

New security technique for real time applications (RTAS)

Ammar Al Roz

ABSTRACT

The Internet has worked so far with a *best effort* traffic model, every packet is treated (forwarded or discarded) equally. This is a very simple and efficient model.

Recently many interactive or real-time services have been introduced and the economical importance of the Internet has grown. The IP phones and services based on that technology is threatening the traditional circuit-switched telephone services, especially on long-distance services.

Transmitting interactive real-time media is the greatest challenge in packet based networks. The end-to-end delay, the delay variations (jitter), and the packet loss must not exceed some time limits; otherwise, usability of the service degrades badly.

Many companies have been deploying Real time applications (RTA) over the internet like VOIP, Video Conferencing and other Multimedia services in recent years. The need to protect users, data and infrastructures becoming more crucial than ever. Encryption is used to provide the security needed for real time applications. Since RTA contain high volume of data, classical encryption techniques are not appropriate. Because most of RTA are implemented on the internet, the encryption and decryption techniques has to take minimal time to achieve acceptable end to end delays. In this paper, a new cryptographic Algorithm is developed to improve the time for encryption and decryption of data (VOIP) of end to end delay.

Keywords: Real time applications security, VOIP, Encryption, Decryption.

Introduction

Cryptography is the science of transforming the data so that nobody can read it or change it except an authorized user. So, changing the original data to a secret message is called Encryption, while decryption is the reverse. The process of encryption and decryption of the data is based of a mathematical procedure called the algorithm.

The major security goals that are of concern to cryptography are, first, confidentiality which is the function of allowing only authorized users to access the information, second, authentication which is the function of the receiver verifying the sender and trust that the sender is actually who he claims to be, third, integrity, which means that the receiver should be able to trust that the message has not been altered during transmission, fourth, non-repudiation which is the inability of the sender or receiver to deny that the message has been sent or received, finally, access control which is to restrict the availability of encrypted information. These five elements has been the focus of a well configured cryptography system.

There are three categories of cryptographic algorithms : Hashing,

Many algorithms have been proposed to implement hashing. One- way hash is one of these algorithms which is widely used in the ATM. Message digest is

another algorithm which has three versions MD2, MD, MD5. The basic idea of the message algorithm is to take a plain text of any length and creates a hash of various length depends on the version. Hash functions has proven to have weaknesses and should be replaced by a more secure methods.

Symmetric encryption algorithms which the major category of cryptographic algorithms and the most common type uses a single key to encrypt and decrypt a message^[3,4].

The purpose of the symmetric algorithms is to decrypt the ciphertext compared to hashing algorithms where they never intended to decrypt the information so it is also called **Private Key Cryptography**. The most common Algorithm is called Data Encryption Standard (DES), it uses a key length of 56 bits and it can be implemented in hardware and software by executing the algorithm 16 times, but the key is considered un-secure because of the length. Triple DES is an improvement over the DES because it employs three times the executing of the algorithm and it uses a key of 192 bit long, which makes the computation time too long and not suitable for real time applications (VOIP)^[5].

Advanced Encryption Standard (AES) is another algorithm which provides much higher security level than DES and perform it in 3 to 10 less computational power than tripleDES^[3,6].

AES performs three steps on every block of 128 bits of plain text, if the 128 bit key is used, it performs 9 rounds, if 192 bit key is used, it performs 11 rounds and a 256 bit key performs 13 rounds, which makes this protocol ideal for real time encryption of voice and signaling^[6].

RC4 algorithm which was invented by Ron Rivest (Ref) is a stream cipher that accepts keys up to 128 bits in length and it is used as part of Wired Equivalent Privacy (WEP) encryption standard on IEEE 802.11a, b and g wireless networks. Implementations can be very computational intensive^[1,2].

The primary weakness in the symmetric encryption algorithm is keeping a single key secure, which yields another approach to cryptography which is called asymmetric encryption or **Public Key** that which uses two keys instead of one. One of the keys is used to encrypt the message and is called **private key** and the second is used to decrypt the message and is called **Public key**. The two keys are mathematically related. Three asymmetric algorithms are commonly used which are RSA, Diffie-Hellman, and Elliptic Curve. RSA is the most commonly used asymmetric algorithm which is built in operating systems by Microsoft, Apple, Sun, and Novell. It is also found in secure telephones, ethernet network cards, and smart cards^[7].

The second, Diffie-Hellman algorithm does not encrypt or decrypt data, rather it allows two users to share a secret key securely over the public networks. Once the key is being shared then both parties can use it to encrypt and decrypt messages using symmetric cryptography. This algorithm is used in IPSec and Secure Shell (SSH) protocols.

