

EE 4215

Security Technology

Lecture 3: Number Theory & DES

2018/19 Semester B

Course Structure

- Week 1: Security Fundamental
- Week 2: Classical Cryptography (116 students)
- **Week 3: Number Theory & DES**
- Week 4: Secret-key Cryptography
- Week 5: Public-key Cryptography
- Week 5: Quiz 1
- Week 6: Hashing Algorithm
- Week 7: PKI, SSL/TLS
- Week 8: Web Security
- Week 9: Wi-Fi & P2P Security
- Week 10: Cloud Computing Security
- Week 10: Quiz 2
- Week 11: Smart-card Security
- Week 12: Blockchain Security Technology
- Week 13: Secure-code, Trusted Hardware, Review

Lecture 3: Number Theory

- Number Theory
- Feistel Cipher Structure
- DES

Divisibility

- We say that a nonzero b **divides** a if $a = mb$ for some m , where a , b , and m are integers
- b divides a if there is no remainder on division
- The notation $b \mid a$ is commonly used to mean b divides a
- If $b \mid a$ we say that b is a **divisor** of a

The positive divisors of 24 are 1, 2, 3, 4, 6, 8, 12, and 24
 $13 \mid 182$; $-5 \mid 30$; $17 \mid 289$; $-3 \mid 33$; $17 \mid 0$

Properties of Divisibility

- If $a \mid 1$, then $a = \pm 1$
- If $a \mid b$ and $b \mid a$, then $a = \pm b$
- Any $b \neq 0$ divides 0
- If $a \mid b$ and $b \mid c$, then $a \mid c$
- If $b \mid g$ and $b \mid h$, then $b \mid (mg + nh)$ for arbitrary integers m and n

$$11 \mid 66 \text{ and } 66 \mid 198 = 11 \mid 198$$

Properties of Divisibility

- To see this last point, note that:
 - If $b \mid g$, then g is of the form $g = b * g_1$ for some integer g_1
 - If $b \mid h$, then h is of the form $h = b * h_1$ for some integer h_1
- So:
 - $mg + nh = mbg_1 + nbh_1 = b * (mg_1 + nh_1)$
and therefore b divides $mg + nh$

$$b = 7; g = 14; h = 63; m = 3; n = 2$$

$$7 \mid 14 \text{ and } 7 \mid 63.$$

$$\text{To show } 7 \mid (3 * 14 + 2 * 63),$$

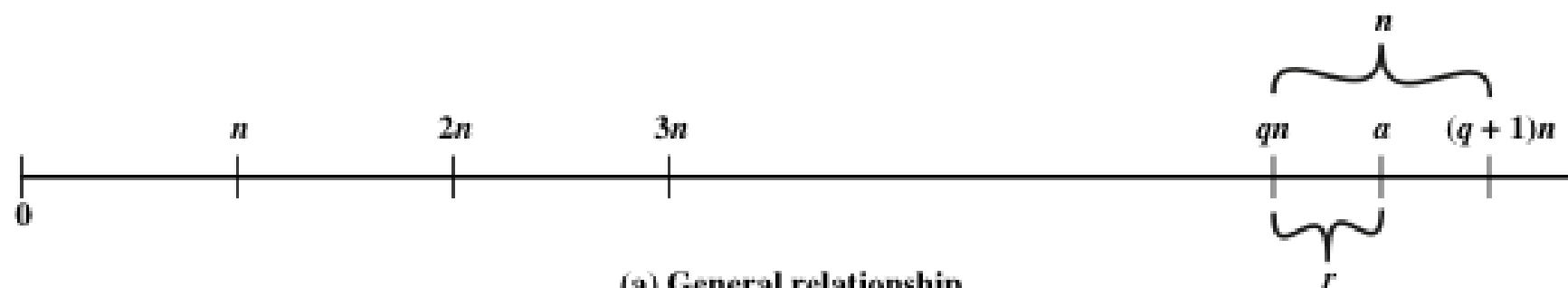
$$\text{we have } (3 * 14 + 2 * 63) = 7(3 * 2 + 2 * 9),$$

