



# AES Finalist Algorithm: The Rijndael Block Cipher

Mel Tsai

University of California at Berkeley



## *Introduction and History*

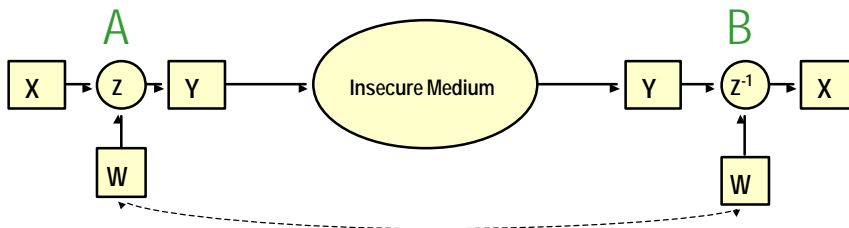


- ◆ **National Institute of Science and Technology**
  - DES is an aging standard that no longer addresses today's needs for strong encryption
  - Triple-DES: Endorsed by NIST as today's defacto standard
- ◆ **AES: The Advanced Encryption Standard**
  - To be finalized in 2001
  - Goal is to define the Federal Information Processing Standard (FIPS) by selecting a new powerful encryption algorithm suitable for encrypting government documents
  - AES candidate algorithms must be:
    - ◆ Symmetric-key ciphers supporting 128, 192, and 256 bit keys
    - ◆ Royalty-Free
    - ◆ Unclassified (i.e. public domain)
    - ◆ Available for worldwide export

## Overview of Secret Key Cryptography



- ◆ To transmit data securely over an insecure medium, two parties agree on a key in which to encrypt data.
  - This key is usually exchanged through public-key cryptographic methods
- ◆ User A encrypts a block of data  $X$  with key  $W$  and sends this data to user B.
- ◆ By using the same key  $W$ , user B decrypts the *ciphertext*  $Y$  back into  $X$



Fundamental Concept:

Due to algorithm  $Z$ , it's nearly impossible to recover data  $X$  from ciphertext  $Y$  without key  $W$ . "Guessing" the key  $W$  through exhaustive search is generally infeasible.

10/18/2000

3

## Introduction and History (cont.)



### ◆ AES Round-3 Finalist Algorithms:

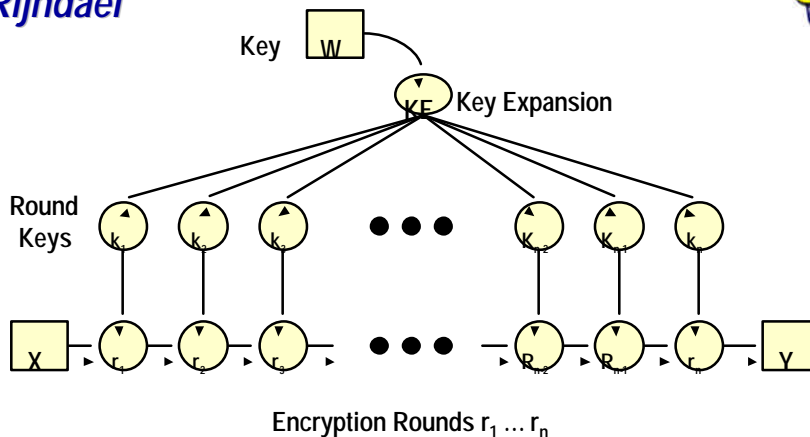
- ◆ MARS
  - ◆ Candidate offering from IBM
- ◆ RC6
  - ◆ Developed by Ron Rivest of RSA Labs, creator of the widely used RC4 algorithm
- ◆ Twofish
  - ◆ From Counterpane Internet Security, Inc.
- ◆ Serpent
  - ◆ Designed by Ross Anderson, Eli Biham and Lars Knudsen
- ◆ Rijndael
  - ◆ Designed by Joan Daemen and Vincent Rijmen

