Agent Command Manipulation System Using Two Keys Encryption Model

Sarah Sohana
Dept. of Computer Science & Engineering
Chittagong University of Engineering & Technology
Chittagong, Bangladesh
Email: sarahsohana@gmail.com

Rahma Bintey Mufiz Mukta
Dept. of Computer Science & Engineering
Chittagong University of Engineering & Technology
Chittagong, Bangladesh
Email: rahmamukta@gmail.com

***Abstract*— Cryptography plays a vital role in information security system against malicious attacks. This security system uses some algorithms to scramble data into scribbled text which can only be decrypted by party those possess the associated key. Keys play the most important role in ensuring security of those algorithms. A two keys based method which uses AES as core algorithm is proposed here. This improved method has been developed as application on Android platform which allows the user to encrypt confidential messages before it is transmitted over the network. This application provides a secure, fast and strong encryption of data. There is a huge amount of confusion and diffusion of data during encryption that makes it very hard for an intruder to interpret the encryption pattern and the plain text form of the encrypted data. Comparison between the proposed system and traditional systems show that the proposed one provides enhanced security in data transmission with good performance.**

Keywords—cryptography; encryption; keys; symmetric key encryption; AES; DES; Tripple DES; security

I. INTRODUCTION

The importance of network security has increased dramatically over the past decade with the development of technology. Security of these confidential data is a critical issue. Plenty of safety approaches are available to transfer data within the organization's premises. But when data need to be transferred in the network companies premises, then a good protection technique is needed which is not only secure but also efficient for transferring the data quickly and efficiently. Cryptography is a technique for data protection and security. It has long been used by militaries and governments to facilitate secret communication. Encryption is now commonly used in protecting information within many kinds of civilian systems. According to [18], the Computer Security Institute reported that in 2007, 71% of companies surveyed utilized encryption for some of their data in transit, and 53% utilized encryption for some of their data in storage.

Many research works have been done on encryption algorithms. Some of them are very efficient and able to secure the protected data against unwanted attacks. According to Kerckhoffs's principle, security of a cryptosystem mainly depends on keys. Symmetric key algorithms use one key for encryption. In public key cryptography two keys are used but they are computationally expensive and CPU intensive.

Key can be defined as the rules which are responsible for converting plain text into cipher text. Most of the algorithms use one key. In this paper a system is proposed which uses two keys instead of using one key. It provides more security than the existing symmetric key encryption system. Using the system two agents can share secret and confidential information without any intrusion.

So keeping in mind the improved security and less CPU usage we propose an Android based application using two keys model which allows the user to encrypt the messages before it is transmitted over the network. And the goal of such application is end to end secure message transmission between two agents.

II. SYSTEM DESCRIPTION

A. Overview of System

The total system consists of a smartphone application that is divided into three parts. After the app is launched the user has to log in with a correct username and password to have access to the system. If correct username and password are given he will be able to log on to the system. Otherwise, access is denied. A session for that user is created. A user can either send a new message or read a message from his inbox. While writing the secret message, the agent gives the input data or secret information he wishes to send to another agent. Two different secret keys for encrypting the message are given along. Each command also contains a command ID which helps to trace which message has been encrypted and will be received by the receiver agent. Next step is the encryption of the message into ciphertext using the improved encryption algorithm. When it is completed a confirmation message is be shown to the user. The secured messages in the inbox are shown in encrypted form along with the command ID.

If the agent is reading a ciphertext after having access to the system, the first step will be the selection of the ciphertext that he wishes to decrypt. Next, he input two keys for decrypting the message selected seeing the message code or choosing a random message. Decryption is done using the same algorithm as the encryption. A code is generated depending on the given secret keys. The receiver needs to give both keys correctly. If the correct encryption key is given twice, the agent is able to see the secret code or information sent by the other agent.

A code is also shown in the case of incorrect keys typed, but in that case, the code is a garbage code i.e. the reader will not be able to see the meaningful information that has been sent and thus secrecy is maintained.

It is difficult to remember all the keys that are needed to decrypt. For this reason, a key exchange mechanism is required which is secured. In the key management part, keys of the messages can be found. As keys are the most sensitive elements in the whole encryption process it must be properly managed and exchanged. For accessing keys a user must have to log in using a one-time password. When a user requests this password is generated for a particular session and the code is delivered to the user's personal phone number which is presented in the user's database. However, this code can be used for only one session and expires when the session ends. If the correct code is given the user may have access to the keys for the selected messages. The total system architecture is shown in fig 1.

