Block ciphers (AES)

Hyoungshick Kim

Department of Software

College of Software

Sungkyunkwan University

AES contest

- 1997 : Call For AES Candidate Algorithms by NIST (National Institute of Standards and Technology)
  - ✓ 128-bit Block cipher
  - ✓ 128/192/256-bit keys
  - ✓ Worldwide-royalty free
  - ✓ More secure than Triple DES
  - ✓ More efficient than Triple DES
- 1998 : 1st Round Candidates – 15 Algorithms
  - ✓ Mars, Twofish, RC6, SAFER+, HPC, CAST256, DEAL, Frog, Magenta, Rijndael, DFC, Serpent, Crypton, E2, LOKI97
- 1999 : 2nd Round Candidates – 5 Algorithms
  - ✓ MARS, RC6, Rijndael, Serpent, and Twofish
- 2000. 10 : Rijndael selected as the finalist
- 2001. 12: official publication as FIPS PUB 197

(http://www.moserware.com/2009/09/stick-figure-guide-to-advanced.html, http://competitions.cr.yp.to/aes.html)

1st Round candidates

Cipher	Submitted by	Country
CAST-256	Entrust	Canada
Crypton	Future Systems	Korea‡
Deal	Outerbridge	Canada†
DFC	ENS–CNRS	France
E2	NTT	Japan
Frog*	TecApro	Costa Rica
HPC*	Schroeppel	USA
LOKI97*	Brown, Pieprzyk, Seberry	Australia
Magenta	Deutsche Telekom	Germany
Mars	IBM	USA†
RC6	RSA	USA†
Rijndael*	Daemen, Rijmen	Belgium‡
Safer+*	Cylink	USA†
Serpent*	Anderson, Biham, Knudsen	UK, Israel, Norway
Twofish*	Counterpane	USA†

