European Journal of Scientific Research
ISSN 1450-216X Vol.14 No.2 (2006), pp. 509-516

http://www.eurojournals.com/ejsr.htm

MIKEY and SRTP Integration for Multicast Streaming

Mamoona Asghar
Department of Computer Science, Faculty of Applied Sciences
International Islamic University, Islamabad
E-mail: mamoona16@gmail.com

Moizza Sharmin
Department of Computer Science, Faculty of Applied Sciences
International Islamic University, Islamabad
E-mail: muiza_khan@yahoo.com

Shiraz Baig
Department of Computer Science, Faculty of Applied Sciences
International Islamic University, Islamabad
E-mail: shiraz_baig@yahoo.com

Qaisar Javaid
Department of Computer Science, Faculty of Applied Sciences
International Islamic University, Islamabad
E-mail: qjccie@yahoo.com

Khalid Rashid
Department of Computer Science, Faculty of Applied Sciences
International Islamic University, Islamabad
E-mail: drkhalid@iiu.edu.pk

Abstract

This paper presents the design and implementation of key management architecture for secure multimedia audio streaming to multicast receivers. It describes the methods and techniques which are used in making a multicast multimedia streaming application (MMKM). It also highlights the security issues involved, implementation of Multimedia Internet Keying (MIKEY), integration with Secure Real time Transport Protocol (SRTP) and then using it for transmitting real time multimedia data in Multicast environments. The main emphasis has been on security rather than on data transmission and compression. In case of multimedia, only audio data is chosen.

Keywords: SRTP, MIKEY, AES, Diffie-Hellman, HMAC, Multicasting, Cryptography, Security, Confidentiality, Authentication, Multimedia.

Introduction

There has been work to define a security protocol for the protection of real-time applications running over RTP [1]. However, a security protocol needs a key management solution to exchange keys and

related security parameters. The focus is on how to set up key management for secure multimedia sessions such that requirements in a heterogeneous environment are fulfilled. MIKEY describes a key management solution that addresses multimedia scenarios for unicast and Multicast environment. [2]

SRTP provides a framework for encryption and message authentication of RTP and RTCP streams. It defines a set of cryptographic transforms with appropriate key management for unicast and multicast RTP applications. [3]

Although SRTP and MIKEY have been discussed in detail in their respective RFCs and other related documents. Some contributions found related to working of both Protocols to make secure multimedia transmission in Unicast environment but there is little on their integration and then implementation for secure Multicast transmission.

Multicast is inherently a receiver-based concept. The sender is not consulted about the addition of a multicast receiver. Receivers can join and/or leave a particular multicast session, whether or not it is currently active, at will. [4] However, this aspect has been addressed in Multicast Multimedia Key Management (MMKM) application and we have made arrangements to make up for this shortcoming by maintaining a list of receivers.

We believe that Diffie-Hellman (DH) key agreement method is the most secure method among the three methods of distributing keys namely pre-shared, public key & Diffie-Hellman. This needs a separate discussion, as to why it is the most secure method. However, one point may be highlighted that two parties agree upon a shared secret key in such a way that the key is unavailable to eavesdroppers. [5] We have used this technique in MIKEY, making data transmission safer.

DH is basically a peer to peer technique [2], but MMKM application implements it in one to many scenarios. This is one of the unique features of this study.

We have developed a workable application which not only evades many security threats (discussed in section 2), but also integrates SRTP and MIKEY, successfully manages multicast transmission of multimedia and gives a method of implementation for future enhancement and application development. There are number of options and choices, laid down in their RFCs. MMKM application and this paper make those decisions and lays down a clear path of implementing transmission of multicast multimedia.

Threat Model

The threat model that is perceived and implemented in our application is discussed below:

2.1 Passive Attacks

Traffic generation Key (TGK) has been generated through Diffie-Hellman technique. It never travels on the network, so there is no possibility of its interception. Then Traffic Encryption Key (TEK) also called Master key is generated. Advance Encryption Standard (AES) [6] is used to encrypt this key and transport it across the receiver, using DH key. If an un-authorized interceptor captures the key he/she cannot use it. 128 bit AES encryption is used, and we rely on the strength of un-breakability of this encryption scheme in a finite time.

