International Journal of Educational Researchers

Volume 9, Issue 4 December 2018

ijer.penpublishing.net

ISSN: 1308 - 9501 (Print)

Implementation of Broadcast Authentication Protocol For Multimedia&nbsp;Communication

A.Wasim Raja, M.Praveen Kumar & S. P. Santhoshkumar

To cite this article

Raja, A., Kumar, M. & Santhoshkumar, S.P. (2018). Implementation of Broadcast Authentication Protocol For Multimedia&nbsp;Communication. International Journal of Educational Researchers , 9(4), 20-24.

Published Online	December 22, 2018
Article Views	3 single - 3 cumulative
Article Download	24 single - 35 cumulative

Pen Academic Publishing, Canakkale/Turkey
Telephone: +90 286 243 06 66 | Fax: +90 286 213 08 00 | info@penpublishing.net | www.penpublishing.net

http://www.eab.org.tr

Educational Research Association
The International Journal of
Educational Researchers 2018, 9(4): 20-24
ISSN: 1308-9501

http://ijer.eab.org.tr

Implementation of Broadcast Authentication Protocol For Multimedia Communication

A.Wasim Raja[1]
M.Praveen Kumar[2]
S.P.Santhoshkumar[3]

Abstract

One of the key challenge of securing broadcast statement is source authentication, that is enabling receivers of broadcast facts to validate that the received data actually originates from the claimed source and was not modified on the way. This problem becomes more complex in real time communication where other receivers of the data are not trusted and where lost packets are not retransmitted.This application is mainly developed for assuring authenticity of multimedia data. It is implemented in java using java media arrangement to send audio video information over real time transport protocol. Message authentication code (MAC), which is a symmetric cryptographic primitive, is used for providing authentication. Delayed key disclosure technique is used to overcome the drawback of MAC. Elements of one-way hash chains are used as keys in order to implement the above technique. This is a capable protocol with low communication transparency, tolerates packet loss and it is primarily based on loose time harmonization between the sender and the receivers.

Keywords: Authentication, Broadcast, Multimedia, Delayed Key

[1] Assistant Professor, Department of Computer Science, Rathinam Technical Campus, Coimbatore, Tamilnadu, India.

[2] Assistant Professor, Department of Information Technology, Rathinam Technical Campus, Coimbatore, Tamilnadu, India.

[3] Assistant Professor, Department of Computer Science, Rathinam Technical Campus, Coimbatore, Tamilnadu, India.

1. Introduction

Now a days for large-scale data dissemination, we use 'broadcasting', that is simultaneous transmission of data to multiple receivers, as the mode of communication. There exits lot of broadcast distribution networks like satellite broadcasts, wireless radio broadcast. While these networks distribute data to multiple receivers efficiently they also encounter a problem called 'packet injection attack', where a malicious user impersonates the sender and injects broadcast packets. So the receiver has to ensure that the broadcast packets they receive really originate from the claimed source. A broadcast authentication protocol enables the receivers to verify that a received packet was really sent by the claimed sender. Sending audio video message is a common demand. Assuring authenticity of media data by cryptographic techniques is quite often neglected. As a result the degree of trust in audio video information sent over public networks is limited since there is no proof that the received information was not altered by malicious adversaries during transmission. In this context assuring authenticity of multimedia message during broadcast is a subject of great interest.

For this purpose cryptographic techniques are the only alternative, since cryptography is the only security guarantee when we are working with information.This paper is organized as follows. In section 2 we review about the cryptographic primitives that can be used to guarantee the authenticity of information. Section 3 holds details about the broadcast authentication protocol. Section 4 holds the conclusion.

2. Cryptographic primitives

There are two constructions for assuring the guarantee over the source of information.

