Security (CS4028)

Lecture 3. Symmetric Cryptography

Chunyan Mu

chunyan.mu@abdn.ac.uk

The School of Natural and Computing Sciences

Schedule

	Week	Lecture 1	Lecture 2	Tutorial
	1	Intro to course & security	Intro to Crypto	-
⇒	2	Symmetric Crypto	Hash	Math for crypto
	3	Asymmetric Crypto-1	Asymmetric Crypto-2	Symmetric Crypto
	4	Signatures	Zero Knowledge Proof	Asymmetric Crypto
	5	Certificates	Authentication	Signature & certificates
	6	Access Control	AC models	Authentication
	7	Information flow control	Information flow control	Access control
	8	Management	Protocols	Concepts & management
	9	Network security	Network security	Protocols and communications
	10	Advanced Topic	Advanced Topic	Network
	11	Revision		

Lecture 3: Symmetric cryptography

Last lecture

- Cryptography definitions, goals, and terminologies
- Classical ciphers
- Cryptographic modes

This lecture

After having attended this session, you will be able to :

- explain what is block ciphers
- describe Feistel structure
- understand how the DES algorithm works
- understand how the AES algorithm works

Outline

Outline

Block ciphers

What is a block cipher?

A block cipher is a symmetric-key cipher that breaks up the plaintext message into blocks of a fixed length and encrypts one block at a time.

- Break the plaintext P into successive blocks:
  - $P_1, P_2, \ldots$ of size n bits each
- Encrypt each P_i with the same key K of size k bits
  - $E_K(P) = E_K(P_1)E_K(P_2)\ldots$

Block ciphers

Main types of block ciphers

- substitution cipher: replace symbols (or groups of symbols) by other symbols (or groups of symbols);
- transposition cipher: permutes the symbols within the block;
- product ciphers: is a composite of substitution and transposition ciphers.

DES is a block cipher based on Feistel structure!

Outline

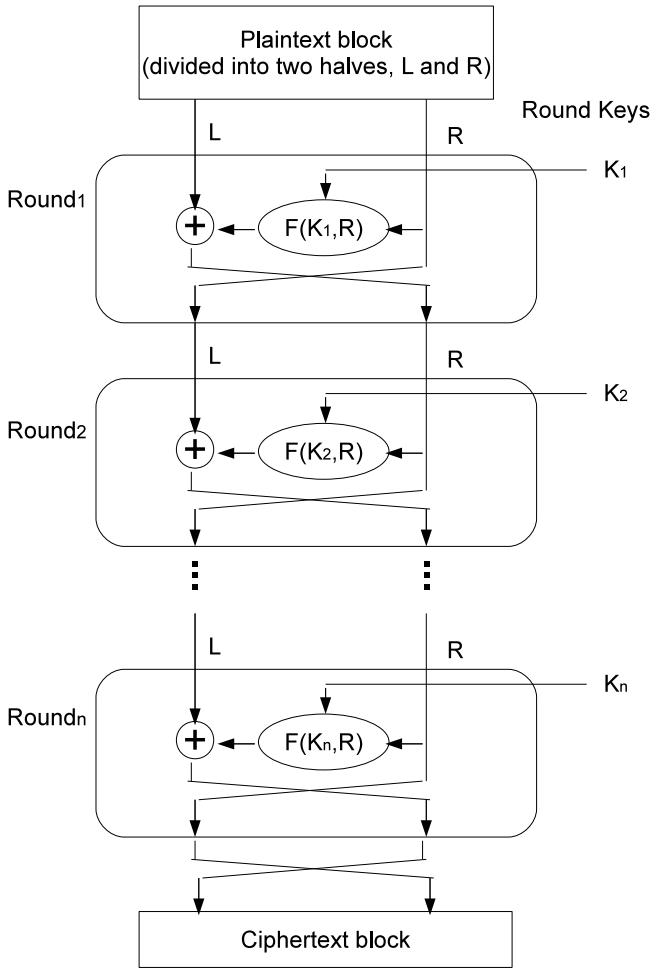
Feistel structure for block ciphers

The Feistel structure

- Named after the IBM cryptographer Horst Feistel and first implemented in the Lucifer cipher by Horst Feistel and Don Coppersmith
- A cryptographic system based on Feistel structure uses the same basic algorithm for both encryption and decryption
- Consists of multiple rounds of processing of the plaintext, with each round consisting of a "substitution" step followed by a permutation step: combination of P and S-boxes

