_i}$ attached with its digest $\mathrm{MAC(K_i, P_i)}$ at time T. Next, after $\mathrm{T + delay}$, he has to disclose the key $\mathrm{K_i}$ corresponding to the packet P_i. The $delay$ is the period of time within which the packet $\mathrm{P_i}$ must arrive to the all recipients. The receivers have to buffer the packet $\mathrm{P_i}$ and wait for the corresponding key K_i to decide if the packet is trusted.

The advantages of Bergadano et al.'s protocol are that it requires only a single MAC to authenticate each packet, and it tolerates packet-loss. However, this protocol is not suitable for those applications that need to ensure the confidentiality of data as an addition to the authentication process. If the disclosed key $\mathrm{K_i}$ arrives late to the recipients, it will introduce other problems causing receivers to distrust this key and reject the buffered packet $\mathrm{P_i}$.

C. *The TESLA Protocol*

Perrig et al. (2002) designed the TESLA (Time Efficient Stream Loss-tolerant Authentication) protocol [4,5]. Their proposal is similar to that of Bergadano et al.'s protocol in that both use the one-way chain to generate the MAC keys. The difference between these two protocols is that the TESLA protocol uses a different key in each interval to authenticate the transmitted messages [3]. Figure 2 clarifies the principles of the TESLA protocol.

The TESLA protocol reduces the authentication size to one MAC per packet and unlike Bergando et al. it uses a different key for each time interval. It also tolerates packet-loss so that it does not affect the authentication process. As for its drawbacks, TESLA does not treat the confidentiality problem of the transmitted data. Thus, it is not appropriate for those applications that require confidentiality of information. Moreover, all the receivers need to be synchronized with the source. Otherwise, a potential security hole will be created. Furthermore, the latency in the receivers' side renders this protocol not suitable for real-time applications.

III. The proposed scheme

For authentic and confidential messages on a multicast network; a reliable protocol was proposed basing on TESLA protocol and the simple symmetric encryption XOR. The reasons for using TESLA protocol are (1) the protocol is robust to packet-loss, (2) the protocol can work on low bandwidth communication, and (3) the protocol is suitable for low computation power devices. The XOR encryption is chosen due to its speedy encryption and decryption of data.

Nevertheless, the protocol requires certain conditions: (1) the source and the receivers should be loosely time-synchronized [9,10], (2) the source and receivers have to store some messages, and (3) the key used for the XOR encryption has to have already been safely shared between the source and the receivers (e.g. the BB84 quantum key distribution protocol) [11].

A. *The Outline of TESLA Confidential Protocol*

(1) The sender divides the time of transmitting into small equal intervals. Next, the sender uses a hash function to generate a one-way chain of keys, and he assigns each key to one-time intervals, in which, the last generated key is assigned to the first time interval and the penultimate generated key is assigned to the second time interval. The sender then defines a discloser time for the one-way chain values.

(2) The sender computes the cipher of each message using the XOR encryption and the shared key. Then, he computes the MAC of the obtained cipher using one of the chain keys that corresponds to the time interval in which the message will be sent. Finally, the sender constructs the packet by concatenating the cipher and the MAC along with disclosing the recent key.

(3) The receiver knows the disclosed time so when he receives the packet he can decide whether the packet should be authenticated or should be buffered until receiving the corresponding disclosed key.

(4) When the receiver receives the corresponding key, he checks its validity by using a one-way chain function of the source and compares the obtained key with the previously disclosed key. After that, the receiver checks the correctness of the MAC. If the MAC is correct the receiver decrypts the cipher otherwise, he discards the packet.

B. *The Sender Side*

Before broadcasting

The sender splits the expected transmitting time into N small uniform time intervals $\mathrm{T_{int}}$. The first time interval of transmitting stars at $\mathrm{T_0}$ and the second starts at $T_1 = T_0 + T_{int}$ and so on. Next, the sender generates a random key $\mathrm{K_n}$ by using a pseudo-random function and uses a hash function H to produce the remaining keys of the one-way chain. The source assigns one key from the chain recursively to each interval of time in which K_n will be attached to T_0, K_{n-1} will be attached to T_1, etc.

During Broadcasting

(1) First, the source agrees with the receivers about a key disclosure delay d which is in the order of the time interval T_{int}.
(2) Next, the source computes the cipher text C_j of the message M_j by using the shared key K_{sh} as follow: $C_j = M_j \oplus K_{sh}$.
(3) Then, the sender calculates the MAC of the cipher text C_j by the key K_i corresponding to the interval *i*.
(4) The source encrypts the recent disclosed key K_{i-d} by the shared key K_{sh} as follow: $K_{i-d} \oplus K_{sh}$
(5) Finally, the sender constructs the packet P_j by concatenating C_j, $MAC(C_j, K_i)$ and $K_{i-d} \oplus k_{sh}$ As described in the equation bellow.

$$P_j = \{C_j \| MAC(C_j, k_i) \| K_{i-d} \oplus K_{sh}\}$$

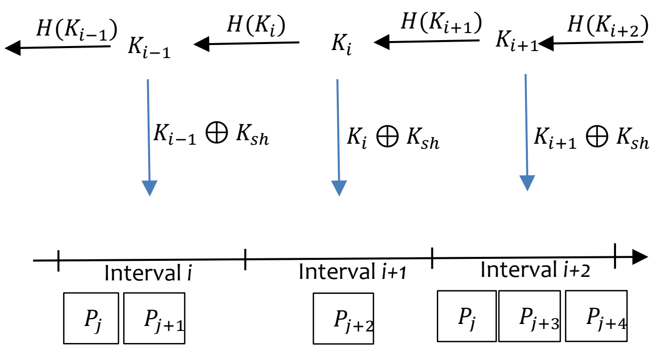

Fig. 2 Adding the confidentiality feature to TESLA protocol

```
Sender Algorithm
t =$t_n$
k=Rand ($2^{64}$)      //generate a random number of length $2^{64}$bits

while t ≥ $t_0$
    OneWayChain (t)= k      // create one-way chain of n+1 keys
    k= H(k)      //H is a hash function
    t=t - int
end while

while Time() ≤($t_n + int$)      // Time: returns the current time
    Load (M)
    C=M⊕$K_{sh}$
    D=MAC(C, CorrespondingKey(Time()))
    $K' = CorrespondingKey(Time() - d)$
    P=C{$C \| D \| K' \oplus K_{sh}$ }
    Send (P);
end while
```