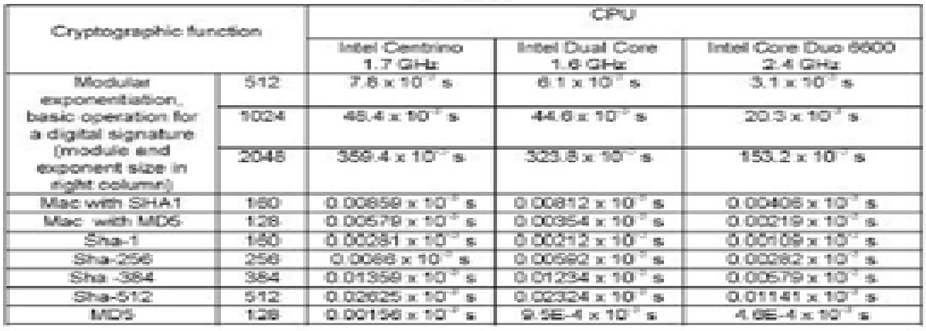
Message Authentication Code (MAC).

A MAC denoted in this paper as $MAC_k(M)$ where k is the key of MAC and M is the message which is to be proved authentic, is a symmetric primitive for assuring authenticity, which uses secret, shared key between the participants. The advantage of MAC codes is that they are easy to compute and require only simple operations. The problem that MAC codes are introducing is that as the number of participants increases so does the number of secret shared keys. For a broadcast scenario where a server broadcasts information to n participants, n different keys are needed and more the server needs to compute n different MAC codes for each broadcasted message even if the message is same for all participants. Fortunately improvements can be done..

Digital Signatures

Digital Signatures are asymmetric primitives that use private key to sign a given message and a public key to verify the signature. The advantage is that the same public key can be used by any number of entities in order to verify the source of a signed message. Therefore the number of keys does not increases with the number of participants, each entity needs to store only its own private key as a secret and has to be informed in an authentic manner of the public keys that are used by the other entities. The drawback of digital signatures in front of MAC codes is that they are computationally intensive. They are thousand times more expensive than MAC codes. Below table shows the expenses of all cryptographic applications.

TABLE 1. COMPUTATIONAL TIME FOR SOME CRYPTOGRAPHIC PRIMITIVES IN JAVA.



Cryptographic function		CPU		
		Intel Centrino 1.7 GHz	Intel Dual Core 1.6 GHz	Intel Core Duo 6600 2.4 GHz
Modular exponentiation, basic operation for a digital signature (module and exponent size in right column)	512	7.8×10^{-3} s	6.1×10^{-3} s	3.1×10^{-3} s
	1024	45.4×10^{-3} s	44.6×10^{-3} s	20.3×10^{-3} s
	2048	359.4×10^{-3} s	323.8×10^{-3} s	153.2×10^{-3} s
Mac with SHA1	160	0.00859×10^{-3} s	0.00812×10^{-3} s	0.00406×10^{-3} s
Mac with MD5	128	0.00579×10^{-3} s	0.00354×10^{-3} s	0.00219×10^{-3} s
Sha-1	160	0.00281×10^{-3} s	0.00212×10^{-3} s	0.00109×10^{-3} s
Sha-256	256	0.0066×10^{-3} s	0.00592×10^{-3} s	0.00282×10^{-3} s
Sha -384	384	0.01359×10^{-3} s	0.01234×10^{-3} s	0.00579×10^{-3} s
Sha-512	512	0.02625×10^{-3} s	0.02324×10^{-3} s	0.01141×10^{-3} s
MD5	128	0.00156×10^{-3} s	9.5E-4 $\times 10^{-3}$ s	4.6E-4 $\times 10^{-3}$ s

3. Broadcast Authentication Protocol.

In order to make such a protocol useful we must avoid the use of secret shared keys on the server side and computing n distinct MAC codes for each message will of course decrease the performance of the protocol. Fortunately, a good solution for this purpose exits: to disclose the key of the MAC only after all the entities that receive information have stored the MAC computed on the particular message MAC $_k$ (M). Of course, after key k is disclosed the problem that we have is that this key cannot be used again. However there is an elegant solution to be used to remove this problem.