10/18/2000

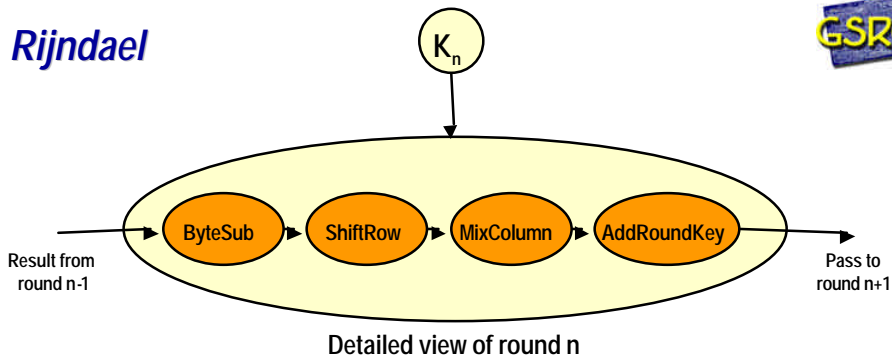
4

## ◆ The Winner: Rijndael

- ◆ Joan Daemen (of Proton World International) and Vincent Rijmen (of Katholieke Universiteit Leuven).
- ◆ (pronounced "Rhine-doll")
- ◆ Allows only 128, 192, and 256-bit key sizes (unlike the other candidates)
- ◆ Variable block length of 128, 192, or 256 bits. All nine combinations of key/block length possible.
  - ◆ A block is the smallest data size the algorithm will encrypt
- ◆ Vast speed improvement over DES in both hardware and software implementations
  - ◆ 8416 bytes/sec on a 20MHz 8051 (@ 12 CPI)
  - ◆ 8.8 Mbytes/sec on a 200MHz Pentium Pro



- ◆ Key is expanded to a set of  $n$  round keys
- ◆ Input block  $X$  undergoes  $n$  rounds of operations (each operation is based on value of the  $n$ th round key), until it reaches a final round.
- ◆ Strength of algorithm relies on the fact that it's very difficult to obtain the intermediate result (or *state*) of round  $n$  from round  $n+1$  without the round key.



◆ Each round performs the following operations:

- ◆ Non-linear Layer: No linear relationship between the input and output of a round
- ◆ Linear Mixing Layer: Guarantees high diffusion over multiple rounds
  - ◆ Very small correlation between bytes of the round input and the bytes of the output
- ◆ Key Addition Layer: Bytes of the input are simply EXOR'ed with the expanded round key

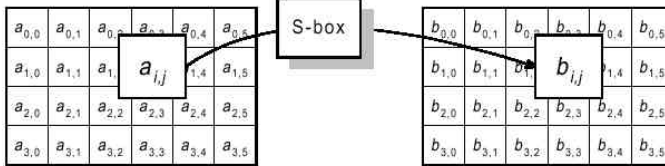
◆ Three layers provide strength against known types of cryptographic attacks: Rijndael provides "full diffusion" after only two rounds

- Linear and differential cryptanalysis
- Known-key and related-key attacks
- Square attack
- Interpolation attacks
- Weak-keys

◆ Rijndael has been shown to be *K-secure*:

- No key-recovery attacks faster than exhaustive search exist
- No known symmetry properties in the round mapping
- No weak keys
- No related-key attacks: No two keys have a high number of expanded round keys in common

## Rijndael: ByteSub



Each byte at the input of a round undergoes a non-linear byte substitution according to the following transform:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

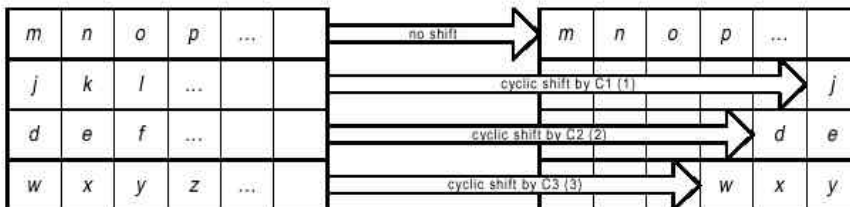
Substitution ("S")-box

## Rijndael: ShiftRow

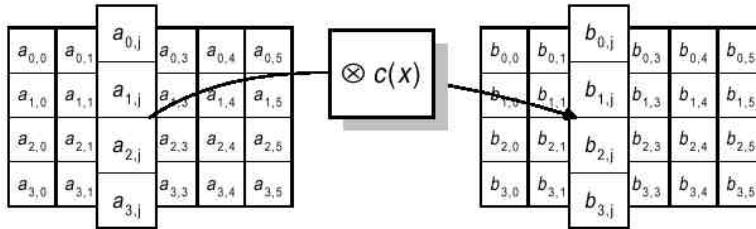


Nb	C1	C2	C3
4	1	2	3
6	1	2	3
8	1	3	4

Depending on the block length, each "row" of the block is cyclically shifted according to the above table