The last algorithm, Elliptic Curve, has not been fully tested because it is still new concept^[8].

Real time applications

As voice service provider's roll out Voice-over IP (VoIP), instant messaging (IM), video, and multimedia conferencing, the need to protect user's privacy against eavesdropping are becoming a more critical issues^[8].

Except for AES protocol, all other protocols mentioned above are not suitable for real time applications because of the delay that they impose on the packets and since the real time packets are sensitive to time delay. We need to design an algorithm that have a better level of security, with minimum encryption delay and minimum message delay.

A security key that is 56 bit long can be discovered easily. A key of 192 bit can consume lots of power processing and is not desirable for RTA. The complexity of the algorithm can impact the encryption and decryption delay.

In our algorithm, we claim that its encryption and decryption contribute to the message time delay at minimum, and it is better than the standards which should be kept to less than one second.

This algorithm has been designed with two major factors in mind. First, the time needed for encryption and decryption. Classical encryption techniques take a lot of time like Triple DES, and AES-Rijndael, so it is not appropriate for real time applications. Second, the level of security should be high enough so attackers can not break the key.

There are two components an attacker needs, the Key and the index table. We use 1024 bit key, and this key is randomly generated which becomes harder to intercepted. This key is delivered in each packet. This key is randomly generate an indexes in an index table. Index tables in sender and receiver, indexes are not send with packets or in any other way, so attacker need to guess 128 indexes. In this approach, he/she needs to we have two levels of security, the attacker needs to decrypt the key, and remove the key from the packet, or else he will hear some noise, and then the conversation will not be clear.

How it will work?

The proposed algorithm is implemented as the following :

1. We need to request a connection. This request is done by sending one packet which is called initial packet. This packet depends on some actions, is done at the sender side. These actions are:

construction of the initial random generator table in the sender and the receiver sides (Table 1). Numbers in this table are generated by mathematical equation. This equation should be initialized in the same way at the sender and receiver sides. The table size is 16*16 as shown below, and numbers in the table are varieng from 0-9.

1	8	1	0	5	6	3	6	5	2	1	8	1	0	5	6
8	8	0	4	0	8	8	0	4	2	8	8	0	4	0	8
1	0	7	2	5	6	5	2	7	2	1	0	7	2	5	6
0	4	2	4	0	0	4	2	4	2	0	4	2	4	0	0
5	0	5	0	5	0	5	0	5	2	5	0	5	0	5	0
6	8	6	0	0	6	8	6	0	2	6	8	6	0	0	6
3	8	5	4	5	8	3	0	9	2	3	8	5	4	5	8
6	0	2	2	0	6	0	2	2	2	6	0	2	2	0	6
5	4	7	4	5	0	9	2	9	2	5	4	7	4	5	0
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
1	8	1	0	5	6	3	6	5	2	1	8	1	0	5	6
8	8	0	4	0	8	8	0	4	2	8	8	0	4	0	8
1	0	7	2	5	6	5	2	7	2	1	0	7	2	5	6
0	4	2	4	0	0	4	2	4	2	0	4	2	4	0	0
5	0	5	0	5	0	5	0	5	2	5	0	5	0	5	0
6	8	6	0	0	6	8	6	0	2	6	8	6	0	0	6

Table 1: Random generator table in the sender and the receiver sides

2. The sender Chooses a large random number (128 Digit) from 1 to 15. The random generator table is then achieved. This table gives indexes. The (128) digits are encrypted in any typical encryption protocol and mixed with some information and data. Then the randomized generator table is sent to the receiver. This initial packet is the connection request.

3. After the receiver gets the request for connection, the initial random number and the data are removed from the packet. Then the receiver decrypts this packet and use this number to get the random generator table which will give the same indexes as in the receiver side.

After the connection is established, we start with the real time data (VOIP). The algorithm works as follows :

at the sender side:

1. Choose a random number of 1024 bit long which will be the key.
2. Encrypt data using the XOR operation with the key (Key XOR data).
3. Mixing key with data by using the indexes that were generated earlier from the random generator table.
4. Send the packet to the receiver side.
5. Use part of the key to generate the random generator table.
6. The Output from the table will be used as the new indexes.

At the receiver side:

1. Split the key from the data, which depends on the key indexes that we got from random generator table.
2. Decrypt data by XOR key with data.
3. Use the same part of the key (as at the sender side/ step #5) to randomize the random generator table.
4. The Output from the table will be used as the new indexes.