Feistel structure for block ciphers

The Feistel structure



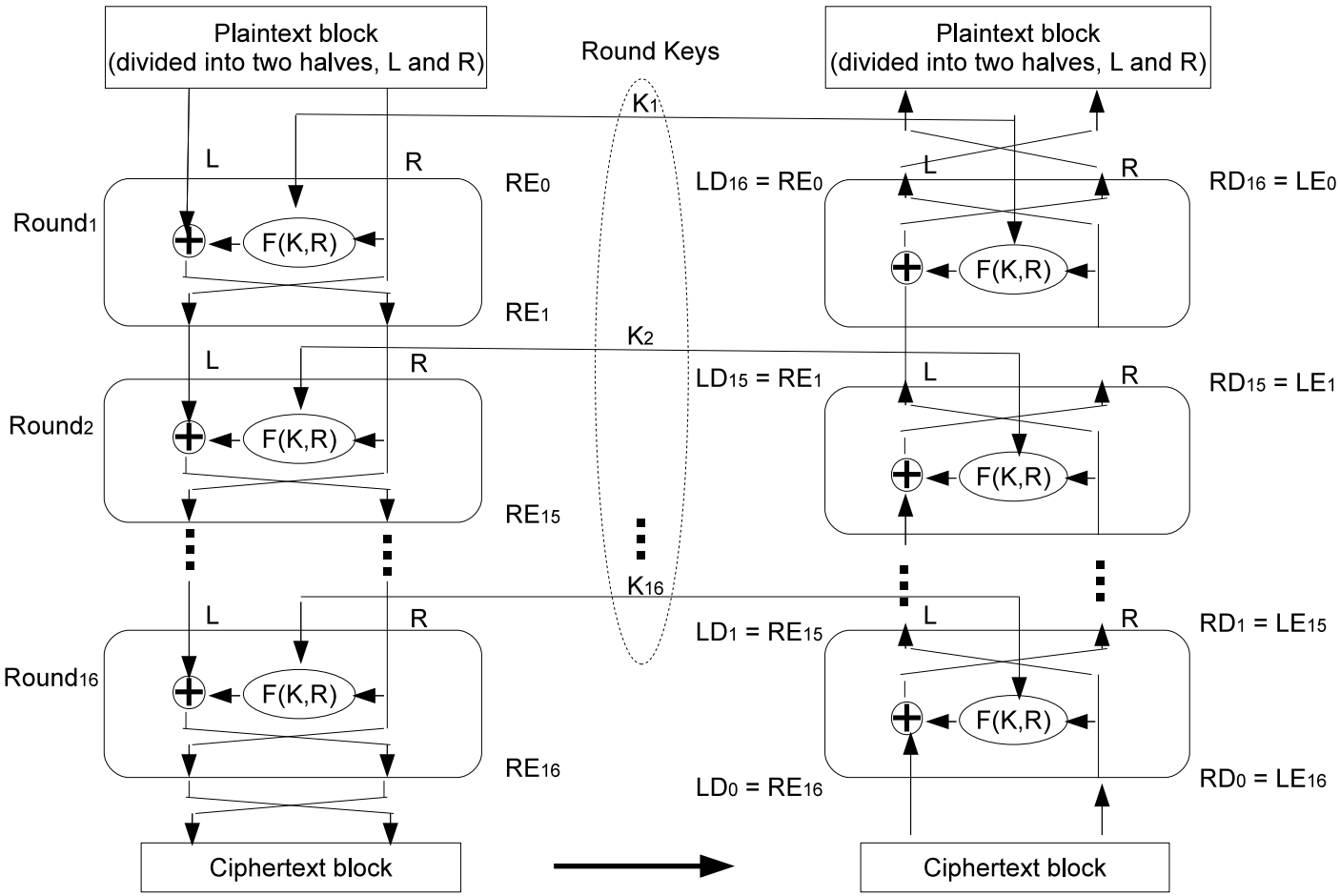
Feistel structure for block ciphers

Each round in the Feistel structure

- Let LE_i and RE_i denote the output half-blocks at the end of the i^{th} round of processing. The letter "E" denotes encryption
- we have:
  - $LE_i = RE_{i-1}$
  - $RE_i = LE_{i-1} \oplus F(RE_{i-1}, K_i)$
- F is referred to as the Feistel function.
- Assuming 16 rounds of processing, the output of the last round of processing is given by:
  - $LE_{16} = RE_{15}$
  - $RE_{16} = LE_{15} \oplus F(RE_{15}, K_{16})$

Feistel structure for block ciphers

Decryption in the Feistel structure



Feistel structure for block ciphers

Decryption in the Feistel structure

- The decryption algorithm is exactly the same as the encryption algorithm with the only difference that the round keys are used in the reverse order.
- The output of each round during decryption is the input to the corresponding round during encryption.

Outline

Data Encryption Standard

history

- 1973-74: NBS (National Bureau of Standards) issued a call for proposals for a standard cryptographic algorithm:
  1. completely specified, efficient and easy to understand;
  2. high level of security;
  3. security on the key, not on the (public) algorithm;
  4. adaptable, economically implementable in electronic devices;
- 1975: IBM's solution is accepted and published in Federal Register, the key size has been reduced from the original 112-bits to 56-bits
- 1976: DES was adopted
- 1981: ANSI approved DES as a private-sector standard
- 1998: The end of DES

DES: facts and remarks

Remarks

- Symmetric: encryption and decryption are using the same key.
- Despite the $2^{56} > 72,000,000,000,000,000$ possible keys, the Electronic Frontier Foundation built in 1998 a machine called DES Deep Crack that could track the DES algorithm by brute-force in an average of 4 days.
- The DES is said to be the most widely used encryption algorithm in the world.

DES: facts and remarks

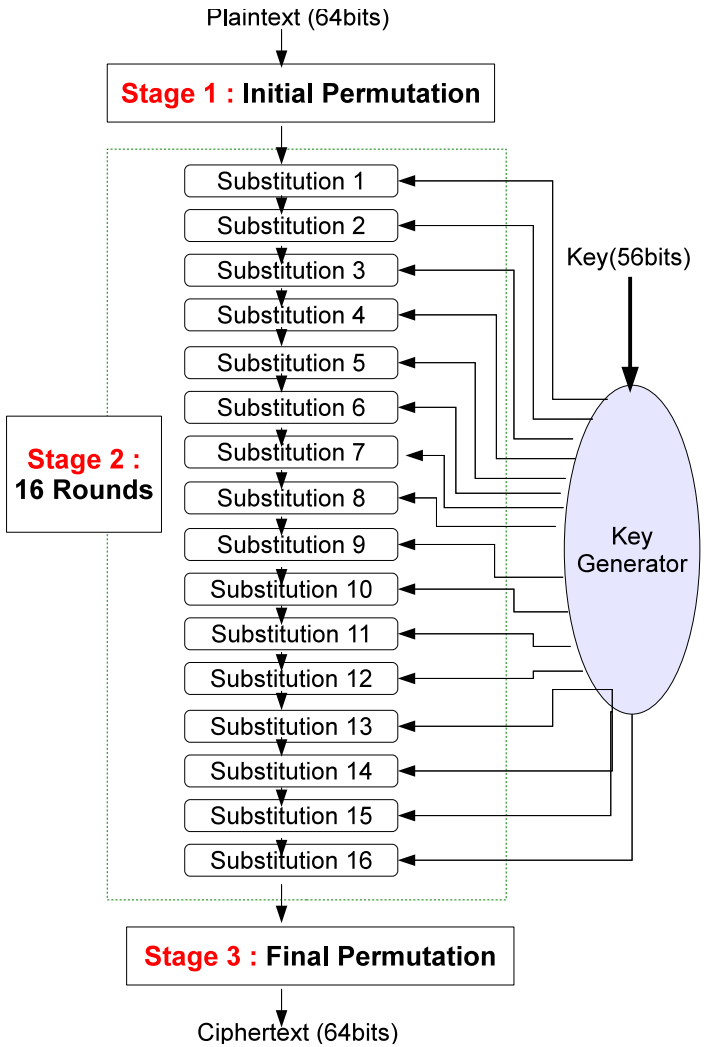
3-stage/18-step algorithm

1. Initial permutation;
2. 16 operations (rounds): DES uses the Feistel cipher structure with 16 rounds of processing;
3. Final permutation;

Sizes

Plaintext blocks	64 bits
Ciphertext blocks	64 bits
Key	56 bits

DES: the three stages



DES: Stage 1 (rearranging the bit order)

The Initial Permutation (IP)

