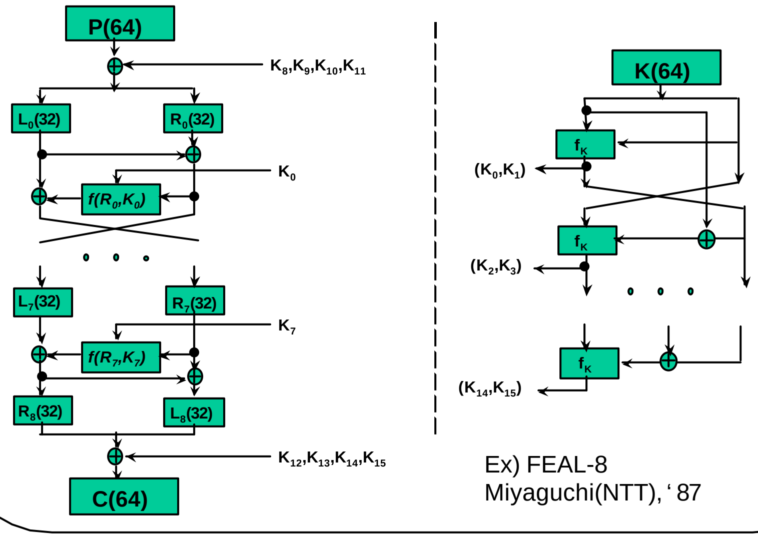
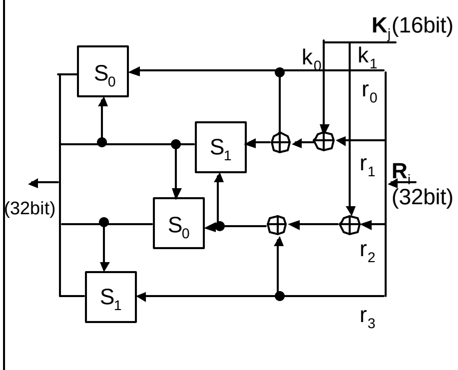
FEAL(Fast data Encipherment ALg.)



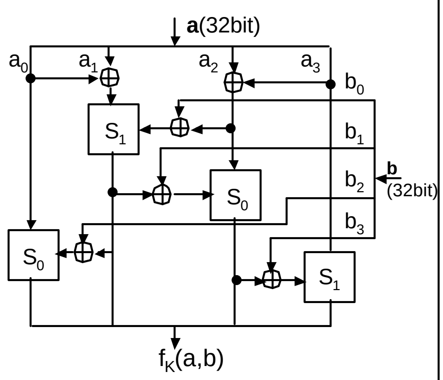
Ex) FEAL-8
Miyaguchi(NTT), ' 87

FEAL(II)

f (R_i,K_j)-function



f $_K$ function



$S_d(X_1,X_2)$=Rot$_2$((X_1+X_2+d) mod256) where, d=0 or 1, X_1 and X_2 : 8bit,
Rot$_2$(Y) : 2-bit left rotation of 8-bit Y

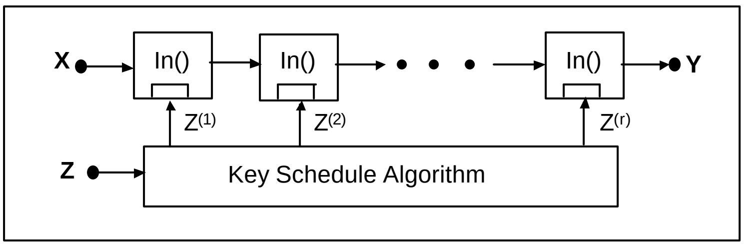
$K_{2(r-1)}$: Left half of f_K(a,b), $K_{2(r-1)+1}$: right half of f_K(a,b)

FEAL(III)

FEAL-n Attack Method	Data Complexity Known	Data Complexity Chosen	Storage Complexity	Processing Complexity
FEAL-4 (LC)	5	-	30Kbytes	6min.
FEAL-6 (LC)	100	-	100Kbytes	40min.
FEAL-8 (LC)	2^{24}			10min.
FEAL-8 (DC)		2^{7} pairs	280Kbytes	2min.
FEAL-16 (DC)		2^{29} pairs		2^{30} op.
FEAL-24 (DC)		2^{45} pairs		2^{46} op.
FEAL-32 (DC)		2^{66} pairs		2^{67} op.

