www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 9 September 2013 Page No. 2861-2865

Sensitive Data Storage in Wireless Devices Using AES Algorithm

Anjali Patil[1], Rajeshwari Goudar[2]

[1]Maharashtra Academy of Engineering, Department of Computer Engineering,
Alandi Road, Pune-411056, India
anjalimpatil21@gmail.com

[2] Maharashtra Academy of Engineering, Department of Computer Engineering,
Alandi Road, Pune-411056, India
rmgoudar66@gmail.com

Abstract: ***Nowadays, Mobile devices like cellular phones are widely used by most people. This rapid growth in mobile technology makes the delivery of healthcare data on mobile phones. The healthcare data is very sensitive and hence, it must be protected against unauthorized access. Mobile device have two constraints: Memory and Processing Power. AES is well-known standard algorithm for encryption. AES is a block cipher.AES provides flexibility, simplicity, easiness of implementation and high throughput.***

Keywords: Security, AES , Encryption, Decryption.

1. Introduction

Mobile devices have reached more people in developing countries as compared to road system, power grid, water works etc.Mobile devices can not carry physically drugs, doctors and equipment between location but can carry and process information in various ways The personal mobile device may also be used to store information. Applications e.g. mobile electronic payment , secure messaging have an inherent need for security. In information security , cryptography algorithms plays an important role. Embedded systems have mainly 2 constraints[1]: Low memory and Low Processor Capacity. Embedded systems are those which are complete devices often including mechanical parts and hardware. Eg: Mobile phones , digital watches , videogame console , MP3 players , wireless sensor nodes, GPS receivers etc.

Many encryption algorithms are widely available and used in information security. They can be categorized into Symmetric (private) and Asymmetric (public) keys Encryption. In Symmetric keys encryption or secret key encryption, only one key is used to encrypt and decrypt data. DES uses one 64-bits key. Triple DES (3DES) uses three 64- bits keys.AES is Advanced Encryption Standards designed from larger collection of Rijndael algorithm. AES has 3 flavors AES-128, AES-192, AES-256 with block size 128 and number of rounds is 10, 12, and 14 respectively. This Paper describes effective security for data storage in mobile devices by implementing AES algorithm for encryption and decryption.

2. Related Work

To give more prospective about the performance of the compared algorithms, this section discusses the results obtained from other resources. In [2] it is found that after only 600 encryptions of a 5 MB file using Triple-DES the remaining battery power is 45% and subsequent encryptions are not possible due to battery dies rapidly. In [3] it is concluded that AES is faster and more efficient than other encryption algorithms. When the transmission of data is considered there is insignificant difference in performance of different symmetric key schemes. Under the scenario of data transfer it would be better to use AES scheme in case the encrypted data is stored at the other end and decrypted multiple times.

3. Current Security Issues

The use of public-key encryption at the application layer has proved to be unsuitable in embedded environments[4]. This is due to the significant amount of processing and resources required. These resource requirements can lead to intolerable response times in enterprise wireless applications. On the other hand, AES symmetric ciphering has proved to be very efficient in terms of speed and size, and suitable for hardware or software implementations on mobile devices [5].

Choosing J2ME as development platform was due to the following reasons: the portability of Java code, the ability to reduce the network traffic by making use of the processing power of the Java phone to process data locally, and the ability

to establish a kind of a differential security policy on the client

that will perform the encryption operations according to the content and sensitivity of network data rather than encrypting everything. This methodology helped to utilize the embedded device processing power very efficiently. Furthermore, J2ME mobile information device applications (MIDlets) can operate over, and make use of the WAP stack to perform HTTP network interaction, without requiring TCP/IP. A side-effect of devising a general application-layer security solution using J2ME is that it provides a feasible solution to the traditional security gap in the WAP gateway [6]. This security gap is due to the security protocol conversion mechanism taking place in the WAP gateway between secure sockets layer (SSL) encryption and the WAP wireless transport layer security (WTLS) encryption protocols. WTLS is used for securing data between the mobile phone and the WAP gateway, while SSL secures the communication between the WAP gateway and the Internet web server.