$$\text{and it is obvious that } 7 \mid (7(3 * 2 + 2 * 9)).$$

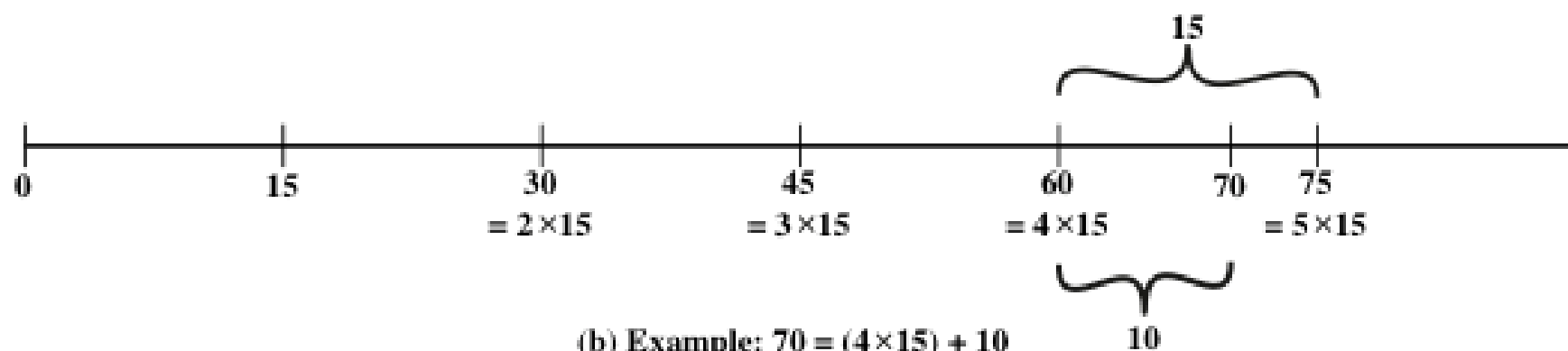
Division Algorithm

- Given any positive integer n and any nonnegative integer a , if we divide a by n we get an integer quotient q and an integer remainder r that obey the following relationship:

$$a = qn + r \qquad 0 \leq r < n; q = [a/n]$$



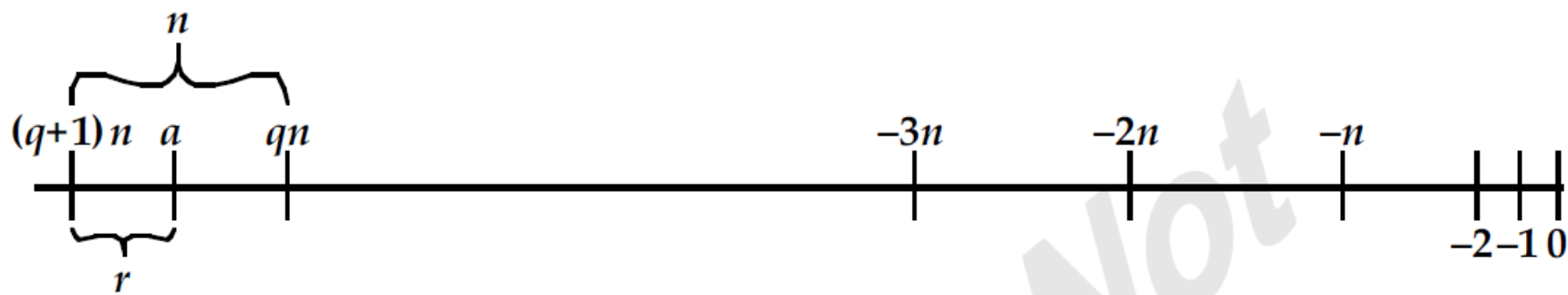
(a) General relationship



(b) Example: $70 = (4 \times 15) + 10$

Figure 2.1 The Relationship $a = qn + r$; $0 \leq r < n$

Draw the same figure for $a < 0$?



Euclidean Algorithm



- One of the basic techniques of number theory
- Procedure for determining the greatest common divisor of two positive integers
- Two integers are **relatively prime** if their only common positive integer factor is 1

Greatest Common Divisor (GCD)

- The greatest common divisor of a and b is the largest integer that divides both a and b
- We can use the notation $\gcd(a,b)$ to mean the **greatest common divisor** of a and b
- We also define $\gcd(0,0) = 0$
- Positive integer c is said to be the gcd of a and b if:
 - c is a divisor of a and b
 - Any divisor of a and b is a divisor of c
- An equivalent definition is:

$$\gcd(a,b) = \max[k, \text{ such that } k \mid a \text{ and } k \mid b]$$

GCD

- Because we require that the greatest common divisor be positive, $\gcd(a,b) = \gcd(a,-b) = \gcd(-a,b) = \gcd(-a,-b)$
- In general, $\gcd(a,b) = \gcd(|a|, |b|)$

$$\gcd(60, 24) = \gcd(60, -24) = 12$$

- Also, because all nonzero integers divide 0, we have $\gcd(a,0) = |a|$
- We stated that two integers a and b are relatively prime if their only common positive integer factor is 1; this is equivalent to saying that a and b are relatively prime if

8 and 15 are relatively prime because the positive divisors of 8 are 1, 2, 4, and 8, and the positive divisors of 15 are 1, 3, 5, and 15. So 1 is the only integer on both lists.