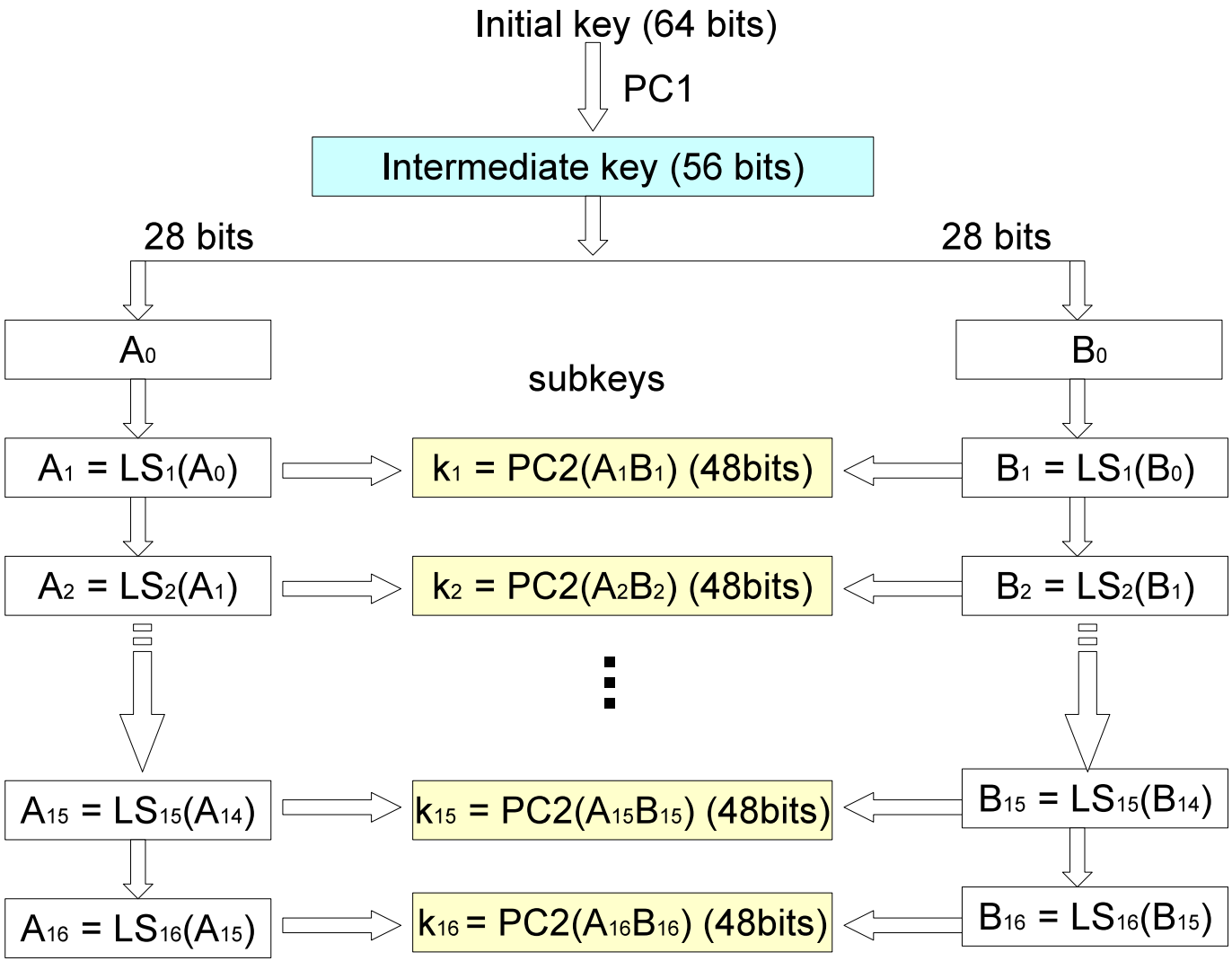
58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5	63	55	47	39	31	23	15	7

$$\text{new } b_1 \leftarrow \text{old } b_{58} \quad \text{new } b_2 \leftarrow \text{old } b_{50}$$
$$\text{new } b_3 \leftarrow \text{old } b_{42} \quad \text{new } b_4 \leftarrow \text{old } b_{34}$$

- We split the permutated block into two halves:
  - L_0 : left-most 32 bits
  - and R_0 : right-most 32 bits
- Remark: if the block is shorter than 64 bits, it should be padded with zeros.

Subkeys generation

Process



Subkeys generation

Process

- **Input:** The original 64-bit key k: the key is usually stored as a 64-bit number originally
- **Algorithm:**
  1. $k' = PC1(k)$ (56-bit intermediate key);
  2. $A_0 = b_{57}\ b_{49}\ \ldots\ b_{36}$ (first half, 28 bits);
  3. $B_0 = b_{63}\ b_{55}\ \ldots\ b_4$ (last half, 28 bits);
  4. 16 generation stages:
     - $A_j = LS_j(A_{j-1})$
     - $B_j = LS_j(B_{j-1})$
     - $k_j = PC2(A_jB_j)$

     where LS_j : Cyclic shift to the left (left-rotations) by 1 if $j \in \{1, 2, 9, 16\}$ and 2 otherwise
- **Output:** A set of 16 keys k_j

Keys generator: Permutation Choice PC1

PC1

- Permutation + Compression
- Permutation Choice 1: $64 \rightarrow 56$

57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

$$\text{new } b_1 \leftarrow \text{old } b_{57} \quad \text{new } b_2 \leftarrow \text{old } b_{49}$$
$$\text{new } b_3 \leftarrow \text{old } b_{41} \quad \text{new } b_4 \leftarrow \text{old } b_{33}$$

- Note that there is no $8, 16, 24, 32, 40, 48, 56$ and $64 \ldots$

Keys generator: Permutation Choice PC2

PC2

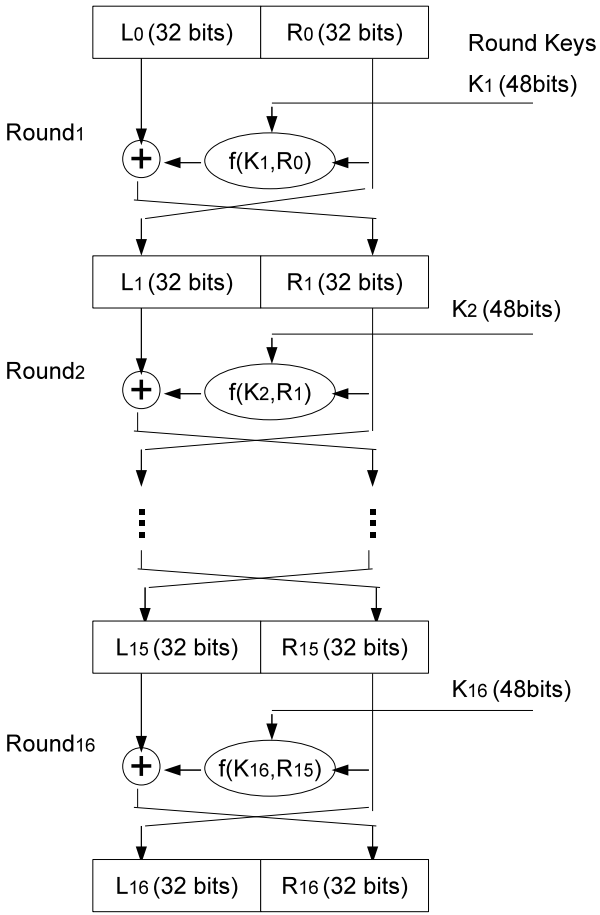
- Permutation + Compression
- Permutation Choice 2: $56 \rightarrow 48$

