International Journal of Advance Research in Science and Engineering
Vol. No.5, Issue No. 03, March 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

# SASA: SMS AUTHENTICITY & SECURITY ON ANDROID ENVIRONMENT USING ASCII CODE CONTROL CHART & ACTIVE KEY

**Ajay Raj Parashar[1], Deepti Mittal[2], Prerana Garg[3]**

[1,2,3] *Information Technology Department, Hindustan College of Science & Technology, (India)*

**ABSTRACT**

*Information security is one of the great concerns at times while transferring few contents publically. SMS (Short Messaging Service) is one of the best and good proven services with a planetary availability in the GSM (Global System for Mobile Communications) networks. SMS is one of the popular ways of communicating as it does not demand cyberspace memory & it is cheap, fast and simple. The main contribution of this work is to introduce SASA model to efficiently embed security & authenticity in SMS communications. The work is about the use of ASCII control code chart & active key to provide SMS security and the message authenticity is provided by computing the message digest of the information to be transmitted & appending it to the data to be sent. This application makes use of stacked in android intents & SMS handler to send & receive messages. The obtained outcome shows the reliable SMS communication on mobile devices.*

***Keywords: Active key, Android, ASCII code control chart, Decryption, Encryption***

## I. INTRODUCTION

SMS (Short Messaging Service) is the most popular data bearer service within GSM (Global System for Mobile Communication) & other cellular networks. It is store-and-forward, easy to use, common & low expenditure service. It is mainly utilized for individual communication, but it has also been utilized in applications where the other party is an information system. The SMS messaging has some security vulnerabilities due to its store-and-forward feature & the problem of forged SMS that can be conducted via cyberspace.

When someone is roaming the SMS content passes through various networks & the internet it to varied vulnerabilities & attacks. While transferring sensitive information via SMS, it is crucial to preserve the content from eavesdroppers & ensure that the message is conveyed by the legitimate sender. SMS provides convenient means for communicating using text messages via mobile devices.

In mobile communication, each message contains a maximum of 140 bytes which is equivalent to 1120 bits. Therefore, one message can contain a maximum of:-

- 160 characters if 7-bit encoding is used.
- 70 characters if 16-bit Unicode character encoding is used.

SMS does not have any stacked in procedure to authenticate the text & render security for data, because most of the applications are designed and developed without taking security into considerations. In this work, main

focus is related to the threats while communicating via SMS in android environment. The work is implemented on android 4.2.0 environment & tested on android mobile.

Now-a-days android mobiles are becoming very popular. Since, android is an open source platform & anyone can develop apps for android, so it is less secure. Hence, android is vulnerable for hacking purpose. The authentication can be ensured by using a hash algorithm. In this work MD5 is used to ensure authenticity. MD5 is a hash algorithm to prepare a message digest of a given plain text.

**Contribution:** In this paper, we are proposing SASA model. SASA model efficiently explains the encryption and decryption of SMS. The transformation of plain text to cipher text and vice-versa is described in order to provide data security.

## II. LITERATURE SURVEY

Mohsen Toorani and Ali A. Beheshti [1] introduced a new secure application layer protocol, called SSMS (A Secure SMS Messaging Protocol for the M-Payment Systems).

It uses an elliptic curve-based public key for the secret key establishment of a symmetric encryption. It efficiently makes the SMS messaging suitable for the m-payment applications.

Chetana Sudhakar, Mina Shivaji and Prof. Santosh Tamboli [2] provided compression & encryption technique for securing SMS data.

This technique compresses the SMS to reduce its length, and then encrypts it using AES and RSA algorithm.

Dinesh P. Baviskar, Sidhhant N. Patil and Onkar K. Pawar [3] proposed message encryption which is character value based encryption in which 3D matrix of message character is replaced with 2D matrix of encryption value.

Poonam Mandavkar, Gauri Patil, Chetna Shetty and Vishal Parkar [4] provide a peer-to-peer SMS security that uses a cryptographic scheme which is a combination of the PBE, SHA, Diffie Hellman Key Exchange, and AES algorithms.