* Placed in the public domain; † and foreign designers; ‡ foreign influence

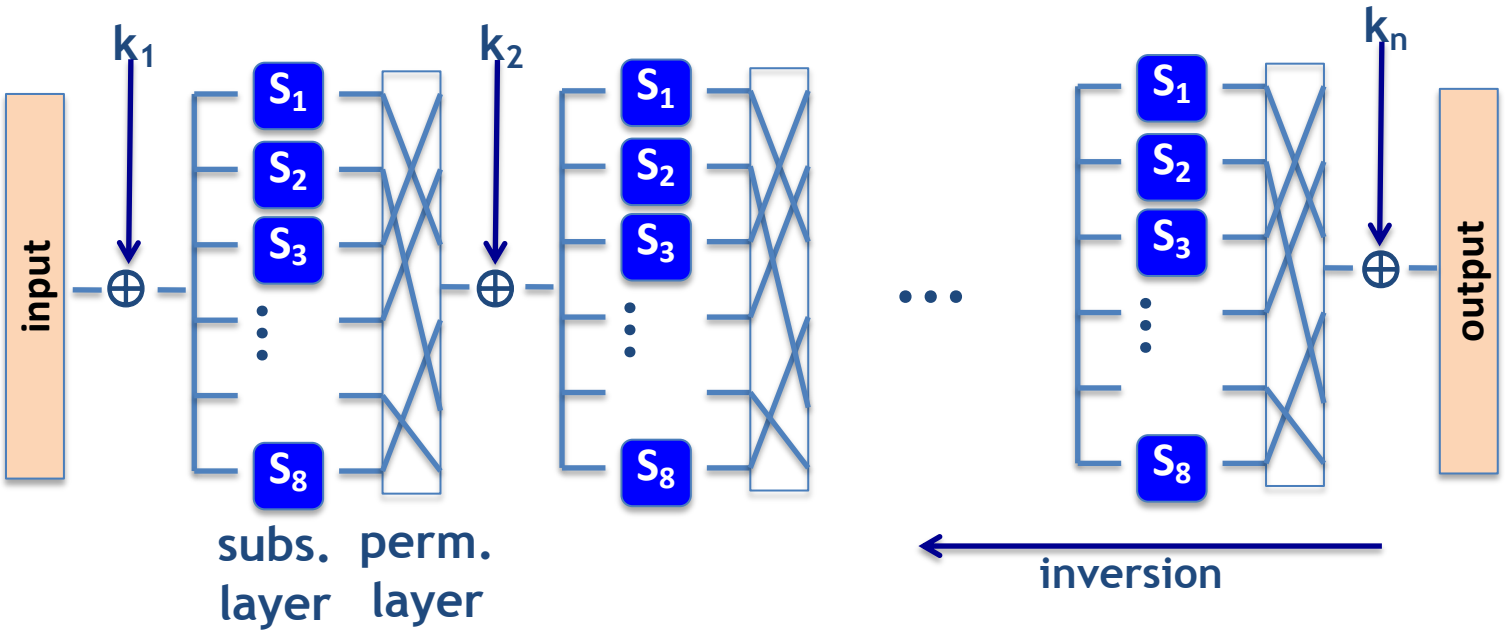
2nd Round candidates

Cipher	Submitter	Structure	Votes
MARS	IBM	Feistel	13 positive, 84 negative
RC6	RSA Lab.	Feistel	23 positive, 37 negative