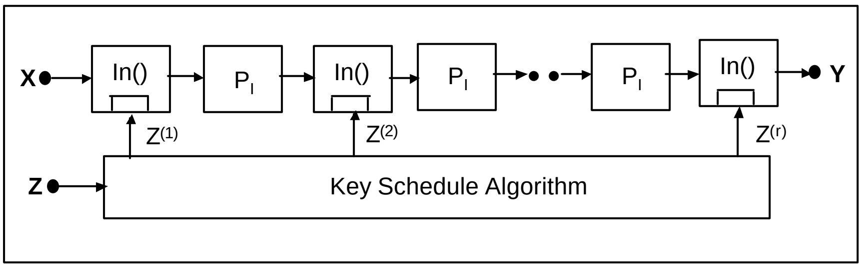
Lai's Classification of "E/D-Similar" Iterated Ciphers

(I) Involution Ciphers Only



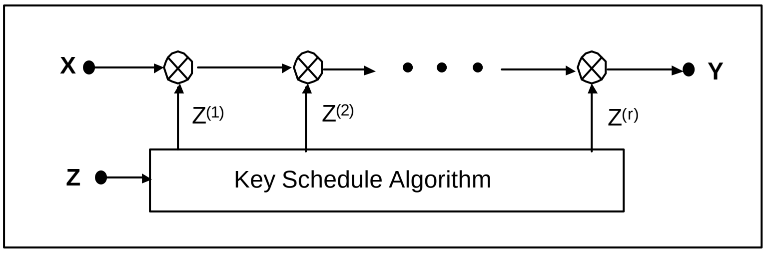
To decrypt : Reverse the Key Schedule

(II) Involution Ciphers and Involutary Permutations



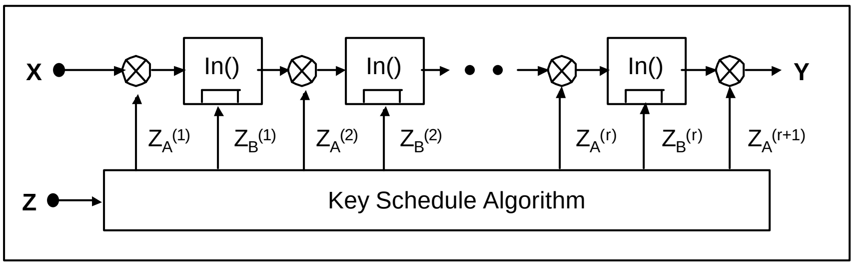
To decrypt : Reverse the Key Schedule
Ex) DES, FEAL, LOKI etc

* Group Ciphers Only



To decrypt : Reverse the Key Schedule and replace each subkey by its group inverse.

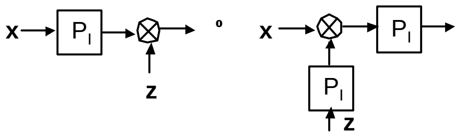
(III) Group Ciphers and Involution Ciphers



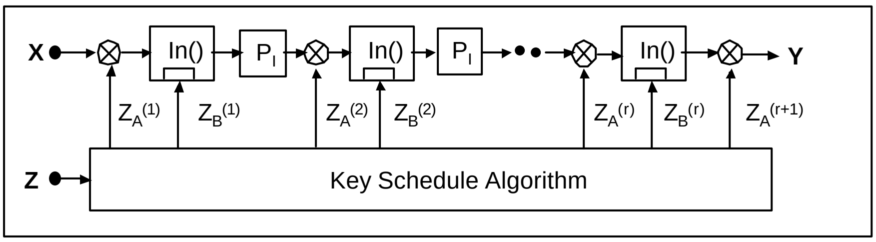
To decrypt : <u>Reverse</u> the Key Schedule and replace each A-key by its group inverse. Ex) PES

(*) Never should be used …equivalent to one encipherment with $Z^{(1)} \otimes Z^{(2)} \otimes .. \otimes Z^{(r)}$

(IV) Group Ciphers, Involution Ciphers and Involutary Permutations such that $P_I(a \otimes b) = P_I(a) \otimes P_I(b)$