Ch.Rupa and P.S.Avadhani [5] proposed a method for evaluating the performance of message encryption scheme using cheating text.

The original message is embedded into meaningful message called as cheating text. In this method the authentication of the message is also possible because it uses modified message digest algorithm.

Subhasis Banerjee and Utpal Roy [6] designed a scheme which uses common private key cryptography for securing SMS by choosing the key randomly from the table. They also used hashing to ensure message integrity.

Namrata A. Kale, S.B. Natikar and S.M. Karande [7] proposed a technique that involves both encryption & compression process. The SMS is encrypted by using 3D-AES and then compressed by Shannon Fano algorithm.

## III. SCOPE OF THE PROJECT

The work involves providing security mechanism for sending data in secure manner. It is useful in many organizations, military areas as they share confidential data among themselves. The objective of the project is that the communication taking place between two individuals remains between them only and no one else in anyways could get access to it.

We are aiming to develop a communication system which would be an android application where people can share their information regardless of the fear of leaking the message. The work ensures that the message that is being transmitted will not be disclosed to any third person i.e. confidentiality will be maintained.

Confidentiality is the most important aspect of information security as we need to protect our data. It involves protecting our confidential data from unauthorized access. Even if the message is accessed by someone, he/she cannot alter the message according to him/her i.e. the work also provides integrity.

Integrity means ensuring that data is not altered in transit from sender to receiver. Integrity ensures that the changes need to be done only by the authorized users i.e. only the authorized user is allowed to make changes to the data.

Since the message digest 5 will be implemented, it will maintain authenticity. Authenticity is the assurance that the information is shared by the legitimate sender and receiver. It verifies the identity of the sender and the receiver i.e. it ensures that the communicating party is the one that it claims to be.

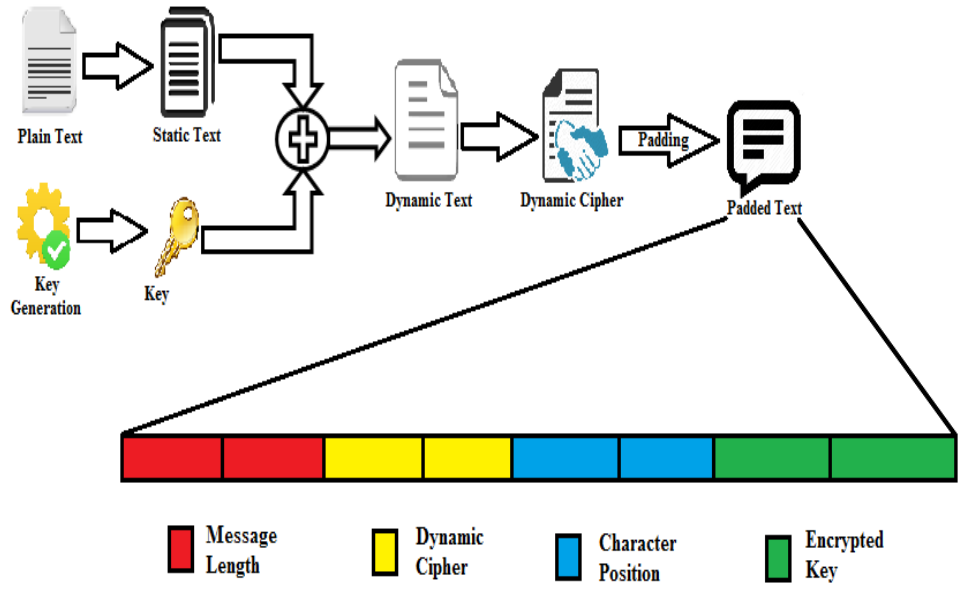
## IV. PROPOSED MODEL

The proposed SASA model is based on ASCII control code chart and active key. It is symmetric cryptography based scheme as it makes use of single key for encryption as well as decryption. SASA model is proposed to provide security to the text messages.