Rijndael	Daemen, Rijmen	SPN	86 positive, 10 negative
Serpent	Anderson, Biham, Knudsen	SPN	59 positive, 7 negative
Twofish	Schneier et. al	Feistel	31 positive, 21 negative

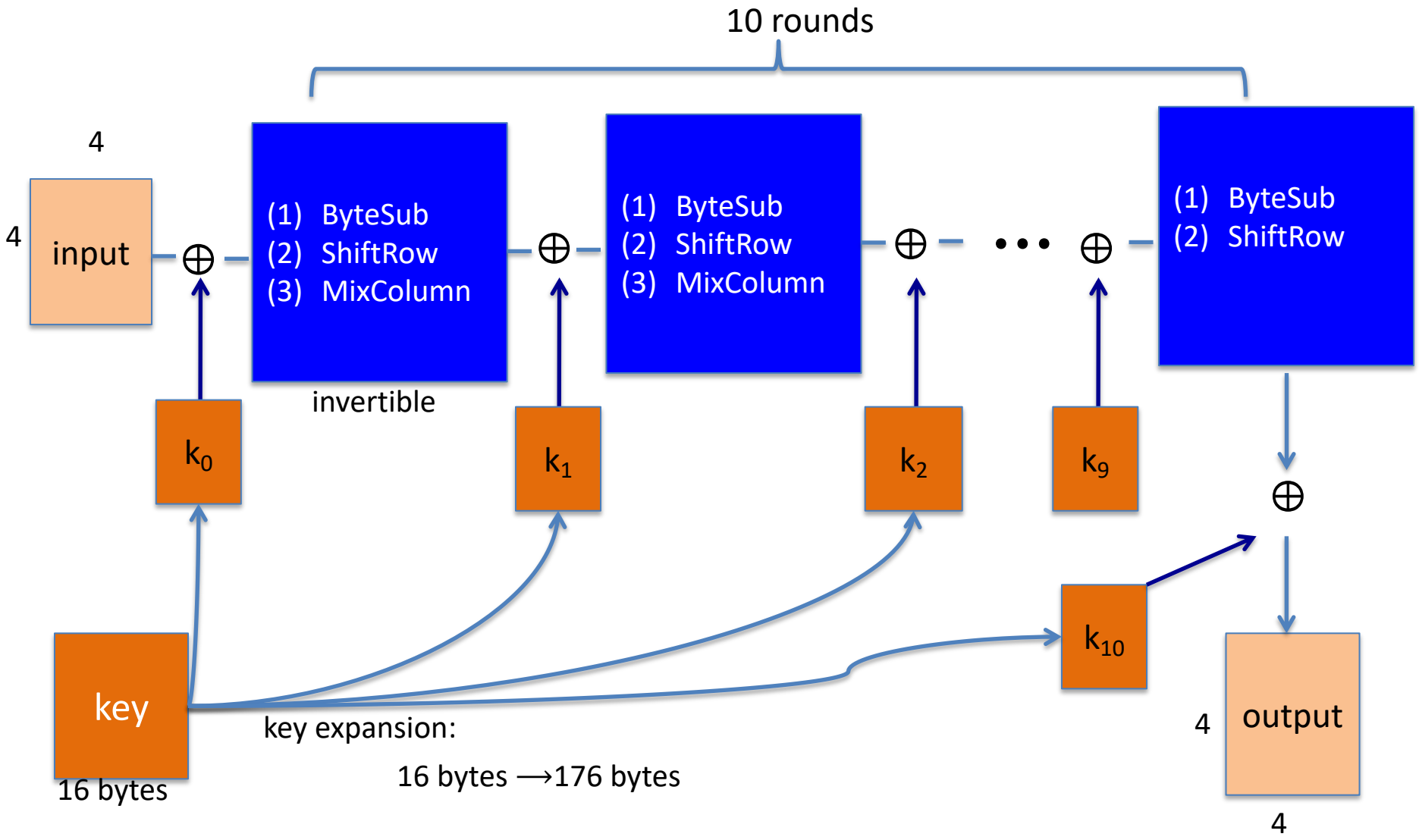
2000. 10 : Rijndael selected as the finalist