The solution is to use as keys, elements of a one-way chain. In this way each disclosed key can be used as a commitment for a new key which is used to compute a new MAC and so on.

A one-way chain is a recurrent array generated by successive composition of a one-way function; each element of a one-way chain can be used as key and is defined as follows:

$\mathbf{K_i=F^{n-i}(x_0), i= 1\ldots n}$

Here k $_i$ is the i th key, n is the length of the one way chain, x $_0$ is a random element value and F is a one way function. Usually in constructing a one-way chain a cryptographic hash function is chosen for implementing F therefore, one-way chains are usually referred as hash chains. Disclosing each key at precise time intervals and using a loose time synchronization which lets each client have an upper bound on the time from the sender's side is probably the best solution for a broadcast protocol. This is the principle used.

The main idea is to have the sender attach to each packet a MAC computed using a key known only to itself. The receiver buffers the received packets without being able to authenticate it.

If the packet received too late, it is discarded. A short while later, the sender discloses k and the receiver is able to authenticate the packet. Consequently, a single MAC per suffices to provide source authentication, provided that the receiver has synchronized its clock with the sender ahead of time. It has some drawbacks. The receiver has to buffer packets, until the receiver authenticates the packets. This may delay delivering the information to the application, may cause storage problems and also generates vulnerability to Dos attacks on the receiver. If the receivers authenticate most packets immediately upon arrival, it reduces the need for buffering at the receiver side and in particular reduces the susceptibility to this type of Dos attacks.

Overview

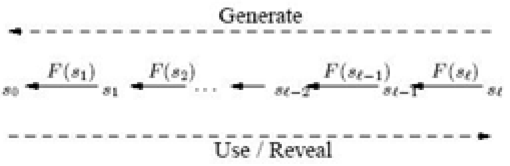
The security property of the protocol guarantees that the receiver never accepts m_i as an authentic message unless m_i **was** actually sent by the sender. It has low space overhead mainly because it is based on symmetric key cryptography. It has a low computation overhead, which is typically only one MAC function computation per packet, for both the sender and receiver. The sender and the receiver need to be loosely time synchronized, which means the synchronization does not need to be precise, but the receiver needs to know an upper bound on the sender's time.

Sender Setup

A sender distributes a stream of data composed of message chunks $\{M_i\}$. Generally, the sender sends each message chunk M_i in one network packet $P_{i.}$

Many multicast packets do not retransmit lost packets. The goal is therefore that the receiver can authenticate each message chunk M_i separately.

The sender splits the time into equal intervals I_i. We denote the duration of each time interval with $T_{int,}$ $_{and}$ the starting time of the interval Ii is $T_{i.}$ Trivially, we have T_i =T $_{0\ +}$ i* $T_{int.}$ In each interval, the sender may send zero or multiple packets. Before sending the first message, the sender determines the sending duration, the interval duration, and the number N of keys of the key chain.



When the receiver has its current time t_r, it computes the upper bound on the current sender's time as

$t_s <= t_r - t_R + t_{S.}$

Bootstrap a new receiver

The protocol requires a data packet to bootstrap a new receiver. The data packet should contain:

□ The beginning time of a specific interval Tj, along with its id Ij.

□ The interval duration T int.

□ Key disclosure delay d.

Sending authenticated packets

Each key of the key chain is used in one time interval. However many messages are sent in each interval, the key which corresponds to that interval is used to compute the MAC of all those messages. The key remains secret for d-1 intervals.

Packets sent in interval Ij can hence disclose key k j-d. As soon as the receivers receive that key, they can verify the authenticity of the packets sent in interval I j-d.

The construction of packet Pj sent in interval Ti is:

{Mj, H (M j +vd), MAC (Ki, Dj), Ki-d}

Where Dj is {M j, H (M j +vd)}, v is the number of packets per time interval,

Ki-d is the key that can be disclosed after delay d.