- **Encryption:**

It refers to the scrambling of information or converting understandable or meaningful text called plain text into non-understandable text called cipher text. Message length is the length of the message entered. Dynamic cipher is the actual cipher pair. Character position is the position of cipher pair in the actual message. Encrypted key is value obtained after encryption of the dynamic key.

Static Character ASCII = 254 – (Plain Character ASCII - 14)




- **ASCII control code chart:**

It consists of characters starting from ASCII value 14 to 254 and 254 to 14 along with the corresponding ASCII values as characters from ASCII values 0 to 14 are non-printable characters.

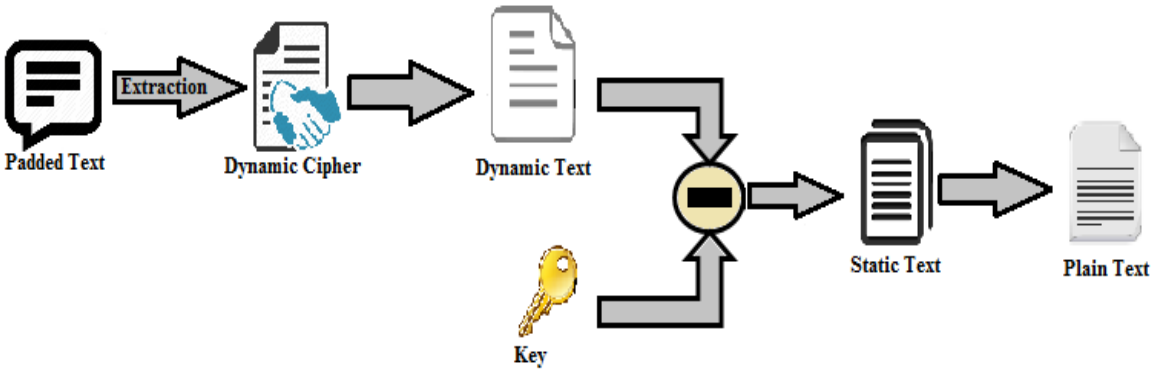
- **Active Key:**

It means that the key used for encryption as well as decryption is dynamic i.e. the key keeps on changing along with the plain text as the key depends upon the message.

- **Decryption:**

The reverse process of encryption i.e. conversion of non-understandable or cipher text into meaningful clear text or plain text is called decryption.

Plain Character ASCII = 254 – (Static Character ASCII - 14)




## V. ALGORITHM

The working of the algorithm proposed in this work is explained below with an example:-

### 1. ENCRYPTION

The work performs encryption by generating active key depending upon the message & using ASCII control code chart.

**Example –** Suppose we have plain text 'HELLO WORLD!'

| **Step 1: Compute static text from plain text using ASCII control code chart** |
|---|
| *Static text (st) = 254 – [Plain Text (pt) – 14]* |
| Plain Text (pt) = HELLO WORLD!<br>pt = {72,69,76,76,79,32,87,79,82,76,68,33}<br>st = {196,199,192,192,189,236,181,189,186,192,200,235} |

| **Step 2: Generation of Active key** |
|---|
| **W** = *Number of consonants in the plain text* |
| **X** = *Number of vowels in the plain text* |
| **Y** = *Number of spaces in the plain text* |
| **Z** = *Number of special symbols in plain text* |
| **N** = *Total number of characters in plain text* |
| *Denominator = (N-W)+(N-X)+(N-Y)+(N-Z)* |

| |
|---|
| *Numerator = 1's compliment of N in 8 bit binary*<br>*K = Numerator / Denominator* |
| Here, W = 7, X = 3, Y = 1, Z = 1, N = 12 = $(00001100)_2 = (0c)_{16}$<br>Denominator = (12–7)+(12–3)+(12–1)+(12–1) = 36<br>Numerator = $(11110011)_2 = 243$<br>K = 243 / 36 = 6 (Approx.) |
| **Step 3: Add the value of key recursively to the value of each static text ASCII to obtain dynamic text ASCII** |
| Dynamic Text (dt) = Static text + Key<br>dt = {202,205,198,198,195,242,187,195,192,198,206,241} |

