Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 11, November 2014, pg.414 – 421

RESEARCH ARTICLE

MINIMIZATION OF DELAY TIME IN DYNAMIC ENCRYPTION ALGORITHM FOR REAL-TIME APPLICATIONS (DEA-RTA)

Kuti Toyin Sadiq (M.Tech)*[1], **Madhavi Kumari** (Associate Professor)[*2]

[#]Jawaharlal Nehru Technological University JNTUH, Hyderabad, India

Electronics and Communication Engineering Department

Abstract— Real-Time Applications (RTA) such as Voice-over IP (VoIP), Instant messages (IM), and video conferencing are sensitive to delay as they are packets base network services. Providing these services over the internet require high level of security to protect user, data and infrastructures. Encryption process is used to provide the security needed for RTA. The encryption and decryption has to take minimal time to achieve acceptable end-to-end delays. The research algorithm proposes a new symmetric encryption algorithm that reduces the encryption and decryption delay time which is known as Dynamic Encryption Algorithm for the Real-Time Applications (DEA-RTA).

Keywords— DEA / RTA, Cryptography, Real -Time Application

I. INTRODUCTION

A Real-Time Application (RTA) is an application program that functions with a time frame that user senses as immediate or current. The latency must be less than a defined value, usually measured in seconds. With the development of network multimedia system, systems will make continuous media streaming. It is very important to secure networked continuous media data from potential threats such as hackers, eavesdroppers etc. The Real-Time streaming Applications (RTSA) like VoIP, IP Telephony, IPTV, Video Conferencing, Internet broadcasting (corporate communication), education (viewing lecture and distance learning) web based channel (IP-TV, internet radio) and video on demand (VOD). In all the streaming applications, high volume of data is transmitted over the network. Since traditional encryption algorithms often fail due to the extra high volume and latency sensitiveness of media data, security becomes a challenging task [5][6].

Security is the major challenge in providing Real-Time Streaming Services and protocols such as RTP, RTCP, SCTP, SRTP, and SRTCP [4] help in providing RSTA services over the internet.

Secure Real-Time Transport Protocol (SRTP) defines a profile of Real-Time Transport Protocol (RTP) intended to provide encryptions, message authentication, integrity, and replay protection to the RTP data in both unicast and multicast applications [10].

Securing RSTA has been an active area of research and many solutions have been proposed to secure the communication keeping in mind the difficulties involved in real-time communication in mind. Since the real-time traffic packets are smaller in size and are processed in real-time they cannot be encrypted like data packets. The most important factor to consider in real-time traffic is limiting the delay of each packet within the maximum acceptable limits. Securing the real-time packets account to a major part of the packet delay in addition to the network and transport delays.

Encryption provides the much needed security for RTSA services over the internet. Encryption/decryption latency is a problem for any cryptographic protocol, because much of it results from the computation time required by underlying encryption algorithm.

AES protocol (AES has been chosen to be used in Universal Mobile Telecommunication System UMTS; third generation networks 3G, and mobile networks [7]. Existing security protocols are not suitable for Real-Time Applications because of long delay they impose on packets. For this reason, the need to design an algorithm that have a better delay time even than AES and a better level of security [5].

One of the major factors that affect delay time and security level is the key length [3], a security key that is 56-bit long takes less delay but can be easily broken, a key of 192-bit long needs more processing time and power but provides higher level of security and not suitable for RTA. Another factor that can affect the encryption and decryption delay is complexity of the algorithm [12].

DEA algorithm has been designed with two major factors in mind, first, time needed for encryption algorithm like 3-DES, AES-Rijndael, and RSA [1] takes longer time and are not appropriate for Real-Time Applications. Second, the level of security should be high enough so attackers cannot obtain the encryption/decryption key easily [5].

II. GENERAL SECURITY CONSIDERATIONS

Cryptography is divided into two main categories, the first and the most common category is called *classical cryptosystem encryption algorithms* (also called *single-key or symmetric*) [3].

