Advanced Encryption Standard (AES)

Raj Jain
Washington University in Saint Louis
Saint Louis, MO 63130
Jain@cse.wustl.edu

Audio/Video recordings of this lecture are available at:
http://www.cse.wustl.edu/~jain/cse571-11/
Overview

1. AES Structure
2. AES Round Function
3. AES Key Expansion
4. AES Decryption

Advanced Encryption Standard (AES)

- Published by NIST in Nov 2001: FIPS PUB 197
- Based on a competition won by Rijmen and Daemen (Rijndael) from Belgium
- 22 submissions, 7 did not satisfy all requirements
- Rijndael allows many block sizes and key sizes
- AES restricts it to:
  - Block Size: 128 bits
  - Key sizes: 128, 192, 256 (AES-128, AES-192, AES-256)
- An iterative rather than Feistel cipher
  - operates on entire data block in every round
- Byte operations: Easy to implement in software
Basic Structure of AES

- # Rounds $N_r = 6 + \max\{N_b, N_k\}$
- $N_b = 32$-bit words in the block
- $N_k = 32$-bit words in key
- AES-128: 10
- AES-192: 12
- AES-256: 14
1. Substitute Bytes

- Each byte is replaced by byte indexed by row (left 4-bits) & column (right 4-bits) of a 16x16 table.
2. Shift Rows

- 1st row is unchanged
- 2nd row does 1 byte circular shift to left
- 3rd row does 2 byte circular shift to left
- 4th row does 3 byte circular shift to left

```
<table>
<thead>
<tr>
<th>87</th>
<th>F2</th>
<th>4D</th>
<th>97</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC</td>
<td>6E</td>
<td>4C</td>
<td>90</td>
</tr>
<tr>
<td>4A</td>
<td>C3</td>
<td>46</td>
<td>E7</td>
</tr>
<tr>
<td>8C</td>
<td>D8</td>
<td>95</td>
<td>A6</td>
</tr>
</tbody>
</table>
```
3. Mix Columns

- Effectively a matrix multiplication in GF($2^8$) using prime polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$

\[
\begin{bmatrix}
02 & 03 & 01 & 01 \\
01 & 02 & 03 & 01 \\
01 & 01 & 02 & 03 \\
03 & 01 & 01 & 02 \\
\end{bmatrix}
\begin{bmatrix}
s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\
s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\
s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\
s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \\
\end{bmatrix}
= 
\begin{bmatrix}
s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\
s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\
s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\
s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \\
\end{bmatrix}
\]

\((\{02\} \cdot \{87\}) \oplus (\{03\} \cdot \{6E\}) \oplus \{46\} \oplus \{A6\} = \{47\}\)
AES Arithmetic

- Uses arithmetic in the finite field GF(2^8) with irreducible polynomial
  \[ m(x) = x^8 + x^4 + x^3 + x + 1 \]
  which is (100011011) or {11B}

- Example:
  \[ \{02\} \cdot \{87\} \mod \{11B\} \]
  \[ = (1 0000 1110) \mod \{11B\} \]
  \[ = (1 0000 1110) \oplus (1 0001 1011) \]
  \[ = (0001 0101) \]
4. Add Round Key

- XOR state with 128-bits of the round key
AES Key Expansion

- Use four byte words called \( w_i \). Subkey = 4 words.
- For AES-128:
  - First subkey (\( w_3, w_2, w_1, w_0 \)) = cipher key
  - Other words are calculated as follows:
    \[
    w_i = w_{i-1} \oplus w_{i-4}
    \]
    for all values of \( i \) that are not multiples of 4.
  - For the words with indices that are a multiple of 4 (\( w_{4k} \)):
    1. RotWord: Bytes of \( w_{4k-1} \) are rotated left shift (nonlinearity)
    2. SubWord: \( SubBytes \) fn is applied to all four bytes. (Diffusion)
    3. The result \( r_{sk} \) is XOR'ed with \( w_{4k-4} \) and a round constant \( r_{conk} \) (breaks Symmetry):
      \[
      w_{4k} = r_{sk} \oplus w_{4k-4} \oplus r_{conk}
      \]
- For AES-192 and AES-256, the key expansion is more complex.
### AES Example Key Expansion

