Analytical Study of AES and Proposed Variant with Enhance Block Length and Key Length

Sayed Tathir Abbas¹, Ravindera Kumar²

¹Al-falah School of Engineering & Technology, Faridabad Haryana, India
²Al-falah School of Engineering & Technology, Faridabad Haryana, India

Abstract: Encryption and decryption are both methods used to ensure the secure passing of messages and other sensitive documents and information. The encryption process plays a major factor in our technology advanced lives. Encryption basically means to convert the message into code or scrambled form. Advanced Encryption Standard (AES) is a specification for the encryption of electronic data. It has been adopted by the U.S. government and is now used worldwide. AES is a symmetric-key algorithm, meaning the same key is used for both encrypting and decrypting the data. This paper defines the method to enhance the block and key length of the conventional AES.

Keywords: Plaintext, Cipher text, key length, block length

1. Introduction

In the United States, AES was announced by National Institute of Standards and Technology (NIST) as U.S. FIPS PUB 197 (FIPS 197) on November 26, 2001 after a five-year standardization process in which fifteen competing designs were presented and evaluated before it was selected as the most suitable. AES is the first publicly accessible and open cipher approved by the National Security Agency (NSA) for top secret information when used in an NSA approved cryptographic module. AES is a symmetric block cipher with a block size of 128 bits. Key lengths can be 128 bits, 192 bits, or 256 bits called AES-128, AES-192 and AES-256 respectively. AES-128 uses 10 rounds, AES-192 uses 12 rounds, and AES-256 uses 14 rounds. In the proposed work the block length and key length are enhanced to two hundred.

2. Architecture

The architecture of this proposal is exactly same as the conventional system. The size of the input key and input data are different than conventional. Here, the block and key length has been enhanced to two hundred. Hence the data matrix used here is 5X5 matrix rather than 4X4 matrix as in conventional AES.

3. Operations Performed

There are several transformations performed to extract the AES cipher. Various transformations are done on byte basis to build AES cipher. The operations performed here are quite different as compared to the conventional addition and multiplication. In AES the addition and multiplication are performed under the “Galois Field” operations. Galois Field is the finite field operation which is named after the brilliant young French mathematician who discovered them.
3.1 G.F Addition

The addition of two bits under GF addition is described in the Table 1.1.

<table>
<thead>
<tr>
<th>+</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Using the above mentioned table, two byte addition is done in the following manner. Given A = \{a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0\} added to B = \{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\} results to C = \{c_7, c_6, c_5, c_4, c_3, c_2, c_1, c_0\}.

3.2 G.F Multiplication

The Multiplication in this field is much more difficult and harder to understand. The first step in multiplying two field elements is to multiply their corresponding polynomials just as in algebra (except that the coefficients are only 0 or 1, and \(1 + 1 = 0\)). The result would be up to a degree 14 polynomial too big to fit into one byte. A finite field now makes use of a fixed degree eight irreducible polynomial (a polynomial that cannot be factored into the product of two simpler polynomials). For the AES the polynomial used is the following

\[C(x) = x^8 + x^4 + x^3 + x + 1\]

The intermediate product of the two polynomials must be divided by \(C(x)\). The remainder from this division is the desired product.

4. AES Rounds

All rounds in proposed AES are identical to the conventional one. The total transformation takes place in one AES round is explained below.

Common Round (State, Round Key)
Byte Sub (State);
Shift Row (State);
Mix Column (State);
Add Round Key (State, Round Key);

Last Round (State, Round Key)
Byte Sub (State);
Shift Row (State);
Add Round Key (State, Round Key);

4.1 Sub Byte

Sub Byte adds confusion by processing each byte through an S-Box. An S-Box is a substitution table, where one byte is substituted for another, based on a substitution algorithm. The S-Box is as

| C(x) = x^8 + x^4 + x^3 + x + 1 |

Table 2: S-Box

<table>
<thead>
<tr>
<th>(x^7)</th>
<th>(x^6)</th>
<th>(x^5)</th>
<th>(x^4)</th>
<th>(x^3)</th>
<th>(x^2)</th>
<th>(x^1)</th>
<th>(x^0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x</td>
<td>63</td>
<td>7c</td>
<td>77</td>
<td>9e</td>
<td>73</td>
<td>4d</td>
<td>6f</td>
</tr>
<tr>
<td>0x</td>
<td>04</td>
<td>58</td>
<td>e7</td>
<td>65</td>
<td>e9</td>
<td>9e</td>
<td>62</td>
</tr>
<tr>
<td>0x</td>
<td>11</td>
<td>67</td>
<td>06</td>
<td>01</td>
<td>86</td>
<td>a2</td>
<td>fb</td>
</tr>
<tr>
<td>0x</td>
<td>77</td>
<td>c5</td>
<td>a5</td>
<td>c1</td>
<td>10</td>
<td>e0</td>
<td>x2</td>
</tr>
<tr>
<td>0x</td>
<td>37</td>
<td>be</td>
<td>5e</td>
<td>a1</td>
<td>f7</td>
<td>d1</td>
<td>a4</td>
</tr>
<tr>
<td>0x</td>
<td>2d</td>
<td>48</td>
<td>03</td>
<td>2b</td>
<td>d0</td>
<td>x7</td>
<td>36</td>
</tr>
</tbody>
</table>

To complete an S-Box operation on an example string of “ABC,” take the hexadecimal value of each byte. ASCII “A” = hex 0x42, “B” = 0x43 and “C” = 0x44. Look up the first (left) hex digit in the S-Box column and the second in the S-Box row. 0x42 becomes 0x2c; 0x43 becomes 0x1a, and 0x44 becomes 0x1b.

4.2 Shift Row

Shift Row provides diffusion by mixing data within rows. Row zero of the State is not shifted, row 1 is shifted 1 byte, row 2 is shifted 2 bytes, and row 3 is shifted 3 bytes, as shown in the fig 2.

Figure 2: Illustration of Shift Row transformation

4.3 Mix Column

Mix column provides diffusion by mixing data within columns. The 4 bytes of each column in the State are treated as a 4-byte number and transformed to another 4-byte number via finite field mathematics, as shown in fig 3. Here \(c(x)\) is the polynomial which is given as

\[C(x) = x^8 + x^4 + x^3 + x + 1\]
4.4 Add Round Key

In this operation, a Round Key is applied to the State by a simple bitwise EX-OR. The Round Key is derived from the Cipher Key by means of the key schedule. The transformation that consists of EXOR a Round Key to the State is denoted by: Add Round Key (State, Round Key).

5. Conclusion

The main application of proposed system is in the places where we can vary the block size according to our need or channel. As we know that any complemented bit in received signal than to transmitted may cause the whole block to vanish. So, we can reduce and increase the block size according to our channel, which can also be named as opportunistic encryption. The proposed model has enhanced the block size as compared to conventional 128 bits. The proposed algorithm can be used in Geographic Information System (GIS) and Satellite Communication when huge data need to be transferred securely. AES with its all variants and proposed model is implemented in Matlab.

References


Authors Profile

Sayed Tathir Abbas received the degree B.Tech. in Electronics and communication Engineering from Veera College of Engineering in 2009 and Pursuing M.Tech in Electronics and communication from Al-Falah School of Engineering and Technology. He has worked in Dr. K. N. Modi Institute of Engineering and Technology in 2009 and joined R.V.Institute of Technology in 2010.

Ravinder Kumar is working as an Assistant Professor in Al-Falah School of Engineering and Technology. He received the degree of B.Tech. and M.Tech in Electronics and communication in 2004 and 2010 respectively. He has more than 9 years of teaching and research experience.