Symmetric encryption also referred to as conventional encryption or single-key encryption is a form of cryptosystem in which encryption and decryption are performed using the same key. Symmetric encryption transforms plaintext into ciphertext using a secret key and an encryption algorithm. Using the same key and a decryption algorithm, the plaintext is recovered from the ciphertext [1]. The most common Algorithm within this category is named Data Encryption Standard (DES), it runs on the key length of 56-bit and it could be implemented inside hardware as well as software by executing this algorithm 16 times, but the key is considered un-secure as a result of the length [9]. The triple DES (3-DES) which is an improvement over the DES has two attractions that assure its widespread use. First, with its 168-bit key length, it overcomes the vulnerability to brute-force attack of DES. Second, the underlying encryption algorithm in 3-DES is the same as in DES. This algorithm has been subjected to more scrutiny than any other encryption algorithm over a longer period of time, and no effective cryptanalytic attack based on the algorithm rather than brute force has been found [1]. 3-DES executes the algorithm 3 times and it runs on the key of 192-bit long, which tends to make the computation time long and not suitable for RTA [13].

Advanced Encryption Standard (AES) is usually another algorithm on the first category which offers much increased security than DES as well as perform it in 3 to 10 less computational power than 3-DES [15].

AES is a block cipher intended to replace DES for commercial applications. It uses a 128-bit block size and a key size of 128, 192, or 256 bits.

AES does not use a Feistel structure. Instead, each full round consists of four separate functions: byte substitution, permutation, arithmetic operations over a finite field, and XOR with a key which tends to make this protocol well suited for RTA encryption/decryption similar to voice as well as signalling [4] [5][1].

The primary weaknesses in the symmetric encryption algorithms are usually keeping this single shared key secure and key distribution which usually yields another method to cryptography known as Asymmetric Encryption or Public key cryptosystem, which represents second category of cryptosystems [8].

The second category is called asymmetric cryptosystem algorithms which uses two keys instead of one [11]. One of the keys is used to encrypt the message and is called public key and the second is used to decrypt the message and called private key. The two keys are mathematically related. Some of the most commonly asymmetric algorithms are the RSA and Diffie-Hellman [5].

In Diffie-Hellman algorithm which is based on the discrete logarithm problem, the users share a common secret key and it is an example of asymmetric key exchange protocol [3]. It allows users to share a secret key securely over the public network (internet). Once the key is being shared then both parties can use it for both encryption and decryption of messages using symmetric cryptography. This algorithm is used in IPSec and Secure Shell (SSH) protocols [5].

III. THE DEA-RTA ALGORITHM

In DEA algorithm, we claim that the algorithm has a minimum message delay which should be kept to less than 400ms in RTA [12]. In our algorithm the key plays the major role in providing higher level of security where; the key is 1024-bit long, the key is randomly chosen which makes it harder to intercept, a new key is delivered in each packet which makes it very difficult to guess [4].

The key is used to randomly generate the indexes of the index table at the sender and the receiver sides. Indexes are not send over the network or in any mean, rather they should be created based on a shared mathematical formula, so the attacker need to guess the 128 entries of the index table in a very short time; which make it very difficult to obtain the key and the attacker will hear noise in best the cases[5].

IV. PROPOSED SYSTEM

The Dynamic Encryption Algorithm for Real-Time Applications DEA-RTA consists of the following processes.

A. Index Generation

Connection is initiated by sending an initial packet to the other communicating party. The initial packet depends on the operations performed at the sender side according to a shared mathematical formula [4]. In this module the initial table size is (32*32) rows and columns, the table entries values are ranging from 0-F (14). Both, the sender and the receiver should have the same copy of the initial table, the initial table is not secret it could be announced to the public. The shared value is randomly generated or selected by the user, the size of this value is not specified, it is recommended use a value that is not less than 10 digits size [5]. The shared value is the most

important component of the system, so it must be kept and exchanged securely; for this reason the Diffie-Hellman or any other key exchange scheme is proposed [4]. The shared value is not fixed during the communication session, the sender or the receiver may choose to change this value at any time during the communication, accordingly the other communicating party must be informed. The system could be configured to change the shared value either manual (i.e. by the user) or periodically according to some statistical information (i.e. size of the sent/received data, time-slice, randomly or any other criteria) [5].