<table>
<thead>
<tr>
<th>Key Words</th>
<th>Auxiliary Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>w0 = 0f 15 71 c9</td>
<td>RotWord(w3) = 7f 67 98 af = x1</td>
</tr>
<tr>
<td>w1 = 47 d9 e8 59</td>
<td>SubWord(x1) = d2 85 46 79 = y1</td>
</tr>
<tr>
<td>w2 = 0c b7 ad</td>
<td>Rcon(1) = 01 00 00 00</td>
</tr>
<tr>
<td>w3 = af 7f 67 98</td>
<td>y1 ⊕ Rcon(1) = d3 85 46 79 = z1</td>
</tr>
<tr>
<td>w4 = w0 ⊕ z1 = dc 90 37 b0</td>
<td>RotWord(w7) = 81 15 a7 38 = x2</td>
</tr>
<tr>
<td>w5 = w4 ⊕ w1 = 9b 49 df e9</td>
<td>SubWord(x4) = 0c 59 5c 07 = y2</td>
</tr>
<tr>
<td>w6 = w5 ⊕ w2 = 97 fe 72 3f</td>
<td>Rcon(2) = 02 00 00 00</td>
</tr>
<tr>
<td>w7 = w6 ⊕ w3 = 38 81 15 a7</td>
<td>y2 ⊕ Rcon(2) = 0e 59 5c 07 = z2</td>
</tr>
<tr>
<td>w8 = w4 ⊕ z2 = d2 c9 6b b7</td>
<td>RotWord(w11) = ff d3 c6 e6 = x3</td>
</tr>
<tr>
<td>w9 = w8 ⊕ w5 = 49 80 b4 5e</td>
<td>SubWord(x2) = 16 66 b4 8e = y3</td>
</tr>
<tr>
<td>w10 = w9 ⊕ w6 = de 7e c6 61</td>
<td>Rcon(3) = 04 00 00 00</td>
</tr>
<tr>
<td>w11 = w10 ⊕ w7 = e6 ff d3 c6</td>
<td>y3 ⊕ Rcon(3) = 12 66 b4 8e = z3</td>
</tr>
</tbody>
</table>
# AES Example Encryption

<table>
<thead>
<tr>
<th>Start of round</th>
<th>After SubBytes</th>
<th>After ShiftRows</th>
<th>After MixColumns</th>
<th>Round Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>01 89 fe 76</td>
<td>ab 8b 89 35</td>
<td>ab 8b 89 35</td>
<td>b9 94 57 75</td>
<td>0f 47 0c af</td>
</tr>
<tr>
<td>23 ab dc 54</td>
<td>05 40 7f f1</td>
<td>40 7f f1 05</td>
<td>e4 8e 16 51</td>
<td>15 d9 b7 7f</td>
</tr>
<tr>
<td>45 cd ba 32</td>
<td>18 3f f0 fc</td>
<td>f0 fc 18 3f</td>
<td>47 20 9a 3f</td>
<td>71 e8 ad 67</td>
</tr>
<tr>
<td>67 ef 98 10</td>
<td>e4 4e 2f c4</td>
<td>c4 e4 4e 2f</td>
<td>c5 d6 f5 3b</td>
<td>c9 59 d6 98</td>
</tr>
<tr>
<td>0e ce f2 d9</td>
<td>65 0f c0 4d</td>
<td>4d 76 ba e3</td>
<td>8e 22 db 12</td>
<td>df 80 f7 c1</td>
</tr>
<tr>
<td>36 72 6b 2b</td>
<td>92 c6 9b 70</td>
<td>c6 9b 70 92</td>
<td>2d c5 1e 52</td>
<td>6b b4 c6 d3</td>
</tr>
<tr>
<td>34 25 17 55</td>
<td>51 16 9b e5</td>
<td>9b e5 51 16</td>
<td>b7 5e 61 c6</td>
<td>b7 5e 61 c6</td>
</tr>
<tr>
<td>ae b6 4e 88</td>
<td>9d 75 74 de</td>
<td>de 9d 75 74</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Washington University in St. Louis  CSE571S  ©2011 Raj Jain
## AES Example Avalanche

<table>
<thead>
<tr>
<th>Round</th>
<th>Number that differ</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>58</td>
</tr>
<tr>
<td>3</td>
<td>59</td>
</tr>
<tr>
<td>4</td>
<td>61</td>
</tr>
<tr>
<td>5</td>
<td>68</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>67</td>
</tr>
<tr>
<td>8</td>
<td>65</td>
</tr>
<tr>
<td>9</td>
<td>61</td>
</tr>
<tr>
<td>10</td>
<td>58</td>
</tr>
</tbody>
</table>
AES Decryption

- AES decryption is not identical to encryption
- But each step has an inverse
Summary

1. AES encrypts 128 bit blocks with 128-bit, 192-bit or 256-bit keys using 10, 12, or 14 rounds, respectively.
2. Is not a Feistel cipher ⇒ All 128 bits are encrypted
3. Each round = 4 steps of SubBytes, ShiftRows, MixColumns, and AddRoundKey.
4. Last round has only 3 steps. No MixColumns.
5. Decryption is not the same as encryption (as in DES). Decryption consists of inverse steps.
5.4 Given the plaintext [0001 0203 0405 0607 0809 0A0B 0C0D 0E0F] and the key [0101 0101 0101 0101 0101 0101 0101 0101]
a. Show the original contents of state, displayed as a 4x4 matrix.
b. Show the value of state after initial AddRoundKey.
c. Show the value of State after SubBytes.
d. Show the value of State after ShiftRows.
e. Show the value of State after MixColumns.
This homework requires two computers with OpenSSH and telnet client and servers installed. You can use CSE571XPC2 client and CSE571XPS server or your own computers.

Start wireshark on the client machine.

telnet to the server and login with your username and password. Logout.

Use “follow the TCP stream option” (right click on the packet) to see your username and password on the screen. Capture the screen and circle your password.

ssh to the server and login with your username and password. Logout.

Stop wireshark and read the trace. Capture the screen. Circle the password characters.

Note the difference in the two logins?