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

$$\text{new } b_1 \leftarrow \text{old } b_{14} \quad \text{new } b_2 \leftarrow \text{old } b_{17}$$
$$\text{new } b_3 \leftarrow \text{old } b_{11} \quad \text{new } b_4 \leftarrow \text{old } b_{24}$$

- Note that there is no $9, 18, 22, 25, 35, 38, 43, 54$

DES: stage 2 (the 16 rounds)



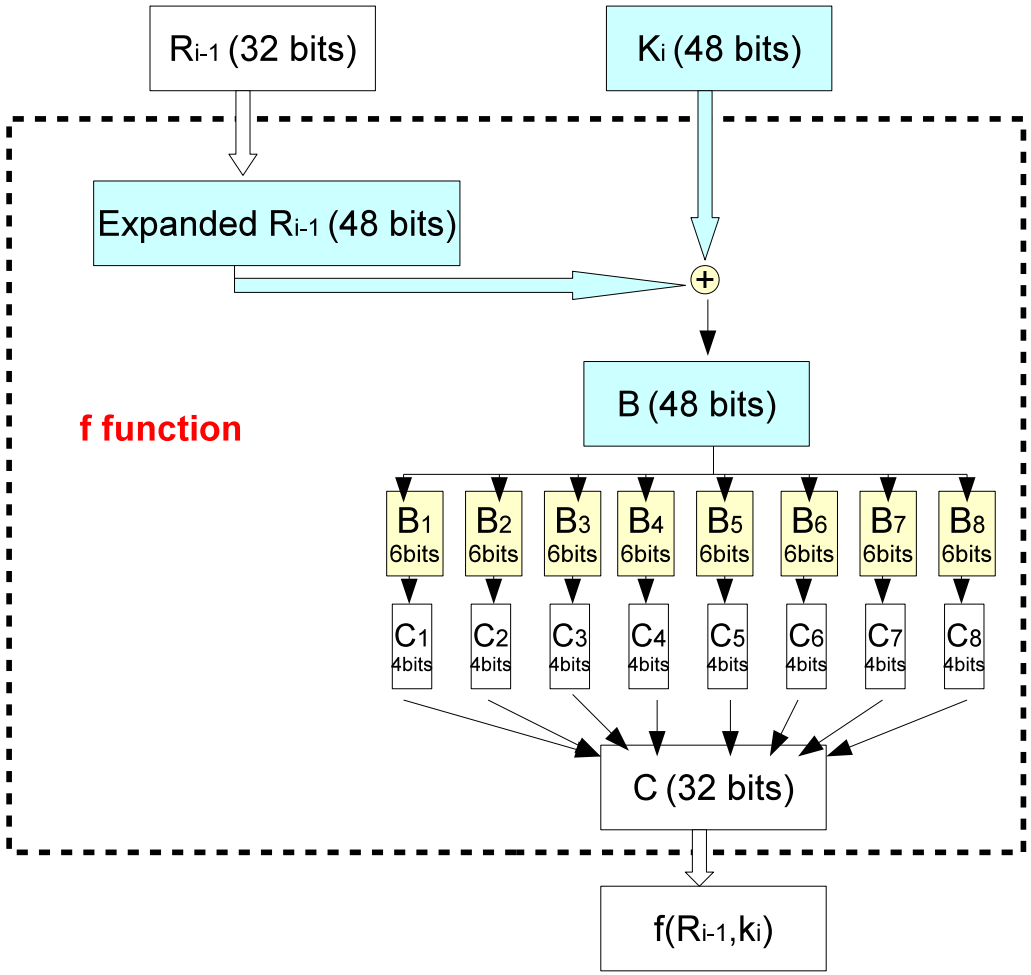
The 16 rounds (Feistel structure)

$L_i R_i$ is computed as follows:

- $L_i = R_{i-1}$;
- $R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$

where $\oplus$ denotes the XOR logical function, f is the specific inner function of DES, and $k_1, k_2, \ldots, k_{16}$ are the subkeys the key generator produced.

DES: the f function



DES: the f function

The S_1 substitution example: 4×6-bit S-Box

- The 8 S-boxes of size 4×6 : $B_i \rightarrow C_i$.
- Substitution: B_j is a 6-bit block $\rightarrow$ C_j is a 4-bit block
  - Given a 6-bit input, the 4-bit output is found by selecting the row using the outer two bits, and the column using the inner four bits.
  - e.g. $B_1 = 101010$, row $= b_1 b_6 = (10)_2 = 2$; column $= b_2 b_3 b_4 b_5 = (0101)_2 = 5 \Leftrightarrow C_1 = 6 = (0110)_2$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	9	0	7	5
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

DES: stage 3

Final Permutation = IP^{-1} (64 bits)

40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25

new $b_1 \leftarrow$ old b_{40} new $b_2 \leftarrow$ old b_8

new $b_3 \leftarrow$ old b_{48} new $b_4 \leftarrow$ old b_{16}

DES: weaknesses

Weaknesses

- Symmetric: the algorithm is symmetric:
  - the same initial key and the same algorithm will decrypt the ciphertext into the original plaintext.
- Public algorithms and functions:
  - the algorithm and the permutations, functions, ... are well known.
  - security depends on the length of the key.
- Initial and final permutations: useless

Variant: 3DES

3DES (Triple Data Encryption System)

- uses a series of DES encryptions/decryptions:
  - there exist multiple variant.
- the official accepted variant is DES-EDE, that is, GIVEN 3 keys k_1, k_2 and k_3, compute:
  - Ciphertext $= E_{k_3}(D_{k_2}(E_{k1}(\texttt{Plaintext})))$
- Note that $k_1 = k_2 = k_3$ make Triple DES compatible with DES.
- It requires three times the computation of normal DES:
  - stronger than DES
- Key size: 168 bits.