4. AES Algorithm

Cryptography plays an important role in the security of data. It enables us to store sensitive information or transmit it across insecure networks so that unauthorized persons cannot read it.

The basic unit for processing in the AES algorithm is a byte (a sequence of eight bits), so the input bit sequence is first transformed into byte sequence. In the next step a two dimensional array of bytes (called the State) is built. The State array consists of four rows of bytes, each containing Nb bytes, where Nb is the block size divided by 32 (number of words). All internal operations (Cipher and Inverse Cipher) of the AES algorithms are then performed on the State array, after which its final value is copied to the output.

The input and output for the AES algorithm each consist of sequences of 128 bits (digits with values of 0 or 1). These sequences will sometimes be referred to as blocks and the number of bits they contain will be referred to as their length. The Cipher Key for the AES algorithm is a sequence of 128, 192 or 256 bits. In AES , there are number of processing rounds. These rounds are based on the key size. If the key length is 128 bits,AES will perform nine processing rounds. If key is of 192 bits, AES perform 12 rounds and if the key size is 256 bits then AES perform 14 processing rounds[7]. Each processing round involves four steps –

· Substitute bytes
· Shift rows
· Mix columns
· Add round key

AES is an iterated symmetric block cipher, which means that:

- AES works by repeating the same defined steps multiple times.
- AES is a secret key encryption algorithm.
- AES operates on a fixed number of bytes.

AES as well as most encryption algorithms is reversible. This means that almost the same steps are performed to complete both encryption and decryption in reverse order. The AES algorithm operates on bytes, which makes it simpler to implement and explain.

The AES encryption procedure is shown in Figure 1. The AES decryption procedure is shown in Figure 2. AES algorithm contain of two parts:

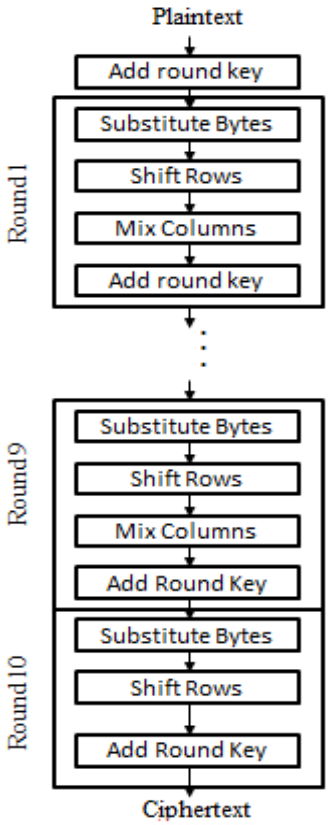

Figure 1: 128 bit encryption AES algorithm

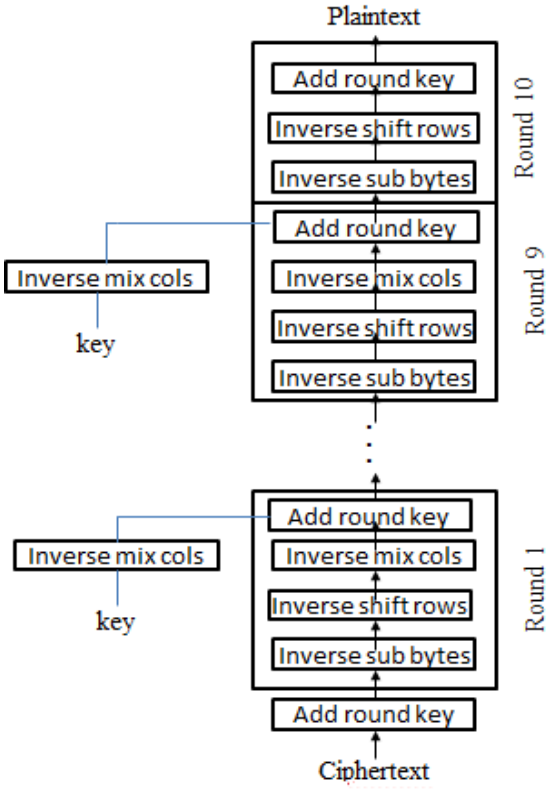

Figure 2: 128 bit decryption AES algorithm

4.1 Round Function

This part consisting of different transformations: subBytes, shiftrows, mixColumns and addroundkey the four transformation are described briefly as follows[8]:

4.2 Structure of Key and Input Data

Both the key and the input data (also referred to as the state) are structured in a 4x4 matrix of bytes. Figure 3 shows how the 128-bit key and input data are distributed into the byte matrices.