Advanced Encryption Standard (AES)

- AES is a standard symmetric encryption algorithm for US federal organizations
- AES has a 128-bit block, arranged as 16 (4-by-4) bytes
- AES is a Substitution-Permutation network (not a Feistel)



AES-128 schematic

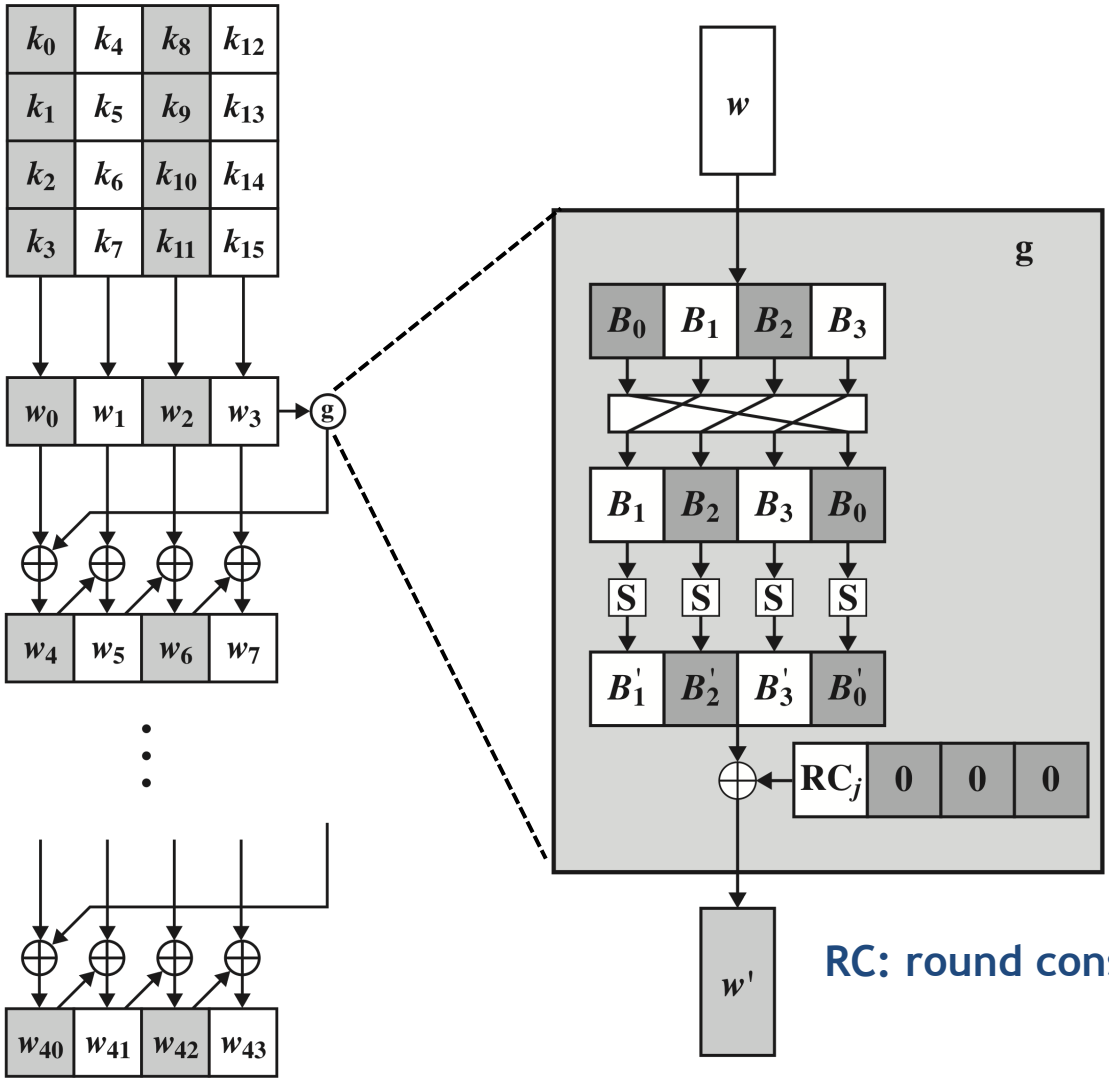


Key generation

before round 1

after round 1

after round 10



RC: round constant

Four operations

1. *Byte Substitution* **confusion**
   - predefined substitution table s[i,j] → s'[i,j]
2. *Shift Row* **diffusion**
   - left circular shift
3. *Mix Columns* **diffusion and confusion**
   - 4 elements in each column are multiplied by a polynomial
4. *Add Round Key* **confusion**
   - Key is derived and added to each column

This step is omitted for the last round

Source code for round functions

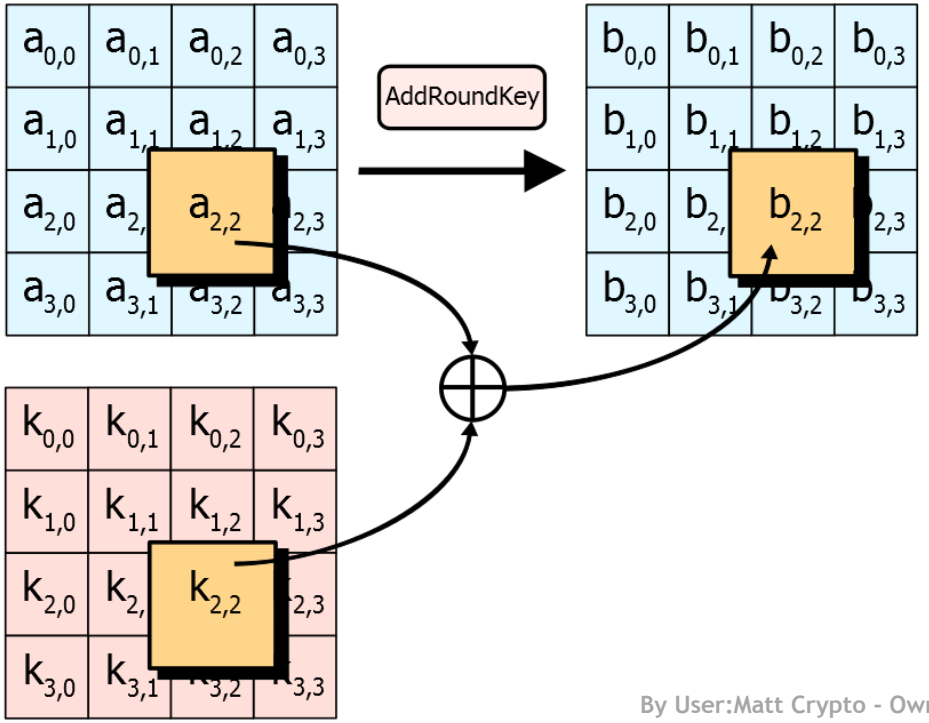
```
// first round
AddRoundKey(0);

// all rounds except for the first and last rounds
for(round=1;round<Nr;round++)
{
    ByteSub();
    ShiftRows();
    MixColumns();
    AddRoundKey(round);
}

// last round
ByteSub();
ShiftRows();
AddRoundKey(Nr);
```