Outline

Advanced Encryption Standard

AES origin

- clearly a replacement for DES was needed
  - Key size is too small
  - The variants are just patches
- can use Triple DES: but slow, has small blocks
- US NIST issued call for ciphers in 1997
- 15 candidates accepted in Jun, 1998
- 5 were shortlisted in Aug, 1999

Advanced Encryption Standard

AES origin: Competition Requirements

- private key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- stronger and faster than Triple-DES
- provide full specification and design details
- both C and Java implementations

AES: the shortlist

The shortlist

MARS	IBM
RC6	RSA Kabiratirues
Rijndael	Joan Daemen (Proton World International) and Vincent Rijment (Katholieke Univ. Leuven)
Serpent	Ross Anderson (University of Cambridge), Eli Biham (Technion), and Lars Knudsen (University of California San Diego)
Twofish	Bruce Schneier, John Kelsey, and Niels Ferguson (Counterpane, Inc.), Doug Whiting (Hi/fn, Inc.), David Wagner (University of California Berkeley), and Chris Hall (Princeton University)

AES: the shortlist

The shortlist

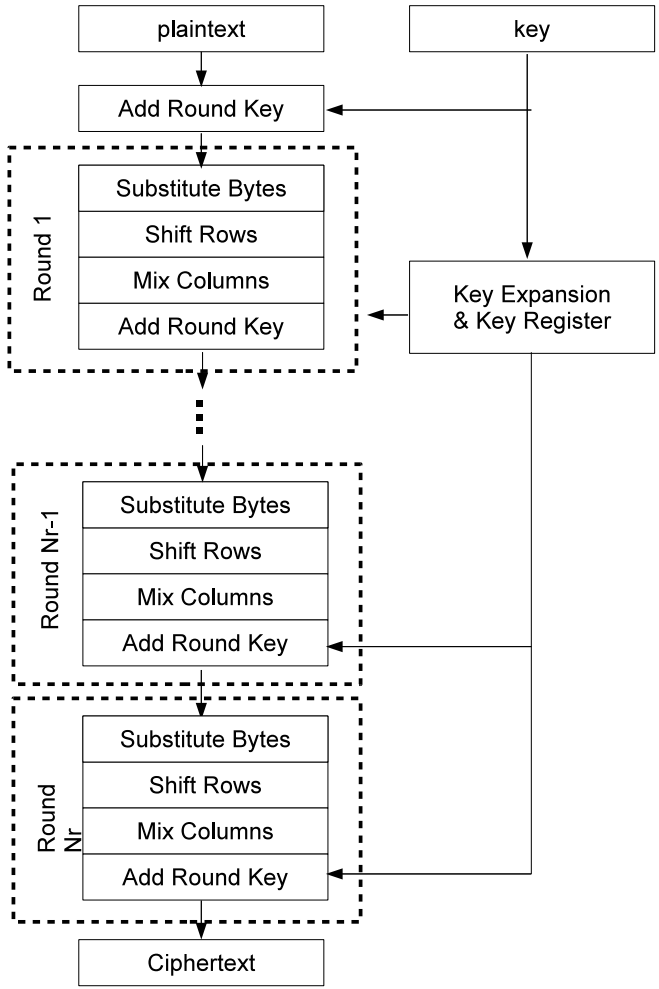
MARS	IBM
RC6	RSA Kabiratirues
Rijndael	Joan Daemen (Proton World International) and Vincent Rijment (Katholieke Univ. Leuven)
Serpent	Ross Anderson (University of Cambridge), Eli Biham (Technion), and Lars Knudsen (University of California San Diego)
Twofish	Bruce Schneier, John Kelsey, and Niels Ferguson (Counterpane, Inc.), Doug Whiting (Hi/fn, Inc.), David Wagner (University of California Berkeley), and Chris Hall (Princeton University)

The winner is ...

- Rijndael was selected: supports more block and key sizes.

AES: overview

Rijndael



AES: overview

Sizes

- plaintext blocks: 128 bits (16 bytes)
- supports also 192 bits (24 bytes) and 256 bits (32 bytes)
- key: 128, 192, 256 bits
- number of rounds ($Nr = 10, 12, 14$) depends on the size of blocks and key

AES: the algorithm

Hight-level description of the algorithm

1. KeyExpansion - round keys are derived from the cipher key using Rijndael's key schedule
2. Initial Round
   - AddRoundKey - each byte of the state is combined with the round key using bitwise xor

AES: the algorithm

Hight-level description of the algorithm

3. Rounds
   - SubBytes - a non-linear substitution step where each byte is replaced with another according to a lookup table.
   - ShiftRows - a transposition step where each row of the state is shifted cyclically a certain number of steps.
   - MixColumns - a mixing operation which operates on the columns of the state, combining the four bytes in each column.
   - AddRoundKey
4. Final Round (no MixColumns)
   - SubBytes
   - ShiftRows
   - AddRoundKey

The state and key bytes arrays

The state arrays

size (bits)	size (bytes)	*Nb* of column
128	16	4
192	24	6
256	32	8

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$	$a_{0,6}$	$a_{0,7}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$	$a_{1,6}$	$a_{1,7}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$	$a_{2,6}$	$a_{2,7}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$	$a_{3,6}$	$a_{3,7}$

$$a_{0,0} = PT[0],\ a_{1,0} = PT[1],\ a_{2,0} = PT[2],\ a_{3,0} = PT[3],$$
$$a_{0,1} = PT[4],\ a_{1,1} = PT[5],\ a_{2,1} = PT[6],\ a_{3,1} = PT[7],$$
$$\ldots\ldots$$

The state and key bytes arrays

The key arrays

size (bits)	size (bytes)	Nk of column
128	16	4
192	24	6
256	32	8

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	$k_{0,6}$	$k_{0,7}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	$k_{1,6}$	$k_{1,7}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$	$k_{2,4}$	$k_{2,5}$	$k_{2,6}$	$k_{2,7}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$	$k_{3,4}$	$k_{3,5}$	$k_{3,6}$	$k_{3,7}$

The state and key bytes arrays

Associated Number of rounds (Nr)

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

- Nb for number of columns in state arrrays
  - AES has a fixed block size of 128 bits - $Nb = 4$
- Nk for number of columns in key arrays
- Nr for number of rounds

AES KeyExpansion

KeyExpansion: generating a series of keys

- In AES, there are a number of rounds, each needing its own key, so the actual key is "stretched out" and transformed to give portions of key for each round.
- The key expansion routine:
  - input: key (denoted key below) of $4 \times Nk$ bytes ($Nk = 4, 6, 8$).
  - output: an expanded key of $Nb \times (Nr + 1)$ bytes, where $Nb = 4$ and Nr is the number of rounds in the algorithm
    - $Nr = 10$ in case $Nk = 4$;
    - $Nr = 12$ in case $Nk = 6$;
    - $Nr = 14$ in case $Nk = 8$.