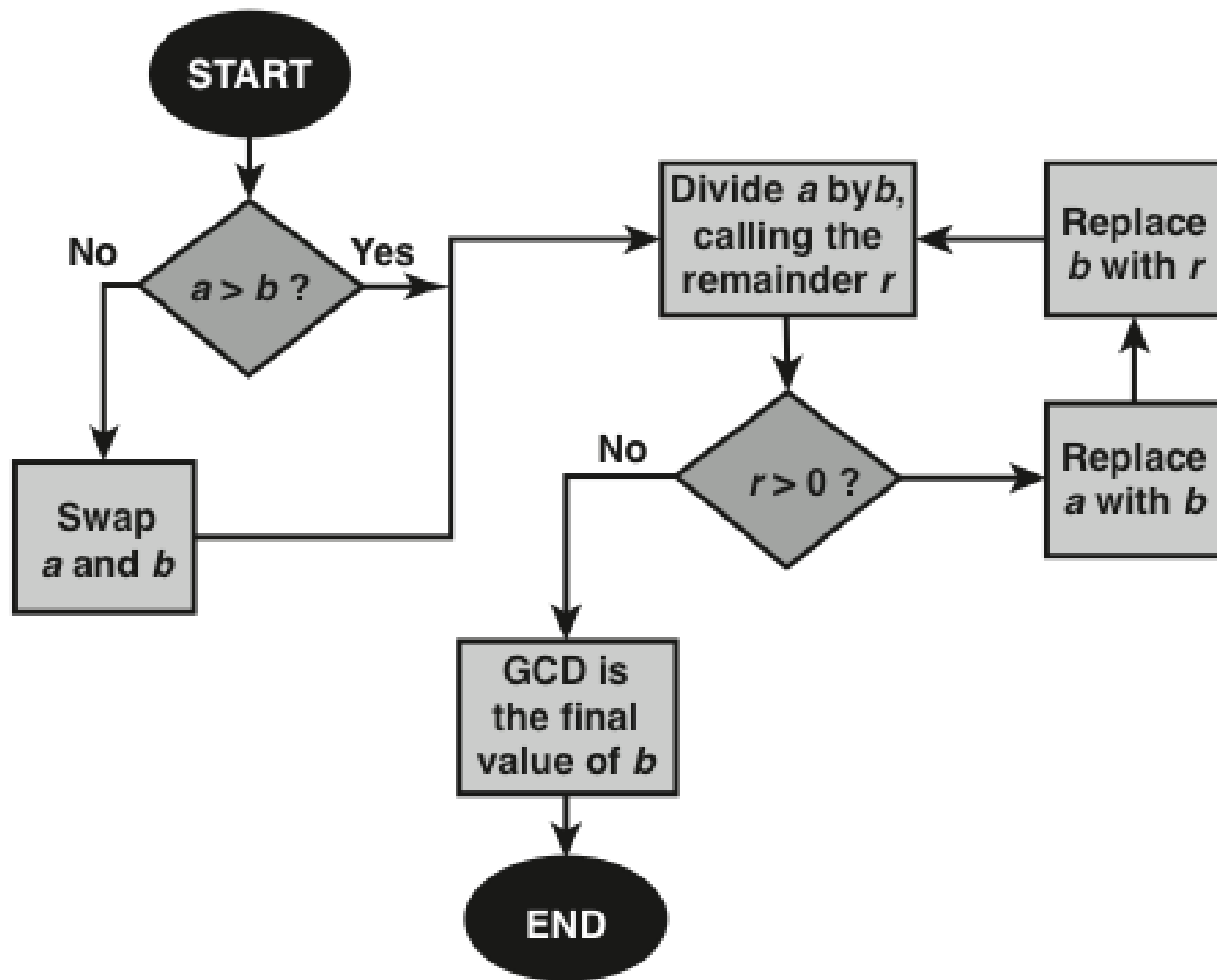


Figure 2.2 Euclidean Algorithm

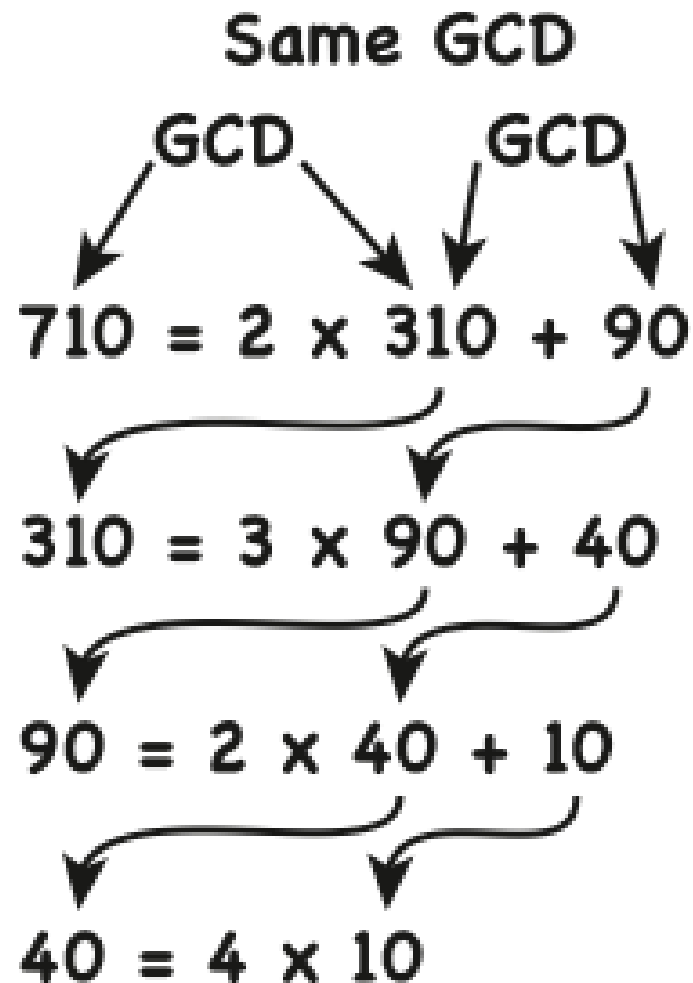


Figure 2.3 Euclidean Algorithm Example: $\text{gcd}(710, 310)$

Euclidean Algorithm Example

Dividend	Divisor	Quotient	Remainder
$a = 1160718174$	$b = 316258250$	$q_1 = 3$	$r_1 = 211943424$
$b = 316258250$	$r_1 = 211943424$	$q_2 = 1$	$r_2 = 104314826$
$r_1 = 211943424$	$r_2 = 104314826$	$q_3 = 2$	$r_3 = 3313772$
$r_2 = 104314826$	$r_3 = 3313772$	$q_4 = 31$	$r_4 = 1587894$
$r_3 = 3313772$	$r_4 = 1587894$	$q_5 = 2$	$r_5 = 137984$
$r_4 = 1587894$	$r_5 = 137984$	$q_6 = 11$	$r_6 = 70070$
$r_5 = 137984$	$r_6 = 70070$	$q_7 = 1$	$r_7 = 67914$
$r_6 = 70070$	$r_7 = 67914$	$q_8 = 1$	$r_8 = 2156$
$r_7 = 67914$	$r_8 = 2156$	$q_9 = 31$	$r_9 = 1078$
$r_8 = 2156$	$r_9 = 1078$	$q_{10} = 2$	$r_{10} = 0$

Modular Arithmetic

- The modulus

- If a is an integer and n is a positive integer, we define $a \bmod n$ to be the remainder when a is divided by n ; the integer n is called the **modulus**
- Thus, for any integer a :

$$a = qn + r \quad 0 \leq r < n; \quad q = [a/n]$$

$$a = [a/n] * n + (a \bmod n)$$

$$11 \bmod 7 = 4; \quad -11 \bmod 7 = 3$$

Modular Arithmetic

- Congruent modulo n