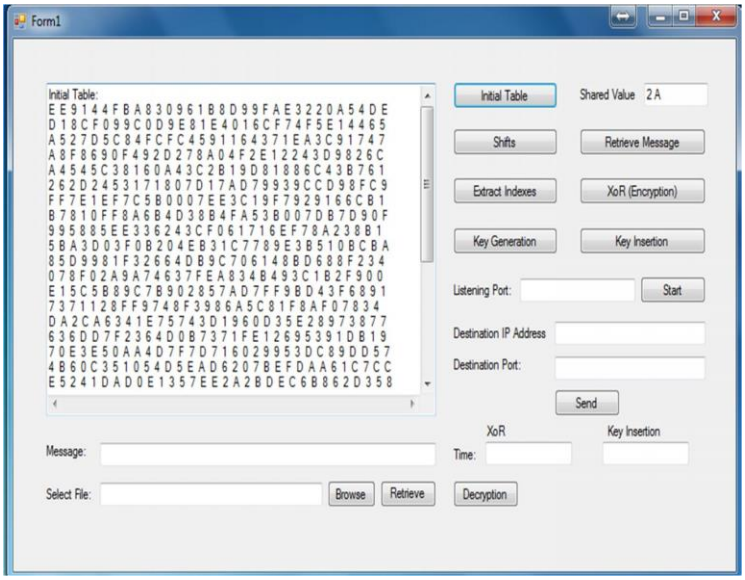

Fig. 1 table of indexes

B. The Extracted Indexes

The circular left shift and circular down shift is performed. The table of indexes generated after performing left and down circular shifts respectively according to the values the extracted from the shared value [5]. The table of extracted indexed is generated out of the shifted initial table starting from the (leftmost and top-corner) and continue row wise until the last element of the table (rightmost, down corner), where the first two digits represent the first index, the next two digits represent the second index and so on, for example, from Figure 3 (C1 is the 1st index, B7 is the 2nd index, 4E is the 3rd index,….., and 8B is the 512th index), in this case we will have 512 indexes. The output table is called the Table of Indexes.

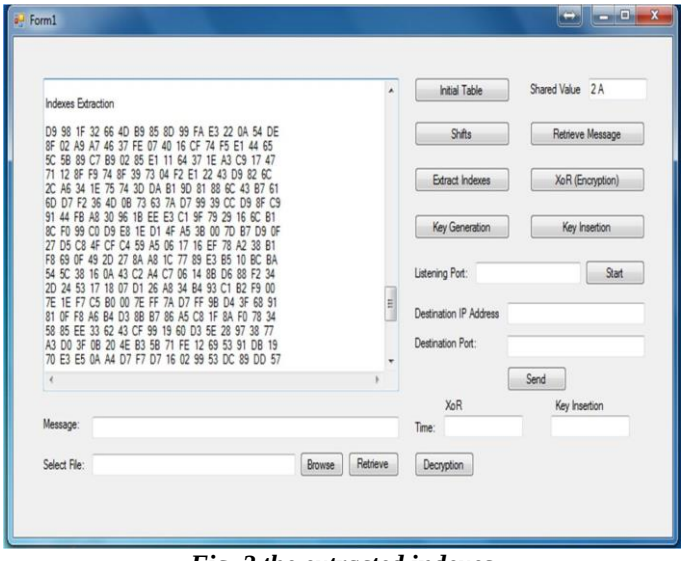

Fig. 2 the extracted indexes

C. Key Insertion Process

The plain text data is the data to be sent; this data could be of any data type and format (i.e. text, audio, video … etc.) The plain text data has no restriction on size, but it is recommended to fit on one packet size minus the key size of the Maximum Transfer Unit (MTU); (Plain-Text-Size = (MTU) - (Key-size)) [5]. The system key is randomly generated or could be selected by the user, the key could be of variable length size, initially it is 1024-bit size used, and the key size can be expanded to 2048 bits or even longer than that. Since the key is dynamic the system can be configured to different keys periodically during the communication session, the users can change the key very frequently and the encryption/decryption speed and operations will not affected, where the key is sent in each single packet. The plain-text-data and the key are XoRed, the first 1024 bits of the data is XoRed with the key, the second 1024 bits of the data is XoRed with the same key and so on until no more plain-text data found. If the plain-text-data is longer or smaller than the key the only matching lengths will be XoRed, this is the reason why we choose a key of variable length size (greater than or less than the plain-text size). The key is inserted in the XoRed table generated, the insertion process is performed according to the Extracted Indexes, the first octet of the key will be inserted in the XoRed table according to the value of the first Index value and the second key octet will be inserted according to the second Index value and so on until the end of the key, where the insertion is performed at the appropriate location as indicted on the index value Converted to decimal value). For more complexity we could insert the first key octet as depending on the first three index octets or the first four index octets. This process adds more complexity and flexibility to the encryption process with no operation cost and makes it harder to the cryptanalysis.