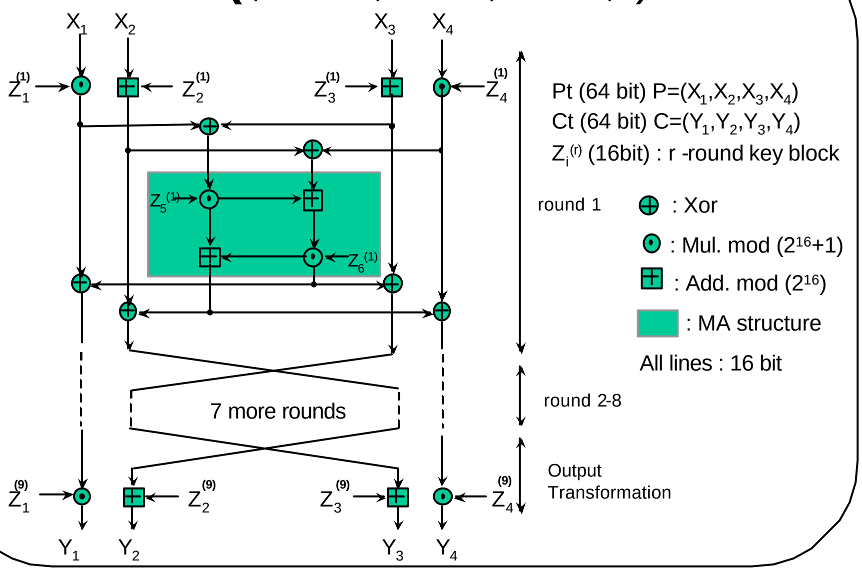
Because
$P_I(x \otimes P_I(z)) = P_I(x) \otimes P_I(P_I(z))$
$= P_I(x) \otimes z$.



There is no P_I in the last round.

To decrypt : <u>Reverse</u> the Key Schedule and replace $Z_A^{(1)}$ and $Z_A^{(r+1)}$ by its group inverse and replace all other A-subkeys by the group inverses of their involutary permuted values [$P_I(Z^{-1}) = (P_I(Z))^{-1}$]. Ex) IDEA

IDEA(Int' l Data Enc. Al.)



IDEA(II)

Divide 128-bit key into 52 16-bit word
(1) Divide 128-bit into 8 blocks and use 6 blocks for round key
(2) 25-bit left rotate of 128-bit
(3) Repeat steps (1) and (2) till 8-round 8

Round Key block for Encryption / Decryption

Round	Encryption Key Block	Decryption Key Block
1	$Z_1^{(1)}\ Z_2^{(1)}Z_3^{(1)}Z_4^{(1)}Z_5^{(1)}Z_6^{(1)}$	$Z_1^{(9)-1}\ -Z_2^{(9)}\ -Z_3^{(9)}Z_4^{(9)-1}Z_5^{(8)}Z_6^{(8)}$
2	$Z_1^{(2)}\ Z_2^{(2)}Z_3^{(2)}Z_4^{(2)}Z_5^{(2)}Z_6^{(2)}$	$Z_1^{(8)-1}\ -Z_2^{(8)}\ -Z_3^{(8)}Z_4^{(8)-1}Z_5^{(7)}Z_6^{(7)}$
3	$Z_1^{(3)}\ Z_2^{(3)}Z_3^{(3)}Z_4^{(3)}Z_5^{(3)}Z_6^{(3)}$	$Z_1^{(7)-1}\ -Z_2^{(7)}\ -Z_3^{(7)}Z_4^{(7)-1}Z_5^{(6)}Z_6^{(6)}$
4	$Z_1^{(4)}\ Z_2^{(4)}Z_3^{(4)}Z_4^{(4)}Z_5^{(4)}Z_6^{(4)}$	$Z_1^{(6)-1}\ -Z_2^{(6)}\ -Z_3^{(6)}Z_4^{(6)-1}Z_5^{(5)}Z_6^{(5)}$
5	$Z_1^{(5)}\ Z_2^{(5)}Z_3^{(5)}Z_4^{(5)}Z_5^{(5)}Z_6^{(5)}$	$Z_1^{(5)-1}\ -Z_2^{(5)}\ -Z_3^{(5)}Z_4^{(5)-1}Z_5^{(4)}Z_6^{(4)}$
6	$Z_1^{(6)}\ Z_2^{(6)}Z_3^{(6)}Z_4^{(6)}Z_5^{(6)}Z_6^{(6)}$	$Z_1^{(4)-1}\ -Z_2^{(4)}\ -Z_3^{(4)}Z_4^{(4)-1}Z_5^{(3)}Z_6^{(3)}$
7	$Z_1^{(7)}\ Z_2^{(7)}Z_3^{(7)}Z_4^{(7)}Z_5^{(7)}Z_6^{(7)}$	$Z_1^{(3)-1}\ -Z_2^{(3)}\ -Z_3^{(3)}Z_4^{(3)-1}Z_5^{(2)}Z_6^{(2)}$
8	$Z_1^{(8)}\ Z_2^{(8)}Z_3^{(8)}Z_4^{(8)}Z_5^{(8)}Z_6^{(8)}$	$Z_1^{(2)-1}\ -Z_2^{(2)}\ -Z_3^{(2)}Z_4^{(2)-1}Z_5^{(1)}Z_6^{(1)}$
Output	$Z_1^{(9)}\ Z_2^{(9)}Z_3^{(9)}Z_4^{(9)}$	$Z_1^{(1)-1}\ -Z_2^{(1)}\ -Z_3^{(1)}Z_4^{(1)-1}$