AES KeyExpansion

KeyExpansion: expanded key size

Expanded Key Sizes in Words		
Nk	Nr	Exp. Key Size ($Nb(Nr+1)$)
4	10	44
6	12	52
8	14	60

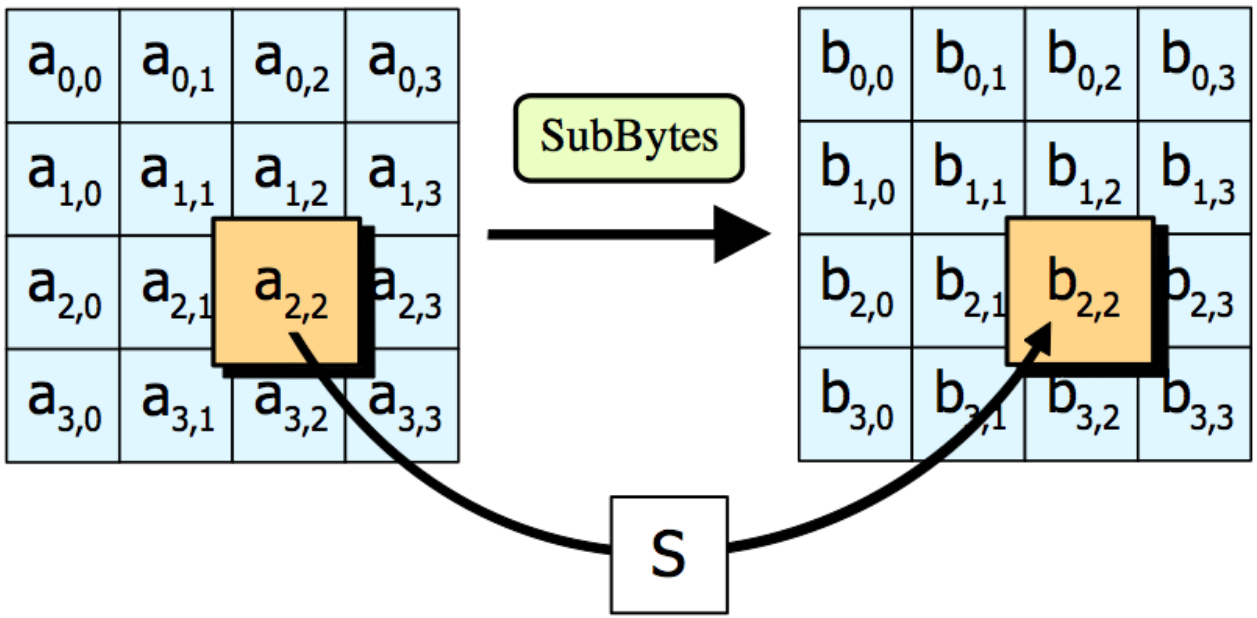
The AES SubBytes step

SubBytes: : S-box and byte substitution

- each byte in the array is updated using an 8-bit substitution box, the Rijndael S-box.
- uses non-linear S-Box represented as a 16×16 array containing a permutation of all 256 8-bit values
- each byte of state is replaced by byte indexed by row left 4-bits) and column (right 4-bits)
  - e.g., byte 57 is replaced by byte in row 5 column 7 which has value $5B$.
- S-box constructed using defined transformation of values in GF(256), based on modular arithmetic with polynomials, can be defined algebraically, not random.

The AES SubBytes step

SubBytes: Byte Substitution



The AES SubBytes step

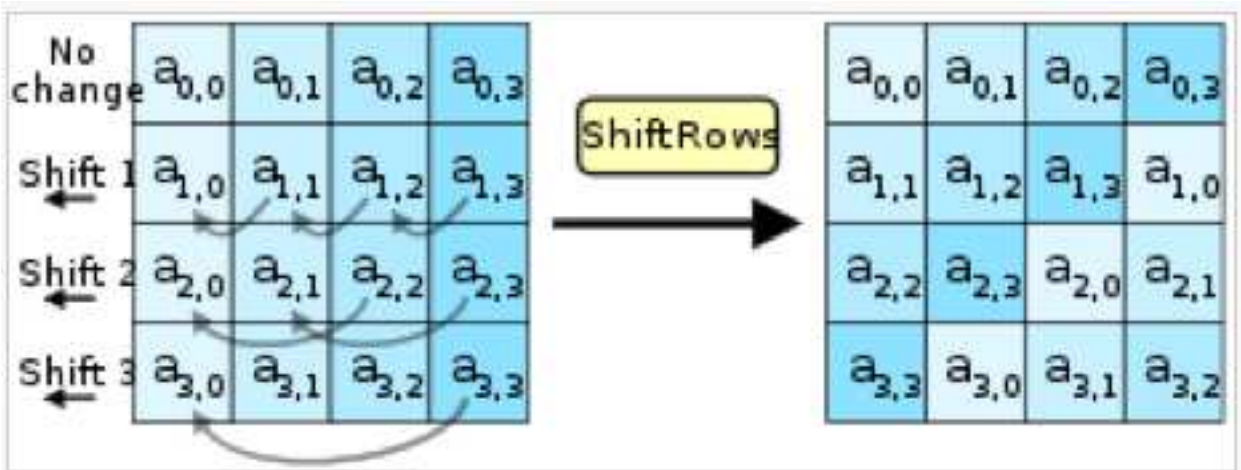
a single S-box: $57 \rightarrow 5B$

	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

AES ShiftRow: Left shifts

ShiftRow: cyclic left shifts

- the ShiftRows step operates on the rows of the state;
- it cyclically shifts the bytes in each row by a certain offset.
- for AES, the first row is left unchanged.
- shifts $= [0, 1, 2, 3]$ if key size $= 128, 192$
- shifts $= [0, 1, 3, 4]$ if key size $= 256$



AES MixColumn: Mixing Columns

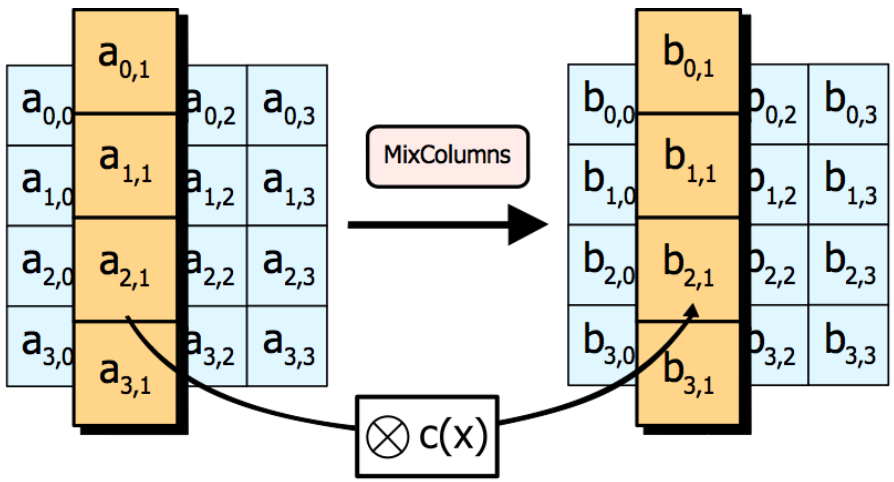
MixColumn: Mixing the content of the columns

- in the MixColumns step, the four bytes of each column of the state are combined using an invertible linear transformation.
- the MixColumns function takes four bytes as input and outputs four bytes, where each input byte affects all four output bytes.
- together with ShiftRows, MixColumns provides diffusion in the cipher.