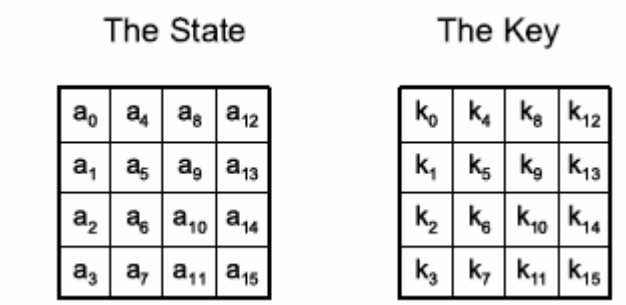

Figure 3: Structure of the Key and the State

4.3 SubstituteBytes(Subbytes Operation)

There are different ways of interpreting the Subbytes operation. In this application , it is sufficient to consider the Subbytes step as a lookup in a table. With the help of this lookup table, the 16 bytes of the state (the input data) are substituted by the corresponding values found in the lookup table (Figure 4.)

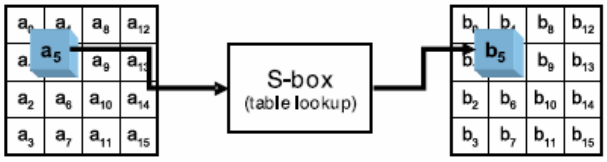

Figure 4: Sub byte Operation

In the S-Box Substitution step, each byte in the matrix is reorganized using an 8-bit substitution box. This substitution box is called the Rijndael S-box[9]. This operation provides the non-linearity in the cipher. The S-box used is derived from the multiplicative inverse over GF (28), known to have good non-linearity properties[10].

4.4 Shift Rows(Shiftrows Operation)

As implied by its name, the Shiftrows operation processes different rows. A simple rotate with a different rotate width is performed. The second row of the 4x4 byte input data (the state) is shifted one byte position to the left in the matrix, the third row is shifted two byte positions to the left, and the fourth row is shifted three byte positions to the left. The first row is not changed.

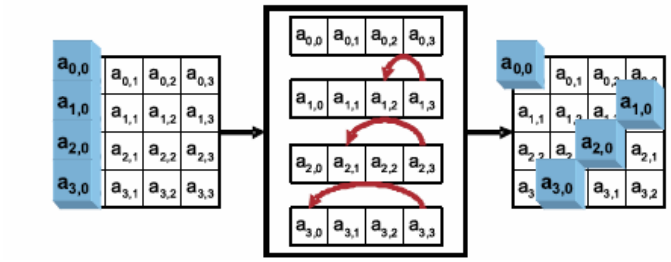

Figure 5: Shiftrows Operation

4.5 Mix Columns(Mix Columns Operation)

Opposed to the Shiftrows operation, which works on rows in the 4x4 state matrix, the Mix columns operation processes columns. Transformation in the Cipher that takes all of the columns of the State and mixes their data (independently of one another) to produce new columns shown in Figure 6.

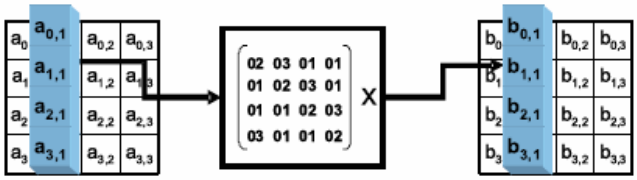

Figure 6: Mixcolumns Operation

4.6 Add Round Key(Addroundkey Operation)

A round key is added to a state. In this operation round key is applied to the state by a simple bit wise XOR. The round key is extracted from the cipher key by means of key schedule. The operation is viewed as a column wise operation between the 4byte of a state column and word of the round key, it can also be viewed as a byte-level operation.