The data is encrypted with an Encryption Key of 128 bit, which has been generated from Master Key in a secure cryptographic manner. Both Master and Encryption keys are modified after every 24 hours. Thus we assume that we are safe against interception of TGK or TEK and data eavesdropping threat.

2.2 Active Attacks

We believe that Diffie-Hellman key is not compromised, as it has never traveled on the network, that's why it prevents impersonation.

To avoid man in the middle attack, an authentication key is generated for the process of authentication. If hacker captures encryption key, then he must capture authentication key to launch

this attack. We believe that this is extremely difficult and this attack cannot be launched. Authentication also detects the security threats related to unauthorized replay, deletion, insertion and manipulation of messages. A mechanism has been incorporated in the application to drop adulterated packets.

The denial of Receipt attack prevention is tackled in a way that the receiver keeps sending a message after a fixed interval (10 seconds) and it verifies that he/she is receiving the data.

2.3 Theft of Data

The user has been given a password and login for its authentication. If user gives his/her login and password to his/her friend, it is possible that data theft takes place. This is basically a billing issue. We expect that a person will not willingly give his /her login and password to an irresponsible person if the data is very confidential. But for the purposes of billing a friend might like to oblige the other friend. We have not kept a self destruct mechanism in implementation. There is no check, if same username logs in on two machines. This check can better work with billing system. However, in this case, the billing agency does not entail a loss, as the party is still paying for the time that is being used by his/her friend.

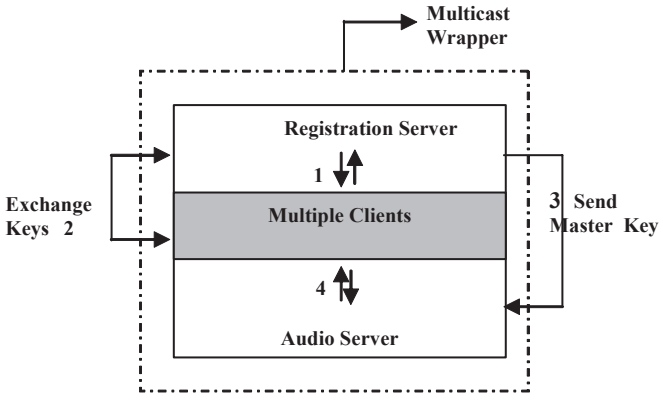
Proposed Architecture

MMKM Architecture is designed to resolve the basic and advance problems of Multicast scenarios. The basic features of multicasting are joining and/or leaving the group by users while the advance features are efficiency in key delivery, error robustness, load distribution, extra network administration, and join/leave notification. In multicast environment the data need to shift from one member to another to travel in network that make the data vulnerable because the attacker can easily attack at several points in the network at the same time. We shall rely on conventional means to cater for this threat.

Proposed MMKM architecture consists of two servers and multiple clients. The application has following three modules;

- Registration Server (RSRVR)
- Audio Server (ASRVR)
- Client (CLNT)

Fig. 1: MMKM Architecture



Communication flow of MMKM components

When any of multicast receiver wishes to listen real time file after joining the group, first the client communicates with Audio Server, then audio server communicates with Registration server for receiving Master keys to establish session among clients and audio server. Moreover the proposed solution introduces a counter to limit the number of audio requests from clients in a multicast group. The MMKM architecture provides complete authentication to registered members which automatically provides security and better Quality of Services to users.

It also maintains a list of join and/or leave notification of all registered users according to their joining and/or leaving time.