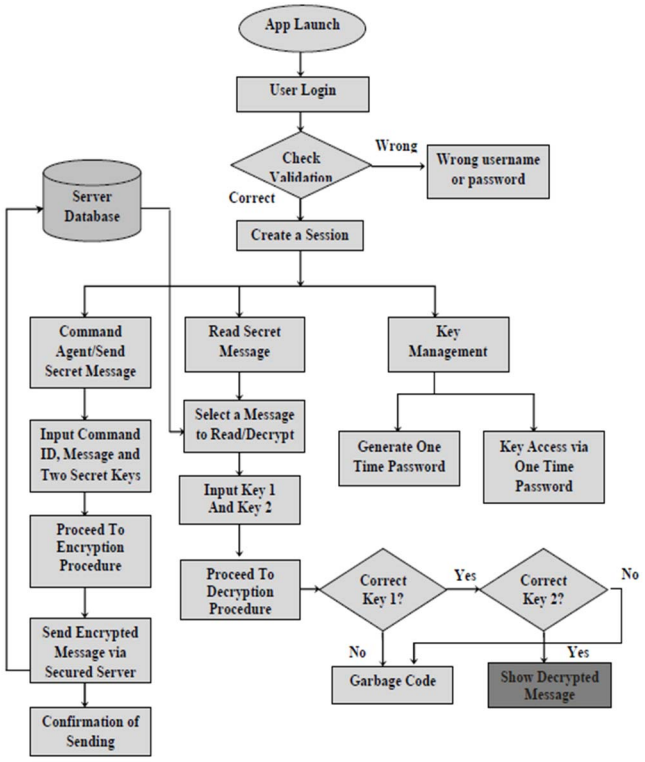

Figure 1: Total System Architecture

B. Improved Encryption Method

The working principle of encryption algorithm is presented in fig 2. Here Key 1 and Key 2 are taken as input. Both keys are converted into binary. Converted keys are bitwise XOR-ed. At the same time, binary keys are encoded using Base-64 encoding method. In the next step Xor-ed keys and Encoded keys are compacted. Keys are scheduled and expanded then fed into AES algorithm scheme.

Plaintext is also taken as input. Here the system allows a 256-bit data length that can be divided into four basic operational blocks. These blocks are organized as a matrix of the order of 4×4 that is called the state. For both encryption and decryption, AddRoundKey is the beginning stage. However, before reaching the final round, this output goes through several rounds. The system goes through 14 rounds for 256-bit keys in order to deliver final result. During each of those rounds four transformations are performed: Sub-bytes, Shift-row, Mix-columns, Add round key.

- Sub-bytes
  It is a non linear byte substitution using a substation table (s box) which is constructed by multiplicative inverse and affine transformation.
- Shift-row
  It is a simple byte transposition, the bytes in the last three rows of state are cyclically shifted; the offset of the left shift varies from one to three bytes.
- Mix-columns
  It is equivalent to a matrix multiplication of columns of the states. Each column vector is multiplied by a fixed matrix.
- Add round Key
  It is a simple XOR between the working state and the round key.

The encryption procedure consists of several steps. A round function is applied to the data block after an initial Addroundkey. It is performed iteratively. In the final round, there is no Mix columns transformation. And as output, a ciphertext is delivered.

The algorithm varies from the traditional AES in terms of keys. Instead of single key, two different keys are used. So randomness is increased.