 - Two integers a and b are said to be **congruent modulo n** if $(a \bmod n) = (b \bmod n)$
 - This is written as $a \equiv b \pmod{n}$
 - Note that if $a \equiv 0 \pmod{n}$, then $n \mid a$

$$73 \equiv 4 \pmod{23}; \quad 21 \equiv -9 \pmod{10}$$

Properties of Congruences

- Congruences have the following properties:
 1. $a = b \pmod{n}$ if $n \mid (a - b)$
 2. $a = b \pmod{n}$ implies $b = a \pmod{n}$
 3. $a = b \pmod{n}$ and $b = c \pmod{n}$ imply $a = c \pmod{n}$
- To demonstrate the first point, if $n \mid (a - b)$, then $(a - b) = kn$ for some k
 - So we can write $a = b + kn$
 - Therefore, $(a \bmod n) = (\text{remainder when } b + kn \text{ is divided by } n) = (\text{remainder when } b \text{ is divided by } n) = (b \bmod n)$

$$23 = 8 \pmod{5} \text{ because } 23 - 8 = 15 = 5 * 3$$

$$-11 = 5 \pmod{8} \text{ because } -11 - 5 = -16 = 8 * (-2)$$

$$81 = 0 \pmod{27} \text{ because } 81 - 0 = 81 = 27 * 3$$

Modular Arithmetic

- Modular arithmetic exhibits the following properties:

1. $[(a \bmod n) + (b \bmod n)] \bmod n = (a + b) \bmod n$

2. $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$

3. $[(a \bmod n) * (b \bmod n)] \bmod n = (a * b) \bmod n$

- We demonstrate the first property:

- Define $(a \bmod n) = r_a$ and $(b \bmod n) = r_b$. Then we can write $a = r_a + jn$ for some integer j and $b = r_b + kn$ for some integer k

- Then:

$$\begin{aligned}(a + b) \bmod n &= (ra + jn + rb + kn) \bmod n \\&= (ra + rb + (k + j)n) \bmod n \\&= (ra + rb) \bmod n \\&= [(a \bmod n) + (b \bmod n)] \bmod n\end{aligned}$$

Remaining Properties:

- Examples of the three remaining properties:

$$11 \bmod 8 = 3; 15 \bmod 8 = 7$$

$$[(11 \bmod 8) + (15 \bmod 8)] \bmod 8 = 10 \bmod 8 = 2$$

$$(11 + 15) \bmod 8 = 26 \bmod 8 = 2$$

$$[(11 \bmod 8) - (15 \bmod 8)] \bmod 8 = -4 \bmod 8 = 4$$

$$(11 - 15) \bmod 8 = -4 \bmod 8 = 4$$

$$[(11 \bmod 8) * (15 \bmod 8)] \bmod 8 = 21 \bmod 8 = 5$$

$$(11 * 15) \bmod 8 = 165 \bmod 8 = 5$$

Arithmetic Modulo 8

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

Multiplication Modulo 8

×	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

Additive
and
Multiplicative
Inverse
Modulo 8

	w	$-w$	w^{-1}
0	0	0	—
1	1	7	1
2	2	6	—
3	3	5	3
4	4	4	—
5	5	3	5
6	6	2	—
7	7	1	7

Properties of Modular Arithmetic for Integers in $\mathbb{Z}_n$

Property	Expression
Commutative Laws	$(w + x) \bmod n = (x + w) \bmod n$ $(w \times x) \bmod n = (x \times w) \bmod n$
Associative Laws	$[(w + x) + y] \bmod n = [w + (x + y)] \bmod n$ $[(w \times x) \times y] \bmod n = [w \times (x \times y)] \bmod n$
Distributive Law	$[w \times (x + y)] \bmod n = [(w \times x) + (w \times y)] \bmod n$
Identities	$(0 + w) \bmod n = w \bmod n$ $(1 \times w) \bmod n = w \bmod n$
Additive Inverse ($-w$)	For each $w \in \mathbb{Z}_n$, there exists a z such that $w + z \equiv 0 \bmod n$

Euclidean Algorithm vs. Extended EA

- Euclidean Algorithm is to find the greatest common divisor d.
 - $\text{GCD}(a, b) = d$
- Extended Euclidean Algorithm
 - $ax + by = d = \text{GCD}(a, b)$
 - Also find two additional integers x, and y, satisfying the above equation.
 - Useful for finite field operations, and encryption algorithm, RSA.
 - Example: input a, b = 240, 46
 - Output d?
 - Output x, y?

Example

- $\text{GCD} = 2, -9 * 240 + 47 * 46 = 2$
- $X, Y = -9, 47$

index i	quotient q_{i-1}	Remainder r_i	s_i	t_i
0		240	1	0
1		46	0	1
2	$240 \div 46 = 5$	$240 - 5 \times 46 = 10$	$1 - 5 \times 0 = 1$	$0 - 5 \times 1 = -5$
3	$46 \div 10 = 4$	$46 - 4 \times 10 = 6$	$0 - 4 \times 1 = -4$	$1 - 4 \times -5 = 21$
4	$10 \div 6 = 1$	$10 - 1 \times 6 = 4$	$1 - 1 \times -4 = 5$	$-5 - 1 \times 21 = -26$
5	$6 \div 4 = 1$	$6 - 1 \times 4 = 2$	$-4 - 1 \times 5 = -9$	$21 - 1 \times -26 = 47$
6	$4 \div 2 = 2$	$4 - 2 \times 2 = 0$	$5 - 2 \times -9 = 23$	$-26 - 2 \times 47 = -120$

Extended Euclidean Algorithm Example

i	r_i	q_i	x_i	Y_i
-1	1759		1	0
0	550		0	1
1	109	3	1	-3
2	5	5	-5	16
3	4	21	106	-339
4	1	1	-111	355
5	0	4		

Result: $d = 1$; $x = -111$; $y = 355$

$$-111 * 1759 + 355 * 550 = 1 = \text{GCD}(1759, 550)$$

Prime Numbers

- Prime numbers only have divisors of 1 and itself
 - They cannot be written as a product of other numbers
- Prime numbers are central to number theory
- Any integer $a > 1$ can be factored in a unique way as

$$a = p_1^{a_1} * p_2^{a_2} * \dots * p_t^{a_t}$$

where $p_1 < p_2 < \dots < p_t$ are prime numbers and where each a_i is a positive integer

- This is known as the fundamental theorem of arithmetic

Primes Under 2000

[illegible]

Fermat's Theorem

- States the following:
 - If p is prime and a is a positive integer not divisible by p then
$$a^{p-1} = 1 \pmod{p}$$
- An alternate form is:
 - If p is prime and a is a positive integer then
$$a^p = a \pmod{p}$$

Example:

Try $13^{(3-1)} = 1 \pmod{3}$ & $15^{(3-1)} = 0 \pmod{3}$

Why?