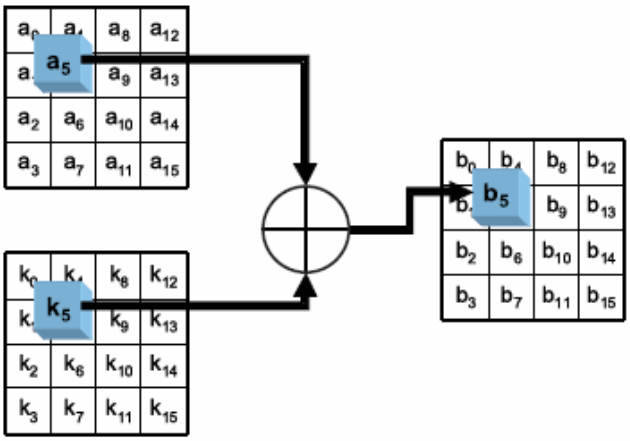

Figure 7: Add Round Key Operation

4.6 Key Expansion

Key expansion refers to the process in which the 128 bits of the original key are expanded into eleven 128-bit round keys. To compute round key (n+1) from round key (n) these steps are performed: Compute the new first column of the next round key as shown in Figure 8:

First all the bytes of the old fourth column have to be substituted using the Subbytes operation. These four bytes are shifted vertically by one byte position and then XORed to the old first column.

The result of these operations is the

- [new second column] = [new first column] XOR [old second column]
- [new third column] = [new second column] XOR [old third column]
- [new fourth column] = [new third column] XOR [old fourth column]

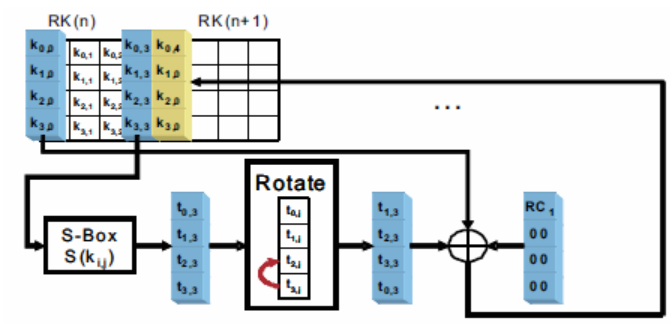

Figure 8: Expanding First Column of Next Round Key

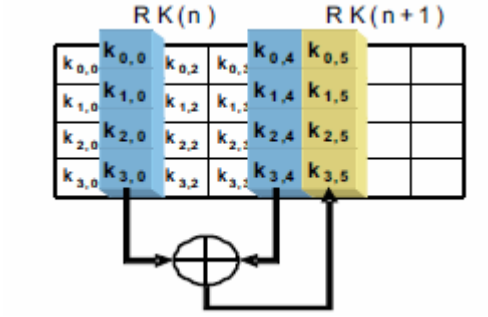

Figure 9: Expanding Other Columns of Next Round Key

5. Notations , Conventions and Mathematical Background

The input and output for the AES algorithm consists of sequences of 128 bits. These sequences are referred to as blocks and the numbers of bits they contain are referred to as their length. The Cipher Key for the AES algorithm is a sequence of 128, 192 or 256 bits. The basic unit of processing in the AES algorithm is a byte**,** which is a sequence of eight bits treated as a single entity[11].

b7 x8 + b6 x7 + b5 x6 + b4 x5 + b3 x4 + b2x3 + b1 x2 + b0 x=Σbixi (1)

Internally, the AES algorithm's operations are performed on a two-dimensional array of bytes called the State. The State consists of four rows of bytes. Each row of a state contains Nb numbers of bytes, where Nb is the block length divided by 32. In the State array, which is denoted by the symbol **S**, each individual byte has two indices. The first byte index is the row number **r**, which lies in the range $0 \le r \le 3$ and the second byte index is the column number **c**, which lies in the range $0 \le c \le Nb-1$. Such indexing allows an individual byte of the State to be referred to as Sr,c or S[r,c]. At the beginning of the Encryption and Decryption the input, which is the array of bytes symbolized by in0in1⋯in15 is copied into the State array. The Encryption or Decryption operations are conducted on the State array[12].