| **Step 4: Find out the binary for the value of each dynamic text ASCII** | | | | | | | |
|---|---|---|---|---|---|---|---|
| 202 = 11001010 | , | 205 = 11001101 | , | 198 = 11000110 | , | 198 = 11000110 |
| 195 = 11000011 | , | 242 = 11110010 | , | 187 = 10111011 | , | 195 = 11000011 |
| 192 = 11000000 | , | 198 = 11000110 | , | 206 = 11001110 | , | 241 = 11110001 |

| **Step 5: Perform left circular shift once on each nibble of dynamic text binary** |
|---|
| 10010101, 10011011, 10011100, 10011100, 10010110, 11110100, 01110111, 10010110, 10010000, 10011100 , 10011101 , 11110010 |

| **Step 6: Compute dynamic cipher by converting each nibble to hexadecimal** |
|---|
| 95 , 9b , 9c , 9c , 96 , f4 , 77 , 96 , 90 , 9c , 9d , f2 |

| **Step 7: Key Encryption**<br>*Convert Key into binary*<br>*Perform left circular shift on each nibble once*<br>*Convert each nibble into hexadecimal* |
|---|
| K = 6 = $(00000110)_2$<br>After shift:- 00001100<br>Encrypted Key = 0c |

| **Step 8: Perform padding of cipher pair in the following format**<br>*Message Length + Dynamic Cipher + Position +Encrypted Key* |
|---|
| Padding of 95 will be 0c95010c so,<br>Message Sent will be as follows:-<br>0c95010c0c9b020c0c9c030c0c9c040c0c96050c0cf4060c0c77070c0c96080c0c90090c0c9c0a0c<br>0c9d0b0c0cf20c0c |

International Journal of Advance Research in Science and Engineering
Vol. No.5, Issue No. 03, March 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

## VI. DECRYPTION

The encrypted key is extracted from the received message and is decrypted to be used to decrypt the dynamic ciphers to plain text.

**Example –** Suppose received message is:-

'0c95010c0c9b020c0c9c030c0c9c040c0c96050c0cf4060c0c77070c0c96080c0c90090c0c9c0a0c0c9d0b0c0cf20c0c'

| **Step 1: Extract dynamic ciphers & encrypted key from the received message** |
|---|
| Extracted cipher pairs are:-<br>95 , 9b , 9c , 9c , 96 , f4 , 77 , 96 , 90 , 9c , 9d , f2<br>Encrypted Key = 0c |

| **Step 2: Convert each hexadecimal value of dynamic cipher to binary** | | | |
|---|---|---|---|
| 95 = 10010101 | 9b = 10011011 | 9c = 10011100 | 9c = 10011100 |
| 96 = 10010110 | f4 = 11110100 | 77 = 01110111 | 96 = 10010110 |
| 90 = 10010000 | 9c = 10011100 | 9d = 10011101 | f2 = 11110010 |

| **Step 3: Perform circular right shift each nibble once** |
|---|
| 11001010, 11001101, 11000110, 11000110, 11000011, 11110010, 10111011, 11000011, 11000000, 11000110 , 11001110 , 11110001 |

| **Step 4: Obtain dynamic text ASCII by converting binary into decimal** |
|---|
| dt = {202 , 205 , 198 , 198 , 195 , 242 , 187 , 195 , 192 , 198 , 206 , 241} |

| **Step 5: Key Decryption**<br>*Convert hexadecimal key into binary*<br>*Perform circular right shift on each nibble once*<br>*Convert binary into decimal* |
|---|
| Encrypted key = 0c = $(00001100)_2$<br>After shift:- 00000110<br>Decrypted Key = 6 |

| **Step 6: Calculate static text ASCII by subtracting key recursively from dynamic text ASCII** |
|---|
| st = {196 , 199 , 192 , 192 , 189 , 236 , 181 , 189 , 186 , 192 , 200 , 235} |

