The Advanced Encryption Standard: Rijndael
In 1997, NIST put out a call for candidates to replace DES. Among the requirements were that the new algorithm should allow key size of 128, 192, and 256 bits, it should operate on blocks of 128 input bits, and it should work on different hardware, e.g., 8-bit (smard cards) and the 32-bit (PCs).

In 1998, the cryptographic community was asked to comment on 15 candidate algorithms.
Introduction

- In August 1999, five finalists were chosen:
  MARS (from IBM)
  RC6 (from RSA Laboratories)
  Rijndael (from Joan Daemen and Vincent Rijmen)
  Serpent (from Anderson, Biham, and Knudsen)
  Twofish (from Schneier, Kelsey, Whiting, Wagner, Hall, and Ferguson)

- On Oct. 2, 2000, Rijndael was selected to be AES.
  - 3 main criteria: security, cost, algorithm and implementation characteristics
The Advanced Encryption Standard

- **Description of AES**
  - **Block length:**
    - 128 bits (Nb=4)
  - **Key length:**
    - 128 bits (Nk=4)
    - 192 bits (Nk=6)
    - 256 bits (Nk=8)
  - **Number of rounds Nr**

<table>
<thead>
<tr>
<th>Nr</th>
<th>Nb = 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nk = 4</td>
<td>10</td>
</tr>
<tr>
<td>Nk = 6</td>
<td>12</td>
</tr>
<tr>
<td>Nk = 8</td>
<td>14</td>
</tr>
</tbody>
</table>
The Advanced Encryption Standard

- **Overview of AES:**
  - $A_{DDROUNDKEY}$, which xors the RoundKey with State.
  - For each of the first $Nr-1$ rounds: perform $S_{UBBYTES}(State)$, $S_{SHIFTROWS}(State)$, $M_{IXCOLUMN}(State)$, $A_{DDROUNDKEY}$.
  - Final round: $S_{UBBYTES}$, $S_{SHIFTROWS}$, $A_{DDROUNDKEY}$.

- All operations in AES are byte-oriented.
  - The plaintext $x$ consists of 16 byte, $x_0,x_1,\ldots,x_{15}$.
  - Initially State is plaintext $x$ (for 128-bit case):
The Advanced Encryption Standard

- **S**UB**B**YTE**S**: It performs a substitution on each byte of State using an S-box, say $\pi_S$.
- $\pi_S$ is a 16x16 array (Figure A). A byte is represented as two hexadecimal digits XY. So XY after substitution is $\pi_S(XY)$. 

![S-box diagram](image)
| X | Y  | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | A  | B  | C  | D  | E  | F  |
|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0 | 63 | 7C | 77 | 7B | F2 | 6B | 6F | C5 | 30 | 01 | 67 | 2B | FE | D7 | AB | 76 |
| 1 | CA | 82 | C9 | 7D | FA | 59 | 47 | F0 | AD | D4 | A2 | AF | 9C | A4 | 72 | C0 |
| 2 | B7 | FD | 93 | 26 | 36 | 3F | F7 | CC | A5 | E5 | F1 | 71 | D8 | 31 | 15 |
| 3 | 04 | C7 | 23 | C3 | 18 | 96 | 05 | 9A | 07 | 12 | 80 | E2 | EB | 27 | B2 | 75 |
| 4 | 09 | 83 | 2C | 1A | 1B | 6E | 5A | A0 | 52 | 3B | D6 | B3 | 29 | E3 | 2F | 84 |
| 5 | 53 | D1 | 00 | ED | 20 | FC | B1 | 5B | 6A | CB | BE | 39 | 4A | 4C | 58 | CF |
| 6 | D0 | EF | AA | FB | 43 | 4D | 33 | 85 | 45 | F9 | 02 | 7F | 50 | 3C | 9F | A8 |
| 7 | 51 | A3 | 40 | 8F | 92 | 9D | 38 | F5 | BC | B6 | DA | 21 | 10 | FF | F3 | D2 |
| 8 | CD | 0C | 13 | EC | 5F | 97 | 44 | 17 | C4 | A7 | 7E | 3D | 64 | 5D | 19 | 73 |
| 9 | 60 | 81 | 4F | DC | 22 | 2A | 90 | 88 | 46 | EE | B8 | 14 | DE | 5E | 0B | DB |
| A | E0 | 32 | 3A | 0A | 49 | 06 | 24 | 5C | C2 | D3 | AC | 62 | 91 | 95 | E4 | 79 |
| B | E7 | C8 | 37 | 6D | 8D | D5 | 4E | A9 | 6C | F4 | EA | 65 | 7A | AE | 08 |
| C | BA | 78 | 25 | 2E | 1C | A6 | B4 | C6 | E8 | DD | 74 | 1F | 4B | BD | 8B | 8A |
| D | 70 | 3E | B5 | 66 | 48 | 03 | F6 | 0E | 61 | 35 | 57 | B9 | 86 | C1 | 1D | 9E |
| E | E1 | F8 | 98 | 11 | 69 | D9 | 8E | 94 | 9B | 1E | 87 | E9 | CE | 55 | 28 | DF |
| F | 8C | A1 | 89 | 0D | BF | E6 | 42 | 68 | 41 | 99 | 2D | 0F | B0 | 54 | BB | 16 |