4. Implementation

The implementation scenario has three phases. First MIKEY is used for key management, key distribution and identification of party through digital signatures. Second phase encompass of the encryption and authentication of SRTP packets. Final phase is the transmission and playing of real time audio data. A number of practical issues like request of audio file from the recipients, password authentication and transmission of multimedia file with its characteristics i.e. bit depth, bitrate, channels are tackled during implementation. Multithreaded application is employed for this purpose. Two servers are used, one for key management and the other for handling of audio transmission.

4.1 Key Terminology and Role of MIKEY viz SRTP

It is considered prudent to clarify certain terminology of the keys, because SRTP and MIKEY uses different terms for certain keys which perform same functions. MIKEY used the terms of TEK and TGK. It distributes either a TEK or TGK. If it distributes TGK, then it is used to generate TEK. If it distributes TEK, security protocol (SRTP) directly uses it. The SRTP calls the TEK of MIKEY as its Master Key. The Master key is then used to generate six Session keys. These keys are Encryption Key, Authentication Key and Salting Key on the sender side and same three keys on the receiver side, making a total of six keys for SRTP packets, while six same keys are also generated for SRTCP packets.

MIKEY is being used to generate a TGK. This TGK is used as TEK and is the Master Key for the SRTP. The Master Key is transported to the receiver side in a cryptographically secure manner. In this way, the same Master Key is available with sender and receiver to generate further keys.

4.2 Multimedia Internet Keying (MIKEY)

A Registration server (RSRVR) is constantly listening to requests of a new user who wishes to register, by sending a registration request. The request follows the pattern of MIKEY, in which digital certificate and other relevant information is provided. A user name and login is accepted and then a Diffie-Hellman key agreement takes place. A 72 byte (288 bit) string is generated, a random number is chosen and the algorithm is applied. After completion of one round of transmission, a TGK is generated. Then server uses timer as a seed, and generates a 16 byte (128 bit) random number, which is treated as TEK or Master Key. This key is encrypted with DH and transmitted to the receiver. Thus TGK and TEK have been generated and distributed. The TEK is now available for generating the six keys used by SRTP.

MIKEY deals with three types of scenarios, peer to peer, one to many, and many to many. On the other hand, MIKEY has three techniques that it uses, namely Pre-shared, Public Key, and Diffie Hellman. The DH technique is mostly used in peer to peer scenario. But we have used it for one to many scenarios. This is how we do it. Each receiver exchanged DH key parameters with the RSRVR at the time of registration. Both parties, then, arrive at a final key, which is symmetric in nature. A database is maintained at the RSRVR. The key of each user is stored in the database along with its username. The server generates a TEK which is same for every user and transfer to the receiver.. At the

time of login, the DH key and TEK both are used. This TEK is then used by the receiver to generate his/her three keys, which are subsequently used by SRTP and SRTCP.

4.3 Secure Real time Transfer Protocol (SRTP)

The TEK is now available to SRTP. The SRTP uses various algorithms to derive the keys. Key derivation reduces the burden on the key establishment. Six different keys for both SRTP and SRTCP are needed per crypto context. All these are derived from a single Master key by using Keyed-Hashing for Message Authentication Code (HMAC) Method. [7] The Pseudo Random Function (PRF) used is described in the following table.

	Parameter	Technique Uses
a.	TGK Generation	Diffie Helman
b.	TEK Generation	HMAC-SH1
a.	Derivation of Encryption Key	HMAC-SH1
b.	Derivation of Authentication Key	HMAC-SH1
c.	Derivation of Salting Keys	HMAC-SH1
d.	Encryption of Data	AES_CM
e.	Master key length	128 bits
f.	Encryption (Session key length)	128 bits
g.	Authentication (Session key length)	160 bits
h.	Master salt key	112 bits
i.	Session salt key length	112 bits
j.	Key derivation rate	0
k.	Key Life Time	24 hours
l.	MKI indicator	0

4.4 Data Encryption

We have used a sound file for the streaming media. The server would open the file and advertise 44 bytes of the wave file characteristics. Then it would go on transmitting the packets, in a multicast manner. These packets would be released at fixed intervals. All the clients who are listening will receive the audio file characteristics and audio file packets. SRTP packets will be encrypted and transported. These packets will be decrypted and the played on each client site. SRTCP packets would also be transmitted to generate specific reports. All these transmissions are being handled in different threads, and on different socket.