$Z_i^{(j)-1}$: multiplicative inverse of $Z_i^{(j)}$ mod $(2^{16}+1)$, $-Z_i^{(j)}$: additive inverse of $Z_i^{(j)}$ mod 2^{16}

LOKI*

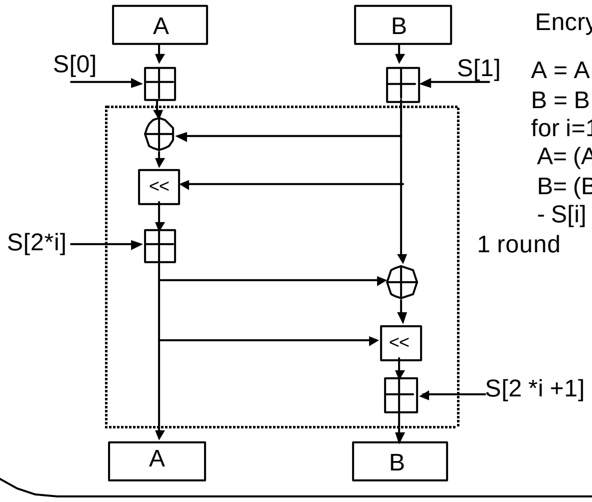
- Brown, Pieprzyk, Seberry(Australia)
- Modify f-function and Key Schedule of DES
  - S -box : 12 bit -> 8 bit
  - Rotation : 12-bit or 13-bit bit rotation
  - Others are same as DES
- LOKI89, LOKI91

* Name of strong and capricious god appeared in Greek myth

RC-5(I)

- Ron Rivest : Ron's Code
- Ease for H/W & S/W Implementation
- RC5 - w/r/b
  - w : word length (16, 32)
  - r : number of round (0-255)
  - b : key byte (0 -255)
  - Ex) RC5-32/12/16 : 100Kb, 586PC
- To prevent weak key, user magic constant (e, p) in key scheduling

RC-5(II)



Encryption

A = A + S[0]
B = B + S[1]
for i=1 to r do
 A= (A ⊕ B)<<B) + S[2*i]
 B= (B ⊕ A)<<A) + S[2*i +1]
 - S[i] : Round Key

⊕ : XOR

<< : Rotation

⊞ : addition mod 2^w

GOST

- **Russia: GOvernment STandard 28147-89**
- **internal specification(unpublished)**
- **64-bit block cipher, 32 round**
- **256-bit main key & 512-bit subkey**
- **No Key Scheduling**
- **Modify f-function of DES : 4x4 k-box**

MISTY

- Matsui, Ichigawa,Sorimachi,Tokita, Yamagishi in Mitsubishi co.
- Pt/Ct:64-bit, Key:128-bit, 8 Round
- Provable security against DC and LC
- By using recursive structure 30Mb/s (HP9735, PA7150-125MHz)
- To be KASUMI for 3GPP group