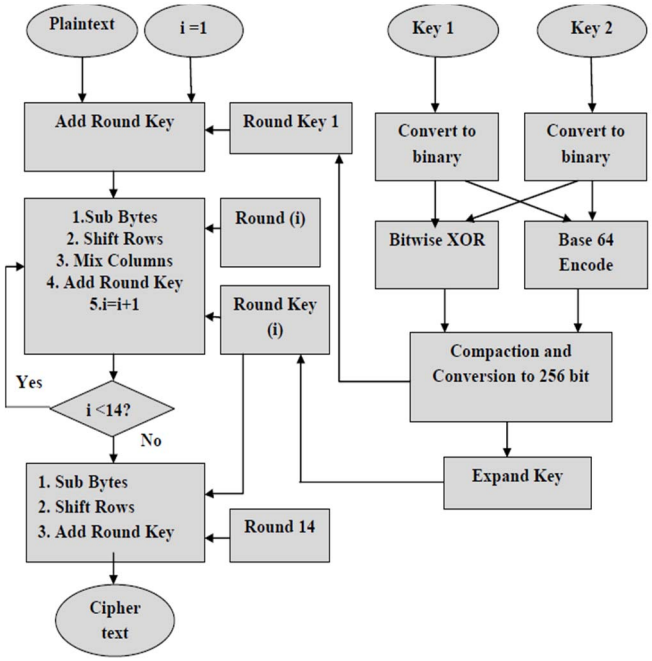

Figure 2: Improved Encryption Method

C. Improved Decryption Method

The decryption method is shown in fig 3. It is similar to the encryption method. In this case, the system requires two keys which receiver gives as inputs and fetches the ciphertext to be decrypted from the database as another input. Unlike

encryption process keys are Xor-ed and encoded and then compacted. They are fed into decryption scheme. Decryption is the reverse process of encryption and it uses inverse transformations: Inverse Substitute Bytes, Inverse Shift Rows, and Inverse Mix Columns. As output, the desired plaintext is delivered. If the input keys are given wrong instead of desired plaintext meaningless texts are shown as output.

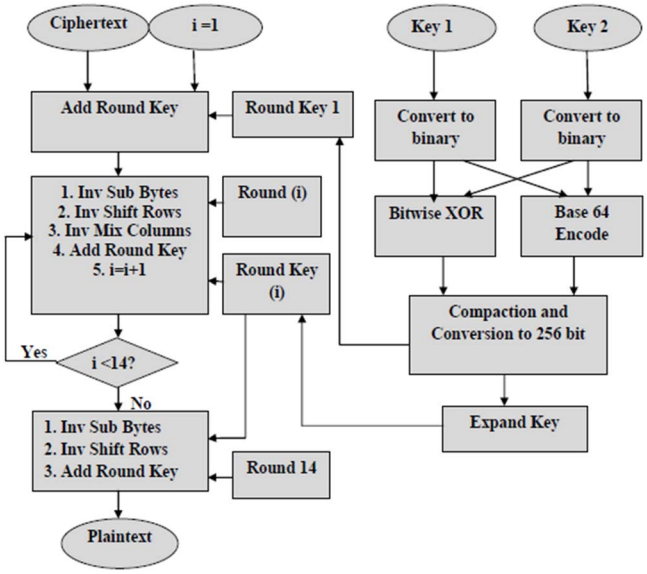

Figure 3: Improved Decryption Method

D. Key Management

Key management consists of two parts: Generate a One Time Password and Key access via One Time Password.

When a user wants to access the keys he needs a password. A one-time password (OTP) is an automatically generated alphanumeric or numeric string of characters that authenticates the user for a single transaction or session. This password is generated one time and expires after the session. A new one is generated for a new session. Generated code is sent via SMS protocol to user's personal phone number that is stored in the server.

After user logs into the system to access the keys he needs to enter the generated code delivered into his phone via SMS. The system then checks if the entered code matches with the code that is stored in the server database. If both codes match, user's access is granted and he can now see the keys for the messages he intends to read. The user can not use the same code again to have access because after the session expires the code cannot be used again.

E. Mobile Application

The system is implemented on Android platform. As it a dynamic app it needs to access a server in order to properly function, i.e. it needs a data connection either through Wi-Fi or phone's carrier.

So a prerequisite for launching the app is data connection. When a connection is established, the application connects to the server after the splash screen. If the connection is failed somehow, then it shows the error message just after the splash screen.

To have access to the system, a user must log in using a username and a password. This user account is created on the server side by the administrator. Unauthorized users can not have access to the system. When logged on with correct username and password, a user session is created by the server.