Every byte in the AES algorithm is interpreted as a finite field element using the notation. All Finite field elements can be added and multiplied. The addition of two elements in a finite field is achieved by "adding" the coefficients for the corresponding powers in the polynomials for the two elements. The addition is performed through use of the XOR operation. Multiplying the binary polynomial defined in equation with the polynomial *x* results, can be implemented at the byte level as shift and a subsequent conditional bitwise XOR with left . This operation on bytes is denoted by xtime( ). Multiplication by higher powers of x can be implemented by repeated application of xtime(). Through the addition of intermediate results, multiplication by any constant can be implemented [13].

Conclusion

Embedded systems like Mobile phones, GPS receivers, Wireless Sensor Nodes etc handle sensitive data, hence requires data security mechanisms. AES algorithm which is a standard algorithm for data encryption is suitable for such scenarios where memory and processing power constraints are very high.

References

[1] Embedded Software, Edward A. Lee, Advances in Computers, Academic Press London 2002

[2] Ruangchaijatupon, P. Krishnamurthy, „" Encryption and Power Consumption in Wireless LANs-N,"" The Third IEEE Workshop on Wireless LANs -September 27-28, 2001- Newton, Massachusetts.

[3] Nagesh Kumar, Jawahar Thakur, Arvind Kalia on "PERFORMANCE ANALYSIS OF SYMMETRIC KEY CRYPTOGRAPHY ALGORITHMS:DES , AES and BLOWFISH " in An International Journal of Engineering Sciences ISSN: 2229-6913 Issue Sept 2011, Vol. 4 ,pp.28-37

[4] Harbitter and D. Menasce, "The Performance of Public Key-Enabled Kerberos Authentication in Mobile Computing Applications," *in Proceedings of the 8th ACM conference on Computer and Communications Security*,2001

[5] G. Keating, "Performance Analysis of AES Candidates on the 6805 CPU Core," *URL http://members.ozemail.com.au/~geoffk/aes-6805/paper.pdf*, April 1999

[6] M. Soriano and D. Ponce, "A Security and Usability Proposal for Mobile Electronic Commerce," *IEEE Communications*, August 2002

[7] Himani Agrawal and Monisha Sharma, "Implementation and analysis of various symmetric cryptosystems", Indian Journal of Science and Technology Vol. 3 No.12, December 2010

[8] D. Atnafu, "Optimizing AES Implementation for High Speed Embedded Application," Feb 2008 Addis Ababa

[9] Rohan Rayrikar,Sanket Upadhyay and Priyanka Pimpale, "SMS Encryption using AES Algorithm on Android", International Journal of Computer Applications Vol. 50– No.19, July 2012

[10] J. Yenuguvanilanka and O. Elkeelany, "Performance Evaluation of Hardware Models of Advanced Encryption Standard (AES) Algorithm, ," *Southeastcon, 2008 IEEE*, pp.222-225,2008

[11] William Stallings, "Cryptogrphy and network Security principles and practices",2007 pp 134-165

[12] Luby, M. G., Mitzenmacher, M., Shokrollahi, M. A.,and Spielman, D. A. Efficient erasure correcting codes. *IEEE Transactions on Information Theory*,47,2(Feb.2001)569-583

[13] FIPS 197,"Advanced Encryption Standard *(AES)"*, November 26, 2001 http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

Author Profile

Anjali M. Patil is currently pursuing masters degree program in computer engineering in Maharashtra Academy of Engg , Pune University, India .
E-mail: anjalimpatil21@gmail.com

Prof.R.M.Goudar M.E Computer Engg., Maharashtra Academy of Engg. Pune University, India..
E-mail: rmgoudar66@gmail.com