Advanced Encryption Standard (AES)

Cryptography - CS 411 / CS 507

Erkay Savaş

Department of Computer Science and Engineering
Sabancı University

October 17, 2019

AES Selection Process

- Successor to DES
- The selection process is administered by NIST
  - AES selection was an open process.
  - 1997, NIST called for candidates to replace DES.
  - Requirements were
    - Block cipher with 128-bit block size
    - Support for 128, 192, 256 bits of key sizes
    - Efficient software and hardware implementation.
  - Cryptographic community was asked to comment on five finalists: MARS(IBM), RC6(RSA), Rijndael, Serpent, Twofish.
  - NIST chose Rijndael as AES in 2000.

Rijndael for AES

Joan **Dae**men &

Vincent **Rij**men

- Likely to be the most commonly used algorithm in the next decade.
- See `http://www.nist.gov/aes` for more information

Algorithm	Pentium Pro 200 Mhz Mbit/s	FPGA hardware Gbit/s
MARS	69	-
RC6	105	2,4
Rijndael	71	1,9
Serpent	27	4,9
Twofish	95	1,6

Performance : AES vs DES

- Hardware ASIC (0.12 μm)

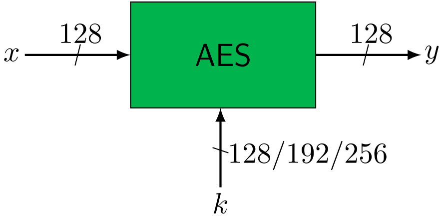
Cipher	Area(# of gates)	Time Performance
TDES	5.5K/16.954K	334 Mbps/1.067 Gbps
AES	5.4K/20.328K/36.9K	311 Mbps/2.8 Gbps/4.459 Gbps

- Hardware FPGA (Virtex-E xcv1000E-8)

Cipher	Area(# of slices)	Time Performance
TDES	668/1122	136 Mbps/290 Mbps
AES	956/2529	109 Mbps/833 Mbps

- Software (AMD Opteron 8354 2.2 GHz processor under Linux)

Cipher	Mode	Time Performance
TDES	CTR	13 MiB/s
AES	CTR($128/192/256$)	139/113/96 MiB/s

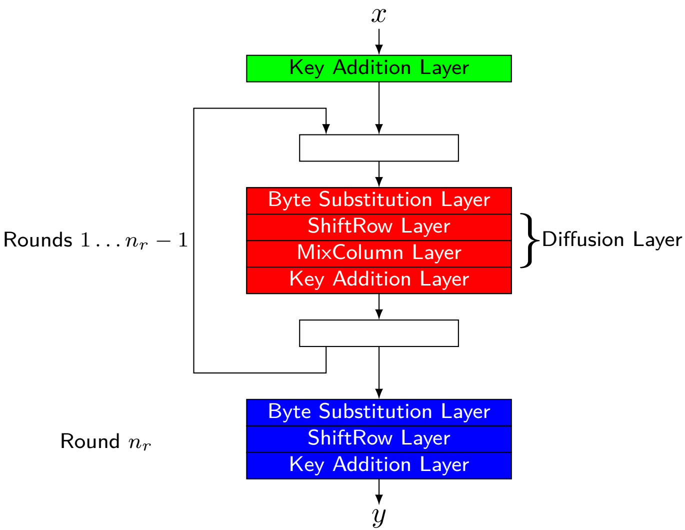


- Rijndael block size is also variable $(128/192/256)$
- Number of rounds (n_r) is a function of the key length:

Key length(in bits)	n_r
128	10
192	12
256	14

Rijndael Overview

- Not a Feistel cipher.
  - Recall: Feistel ciphers do not process the whole block in each iteration.
  - This explains why Rijndael has fewer number of rounds.
- Rijndael has three basic steps (or layers):
  - Key Addition Layer: XORing the block with the round key.
  - Byte Substitution Layer: 8-by-8 substitution (s-box). Nonlinear operation (confusion).
  - Diffusion Layer: provides the diffusion of the bits of a block. Linear operations
    - ShiftRow Layer
    - MixColumn Layer


x
Key Addition Layer
Rounds $1 \ldots n_r - 1$
Byte Substitution Layer
ShiftRow Layer
MixColumn Layer
Key Addition Layer
Diffusion Layer
Round n_r
Byte Substitution Layer
ShiftRow Layer
Key Addition Layer
y

The Layers

- We will assume the block and key lengths are fixed to 128-bit (16 bytes).
  - 16 bytes (128 bit) are arranged into a 4×4 matrix

$$S = \begin{pmatrix} s_0^{i-1} & s_4^{i-1} & s_8^{i-1} & s_{12}^{i-1} \\ s_1^{i-1} & s_5^{i-1} & s_9^{i-1} & s_{13}^{i-1} \\ s_2^{i-1} & s_6^{i-1} & s_{10}^{i-1} & s_{14}^{i-1} \\ s_3^{i-1} & s_7^{i-1} & s_{11}^{i-1} & s_{15}^{i-1} \end{pmatrix}$$

- Each matrix entry can be thought an element of $GF(2^8)$ with $x^8 + x^4 + x^3 + x + 1$.
  - We will occasionally do arithmetic in $GF(2^8)$.