Some Values of Euler's Totient Function $\phi(n)$

It counts the positive integers up to a given integer n that are **relatively prime** to n

the only positive integer that divides both of (a, b) is 1 $\rightarrow$ co-prime

n	$\phi(n)$
1	1
2	1
3	2
4	2
5	4
6	2
7	6
8	4
9	6
10	4

n	$\phi(n)$
11	10
12	4
13	12
14	6
15	8
16	8
17	16
18	6
19	18
20	8

n	$\phi(n)$
21	12
22	10
23	22
24	8
25	20
26	12
27	18
28	12
29	28
30	8

Example:

$$\phi(21) = 3 * 7$$

Co-primes are:

1, 2, 4, 5, 8, 10,
11, 13, 16, 17, 19, 20

Total: 12 of them.

Euler's Theorem

- States that for every a and n that are relatively prime:

$$a^{\phi(n)} = 1 \pmod{n}$$

- An alternative form is:

$$a^{\phi(n)+1} = a \pmod{n}$$

**Important for RSA and
other crypto standards!**

Example: $a = 13$, $n = 5$, $13^4 = 1 \pmod{5}$?

$$13^4 = 28,561 \pmod{5} = 1$$

Example: $a = 15$, $n = 5$, $15^4 = ? \pmod{5}$

$$15^4 = 50,625 \pmod{5} = 0$$

Miller-Rabin Algorithm

- Typically used to test a large number for primality

- Algorithm is:

A **composite number** is a **positive integer** that can be formed by multiplying two smaller positive integers.

TEST (n)

1. Find integers k, q , with $k > 0, q$ odd, so that $(n - 1) = 2^k q$;
2. Select a random integer $a, 1 < a < n - 1$;
3. **if** $a^q \bmod n = 1$ **then** return ("inconclusive") ;
4. **for** $j = 0$ **to** $k - 1$ **do**
5. **if** $(a^{2^j q} \bmod n = n - 1)$ **then** return ("inconclusive") ;
6. return ("composite") ;

Example

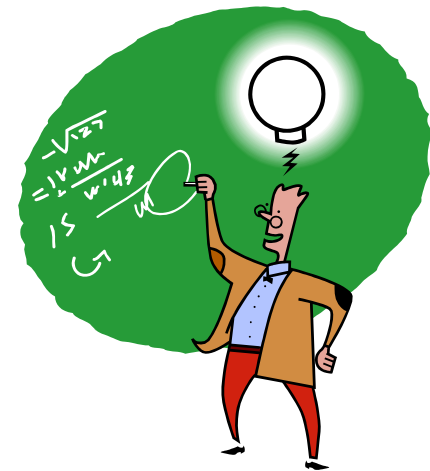
- If $n = 29$ is a prime number
- $(n - 1) = 28 = 2^2(7) \rightarrow k = 2, q = 7$
- $a = 10, 10^7 \bmod 29 = 17$
- $a = 2, 2^7 \bmod 29 = 12, 2^{14} \bmod 29 = 28$
 - All inclusive results

- If $n = 13 * 17 = 221$, a composite number
- $(n-1) = 220 = 2^2(55) \rightarrow k = 2, q = 55$
- $a = 5, 5^{55} \bmod 221 = 112$ (return composite)
- $a = 21, 21^{55} \bmod 221 = 200$ (return inclusive)
- From 2 to 219 for the number 221
 - We have 21, 47, 174, 200, shows that the test is inclusive!

Deterministic Primality Algorithm

- Prior to 2002 there was no known method of efficiently proving the primality of very large numbers
- All of the algorithms in use produced a probabilistic result
- In 2002 Agrawal, Kayal, and Saxena developed an algorithm that efficiently determines whether a given large number is prime
 - Known as the AKS algorithm
 - Does not appear to be as efficient as the Miller-Rabin algorithm