Add round key

- Bitwise XOR state *s* with key k_0



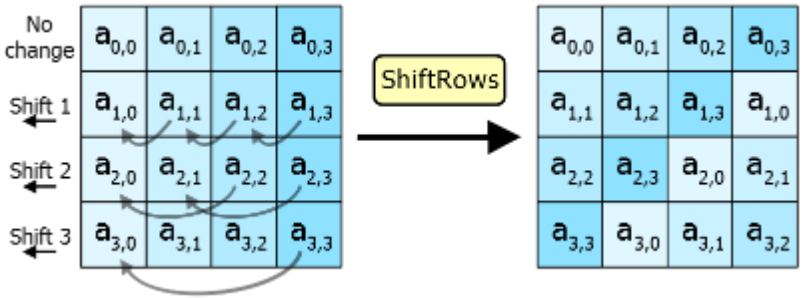
Three functions for each round

- **ByteSub**: a 1 byte S-box. 256 byte table
  (easily computable)

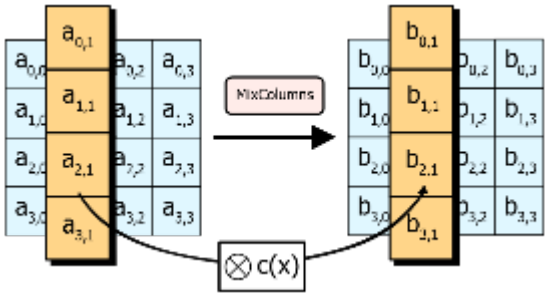
  The only nonlinear elements:

  $\text{ByteSub}(A_i) + \text{ByteSub}(A_j) \neq \text{ByteSub}(A_i + A_j)$, for i, j = 0, … ,15

- **ShiftRows**:



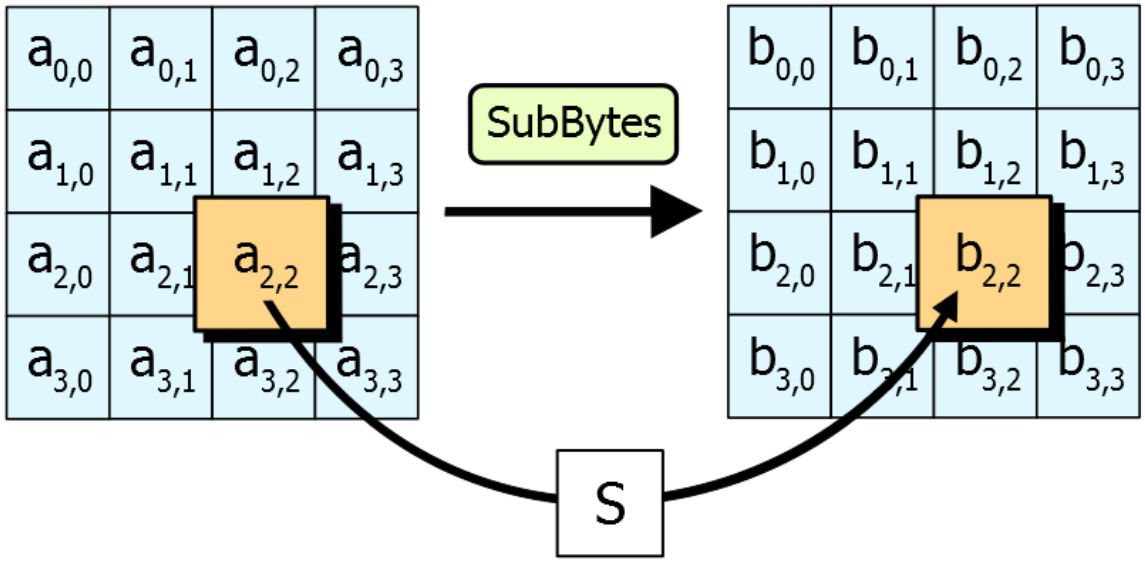
- **MixColumns**:



Multiply by constant matrix c(x) = $\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$

ByteSub

- For each round… (10 rounds total)
  - Substitute bytes
    - Use lookup table to switch positions



ByteSub

- A simple substitution of each byte.
- Uses *one table* of 16x16 bytes containing a permutation of all *256 8-bit values*.
- *Each byte of state* is replaced by *byte in row* (*left 4-bits*) and column (*right 4-bits*).
  - ✓ e.g., *byte {95}* is replaced by *row 9 col 5 byte, w*hich is the value *{2A}*.
- *S-box* is constructed using a defined transformation of the values in $GF(2^8)$.
- Designed to be resistant to all known attacks.