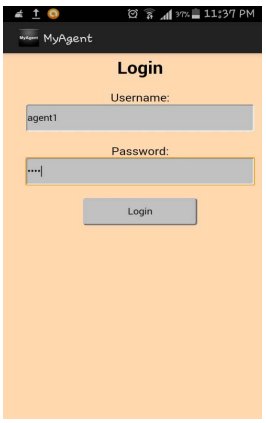

Figure 4: User Login

The Main screen of the app has three main options

- Command an Agent
- Secured Messages
- Key Management

The first option is for sending secret message to an agent. The second option is for reading a secret message. The last one is for accessing keys for decrypting. It has also a logout option to end the session. Screenshot of the main screen is shown in fig 5.

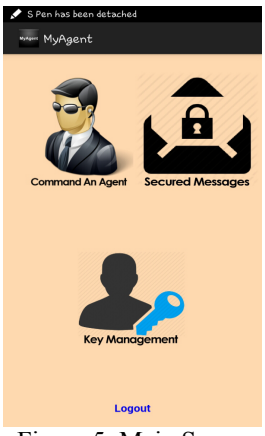

Figure 5: Main Screen

While sending message the user needs to fill the field boxes for message ID, the message, key 1 and key 2. He also needs to specify the user he wants to send the message to. After clicking send button the message is encrypted and transmitted via server. Encryption is done before transmitting. Encrypted messages are then sent to the specific users account. A confirmation message is shown after the message is sent. If any

field is kept blank, no message is transmitted and an error message is shown.

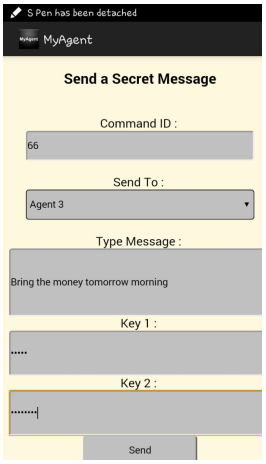

Figure 6: Command an Agent

Encrypted messages that are sent to a user are stored database. When this option in the main screen is selected the encrypted messages along with their message IDs are visible to the user. He can only see the messages that have been sent to his account. The user can then select a message to decrypt and read the secret information. User needs to fill in the field boxes for key 1 and key 2. The keys are identical to the keys that have been used to encrypt the message. Using these two keys the message is then decrypted. This is shown in fig. 7.

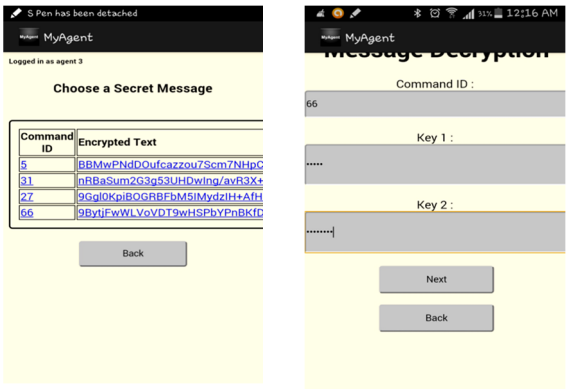

Figure 7: Reading a Secret Message

If correct keys are given the desired plaintext is shown to the user. Otherwise, a garbage code is found.

In our app one time password is used to access the keys. In the key management option, there is a button named 'Generate Onetime Password'. When this button is clicked a one-time password is generated and the password is sent to the user's phone number via SMS. The user can then enter the code to have access to the keys. This code only works for that session. When the session expired the code can no longer be used to access the keys.

III. Experimental Result

This study evaluates different encryption algorithms namely; AES, DES, 3DES and the Proposed Two Key Encryption method. After implementing the system we compare the system with these existing symmetric key systems. The parameters we used for comparing are as follows.

A. Avalanche Effect

A desirable property of any encryption algorithm is that a tiny change in either the plaintext or the key will produce a big change in the cipher text. That means a change in one bit of the either plaintext or the key should produce a change in many bits of the cipher texts. This property is known as Avalanche Effect. As in [15], [16], the more change in avalanche effect the more secure the system.