4.5 Other Services

Furthermore, there is a need that when a client leaves, i.e. sends a BYE packet, it must be recorded at the ASRVR. Normally, in multicast environment, we cannot keep a record, of who is coming and going. [4] We were required to keep a record of that. Each client, after login, sends “I am alive” message after every 10 seconds. If for six consecutive turns, this message is not received, it is assumed that the client has died. Of course, if the client leaves in the normal manner, it will send a BYE packet and the server would come to know that a particular client has left.

Another facility of Audio files queue has been provided. In this case, a client, after login, gets a list of songs available with the server. The client can choose any file of his choice and send that choice to the server. The request will join the queue and the file will be played on its own turn.

5. Results

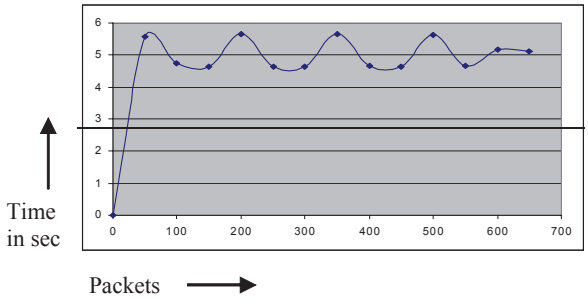
The function gettimeofday() is used to determine the critical timings of the transmission of every 50th packet in seconds and microseconds (This time is included in the Time Stamp of the SRTP header).

The main purpose is to calculate the timing difference between non-encrypted packets and the encrypted packets. Two scenarios are examined, which are as follows

5.1 Scenario I

In this case full encryption of data packets and key generation is carried out. When MMKM application transmits encrypted audio packets, there is 5 seconds delay to transmit every 50th packet i.e. the values returned by function gettimeofday varies from 1132844404 to 1132844409 in seconds. The last two digits of the seconds and all the digits of the microseconds have been taken to show the results.

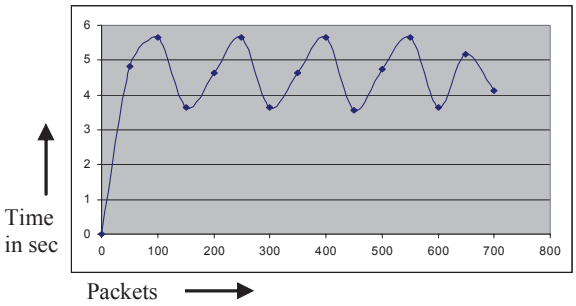
Fig. 2: Encrypted Audio Packets



5.2 Scenario II

In this case no key generation is done and the non-encrypted data packets are dispatched. When MMKM application sends audio packets without any key generation and without encryption, the time varies in order of 4 to 5 seconds and vice versa. However, the quality does not suffer.

Fig. 3: Encryption without Audio Packets



Generally, it is considered best to use buffering technique in the multimedia applications. What we have done, we fill 50 buffers each of 16384 bytes and then start playing it. It find that after some time (less than a minute) all the buffers would become empty and playing would have to wait to fill in the buffers again. Thus the audio streams does not play continuously, it stops at intervals. This is an undesirable situation. It happens in certain audio streams while other audio file plays fine. Later we found, that the audio streams, which have been sampled at 48000 per second, at the record time, played fine during playtime. While, the audio streams that are sampled at 44100 per second, at record time, stops at intervals during playtime. This problem is briefly discussed below.