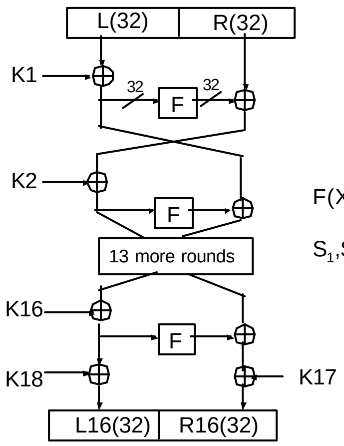
Skipjack

- Classified 64-bit block for replacing -> Internal structure published '98
- 80-bit key, 32 round
- Used for Clipper Project with Key Escrow
- ECB, CBC, 64 bitOFB,{1/8/16/ 32} bit CFB

Blowfish

- Bruce Schneier, et al
- XOR, Addition for 32 bit CPU
- 64bit block, 448 bit key, 8 x 32 S-box
- 16 round
- Takes long time for key scheduling

Blowfish(II)



$F(X_L)=(((S_1 + S_2 \bmod 2^{32}) \oplus S_3) + S_4) \bmod 2^{32}$

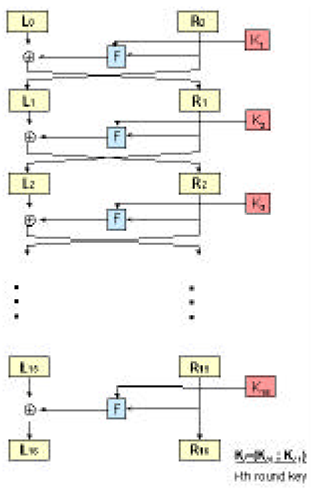
S_1, S_2, S_3, S_4 : 4 8 x 32 S-boxes

SEED(I)

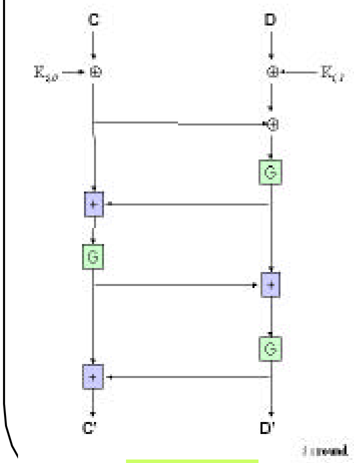
- (Korean Data Encryption Standard)
  - 128-bit symmetric block cipher
  - developed by KISA, ETRI, ADD + a
- Reference
  - KRyptoGate (Korean cRYPTOgraphers' GATEway) http://www.cryptogate.com
  - http://dosan.skku.ac.kr/~sjkim/Kr-Standard.html

SEED(II)

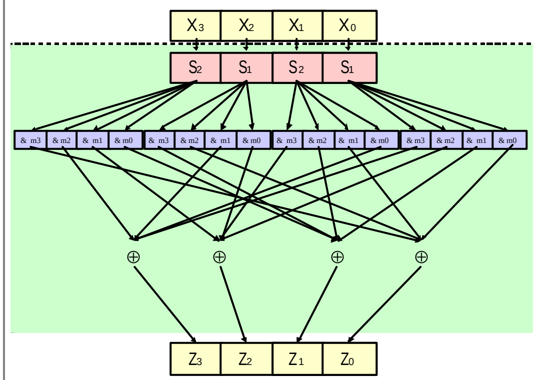
- Introduction
  - Symmetric Block Cipher
  - Block size, key size : 128-bit
  - Feistel structure
  - 16 rounds
- Design criteria
  - Provide security proof
  - Secure against DC, LC, High order DC, Related key attacl
  - Faster than 3-DES in softwa
  - Use nonlinear function for round-key generation



SEED(III)



F-function



G-function

SEED(IV)