**Figure A**
The AES S-box
The Advanced Encryption Standard

- The **AES** S-box can be defined algebraically. The permutation $\pi_S$ incorporates operations in the finite field $\mathbb{F}_{2^8} = \mathbb{Z}_2[x]/(x^8 + x^4 + x^3 + x + 1)$.

- $\text{FIELD}_{\text{INV}}$: the multiplicative inverse of a field element
- $\text{BINARY}_{\text{TOFIELD}}$: convert a byte to a field element
- $\text{FIELD}_{\text{TOBINARY}}$: inverse operation

\[
\sum_{i=0}^{7} a_i x^i
\]

corresponds to the byte $a_7a_6a_5a_4a_3a_2a_1a_0$
The Advanced Encryption Standard

Algorithm B: $\text{SUBBYTES}(a_7a_6a_5a_4a_3a_2a_1a_0)$

- external $\text{FIELDINV, BINARYTOFIELD, FIELDTOBINARY}$
- $z \leftarrow \text{BINARYTOFIELD}(a_7a_6a_5a_4a_3a_2a_1a_0)$
- if $z \neq 0$
  - then $z \leftarrow \text{FIELDINV}(z)$
- $(a_7a_6a_5a_4a_3a_2a_1a_0) \leftarrow \text{FIELDTOBINARY}(z)$
- $(c_7c_6c_5c_4c_3c_2c_1c_0) \leftarrow (01100011)$

**comment:** In the following loop, all subscripts are to be reduced modulo 8

- for $i \leftarrow 0$ to 7
  - do $b_i \leftarrow (a_i + a_{i+4} + a_{i+5} + a_{i+6} + a_{i+7} + c_i) \mod 2$
- return $(b_7b_6b_5b_4b_3b_2b_1b_0)$
Operations in Algorithm B

- Take a byte \((a_7a_6a_5a_4a_3a_2a_1a_0)\) as \(\sum_{i=0}^{7} a_ix^i\) in \(\mathbb{F}_{2^8}\)
- for \(\sum_{i=0}^{7} a_ix^i \neq 0\), Find its multiplicative inverse \(\sum_{i=0}^{7} d_ix^i\)
- \((d_7d_6d_5d_4d_3d_2d_1d_0) \rightarrow (a_7a_6a_5a_4a_3a_2a_1a_0)\)

\[
\begin{bmatrix}
 b_0 \\
 b_1 \\
 b_2 \\
 b_3 \\
 b_4 \\
 b_5 \\
 b_6 \\
 b_7 \\
\end{bmatrix} =
\begin{bmatrix}
 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\
 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\
 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\
 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
\end{bmatrix}
\begin{bmatrix}
 a_0 \\
 a_1 \\
 a_2 \\
 a_3 \\
 a_4 \\
 a_5 \\
 a_6 \\
 a_7 \\
\end{bmatrix}
+ \begin{bmatrix}
 c_0 \\
 c_1 \\
 c_2 \\
 c_3 \\
 c_4 \\
 c_5 \\
 c_6 \\
 c_7 \\
\end{bmatrix}
\]
The Advanced Encryption Standard