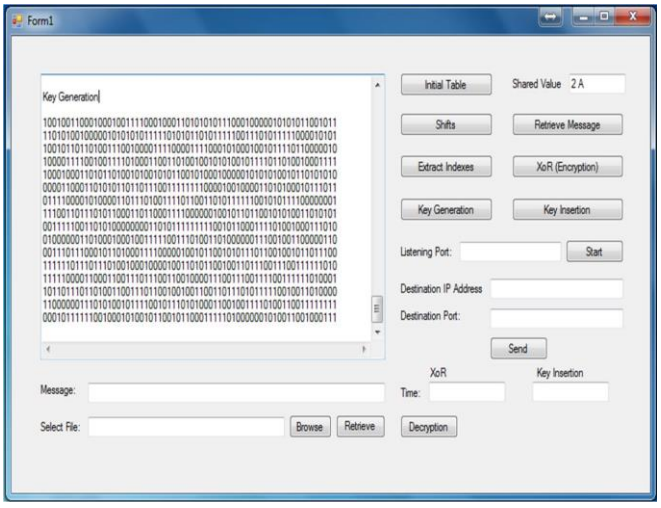

Fig. 3 key insertion process

D. Encryption Process

The index generation process is performed at the beginning of the encryption/decryption process and repeated when there is a need to change the shared value this means that step is less operated than the other step. The output of the Index Generator step is the Extracted Index. The Extracted Index resulted from performing a regular left-circular column shift followed by a down-circular column shift according to the Index values. The Extracted Indexes are taken out of the final result after performing the left/down circular shifts; the Indexes are 128 different indexes, these indexes will remain unchanged until the shared value is changed, changing the shared value depends on many factors for example to change the shared value very frequent will add more complexity over the plain-text and makes it harder to the attacker to guess the key, at the same time this may affect the performance by adding more delay to the packet. The key insertion process is the most used step in this algorithm, the flexibility of changing the key and use a variable key length size is the corner stone in the algorithm strength, the user may stick to a single key during the communication

process and change the shared value or stick to the same shared value and change the key, the result is the same, but it is recommended to change both of the shared value and the key, this will make it more difficult to cryptanalysis either to guess or extract the key. The XoR operations used to generate the XoRed table doesn't consume the machine resources and takes less operations, the XoR operation is very fast to perform and more enhancements like S-Boxes may be used to add more complexity and gives confusion and diffusion to the algorithm.

E. Decryption Process

The index generation process is performed exactly as in the encryption part explained above. Since the process is identical at the sender and the receiver sides the explanation above is enough. The key extraction is there verse of the key insertion process; the extraction starts from the last index value (the 512th index) and ends with the first index value (the 1st index) in reverse order. When pointing to the key position the algorithm will extract the next one octet, the first extraction part represents the last key octet and so on until the whole key recovered back from the XoRed table. The decryption process is easy and straight forward, just perform an XoR operation over the received data and the recovered key, the output is exactly the original plain-text-data after get decrypted.

V. RESULTS

The concept of this paper is implemented and different results are shown below, The proposed paper is implemented in .Net technology on a Pentium-IV PC with minimum 20 GB hard-disk and 1GB RAM. The propose paper's concepts shows efficient results and has been efficiently tested on different Datasets.