F-function

$$C' = G[G[G\{(C \oplus K_{i,0}) \oplus (D \oplus K_{i,1})\} \boxplus (C \oplus K_{i,0})] \boxplus G\{(C \oplus K_{i,0}) \oplus (D \oplus K_{i,1})\}]$$
$$\boxplus G[G\{(C \oplus K_{i,0}) \oplus (D \oplus K_{i,1})\} \boxplus (C \oplus K_{i,0})]$$
$$D' = G[G[G\{(C \oplus K_{i,0}) \oplus (D \oplus K_{i,1})\} \boxplus (C \oplus K_{i,0})] \boxplus G\{(C \oplus K_{i,0}) \oplus (D \oplus K_{i,1})\}]$$

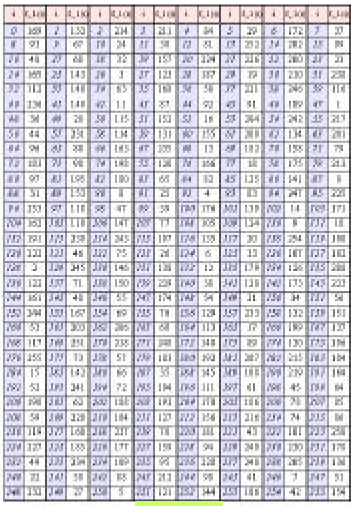
G-function

$Y_3 = S_2(X_3)$, $Y_2 = S_1(X_2)$, $Y_1 = S_2(X_1)$, $Y_0 = S_1(X_0)$,

$$Z_3 = (Y_0 \& m_3) \oplus (Y_1 \& m_0) \oplus (Y_2 \& m_1) \oplus (Y_3 \& m_2)$$
$$Z_2 = (Y_0 \& m_2) \oplus (Y_1 \& m_3) \oplus (Y_2 \& m_0) \oplus (Y_3 \& m_1)$$
$$Z_1 = (Y_0 \& m_1) \oplus (Y_1 \& m_2) \oplus (Y_2 \& m_3) \oplus (Y_3 \& m_0)$$
$$Z_0 = (Y_0 \& m_0) \oplus (Y_1 \& m_1) \oplus (Y_2 \& m_2) \oplus (Y_3 \& m_3)$$

($m_0 = 0xfc$, $m_1 = 0xf3$, $m_2 = 0xcf$, $m_3 = 0x3f$)

SEED(V)



S_1 box

S_2 box

SEED(VI)

Round-key generation

Input 128-bit key *(A,B,C,D)*

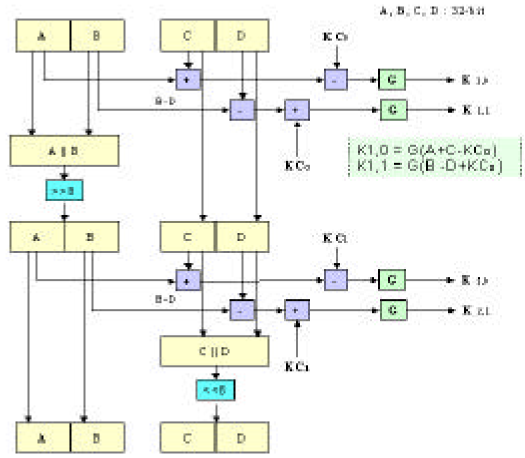
```
for( i=1; i<=16; i++) {
    K_{i,0} ← G(A+C-KC_i);
    K_{i,1} ← G(B-D+KC_i);
    if( i%2==1 ) A||B ← (A||B)>>8;
    else C||D ← (C||D)<<8;
}
```

Output round keys $K_{i,j}$

Round Constants	
KC_0 = 0x9e3779b9	KC_8 = 0x3779b99e
KC_1 = 0x3c6ef373	KC_9 = 0x6ef3733c
KC_2 = 0x78dde6e6	KC_{10} = 0xdde6e678
KC_3 = 0xf1bbcdcc	KC_{11} = 0xbbcdccf1
KC_4 = 0xe3779b99	KC_{12} = 0x779b99e3
KC_5 = 0xc6ef3733	KC_{13} = 0xef3733c6
KC_6 = 0x8dde6e67	KC_{14} = 0xde6e678d
KC_7 = 0x1bbcdccf	KC_{15} = 0xbcdccf1b

SEED(VII)

Round-key generation



Others

- Multi2
- SAFER(Secure And Fast Encryption Routine)-64,128
- Lion& Bear
- TEA (Tiny Encryption Algorithm)
- CAST(Carlisle Adams & Stafford Tavares)