- **Example C: (illustrates Algorithm B)**
  - Suppose we begin with (hex) 53. In binary, it’s 01010011, which represents the field element
    \[ x^6 + x^4 + x + 1 \]
  The multiplicative inverse (in \( \mathbb{F}_{2^8} \)) can be shown to be
  \[ x^7 + x^6 + x^3 + x \]
  Thus we have
  \[(a_7a_6a_5a_4a_3a_2a_1a_0) = (11001010).\]
The Advanced Encryption Standard

\[ b_0 = a_0 + a_4 + a_5 + a_6 + a_7 + c_0 \mod 2 \]
\[ = 0 + 0 + 0 + 1 + 1 + 1 \mod 2 \]
\[ = 1 \]

\[ b_1 = a_1 + a_5 + a_6 + a_7 + a_0 + c_1 \mod 2 \]
\[ = 1 + 0 + 1 + 1 + 0 + 1 \mod 2 \]
\[ = 0, \]

etc. The result is

\[ (b_7b_6b_5b_4b_3b_2b_1b_0) = (11101101). \]

which is \textbf{ED} in hex.

- This computation can be checked by verifying the entry in row 5 and column 3 of \textit{Figure A}.\hfill \Box
The Advanced Encryption Standard

- **SHIFTROWS:**

<table>
<thead>
<tr>
<th>Row 0: no shift</th>
<th>Row 1: shift 1</th>
<th>Row 2: shift 2</th>
<th>Row 3: shift 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{0,0}$</td>
<td>$S_{0,1}$</td>
<td>$S_{0,2}$</td>
<td>$S_{0,3}$</td>
</tr>
<tr>
<td>$S_{1,0}$</td>
<td>$S_{1,1}$</td>
<td>$S_{1,2}$</td>
<td>$S_{1,3}$</td>
</tr>
<tr>
<td>$S_{2,0}$</td>
<td>$S_{2,1}$</td>
<td>$S_{2,2}$</td>
<td>$S_{2,3}$</td>
</tr>
<tr>
<td>$S_{3,0}$</td>
<td>$S_{3,1}$</td>
<td>$S_{3,2}$</td>
<td>$S_{3,3}$</td>
</tr>
</tbody>
</table>

- **Row 0:** no shift
- **Row 1:** shift 1
- **Row 2:** shift 2
- **Row 3:** shift 3
The Advanced Encryption Standard

- **MixColumns**: (Algorithm D)
  - It is carried out on each of the four columns of \( \text{State} \).
  - Each column of \( \text{State} \) is replaced by a new column which is formed by multiplying that column by a certain matrix of elements of the field \( \mathbb{F}_{2^8} \).
  - \( \text{FieldMult} \) computes two inputs product in the field.

Note: 2 is \( x \) in \( \mathbb{F}_{2^8} \) and 3 is \( x+1 \) in \( \mathbb{F}_{2^8} \).
The Advanced Encryption Standard

Algorithm D: \texttt{MIXCOLUMN}(c)

\textbf{external} \texttt{FIELDMULT}, \texttt{BINARYTOFIELD}, \texttt{FIELDTOBINARY}

\texttt{for} \ i \leftarrow 0 \ \texttt{to} \ 3

\texttt{do} \ t_i \leftarrow \texttt{BINARYTOFIELD}(s_{i,c})