AES MixColumn: Mixing Columns

MixColumn: Mixing the content of the columns

- carried out for each of the four columns of State.
- each column of State is replaced by a new column which is formed by multiplying that column by a certain matrix of elements of the field



AES MixColumn: Mixing Columns

MixColumn: Mixing the content of the columns

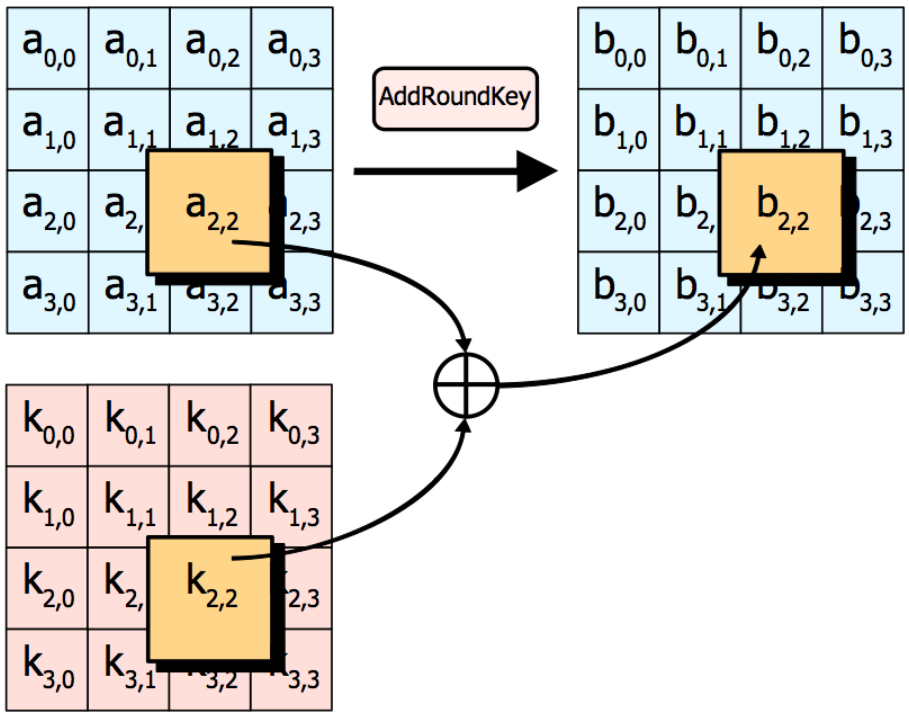
$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

AES AddRoundKey

AddRoundKey: xor

- In the AddRoundKey step, the subkey is combined with the state.
- For each round, a subkey is derived from the main key using Rijndael's key schedule; each subkey is the same size as the state.
- The subkey is added by combining each byte of the state with the corresponding byte of the subkey using bitwise XOR.

AES AddRoundKey



AES AddRoundKey

AddRoundKey: xor

We are considering a position-by-position $a \oplus k$, i.e.

$$\begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \oplus \begin{pmatrix} k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} \end{pmatrix}$$

$$=$$

$$\begin{pmatrix} a_{0,0} \oplus k_{0,0} & a_{0,1} \oplus k_{0,1} & a_{0,2} \oplus k_{0,2} & a_{0,3} \oplus k_{0,3} \\ a_{1,0} \oplus k_{1,0} & a_{1,1} \oplus k_{1,1} & a_{1,2} \oplus k_{1,2} & a_{1,3} \oplus k_{1,3} \\ a_{2,0} \oplus k_{2,0} & a_{2,1} \oplus k_{2,1} & a_{2,2} \oplus k_{2,2} & a_{2,3} \oplus k_{2,3} \\ a_{3,0} \oplus k_{3,0} & a_{3,1} \oplus k_{3,1} & a_{3,2} \oplus k_{3,2} & a_{3,3} \oplus k_{3,3} \end{pmatrix}$$

Outline

Summary

This lecture

- Block ciphers recap
- Feistel Structure and DES in detail
- AES in detail

Next lecture

- Hash functions

Outline

Mathematics for AES: GF(2^8) (Galois Fields)

AES is using binary polynomials of degree at most 7

	binary polynomial of degree at most 7
Generic	$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$
Example 1	$x^6 + x^4 + x^2 + x + 1$
Example 2	$x^7 + x^5 + x + 1$

Equivalent binary, decimal and hexadecimal representations

	binary	decimal	hexa
Generic	$b_7\ b_6\ b_5\ b_4\ b_3\ b_2\ b_1\ b_0$	$d_1\ d_2\ d_3$	$h_1\ h_2$
Example 1	0 1 0 1 0 1 1 1	$8\ 7_d$	$5\ 7_h$
Example 2	1 0 1 0 0 0 1 1	$1\ 6\ 3_d$	$A\ 3_h$
Example 3	[00000000, 11111111]	$[0, 255_d]$	$[0, FF_h]$

Galois Fields: additions

"Normal" polynomial addition

$$
\begin{array}{rrrrrrrrrrrrrrr}
 & & & & x^6 & + & & & x^4 & + & x^2 & + & x & + & 1 \\
+ & x^7 & + & & & & x^5 & + & & & & & x & + & 1 \\
\hline
= & x^7 & + & x^6 & + & x^5 & + & x^4 & + & x^2 & + & 2x & + & 2 &
\end{array}
$$

"Normal" polynomial addition $\rightarrow$ binary polynomials?