Summary of block ciphers

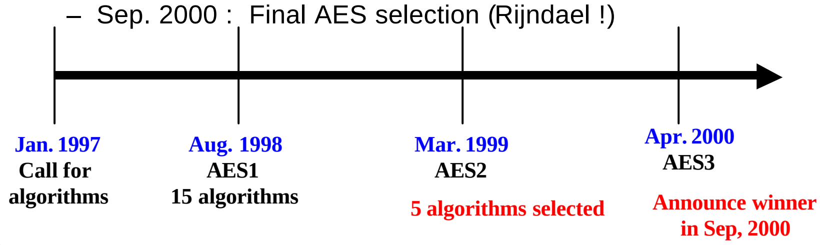
Algorithm	Year	Country	Pt/Ct	Key	Round
DES	1977	USA	64	56	16
FEAL	1987	Japan	64	64	4,8,16,32
GOST	1989	Russia	64	256	32
IDEA	1990	Swiss	64	128	8
LOKI	1991	Australia	64	64	16
SKIPJACK	1990	USA	64	80	32
MISTY	1996	Japan	64	128	>8
SEED	1998	Korea	128	128	16

AES requirements

- **Block cipher**
  - 128-bit blocks
  - 128/192/256-bit keys
- **Worldwide-royalty free**
- **More secure than Triple DES**
- **More efficient than Triple DES**

AES Calendar

- Jan. 2, 1997 : Announcement of intent to develop AES and request for comments
- Sep. 12, 1997 : Formal call for candidate algorithms
- Aug. 20-22, 1998 : First AES Candidate Conference and beginning of Round 1 evaluation (15 algorithms), Rome, Italy
- Mar. 22-23, 1999 : Second AES Candidate Conference, NY, USA
- Sep. 2000 : Final AES selection (Rijndael !)



AES1 algorithms

- 15 algorithms are proposed at AES1 conference

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

AES Round 2 Algorithms

- After AES2 conference, NIST selected the following 5 algorithms as the round 2 candidate algorithm.

Algorithm Name	Submitter Name(s)
MARS	IBM *(represented by Nevenko Zunic)*
RC6™	RSA Laboratories *(represented by Burt Kaliski)*
Rijndael	Joan Daemen, Vincent Rijmen
Serpent	Ross Anderson, Eli Biham, Lars Knudsen
Twofish	Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, Niels Ferguson

Rijndael(I)

- Proposed by Joan Daemen, Vincent Rijmen(Belgium)
- Design choices
  - Square type
  - Three distinct invertible uniform transformations(Layers)
    - Linear mixing layer : guarantee high diffusion
    - Non-linear layer : parallel application of S-boxes
    - Key addition layer : XOR the round key to the intermediate state
  - Initial key addition, final key addition
- Representation of state and key
  - Rectangular array of bytes with 4 rows (square type)
  - Nb : number of column of the state
  - Nk : number of column of the cipher key

Rijndael(II)

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$

State (Nb=6)

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$

Key (Nk=4)

Nr	Nb = 4	Nb = 6	Nb = 8
Nk = 4	10	12	14
Nk = 6	12	12	14
Nk = 8	14	14	14

Number of rounds (Nr)

Rijndael(III)

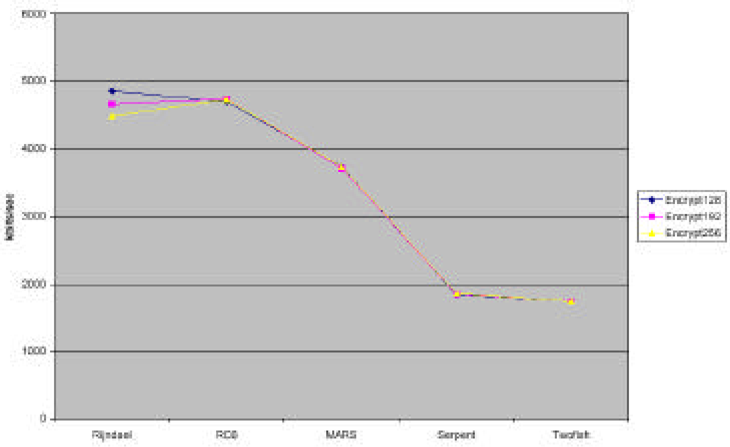
```
Rijndael(State,CipherKey)
{
  KeyExpansion(CipherKey,ExpandedKey) ;
  AddRoundKey(State,ExpandedKey);
  For( i=1 ; i<Nr ; i++ ) Round(State,ExpandedKey + Nb*i) ;
  FinalRound(State,ExpandedKey + Nb*Nr);
}
```

```
Round(State,RoundKey)
{
  ByteSub(State);
  ShiftRow(State);
  MixColumn(State);
  AddRoundKey(State,RoundKey);
}
```

```
FinalRound(State,RoundKey)
{
  ByteSub(State) ;
  ShiftRow(State) ;
  AddRoundKey(State,RoundKey);
}
```