Avalanche Effect (%) = (Number of Changed Bits in Cipher text) / (Total number of bits in cipher text) * 100

TABLE I. Change in bits in ciphertext (binary) if the plaintext is changed one bit

Algorithm	Ciphertext for plaintext "security"	Ciphertext for plaintext "secusity"	Change In bits
DES	unDK1dCTj8I=	0MtUQ5pFO7I=	30
3DES	8TdxLKYcsS66cMrV0JOPwg==	kUIT6JYc2CPQy1RDmkU7sg==	87
AES	9JodZFHwwZKCqjFpc7IjAa4AGje4HPrUwerhcEDB8pw=	tVWfXk2qyZcnXU+h5zAYKdO/00XrlQ2INsuwKTzSQE0=	138
Proposed System	SE/B1rXPrX7QPzUFtt+iTgpeOWzAD51VHe56HC0M1jo=	vaEstCM40c2r4w717WlvfwgHe9BsEEv+QG2 d0y+olc=	156

The Table I shows the avalanche effect for each algorithm when we took input plaintext as "security" and flipping one bit from the plaintext to get "secusity" (on flipping r (0111 0010) to s (0111 0011)).

TABLE II. Comparison based on Avalanche Effect

Technique	Avalanche Effect (%)
DES	31
Triple DES	45
AES	53
Proposed Algorithm	60

Fig. 8 shows comparison among DES, 3DES, AES and Proposed algorithm based on Avalanche effect. By analyzing the table 2 and fig. 8, it can be noticed that in the proposed algorithm avalanche effect is significantly high i.e. it is 62 due to one-bit change in plaintext keeping the key unchanged, whereas for DES it is 35. Higher avalanche effect means higher security. Therefore, it shows that our proposed system offers higher security.

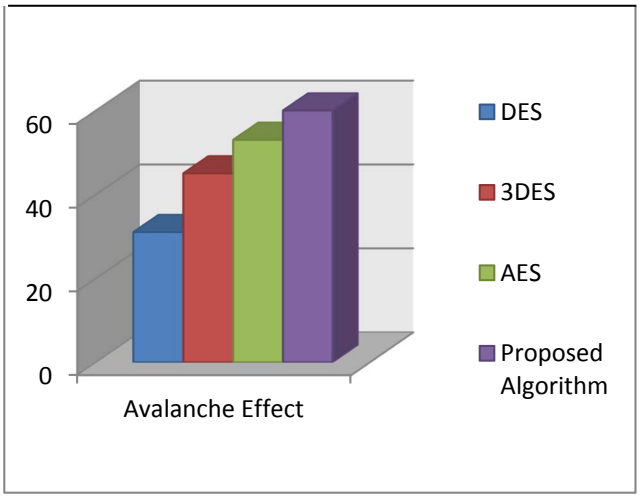

Figure 8: Comparison based on Avalanche Effect

B. Encryption/Decryption Time

It indicates the time required by algorithm for processing a particular length of data. In this experiment the text files sizes range from 49 KB to 7310 KB.

C. CPU Process Time – in the form of Throughput

Encryption time is used to calculate the throughput of an encryption scheme. The throughput is calculated by dividing the total plaintext in MB by total encryption time in Second for each algorithm. Similar procedure has been followed to calculate the throughput of decryption scheme.

TABLE III. COMPARATIVE EXECUTION TIMES (IN MILLISECONDS) OF ENCRYPTION ALGORITHMS WITH DIFFERENT PACKET SIZE

Input Size (KB)	DES	3DES	AES	Proposed Method
49	50	53	63	65
321	74	87	149	151
963	164	177	164	165
5345	783	835	655	663
7310	953	1101	882	894
Average Time	**404.8**	**450.6**	**382**	**387.6**
Through put	**3.374**	**3.031**	**3.55**	**3.524**

Fig. 9 shows the throughput by different encryption algorithms. From table III, we can see that 3DES needs more time to encrypt different file sizes than all other algorithms. So it's throughout is the lowest. AES and our system need almost equal time to encrypt a message. So the throughput for both AES and the two key encryption method are quite similar, despite of the fact two keys are being used in our system.