Source code for ByteSub()

```
// The ByteSub Function Substitutes the values in the
// state matrix with values in an S-box.
void ByteSub()
{
        int i,j;
        for(i=0;i<4;i++)
        {
                for(j=0;j<4;j++)
                {
                        state[i][j] = getSBoxValue(state[i][j]);

                }
        }
}
```

S-Box

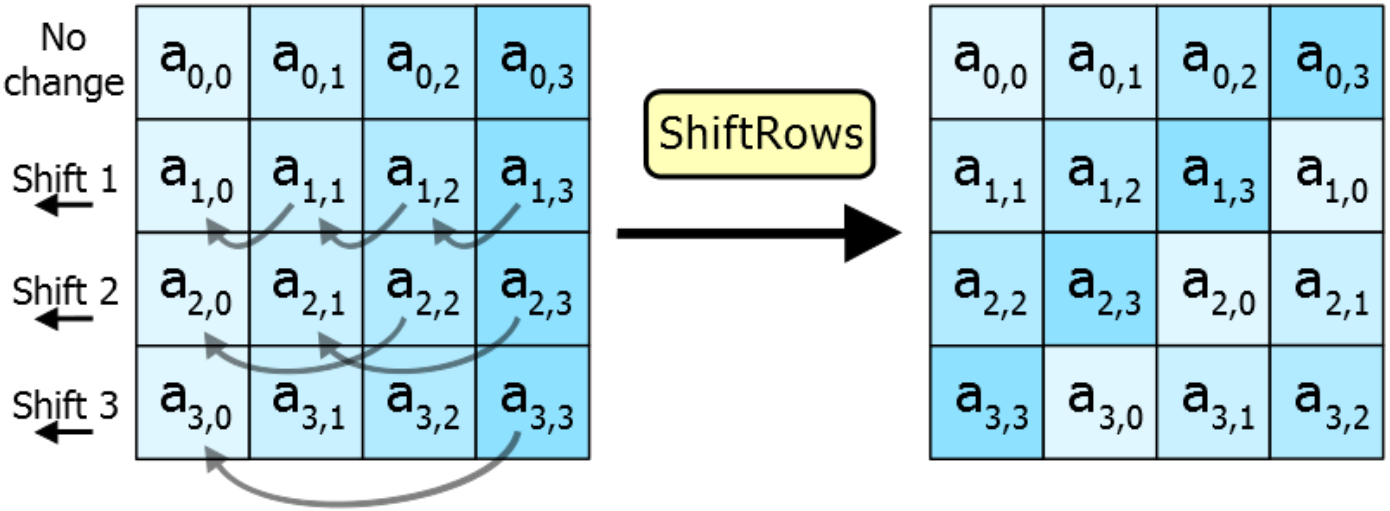
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	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	56	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

HEX 19 would get replaced with HEX D4

AES is a byte-oriented cipher

ShiftRows

- For each round…
  - Shift rows



Example of ShiftRows

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

The ith row is shifted (i-1) times.