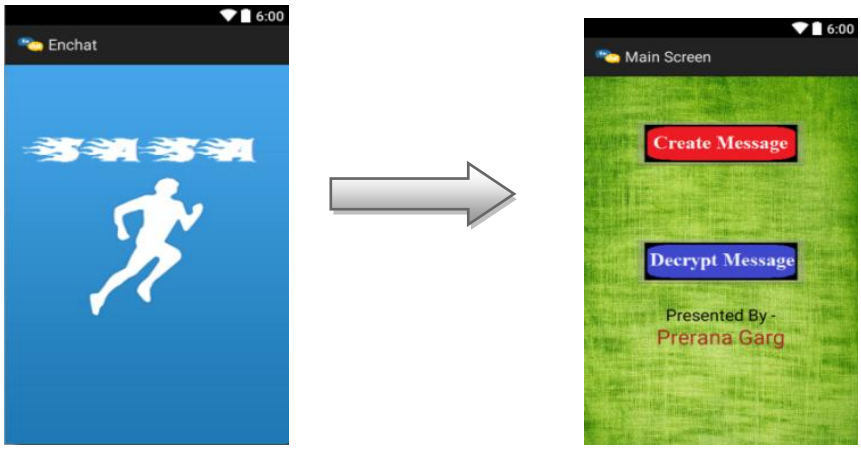
| **Step 7: Compute each plain text character from static text by using**<br>*Plain text ASCII = 254 – [Static text ASCII – 14]*<br>*Convert each plain text ASCII into character by using ASCII control code chart* |
|---|
| Plain text ASCII:-<br>72 , 69 , 76 , 76 , 79 , 32 , 87 , 79 , 82 , 76 , 68 , 33<br>Plain Text is:-<br>HELLO WORLD! |

## VII. PERFORMANCE ANALYSIS

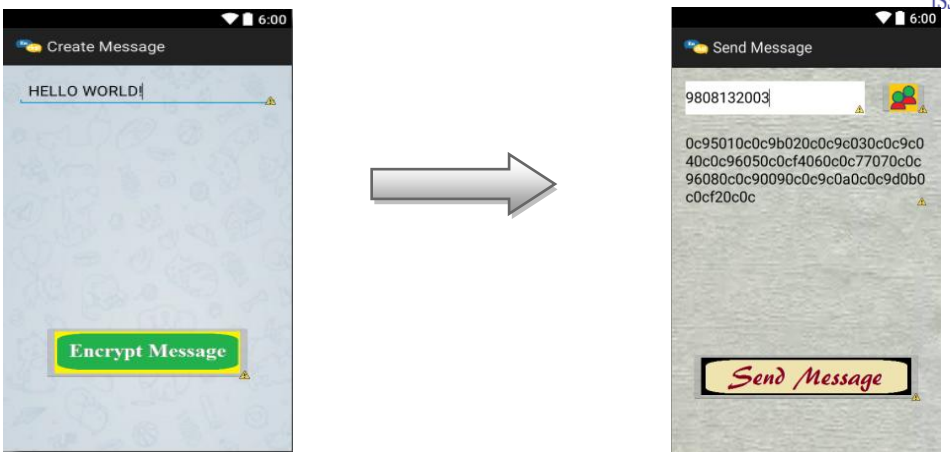
| S No. | Feature | Existing System | Proposed System |
|---|---|---|---|
| 1 | Security | Less Secure | Uses independently developed algorithms |
| 2 | Confidentiality | Low | Algorithm for implementing secrecy is used. |
| 3 | Key Concept | No key used | A new key is generated for each message |
| 4 | Storage Efficiency | Less efficient | More efficient, as ASCII control code chart is computed not stored. |
| 5 | Processor Utilization | More | Less |
| 6 | Attack Prevention | Low | Prevents attacks and tapping |
| 7 | Authenticity | Low | Uses message digest 5 to provide authenticity |

## VIII. INTERFACE

In this application, the interface that appears firstly is splash screen i.e. the interface which disappears after a certain amount of time.