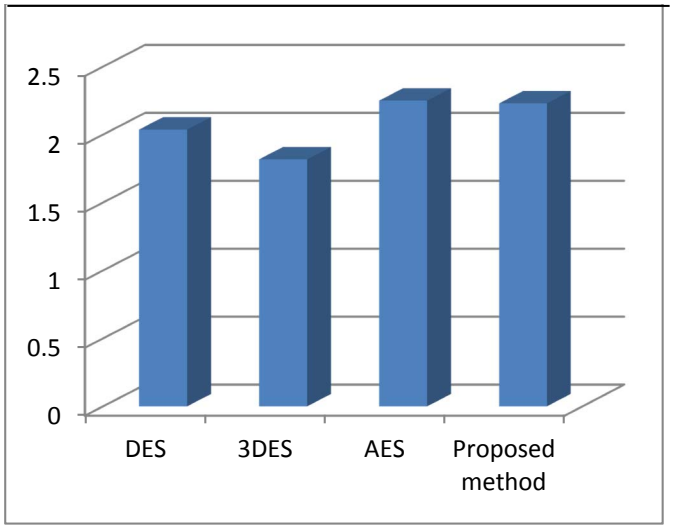

Figure 9: Throughput of each encryption algorithm (Megabyte/Sec)

TABLE IV. COMPARATIVE EXECUTION TIMES (IN MILLISECONDS) OF DECRYPTION ALGORITHMS WITH DIFFERENT PACKET SIZES

Input Size (KB)	DES	3DES	AES	Proposed System
49	29	54	56	57
321	82	167	164	165
963	250	283	208	211
5345	1296	1466	1237	1245
7310	1695	1786	1366	1385
Average Time	670.4	751.2	606	613
Through put(MB/sec)	**2.037**	**1.818**	**2.25**	**2.23**

Table IV depicts the same result for decryption as for encryption. 3DES takes much time to decrypt so its throughput is less than others. AES and our proposed system show almost same output as observed in the both charts.

By analyzing fig. 10, we can see that though our proposed system is using two keys its processing time both is similar to the traditional AES and it is faster than the processing time of

DES. But it can be shown that its security level and encryption intensity is higher than other algorithms. Here, the performance of 3DES is the slowest because of its triple phase encryption characteristics.

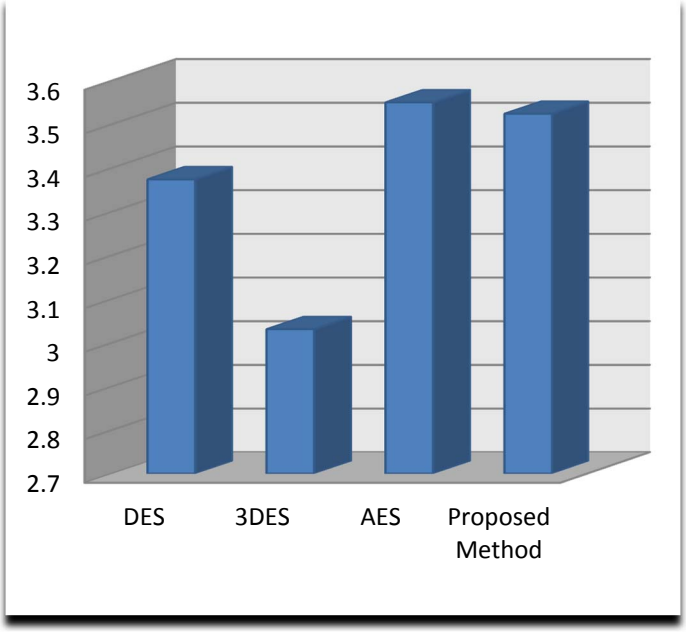

Figure 10: Throughput of each decryption algorithm (Megabyte/Sec)

IV. CONCLUSION

In this paper, we proposed a new application with an improved encryption scheme. We developed the system in a way so that the requirements for speed and compactness can meet. The program size is very small and it can be installed into any cell-phone running on Android platform. The user experiences no delays while using the program, which is a clear indication that the speed requirement is met. We analyzed the security measurements and performance in this paper which is satisfactory. We made sure that the user interface is simple and easy to handle. Most importantly, in this application the messages having confidential information are stored securely and remain undisclosed even when the device is accessed by an adversary. So it can be concluded that this application guarantees a secure end to end transfer of confidential data with ease.

REFERENCES

[1] F. Jing, and Zhu Xian,*"Data Encryption by Two Keys",* Information Science and Engineering (ICISE), 1st International Conference on. IEEE, pp. 1683-1686, 2009.

[2] A.K. Al Tamimi, *"Swati, Performance Analysis of Data Encryption Algorithms'."*, International Journal of Advanced Research in Computer Science and Software Engineering, 3.2, 147-149, 2013.