## Rijndael: MixColumn



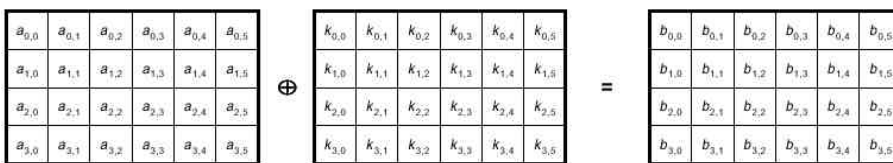
Each column is multiplied by a fixed polynomial

$$C(x) = '03' * X^3 + '01' * X^2 + '01' * X + '02'$$

This corresponds to matrix multiplication  $b(x) = c(x) \cdot \ddot{A}a(x)$ :

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

## Rijndael: Key Expansion and Addition



Each word is simply EXOR'ed with the expanded round key

Key Expansion algorithm:

```

KeyExpansion(int* Key[4*Nk], int* EKey[Nb*(Nr+1)])
{
    for(i = 0; i < Nk; i++)
        EKey[i] = (Key[4*i], Key[4*i+1], Key[4*i+2], Key[4*i+3]);
    for(i = Nk; i < Nb * (Nr + 1); i++)
    {
        temp = EKey[i - 1];
        if (i % Nk == 0)
            temp = SubByte(RotByte(temp)) ^ Rcon[i / Nk];
        EKey[i] = EKey[i - Nk] ^ temp;
    }
}
    
```

## Rijndael: Implementations



- ◆ Rijndael is well suited for software implementations on 8-bit processors (important for “Smart Cards”)
  - Atomic operations focus on bytes and nibbles, not 32 or 64 bit integers
  - Layers such as ByteSub can be efficiently implemented using small tables in ROM (e.g. < 256 bytes).
  - No special instructions are required to speed up operation, e.g. barrel rotates
- ◆ For 32-bit implementations:
  - An entire round can be implemented via a fast table lookup routine on machines with 32-bit or higher word lengths
  - Considerable parallelism exists in the algorithm
    - ◆ Each layer of Rijndael operates in a parallel manner on the bytes of the round state, all four component transforms act on individual parts of the block
    - ◆ Although the Key expansion is complicated and cannot benefit much from parallelism, it only needs to be performed *once* until the two parties switch keys.

## Rijndael: Implementations



- ◆ Hardware Implementations
  - Rijndael performs very well in software, but there still exists cases where more performance is required (e.g. server and VPN applications).
  - Multiple S-Box engines, round-key EXORs, and byte shifts can all be implemented efficiently in hardware when absolute speed is required
  - Small amount of hardware can vastly speed up 8-bit implementations
- ◆ Inverse Cipher
  - Except for the non-linear ByteSub step, each part of Rijndael has a straightforward inverse and the operations simply need to be undone in the reverse order.
  - However, Rijndael was specially written so that the same code that encrypts a block can also decrypt the same block simply by changing certain tables and polynomials for each layer. The rest of the operation remains identical.

## Conclusions and The Future



- ◆ Rijndael is an extremely fast, state-of-the-art, highly secure algorithm
- ◆ Rijndael has efficient implementations in both hardware and software; it requires no special instructions to obtain good performance on any computing platform
  
- ◆ Despite being the chosen by NIST as the AES candidate winner, Rijndael is not yet automatically the new encryption standard
  - Rijndael will soon be formally announced in the Federal Register
  - NIST will then undergo public review and comments on the draft Federal Information Processing Standard for 90 days
  - Triple-DES, still highly secure and supported by NIST, is expected to be common for the foreseeable future.