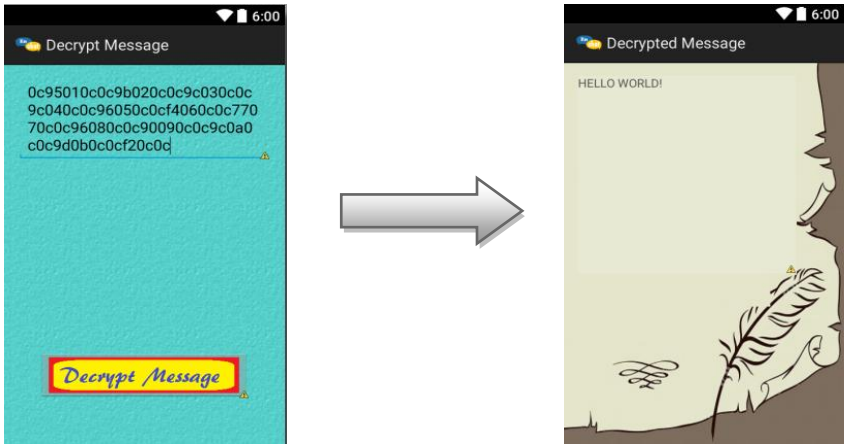



On clicking the create message button as show in the main screen interface following interfaces appears:-




As the user click on send message button, the message sent successfully toast appears & the main screen reappears.

If the user clicks on decrypt message button on the main screen then following interfaces appears:-




## IX. FUTURE DEVELOPMENT

We can improve GUI (Graphical User Interface) of the application. Word document, pdf (portable document file) files, images encryption support can be provided in future.

## X. CONCLUSION

As the messages comprises of confidential data like banking information, transaction details, some military information etc hence, security of messages is of great concern. In this work an algorithm is proposed for messages encryption & decryption which is a symmetric key cryptographic scheme based on ASCII control code chart & active key. The work involves use of built in android Intents and SMS handler & is cost effective, simple and easy to use.

## REFRENCES

**[1]** Mohsen Toorani and Ali A. Beheshti SSMS - A Secure SMS Messaging Protocol for the M-Payment Systems, IEEE Symposium on Computers and Communications, 2012, 700-705.

**[2]** David Lisoněk and Martin Drahanský, SMS Encryption for Mobile Communication, IEEE International Conference on Security Technology, 2008, 198 – 201.

**[3]** Dinesh P. Baviskar, Sidhhant N. Patil and Onkar K. Pawar, for Android based Message Encryption/Decryption using Matrix, International Journal of Research in Engineering & Technology eISSN: 2319-1163 | pISSN: 2321-7308.

**[4]** Punam V. Maitri, Rekha V. Sarawade, Mayuri P. Patil and Sarika T. Deokate for MSC : Mobile Secure Communication Using SMS in Network Security : A Survey, International Journal of Engineering Research & Technology (IJERT) Vol. 2 Issue 11, November – 2013 IJERTIJERT ISSN: 2278-0181.

**[5]** Ch. Rupa and P.S. Avadhani, Message Encryption Scheme Using Cheating Text, IEEE International Conference on Information Technology, 2009, 470-474.

**[6]** Subhasis Banerjee and Utpal Roy for Secure SMS Communication in Android based System with Two Stage Protection, International Journal of Computer Science and Mobile Computing June 2015 ISSN 2320–088X.

**[7]** Namrata A. Kale, S.B. Natikar and S.M. Karande for Secured Mobile Messaging for Android application, International Journal of Advance Research in Computer Science and Management Studies ISSN: 2327782.