V · T · E	Number-theoretic algorithms	[hide]
Primality tests	AKS · APR · Baillie–PSW · Elliptic curve · Pocklington · Fermat · Lucas · Lucas–Lehmer · Lucas–Lehmer–Riesel · Proth's theorem · Pépin's · Quadratic Frobenius · Solovay–Strassen · Miller–Rabin	
Prime-generating	Sieve of Atkin · Sieve of Eratosthenes · Sieve of Sundaram · Wheel factorization	
Integer factorization	Continued fraction (CFRAC) · Dixon's · Lenstra elliptic curve (ECM) · Euler's · Pollard's rho · $p - 1$ · $p + 1$ · Quadratic sieve (QS) · General number field sieve (GNFS) · Special number field sieve (SNFS) · Rational sieve · Fermat's · Shanks's square forms · Trial division · Shor's	
Multiplication	Ancient Egyptian · Long · Karatsuba · Toom–Cook · Schönhage–Strassen · Fürer's	
Discrete logarithm	Baby-step giant-step · Pollard rho · Pollard kangaroo · Pohlig–Hellman · Index calculus · Function field sieve	
Greatest common divisor	Binary · Euclidean · Extended Euclidean · Lehmer's	
Modular square root	Cipolla · Pocklington's · Tonelli–Shanks	
Other algorithms	Chakravala · Cornacchia · LLL · Integer square root · Modular exponentiation · Schoof's	



Chinese Remainder Theorem (CRT)

- Believed to have been discovered by the Chinese mathematician Sun-Tsu in around 100 A.D.
- One of the most useful results of number theory
- Says it is possible to reconstruct integers in a certain range from their residues modulo a set of pairwise relatively prime moduli
- Can be stated in several ways

Provides a way to manipulate (potentially very large) numbers mod M in terms of tuples of smaller numbers

- This can be useful when M is 150 digits or more
- However, it is necessary to know beforehand the factorization of M



Powers of Integers, Modulo 19

[illegible]

Primitive roots of a prime number

- All sequences end in 1.
- Some of the sequences are of length 18. We call them this number “a primitive root of the modulus 19.”
- They are: 2, 3, 10, 13, 14, 15.
- Exercise: Try prime number 13.

Tables of Discrete Logarithms, Modulo 19

Table 8.4 Tables of Discrete Logarithms, Modulo 19

(a) Discrete logarithms to the base 2, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{2,19}(a)$	18	1	13	2	16	14	6	3	8	17	12	15	5	7	11	4	10	9

(b) Discrete logarithms to the base 3, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{3,19}(a)$	18	7	1	14	4	8	6	3	2	11	12	15	17	13	5	10	16	9

(c) Discrete logarithms to the base 10, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{10,19}(a)$	18	17	5	16	2	4	12	15	10	1	6	3	13	11	7	14	8	9

(d) Discrete logarithms to the base 13, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{13,19}(a)$	18	11	17	4	14	10	12	15	16	7	6	3	1	5	13	8	2	9

(e) Discrete logarithms to the base 14, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{14,19}(a)$	18	13	7	8	10	2	6	3	14	5	12	15	11	1	17	16	4	9

(f) Discrete logarithms to the base 15, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_{15,19}(a)$	18	5	11	10	8	16	12	15	4	13	6	3	7	17	1	2	14	9

Discrete Logarithm Example

- $\text{Log}_{2,19}(2)$
- $2^{\text{exp}} = 2 \pmod{19}$, $\text{exp} = 1$

- $\text{Log}_{2,19}(3)$
- $2^{\text{exp}} = 3 \pmod{19}$, $\text{exp} = 13$

- $\text{Log}_{3,19}(2)$
- $3^{\text{exp}} = 2 \pmod{19}$, $\text{exp} = 7$

- $\text{Log}_{3,19}(3)$
- $3^{\text{exp}} = 3 \pmod{19}$, $\text{exp} = 1$

- Calculator: <https://www.alpertron.com.ar/DILOG.HTM>

Lecture 3: Number Theory

- Number Theory
- Feistel Cipher Structure
- DES

Stream Cipher

Encrypts a digital data stream one bit or one byte at a time

Examples:

- Autokeyed Vigenère cipher
- Vernam cipher

In the ideal case, a one-time pad version of the Vernam cipher would be used, in which the keystream is as long as the plaintext bit stream

If the cryptographic keystream is random, then this cipher is unbreakable by any means other than acquiring the keystream