Generally Bytes per second (BPS) being played can be calculated by the formula:
Sample Rate * Depth * channels / 8
In case of 48000 sample rate the BPS is
48000 * 16 * 2 / 8 = 192000 BPS

While with 44100 sample rate the BPS is
44100 * 16 * 2 / 8 = 176400 BPS
Thus in every second we will be losing
192000 – 176400 = 15600 BPS

Because the bytes would be playing much faster i.e. at 48000 sample rate, while the bytes are being dispatched by server at a slower rate i.e. 44100 sample rate. The fifty buffers, that we have used, contained 50 * 16384 = 819200 bytes. In every second, we would be losing 15600 bytes. So, we would lose all the bytes in 819200/15600 = 52.5 seconds. All 50 buffers would become empty in this time and the playing machine will have to wait to receive further data. There is no remedy to this problem except that we should use a better sound card.

To overcome buffering problem we use an artificial local client, which is running with ASRVR on same machine. The client is working on the principle that it sends request for every audio packet to play and also uses to monitor the transmission and playing of audio session. There is no buffering process involved. This technique ensures that any other client (on the Internet or on LAN) can not play the data faster than the local client. This system addresses the problem of playing too fast by the client. This solution will work beautifully on LAN or WLAN where there are no network delays and no problems of bandwidth. This solution has the limitation on the Internet, because it will not buffer the data. If there are network delays, the quality of sound will suffer.

Conclusion

The emphasis of this paper is to develop an environment that performs secure multimedia streaming to multicast receivers, which prevent security threats and also provides better quality of services.

MIKEY and SRTP RFCs [2, 3] are both related to each other in the security context, but their roles are not clearly defined in the RFCs, which causes implementation ambiguities.

MIKEY uses the words TGK and TEK for keys but does not clearly define other session keys. On the other hand, SRTP discusses two keys, the Master Key and the Master Salt Key. MIKEY does not discuss Salt Keys. Yet, it derives all the keys, but it is not clear that the derivation of keys is only for initial security context parameters exchange or subsequently, it will also be used for the actual data transmission. The confusion arises in deciding how do MIKEY and SRTP integrate with each other.

The RFC of SRTP says that we should derive six keys from our master key. We have three choices, i) we can generate Master key by using random number, pre-shared key or Diffie-Hellman. The RFC does provide a choice but is silent on this aspect. According to RFCs we could end up with 12 keys, six for the MIKEY Phase and six for the SRTP (data transmission) phase. It produces too much overhead. We suggest that one key either pre-shared key or generated by Diffie-Hellman or Public key algorithms should be use and employ it for both phases. It will reduce the overhead of calculation of keys and simplify implementation. This concept has been used in our application.

Another point is about Salt Keys. SRTP RFC leaves it to the choice of the user whether to generate the Salt Keys or not. Similarly, if the Salt keys are to be used, the RFC does not clarify how many bits are to be changed at what frequency. In summary, the integration between SRTP and MIKEY should be more clearly defined.

Future Enhancements

MMKM modules (RSRVR, ASRVR, and CLNT) are flexible enough to accommodate and incorporate multiple functions in it depending upon day to day needs and future aspects. Some possible enhancements can be the use of MP3 Audio Format instead of .wav file. It can be better to implement on mp3 format because minimal disk space utilization is preferred now a days.

Billing System may be maintained to enhance security by using credit card number. Implementation of billing system can also eliminate the problem of “stealing” client software, even from a friend.

References

[1] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, Request for Comments: 3550, July 2003

[2] Arkko et. al., MIKEY: Multimedia Internet KEYing,., Request for Comments: 3830, August 2004.

[3] M. Baugher et. al.,SRTP: The Secure Real-time Transport Protocol, Request for Comments: 3711, March 2004

[4] Fern Levitt, Internet Multicast Security, Overview of Issues and Technologies, Doc No. NU-R111, Rel. C, Feb. 8, 1999

[5] E. Rescorla, Diffie-Hellman Key Agreement Method, Request for Comments: 2631, June 1999

[6] NIST, "Advanced Encryption Standard (AES)", FIPS PUB 197, November 26, 2001, http://www.nist.gov/aes/.

[7] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", Request for Comments: 2104, February 1997.