Fig. 3 The sender's algorithm

C. The Receiver Side

When the receiver receives the packet P_j:

(1) First, he decrypts the key appended to the packet ($K_{i-d} \oplus K_{sh}$) by the shared key K_{sh}.
(2) Second, he compares the obtained key with the precedent safe key that was revealed. If they are identic, then the receiver should go to step (3) unless he has to jump to step (4).
(3) In this step, the receiver verifies the expiration date of this key, by checking that the time of receiving the current package belongs to the same time interval *i* of the previously received key. If it does he should buffer the packet until receiving the corresponding disclosed key, otherwise, he has to remove the packet because it was a bogus message created by one of the multicast members.
(4) In this step, the receiver uses the hash function *H* to verify that the current received key generated the previous received key. If it does, then he authenticates and decrypts the buffered packets unless he must reject this packet since a fake key created it.

```
Receiver Algorithm
$K_c = K_{i-d} \oplus K_{sh}$
If $K_c = K_p$      // $K_p$, $K_c$ are the previous and current disclosed keys
    $T_c$=ReceivedTime($K_c$)      // ReveivedTime(K) returns the time
    $T_p$=ReceivedTime($K_p$)      when key K was received
    If ( $T_c, T_p \in Interval\ i$)
        Buffer the received packet $P$;
    else
        Reject the packet $P$;
    end if
else
    If $K_p = H(K_c)$
        Authenticate the buffered packets by using $K_c$;
        Decrypt the obtained messages by using $K_{sh}$;
        $K_p = K_c$
    else
        Eject the packet $P$;
    end if
end if
```

Fig. 4 The receiver's algorithm

IV. Discussion

In this work, we attempted to propose a secure pattern that address the confidentiality and authentication issues in multicast networks. We combined a simple asymmetric encryption technique (XOR) to ensure the confidentiality of the transmitted data with the TESLA protocol to confirm that the received messages at the recipients' side come from a trusted source. However, of all the schemes that have been proposed, no one scheme has been able to fulfil the properties that should be possessed by any multicast authentication scheme for static as well as mobile networks. In detail, there exists around thirteen properties. However, in this paper, we mention those critical to any multicast network. A perfect multicast security scheme should cover a robust, scalable and fast authentication mechanism with DoS resilience, intelligence, minimal storage requirements, time synchronization and jamming resistance. Some of these important properties we did not treat are the fast authentication and minimal storage requirements because every time we need to buffer the received messages until receipt of the corresponding disclosed key. Our blueprint does not provide resistance against jamming because we have used one-way chain function that can allow the disclosed keys to be lost during the jamming periods. This is a drawback of the proposed model that requires further research.

V. Conclusion

Many protocols have been proposed to accomplish multicast security. However, most treat either the authentication or the confidentiality of data, whereas, some multicast applications require both aspects. To deal with this problem, we proposed a reliable scheme. This scheme relies on the TESLA protocol and the simple encryption XOR. The features of this scheme are tolerating packet-loss, requires low bandwidth communication and is suitable for low-performance devices. We conclude that this protocol is secure and efficient for multicast applications.

Acknowledgment

This work is supported by MOSTI (Malaysia) under the e-science Project No. (01-01-08-SF0148).

References

[1] R. Canetti, J. Garayt, G. Itkid, D. Micciancios, M. Naore, and B. Pinkasll, “Multicast Security: A Taxonomy and Efficient Constructions,”, INFOCOM, 1999.

[2] T. Hardjono and G. Tsudik, “IP multicast security: Issues and directions,” Annales de Telecom., 2000.

[3] Y. Challal, H. Bettahar, and A. Bouabdallah, “A taxonomy of multicast data origin authentication: Issues and solutions,” IEEE Commun. Surv. Tutorials, vol. 6, pp. 34-57, 2004.

[4] A. Perrigy, R. Canettiz, J. Tygary, and D. Songy, “Efficient Authentication and Signing of Multicast Streams over Lossy Channels,” IEEE Symp. Security and Privacy, 2000.

[5] A. Perrig, R. Canetti, J. Tygar, and D. Song, “The TESLA Broadcast Authentication Protocol,” RSA CryptoBytes, vol. 5, 2002.

[6] L. Lamport, “Password authentication with insecure communication.” Commun. ACM, vol. 24, no. 11, pp. 770-772, 1981.

[7] N. Haller, “The S/Key™ One-time Password System,” ISOC Symp. Network and Distributed Security. Rep. 99-02, 1994.

[8] F. Bergadano, D. Cavagnino and B. Crispo, “Individual Single- Source Authentication on the Mbone,” IEEE Int’l. Conf. Multimedia and Expo,2000.

[9] M. Reiter, “A security architecture for fault-tolerant systems,” Ph.D Cornell University, 1993.

[10] M. Reiter, K. Birman, and R. van Renesse, “A security architecture for fault-tolerant systems,” ACM Trans. Comput. Syst, vol. 12, no. 4, pp. 340-371, 1994.

[11] C. Bennet and G. Brassard, “Quantum cryptography: public key distribution and coin tossing,” Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, 1984.