Comparison of AES2 algorithms(I)

- Encryption speed analysis by NIST



Comparison of AES2 algorithms(II)

- Java Implementation by A. Sterbenz(Graz Univ.)

Encryption Speed (kbit/s)	DES (56 bit)	Triple DES (168 bit)	IDEA	MARS	RC6	Rijndael	Serpent	Twofish
128 bit key	10508	4178	12820	19718	26212	19321	11464	19265
192 bit key	n/a	n/a	n/a	19760	26192	10922	11474	19296
256 bit key	n/a	n/a	n/a	19737	26209	14957	11471	19275

Decryption Speed (kbit/s)	DES (56 bit)	Triple DES (168 bit)	IDEA	MARS	RC6	Rijndael	Serpent	Twofish
128 bit key	10519	4173	13018	19443	24338	18868	11519	18841
192 bit key	n/a	n/a	n/a	19670	24382	16484	11514	18841
256 bit key	n/a	n/a	n/a	19489	24279	14468	11533	18806

Encryption Key Setup (keys/s)	DES (56 bit)	Triple DES (168 bit)	IDEA	MARS	RC6	Rijndael	Serpent	Twofish
128 bit key	18128	5150	90571	28680	45603	96234	34729	13469
192 bit key	n/a	n/a	n/a	27928	40625	86773	33516	10556
256 bit key	n/a	n/a	n/a	26683	29069	70494	31973	8500

Decryption Key Setup (keys/s)	DES (56 bit)	Triple DES (168 bit)	IDEA	MARS	RC6	Rijndael	Serpent	Twofish
128 bit key	18039	5136	20737	28743	45709	56017	34687	13469
192 bit key	n/a	n/a	n/a	27917	40625	48324	33560	10550
256 bit key	n/a	n/a	n/a	26731	39028	39063	31973	8531

Comparison of AES2 algorithms(III)

- Smart Card Implementation by F. Sano (Toshiba)

Cipher	RAM (bytes)		ROM (bytes)		Time (clock) Encrypt		Schedule		Encrypt + Schedule		
MARS	572	5	5,468		45,588	4	21,742	2	67,330	3	*
RC6	156	3	1,060	2	34,736	3	138,851	4	173,587	4	
Rijndael	66	1	980	1	25,494	1	10,318	1	35,812	1	only encryption
Serpent	164	4	3,937	4	71,924	5	147,972	5	219,896	5	
Twofish	90	2	2,808	3	31,877	2	28,512	3	60,389	2	
DES	17		772						25,398		
Triple DES	17		849						72,341		
MISTY	44		1,598						25,486		

* : omit to check “weak” in the key schedule

Comparison of AES2 algorithms(IV)

- CMOS ASIC Implementation by Ichikawa (Mitsubishi)

Table4.1 Hardware evaluation results

Algorithm name	area [Gate] Encryption & Decryption	Key Schedule	Total	Key setup time[ns]	Critical-path[ns]	Throughput [Mbps]
DES	42,204	12,201	54,405	-	55.11	1161.31
Triple-DES	124,888	23,207	148,147	-	157.09	407.4
MARS	690,654	2,245,096	2,935,754	1740.99	567.49	225.55
RC6	741,641	901,382	1,643,037	2112.26	627.57	203.96
Rijndael	518,508	93,708	612,834	57.39	65.64	1950.03
Serpent	298,533	205,096	503,770	114.07	137.4	931.58
Twofish	200,165	231,682	431,857	16.38	324.8	394.08