- even coefficients $\rightarrow 0$
- odd coefficients $\rightarrow 1$

Galois Fields: additions

"Binary" polynomial additions are using bitwise xor ($\oplus$)

Polynomial

$$\begin{array}{l} \quad x^6 + \quad x^4 + x^2 + x + 1 \\ \oplus\ x^7 + \quad x^5 + \quad\quad\quad x + 1 \\ \hline = x^7 + x^6 + x^5 + x^4 + x^2 \end{array}$$

"Binary" polynomial additions are using bitwise xor ($\oplus$)

Binary	**Decimal**	**Hexadecimal**
01010111	87_d	57_h
$\oplus$10100011	$\oplus 163_d$	$\oplus A3_h$
$= 11110100$	244_d	$F4_h$

Galois Fields: multiplications

"Normal" polynomial multiplication

$$
\begin{aligned}
& (x^6+x^4+x^2+x+1)\cdot(x^7+x^5+x+1) \\
= & \qquad\qquad\qquad\qquad\quad x^6+\quad\; x^4+\quad\; x^2+x+1 \\
+ & \qquad\qquad\quad x^7+\quad\; x^5+\quad\; x^3+\; x^2\; +x \\
+ & \qquad x^{11}+x^9+\quad\; x^7+\; x^6+\; x^5 \\
+ & \; x^{13}+\; x^{11}+\; x^9+\quad\; x^7 \\
= & \; x^{13}+2x^{11}+2x^9+x^8+3x^7+2x^6+2x^5+x^4+x^3+2x^2+2x+1
\end{aligned}
$$

"Normal" polynomial$\longrightarrow$ binary polynomial multiplication?

- even coefficients $\longrightarrow 0$
- odd coefficients $\longrightarrow 1$

Galois Fields: multiplications

"Binary" polynomial multiplication

$$
\begin{array}{ll}
 & 01110111 \cdot 10100011 = \\
 & \quad\quad\quad 0\,1\,1\,1\,0\,1\,1\,1 \\
\oplus & \quad\quad 0\,1\,1\,1\,0\,1\,1\,1 \\
\oplus & \quad\quad 0\,1\,1\,1\,0\,1\,1\,1 \\
\oplus & \quad 0\,1\,1\,1\,0\,1\,1\,1 \\
\hline
= & \quad 0\,1\,1\,0\,1\,0\,1\,1\,1\,1\,1\,1\,0\,0\,1 \\
= & \quad x^{13} + x^{12} + x^{10} + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + 1
\end{array}
$$

Is the binary polynomial of degree still at most 7?

Galois Fields: division

polynomial division N/D

- A new resulting term NRT is obtainned by dividing the greatest power of N by the greatest power of D
- A new numerator NN is obtained by computing $NN = N - NRT \times D$
- This procedure is repeated until the greatest power of the new numerator NN is less than the greatest power of the denominator D: $GP(NN) < GP(D)$
- The final numerator is the remainder of the polynomial division
- Obtain the desired binary polynomial by applying generalised XOR rules to the remainder:
  - even coefficient $\rightarrow 0$
  - odd coefficient $\rightarrow 1$

Galois Fields: division

polynomial division N/D

```
NN <- N
while deg(NN) > deg(D) do
      NRT <- greatest power(NN)/greatest power(D)
      NN <- NN - NRT * D
done
return(NN)
```

Galois Fields: division

Example: polynomial division $(x^5 + x^3 + x^2 + 1)/(x^2 + x)$

Galois Fields: division

Example: polynomial division $(x^5 + x^3 + x^2 + 1)/(x^2 + x)$

- $N = x^5 + x^3 + x^2 + 1$, $D = x^2 + x$

NN	NRT	$NN - NRT \times D$
$x^5 + x^3 + x^2 + 1$	$x^5/x^2 = x^3$	$(x^5 + x^3 + x^2 + 1) - x^3 \cdot (x^2 + x) = -x^4 + x^3 + x^2 + 1$
$-x^4 + x^3 + x^2 + 1$	$-x^4/x^2 = -x^2$	$(-x^4 + x^3 + x^2 + 1) - (-x^2) \cdot (x^2 + x) = 2x^3 + x^2 + 1$
$x^2 + 1$	$x^2/x^2 = 1$	$(x^2 + 1) - 1 \cdot (x^2 + x) = -x + 1$
$x + 1$		

- $x^5 + x^3 + x^2 + 1 = (x^2 + x) \cdot (x^3 - x^2 + 1) + x + 1$
- $x^5 + x^3 + x^2 + 1 \equiv (x + 1) \bmod (x^2 + x)$

Galois Fields: division

Example: the Rijndael case - "reduced" binary polynomial multiplication

- $x^{13} + x^8 + x^7 + x^4 + x^3 + 1$: a binary polynomial of degree greater than 7
- instead, we consider the remainder of the division of this polynomial by a **given** irreducible binary polynomial of degree 8:

$$m_x = x^8 + x^4 + x^3 + x + 1 \equiv 1000110011 \equiv 283_d \equiv 133_h$$

- we have:

$$x^{13}+x^8+x^7+x^4+x^3+x^2+x+1 \equiv x^7+x^6+x^3+x^2+x+1 \bmod m_x$$

- try: $N = x^{13} + x^{12} + x^{10} + x^8 + x^7 + x^6 + x^5 + x^4 + x^3$

"Reduced" binary polynomial multiplication by x

$$
\begin{aligned}
\texttt{xtime}(P_x) &= P_x \cdot x \\
&= (b_7 x^8 + b_6 x^7 + b_5 x^6 + b_4 x^5 + b_3 x^4 + b_2 x^3 \\
&\quad + b_1 x^2 + b_0 x) \bmod m_x \\
&= (b_6 x^7 + b_5 x^6 + b_4 x^5 + (b_3 \oplus b_7) x^4 \\
&\quad + (b_2 \oplus b_7) x^3 + b_1 x^2 \\
&\quad + (b_0 \oplus b_7) x + b_7) \bmod m_x
\end{aligned}
$$

How do I compute xtime(57)?

57_h	5				7			
binary	0	1	0	1	0	1	1	1
orig. order	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
new order	b_6	b_5	b_4	$b_3 \oplus b_7$	$b_2 \oplus b_7$	b_1	$b_0 \oplus b_7$	b_7
binary	1	0	1	0	1	1	1	0
xtime(57)	A				E			

How do I deduce $57 \cdot 13$?

How do I deduce $57 \cdot 13$?

$$
\begin{aligned}
57 \cdot 01 &= 57 \\
57 \cdot 02 &= \text{xtime}(57) = AE \\
57 \cdot 04 &= \text{xtime}(AE) = 47 \\
57 \cdot 08 &= \text{xtime}(47) = 8E \\
57 \cdot 10 &= \text{xtime}(8E) = 07
\end{aligned}
$$

$$
\begin{aligned}
57 \cdot 13 &= 57 \cdot (10 \oplus 02 \oplus 01) \\
&= 57 \cdot 10 \oplus 57 \cdot 02 \oplus 57 \cdot 01 \\
&= 07 \oplus AE \oplus 57 \\
&= 00000111 \oplus 10101110 \oplus 010100111 \\
&= 11111110 = FE
\end{aligned}
$$