\texttt{u}_0 \leftarrow \texttt{FIELDMULT}(x,t_0) \oplus \texttt{FIELDMULT}(x+1,t_1) \oplus t_2 \oplus t_3

\texttt{u}_1 \leftarrow \texttt{FIELDMULT}(x,t_1) \oplus \texttt{FIELDMULT}(x+1,t_2) \oplus t_3 \oplus t_0

\texttt{u}_2 \leftarrow \texttt{FIELDMULT}(x,t_2) \oplus \texttt{FIELDMULT}(x+1,t_3) \oplus t_0 \oplus t_1

\texttt{u}_3 \leftarrow \texttt{FIELDMULT}(x,t_3) \oplus \texttt{FIELDMULT}(x+1,t_0) \oplus t_1 \oplus t_2

\texttt{for} \ i \leftarrow 0 \ \texttt{to} \ 3

\texttt{do} \ s_{i,c} \leftarrow \texttt{FIELDTOBINARY}(u_i)
The Advanced Encryption Standard

- **$\text{KEY EXPANSION}$**: (for 10-round AES)
  - 10-round, 128-bit key
  - We need 11 round keys, each of 16 bytes
  - Key scheduling algorithm is word-oriented (4 bytes), so a round key consists of 4 words
  - The concatenation of round keys is called the **expanded key**, which consists of 44 words, $w[0], w[1],…, w[43]$.
  - See **Algorithm E**
The Advanced Encryption Standard

- Notations of Algorithm E:
  - Input: 128-bit key, \texttt{key}, \texttt{key[0]},…,\texttt{key[15]}
  - Output: words, \texttt{w}
  - \texttt{ROTWORD}: a cyclic shift of four bytes \(B_0,B_1,B_2,B_3\)
    \[
    \texttt{ROTWORD}(B_0,B_1,B_2,B_3) = (B_1,B_2,B_3,B_0)
    \]
  - \texttt{SUBWORD}: applies the S-box to each byte
    \[
    \texttt{SUBWORD}(B_0,B_1,B_2,B_3) = (B_0',B_1',B_2',B_3')
    \]
    where \(B_i' = \texttt{SUBBYTES}(B_i)\)
  - \texttt{RCon}: an array of 10 words, \texttt{RCon[1]},…,\texttt{RCon[10]}, they are constants defined at the beginning
Algorithm E: \texttt{KEYEXPANSION(key)}

\begin{verbatim}
external \texttt{ROTWORD, SUBWORD}
RCon[1] \leftarrow 01000000
RCon[2] \leftarrow 02000000
RCon[3] \leftarrow 04000000
RCon[4] \leftarrow 08000000
RCon[5] \leftarrow 10000000
RCon[6] \leftarrow 20000000
RCon[7] \leftarrow 40000000
RCon[8] \leftarrow 80000000
RCon[9] \leftarrow 1B000000
RCon[10] \leftarrow 36000000
for \ i \leftarrow 0 \ to \ 3
   do w[i] \leftarrow (key[4i], key[4i+1], key[4i+2], key[4i+3])
for \ i \leftarrow 4 \ to \ 43
   do temp \leftarrow w[i-1]
      if \ i \equiv 0 \ (\text{mod} \ 4)
      then temp \leftarrow \texttt{SUBWORD(ROTWORD(temp))} \oplus \text{RCon}[i/4]
      w[i] \leftarrow w[i-4] \oplus \text{temp}
return (w[0], \ldots, w[43])
\end{verbatim}

round key 0: w[0], w[1], w[2], w[3]
round key 1: w[4], w[5], w[6], w[7]
\ldots
round key 10: w[40], w[41], w[42],..,w[43]
The Advanced Encryption Standard

- Above are the operations need to encrypt in AES.
- To decrypt, we perform all operations and the key schedule in the reverse order.
- Each operation, `SHIFTROWS`, `SUBBYTES`, `MIXCOLUMNS` must be replaced by their inverse operations.
  - `ADDROUNDKEY` is its own reverse.