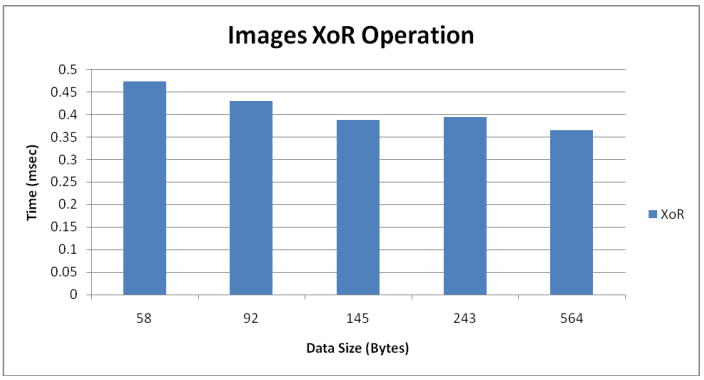

Fig. 4 time Vs data size for XOR images

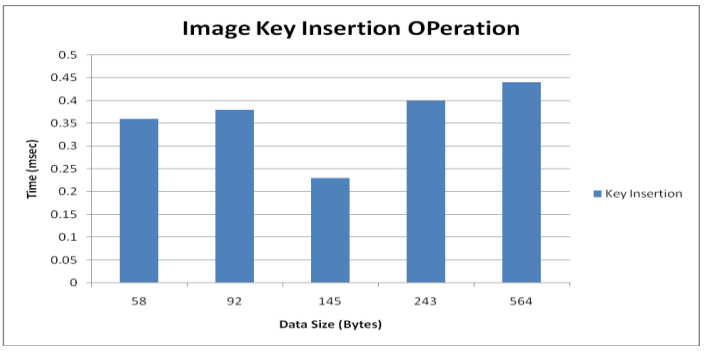

Fig. 5 time Vs data size for key insertion images

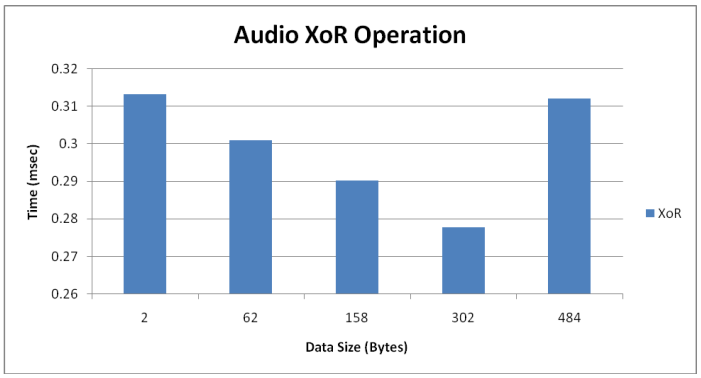

Fig. 6 time Vs data size for XOR audio

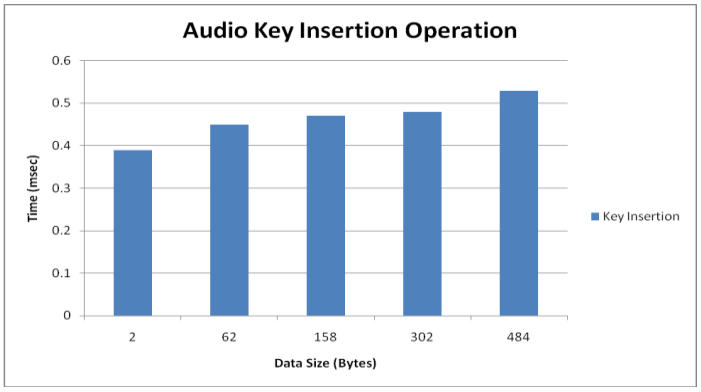

Fig. 7 time Vs data size for key insertion audio

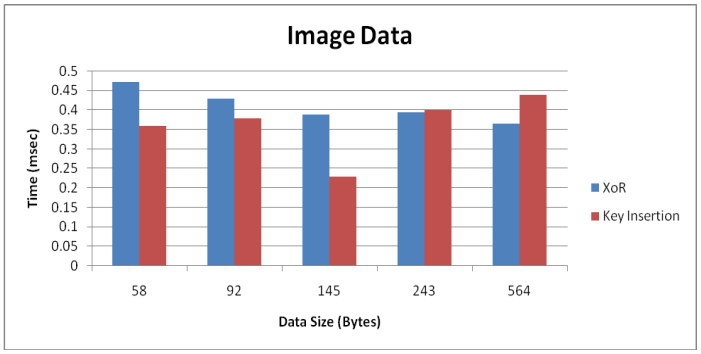

Fig. 8 time Vs data size images

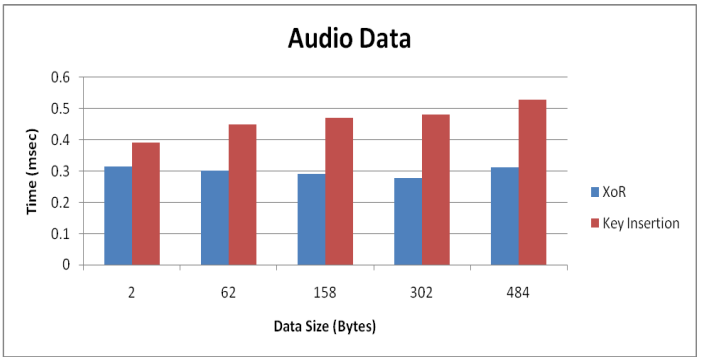

Fig. 9 time Vs data size audio

VI. CONCLUSIONS

The proposed algorithm provided a method for securing the Real-Time traffic in general. The DEA algorithm provides better security than existing encryption/decryption techniques by choosing a key length of 1024-bit long and using a different key for a new packet. The DEA encryption scheme suggests reducing the computational time of underlying encryption algorithm by using an algorithm that is of lesser complexity and takes less computational time.

REFERENCES

[1] Williams Stalling, Cryptography and Network Security *Pinciples and Pratices*, Fouth Edition, Prentice Hall, November 16, 2005.

[2] B. Ontiveros, I. Soto, R. Carrasco, A New Cryptography Algorithm using Cab Curves an LDPC for Wireless Communication Systems, WSEAS Transactions on Mathematics, Vol. 6, No. 1, 2007, pp. 422-425.

[3] D. Stinson, Cryptography Theory and Practice, CRC Press Inc, NY, USA, 1995.

[4] A. H. Omari, B. M. Al-Kasasbeh, R. E. Al-Qutaish and M. I. Al- Muhairat, A New Cryptographic Algorithm for the Real-Time Applications, in Proceedings of the 7th International Conference on Information Security and Privacy - (ISP'08), Cairo, Egypt, from Dec. 29 Dec. 31, 2008.