Receiver's tasks

Since the security of the protocol depends on the keys that remain secret until the predetermined time period, the receiver must verify for each packet that the key, which is used to compute the MAC of that packet, is not yet disclosed by the sender. Otherwise, an attacker could have changed the message data and recomputed the MAC. So the receiver must verify the security condition of the packet it receives. A packet arrived safely, if the receiver is assured that the sender cannot yet be in the time interval in which the corresponding key is disclosed. The intuition is that if packets satisfy the security condition, then no attacker could have altered it in transit, because the corresponding MAC key is not yet disclosed. In case the security condition is not valid, the receiver must drop that packet, because the authenticity is not assured any more. To evaluate the security condition, the receiver computes the highest interval x the sender could possibly in, which is x = [(tj – T0) / T int].

The receiver now verifies that x< Ii +d

(Ii is the interval index), which means that the sender must not have been in the interval in which the key Ki is disclosed, hence no attacker can possibly know the key and spoof the message contents.

Immediate authentication is provided which allows the receiver to authenticate the packets as soon as they arrive. If the sender could buffer packets during one disclosure delay, then it could store the hash value of the data of a later packet in an earlier packet and hence as soon as the earlier packet is authenticated, the data in the later packet is authenticated through hash value as well. The sender buffers packets for the duration of one disclosure delay. The sender sends out a constant number of v packets per time interval. To construct the packet for the message Mj in time interval Ti, the sender appends the hash value of the message chunk

M j +vd with the key Ki. When the packet P j+vd arrives at the receiver which discloses the key Ki it allows the authentication of packet Pj sent in interval Ii. Pj carries a hash data M j +vd in P j+vd . If Pj is authentic, H(M j +vd) is also authentic and therefore the data M j +vd is immediately authenticated. Also note that if Pj is lost or dropped due to violation of the security condition, P j+vd will not be immediately authenticated and can still be authenticated later using the MAC value. If the packet contains a disclosed key K i- d, regardless of whether the security condition is verified or not, the receiver checks whether it can use Ki-d to authenticate previous packets. Clearly, if it has received Ki-d previously, it does not have any work to do. Otherwise, let us assume that the last key value is Kv. The receiver verifies if K i-d is legitimate by verifying that Kv = F $^{i- d-v}$ (Ki-d) . It is clear that the arbitrary packet loss can tolerated, because the receiver can verify the authenticity of all receiver packets that satisfy the security condition eventually.

For dealing with audio video streams java media framework (JMF) API was used [6]. JMF is an application programming interface for incorporating time-based media into java programs. It provides support for media playback, capturing and storing media data and performing custom processing on media data streams. JMF provides interfaces/classes that handle the construction of players, processors, datastores and datasinks describe the location of media stream[6].

4. Conclusion

A protocol is developed for broadcasting multimedia data and authentication is provided. The protocol assumes all members have joined the group and have synchronized with the sender before any transmission starts. In reality, receives may wish to join after the transmission has started. As a future work, we allow the receivers to 'join on the fly' to an ongoing session.

REFRENCES

[1]. Bogdan Groza, Dorina Petrica, Simona Barbu, Mariana Bilanin "Implementation of authentication protocol for sending audio video information in java " IEEE 2007.

[2]. Perrig, R.Canetti, J.D.Tygar, D.Song. The TESLA Broadcast Authentication Protocol "In cryptoBytes, 5.2 Summer/Fall, pp, 2-13 2002.

[3]. L.Lamport," Password authentication with insecure communication", communication of the ACM, 24,770-772.

[4]. FIPS 180-1,National Institute of Standards and Technology " Announcing Secure hash standard ".

[5]. Java.sun.com: The Source for java developers, http://java.sun.com.

[6]. JMF 2.1.1 Solutions from Sun Developer Network, http://java.sun.com/products/java-media/jmf/2.1.1/solutions/.