The Byte Substitution Layer 1/2

- Each byte in the matrix is changed to another byte by the following operations:
  1. Each byte in S is an element of $GF(2^8)$, $A(x)$.
  2. Find the multiplicative inverse of $A(x)$, $T(x) = A^{-1}(x)$.
  3. Apply the affine transformation defined by

$$\begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} t_0 \\ t_1 \\ t_2 \\ t_3 \\ t_4 \\ t_5 \\ t_6 \\ t_7 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

The Byte Substitution Layer - SBOX

- In round i, every byte of the state s_j^{i-1} is substitued by a highly nonlinear transformation
- Namely, $b_j = \mathrm{SBOX}(s_j^{i-1})$ for $j = 0, 1, \ldots, 15$

$$\begin{pmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{pmatrix} \xleftarrow{\mathrm{SBOX}} \begin{pmatrix} s_0^{i-1} & s_4^{i-1} & s_8^{i-1} & s_{12}^{i-1} \\ s_1^{i-1} & s_5^{i-1} & s_9^{i-1} & s_{13}^{i-1} \\ s_2^{i-1} & s_6^{i-1} & s_{10}^{i-1} & s_{14}^{i-1} \\ s_3^{i-1} & s_7^{i-1} & s_{11}^{i-1} & s_{15}^{i-1} \end{pmatrix}$$

- The result is another 4×4 matrix whose entries are bytes.
- You can use a table with 256 entries whose entries are bytes in order to implement this layer.

The Shift Row Layer

- Four rows of the matrix are shifted cyclically to the left by offsets of 0, 1, 2, 31, 2, 3.

$$\begin{pmatrix} c_0 & c_4 & c_8 & c_{12} \\ c_1 & c_5 & c_9 & c_{13} \\ c_2 & c_6 & c_{10} & c_{14} \\ c_3 & c_7 & c_{11} & c_{15} \end{pmatrix} = \begin{pmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_5 & b_9 & b_{13} & b_1 \\ b_{10} & b_{14} & b_2 & b_6 \\ b_{15} & b_3 & b_7 & b_{11} \end{pmatrix} \leftarrow \begin{pmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{pmatrix}$$

The Mix Column Layer

$$\begin{pmatrix} d_0 & d_4 & d_8 & d_{12} \\ d_1 & d_5 & d_9 & d_{13} \\ d_2 & d_6 & d_{10} & d_{14} \\ d_3 & d_7 & d_{11} & d_{15} \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} c_0 & c_4 & c_8 & c_{12} \\ c_1 & c_5 & c_9 & c_{13} \\ c_2 & c_6 & c_{10} & c_{14} \\ c_3 & c_7 & c_{11} & c_{15} \end{pmatrix}$$

- $02 = 0000\ 0010$
- $03 = 0000\ 0011$

The Round Key Addition

- A simple XORing operation

$$\begin{pmatrix} s_0^i & s_4^i & s_8^i & s_{12}^i \\ s_1^i & s_5^i & s_9^i & s_{13}^i \\ s_2^i & s_6^i & s_{10}^i & s_{14}^i \\ s_3^i & s_7^i & s_{11}^i & s_{15}^i \end{pmatrix} = \begin{pmatrix} d_0 & d_4 & d_8 & d_{12} \\ d_1 & d_5 & d_9 & d_{13} \\ d_2 & d_6 & d_{10} & d_{14} \\ d_3 & d_7 & d_{11} & d_{15} \end{pmatrix} \oplus \begin{pmatrix} k_0^i & k_4^i & k_8^i & k_{12}^i \\ k_1^i & k_5^i & k_9^i & k_{13}^i \\ k_2^i & k_6^i & k_{10}^i & k_{14}^i \\ k_3^i & k_7^i & k_{11}^i & k_{15}^i \end{pmatrix}$$

- The matrix whose entries are s_j^i is the output of the round i

The Key Schedule

- The original key consists of 128 bits (16 B)
- We need round keys for the 10 (12 or 14) rounds
- The nonlinear SBOX function is used to generate round keys

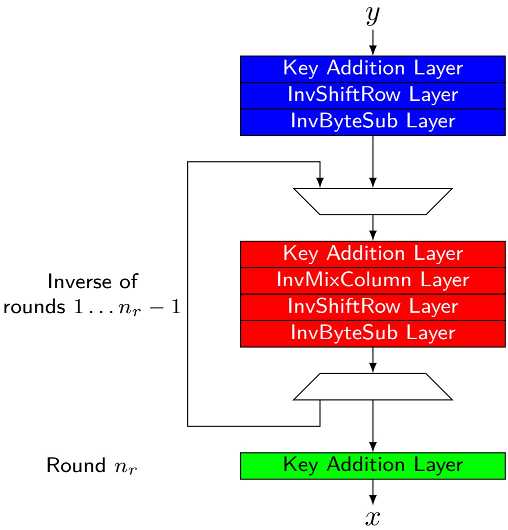
Decryption

- Rijndael is not Feistel cipher;
  - Thus each layer must actually be inverted.
  - Operations in each layer are invertible:
    - InvByteSub
    - InvShiftRow (Shift right instead of left)
    - InvMixColumn
    - The inverse of MixColumn exists because 4×4 matrix used in MixColumn is invertible.

  InvMixColumn matrix

$$\begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \qquad 0E = 0000\ 1110 \rightarrow$$

Rijndael Decryption



Final Remarks

- In every round, each bit in the block are treated uniformly
  - This has the effect of diffusing the input bits faster
  - After two rounds each of the 128 output bits depends on each of the 128 input bits.
- S-box is constructed using a very simple algebraic mapping,
  - $x \to x^{-1}$ in $GF(2^8)$ $\to$highly nonlinear; balanced
  - Its simplicity removes any suspicions about a certain *trapdoor*, which was believed to exist in DES for years.
- Key scheduling utilizes highly nonlinear SubByte mapping.
- No known attacks are better than brute force for seven or more rounds