[5] Ahmad. H. Omari, Basil. M. Al-Kasasbeh, Rafa. E. Al-Qutaish and Mohammad I. Muhairat, DEA-RTA: A Dynamic Encryption Algorithm for Real-Time Applications, International Journal of Computers, Issue 1, Volume 3, 2009.

[6] Saurabh Kumar, Sandeep Kumar, An Enhanced and effective Encrypting Algorithm for High Volume Video Data Streaming Application MANET, International Conference on Recent Advances and Future Trends in Information Technoogy (iRAFIT2012).

[7] B.A Forouzan, Data Communications and Networking, Fouth Edition, McGraw-Hill 2007.

[8] J. Dray, Report on the NIST Java AES Candidate AlgorithmAnalysis,online:http://csrc.nist.gov/encryption/aes/round/r1-java.pdf, 1999, accessed on Sept. 2008.

[9] G. Blelloch, Introduction to Cryptography, online: http://www2.cs.cmu.edu/afs/cs/project/pscicoguyb/realworld/crypto.ps, 2000, accessed on Sept. 2008.

[10] T. Verhoeff, Encryptography, online: http://www pa.win.tue.nl/wstomv/software/AESRijndael/rijndael-test.pas, 2001, accessed on Sept. 2008.

[11] Wikipedia Website, online: http://en.wikipedia.org, accessed on Sept., 2008.

[12] G. Carter, E. Dawsony and L. Nielseny, Key Schedule Classification of the AES Candidates, in Proceedings of the end AES Conference, Rome, Italy, 1999.

[13] V. Hallivuori, Real-Time Transport Protocol (RTP)security, Telecommunications Software and Multimedia Laboratory, T-110.501 Seminar on Network Security, 2004.

[14] J. Dray, Report on the NIST Java AES Candidate Algorithm Analysis, online: http://csrc.nist.gov/ encryption/aes/round/r1-java.pdf, 1999, accessed on Sept. 2008.