Now we show how random generator table works. This table is used to generate the new random numbers by changing the existing number location. This change is done by input numbers to table

Lets take a small number like 2 2 0 1 4 3 and a small table 5 * 5, remember that our table is 16*16 .

Step 1: We take the first number which is and shift the table in vertical way.

Step 2: We take the second number which is and shift the table in horizontal way.

Repeat step 1 and 2 until the input numbers are finished (Figure 1).

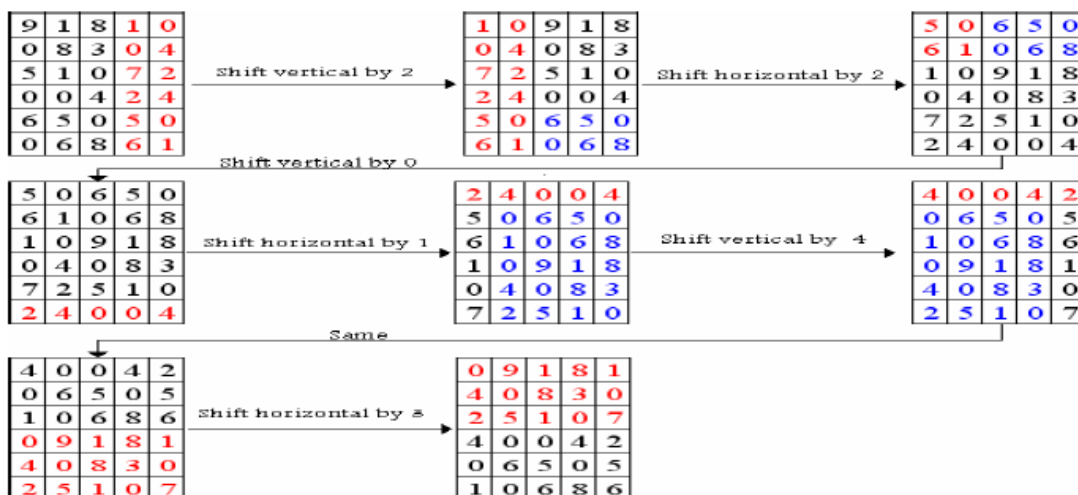


Figure 1 : The random generator.

Actually we have 16×16 table, and the input number is 16 decimal digits. These digits are part of the key (convert bits to decimal digits), each decimal digit mod 15.

Table 2 : Key indexes.

6	7	2	8	5	4	7	3	6	5	2	1	4	5	7	6
8	0	0	8	8	1	0	8	0	4	2	0	2	0	5	6
6	4	5	0	8	0	4	5	2	7	8	7	2	4	0	8
5	2	0	4	0	7	2	6	5	2	7	2	0	2	5	6
7	0	6	0	0	6	7	2	2	4	2	7	2	4	5	2
1	2	1	2	0	4	1	2	9	7	5	6	8	0	2	0
8	2	8	8	5	0	0	7	7	2	6	2	7	0	0	6
1	2	1	6	3	7	7	0	5	0	4	2	8	5	0	0
1	8	1	0	9	4	0	0	0	6	5	2	8	0	6	2
8	8	0	7	7	5	8	8	7	2	5	4	0	5	8	0
1	0	7	7	7	0	6	5	5	5	0	5	0	0	6	6
0	4	2	5	6	9	0	4	0	6	8	6	0	6	2	4
2	6	8	0	8	4	0	5	5	3	8	5	4	8	4	5
5	0	5	5	6	7	5	5	0	6	0	2	2	6	2	0
8	8	0	0	0	0	7	5	0	5	4	4	0	0	8	0
0	1	8	1	0	4	5	6	8	6	7	5	2	0	3	4

Indexes are: 67 28 54 73 65 21 45 7. Table 2 is ensuring key indexes. So it will give us 128 indexes.

Mixing key with data :

It depends on random generator table. We insert every 8 bit from the key to the specific index in the data as show in figure 2.

67 28 54 73 65 21 45 75



Figure 2 : Insert every 8 bit from the key to the specific index in the data.

In the previous table we notice that the key is in the first 1224 bits, so if we need more diffusion we can take 3, 4, 5, or more squares instead of 2 squares. This will give us more key diffusion in the data, and then provide us with the flexibility to minimize or maximize the table size as required by the application. To send a packet, we calculate the delay of encryption packet steps from 1 to 6. Steps 5 and 6 have no such

delay, because the algorithm design process we used allows us to execute steps no. 5 and 6 while the packet is send.