Source code for ShiftRows()

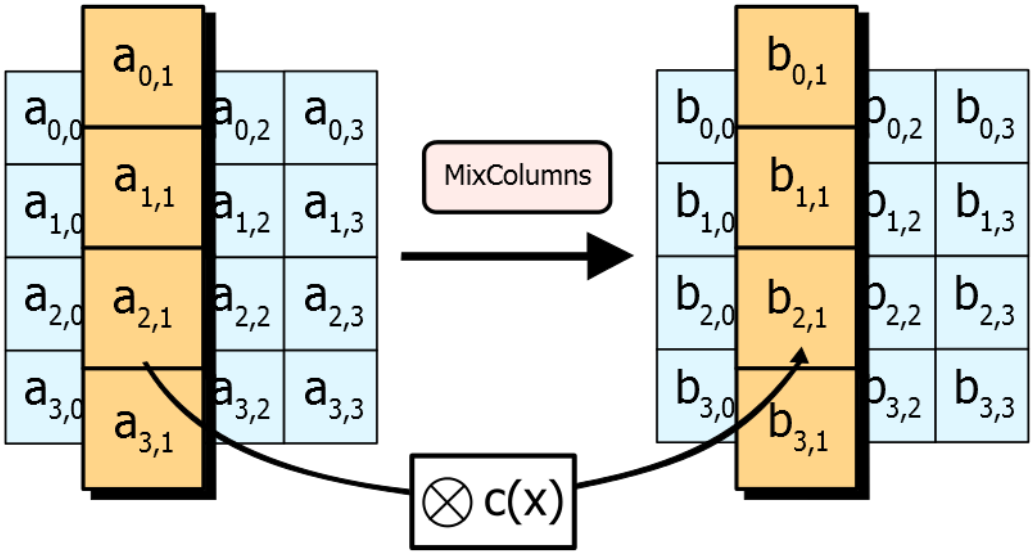
```
// The ShiftRows() function shifts the rows in the state.
// Each row is shifted with different offset.
// Offset = Row number. So the first row is not shifted.
void ShiftRows()
{
        unsigned char temp;

        // Rotate second row 1 columns to left
        temp=state[1][0];
        state[1][0]=state[1][1];
        state[1][1]=state[1][2];
        state[1][2]=state[1][3];
        state[1][3]=temp;

        …
        // Skip the codes for the third and fourth rows
}
```

MixColumns

- For each round…
  - Mix columns
    - Multiply by constant matrix c(x) = $\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$



AES decryption

- To decrypt, process must be invertible
- Inverse of AddRoundKey is easy since "$\oplus$" is its own inverse
- MixColumn is invertible (inverse is also implemented as a lookup table)
- Inverse of ShiftRow is easy (cyclic shift the other direction)
- ByteSub is invertible (inverse is also implemented as a lookup table)

Performance

Cipher	Type	Key size	Speed (MB/sec)
MD5	Hash	--	255
DES	Block cipher	64	32
3DES	Block cipher	168	13
AES-128	Block cipher	128	109

AMD CPU (2.2 GHz), Linux, Crypto++ 5.6.0
(http://www.cryptopp.com/benchmarks.html)

AES in hardware

AES instructions in Intel Westmere:

- **aesenc, aesenclast**: do one round of AES

  128-bit registers: xmm1=state, xmm2=round key

  aesenc xmm1, xmm2 ; puts result in xmm1

- **aeskeygenassist**: performs AES key expansion

- Claim 14 x speed-up over OpenSSL on same hardware

- https://software.intel.com/en-us/articles/download-the-intel-aesni-sample-library

Similar instructions on AMD Bulldozer

Known attacks on the AES

- Best key recovery attack:

  four times better than exhaustive Search

  [Bogdanov, Khovratovich and Rechberger, 2011]

- Related key attack on AES-256:

  Given 2^{99} input/output pairs, we can recover keys in time $\approx 2^{99}$

  [Biryukov and Khovratovich, 2009]

Summary of symmetric key ciphers

- Stream cipher — like a one-time pad
  - Key is stretched into a long **key stream** (using a pseudo random generator)
  - Key stream is used just like a one-time pad
  - Employs “substitution” only
  - Example: RC4, A5/1
- Block cipher
  - Employs both “substitution” and “transposition”
  - Examples: DES, 3DES, AES

Questions?


Sungkyunkwan
Campus
Social
Graduate
Professors
English
Confucian
campus
Zone
Sciences
University
Global
Education
Seoul
Natural
Suwon
Korean
School
university
established
edit
also
Humanities
Science
schools
Administration
ASLE-Korea
Academic
Advanced
constructed
Building
Engineering
located
Joseon
Anniversary
Institute
now
two
Hall
Course
lecturers