[3] S. P. Singh, and Raman Maini, *"Comparison of data encryption algorithms.",* International Journal of Computer Science and Communication 2.1: 125-127, 2011.

[4] A. Kumar, Dr Sudesh Jakhar, and S. Kakkar, *"Comparative Analysis between DES and RSA Algorithms.",* International Journal of Advanced Research in Computer Science and Software Engineering 2.7: 386-391, 2012.

[5] J. Thakur, and Nagesh Kumar, *"DES, AES and Blowfish: Symmetric key cryptography algorithms simulation based performance analysis."*, International journal of emerging technology and advanced engineering 1.2: 6-12, 2011.

[6] M.H.Fei, Fan Jing, and Li Hong Lian, *"Order Exchange Key in Data Encryption."*, Recent Advances in Computer Science and Information Engineering, Springer Berlin Heidelberg, pp. 407-412, 2012.

[7] V. Agrawal, Shruti Agrawal, and Rajesh Deshmukh, "*Analysis and Review of Encryption and Decryption for Secure Communication."*, Analysis 2, no. 2 , 2014.

[8] R. Rayarikar, S. Upadhyay and P. Pimpale, *"SMS Encryption using AES Algorithm on Android."*International Journal of Computer Applications 50(19):12-17, July 2012.

[9] D. Hadapad, and Steven Raj N, *"Android Application For Secure File Transferring Using Data Encryption Standard"*, International Journal of Engineering Research & Technology, Vol.2 - Issue 7, July 2013.

[10] A. Bhushan and P.Dulari. "TORDES-the new symmetric key algorithm."

[11] K.R. Saraf, V.P. Jagtap, A.K. Mishra."*Text and Image Encryption Decryption Using Advanced Encryption Standard.*" International Journal of Emerging Trends & Technology in Computer Science (2014).

[12] P.C. Mandal."Evaluation of performance of the Symmetric Key Algorithms: DES, 3DES, AES and Blowfish." Journal of Global Research in Computer Science 3.8 (2012): 67-70.

[13] D. Salama, H.A. Kader, M. Hadhoud. *"Studying the Effects of Most Common Encryption Algorithms."* International Arab Journal of e-Technology 2.1 (2011): 1-10.

[14] Z.P.Buba, G.M.Wajiga,*"Cryptographic algorithms for secure data communication."* International Journal of Computer Science and Security (IJCSS) 5.2 (2011): 227-243.

[15] A. Kumar, M. N. Tiwari. *"Effective implementation and avalanche effect of AES."* International Journal of Security, Privacy and Trust Management (IJSPTM) 1.3/4 (2012): 31-35.

[16] A.K. Mandal, A. Tiwari. *"Analysis of Avalanche Effect in Plaintext of DES using Binary Codes."* International Journal of Emerging Trends and Technology in Computer Science (IJETTCS) 1.3 (2012): 166-177.

[17] E.K. Kavitha. "Performance Evaluation of Cryptographic Algorithms: AES and DES for Implementation of Secured Customer Relationship Management (CRM) System." Journal of e-technology 2.1 (2011).

[18] A. Kumar, S. Sinha, R. Chaudhary. *"A Comparative Analysis of Encryption Algorithms for Better Utilization."* International Journal of Computer Applications 71.14 (2013): 17-23.

[19] A. Doganaksoy, B. Ege, O. Koçak, F. Sulak. *"Cryptographic Randomness Testing of Block Ciphers and Hash Functions."* IACR Cryptology ePrint Archive 2010 (2010): 564.

[20] One time password. Last day of access (September 5, 2015). [Online]. Available: https://en.wikipedia.org/wiki/One-time_password

[21] One Time Password Token & Security Solution. Last day of access (September 5, 2015). [Online]. Available:

[22] http://www.securemetric.com/one-time-password-security.php

[23] Encryption. Last day of access (April 22, 2015). Available:

[24] https://play.google.com/store/apps/details?id=com.TollerTech.Encryption

[25] G. Singh, S. Kinger. *"Integrating AES, DES, and 3-DES Encryption Algorithms for Enhanced Data Security."* International Journal of Scientific & Engineering Research 4.7 (2013): 2058.