The encryption and decryption times are illustrated in table 3.

Table 4 shows that this technique is superior to others and provide a better security level for RTA.

Table 3: Encryption and decryption times.

	Data Size (Byte)	Key (Bit)	Time (MS)
Encryption	1500	1024	0.88125
	1024	1024	0.79075
	1024	512	0.74075
	1024	256	0.71575
Decryption	1500	1024	9.355
	1024	1024	9.183125
	1024	512	3.7725
	1024	256	1.958125

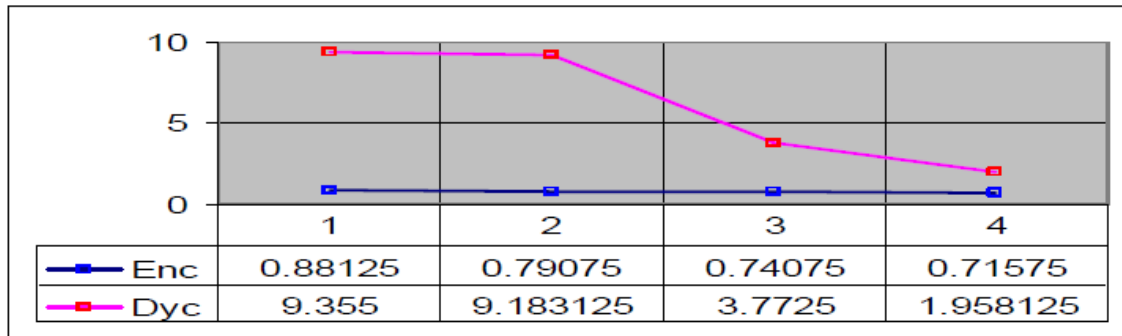


Figure 3 : Comparison between our experiment and the requirement of the RTA.

experiment and the requirement of the RTA. All encryption time is better then and it is less than 1 Millisecond, In decryption process, we notice that at some point it takes more than 9 MS. This number is high with respect to the encryption time, but it is better comparing to AES-Rijndael algorithm. RTAS realize better results, and it is highly recommended.

Table 4 : Compression between AES-Rijndael and RTAS, Data Size = 1024 byte and Key = 256 bit.

Security name	AES-Rijndael	RTAS
Encryption (MS)	10.884	0.71575
Decryption (MS)	10.718	1.958125

Conclusion:

Most of the available encryption/decryption techniques are not suitable to be used to secure voice data over an open network since they were originally built for text data, and due to their extensive computations which result in an unacceptable delays. This work attempts to develop a new encryption approach which adds a minimum delay time to the Real time applications packets end-to-end delay. The distribution of the encryption keys is usually carried out through a trusted agency; this results in a significant delay before the real time application starts. This work attempts also to provide a new method of key exchange without an intermediate party.

Recommendations:

The followings are some recommendation to be considered :

1. If we maximize table and take more squares, we can enhance the security level.
2. we also can enhance the security level by using more than one decimal digit in each square in the table.

Acknowledgement:

This paper received financial support towards the cost of its publication from the Deanship of

Research and Graduate Studies at Applied
Science University, Amman - Jordan.

References:

1. Scott R. Fluhrer, Itsik Mantin and Adi Shamir, 2001. Weaknesses in the Key Scheduling Algorithm of RC4, Selected Areas in Cryptography, pp: 1 – 24.
2. Scott R. Fluhrer and David A. McGrew, 2000, Statistical Analysis of the Alleged RC4 Keystream Generator, FSE, pp: 19 – 30.
3. Xiaoyun Wang, Hongbo Yu, 2005. How to Break MD5 and Other Hash Functions, *EUROCRYPT*. ISBN 3-540-25910-4.
4. RFC 1321 The MD5 Message-Digest Algorithm.
5. Nicolas Courtois, Josef Pieprzyk, 2002. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations, pp: 267–287, ASIACRYPT.
6. Joan Daemen, Steve Borg and Vincent Rijmen, 2002. The Design of Rijndael: AES - The Advanced Encryption Standard. Springer-Verlag,. ISBN 3-540-42580-2.
7. R. Rivest, A. Shamir, L. Adleman, 1978. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21, 2, pp: 120–126. Previously released as an MIT "Technical Memo" in April 1977. Initial publication of the *RSA* scheme
8. R. Barbieri, D. Bruschi, E Rosti, 2002. Voice over IPsec: Analysis and Solutions. *Proceedings of the 18th Annual Computer Security Applications Conference*.