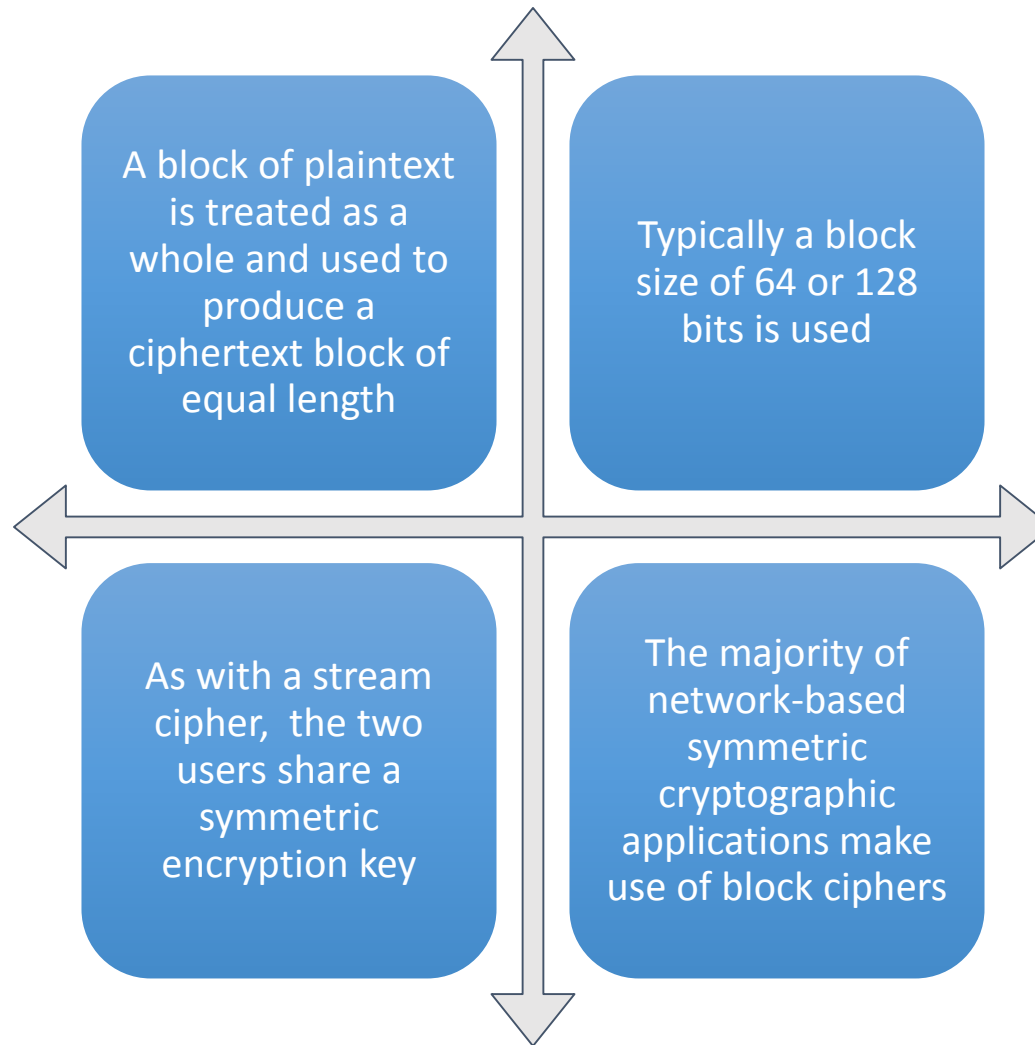
- Keystream must be provided to both users in advance via some independent and secure channel
- This introduces insurmountable logistical problems if the intended data traffic is very large

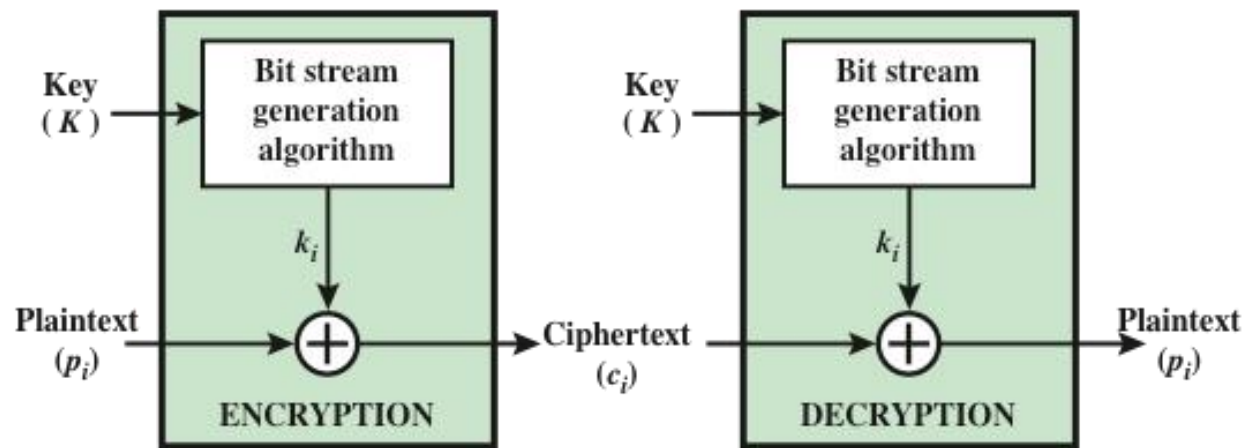
For practical reasons the bit-stream generator must be implemented as an algorithmic procedure so that the cryptographic bit stream can be produced by both users

It must be computationally impractical to predict future portions of the bit stream based on previous portions of the bit stream

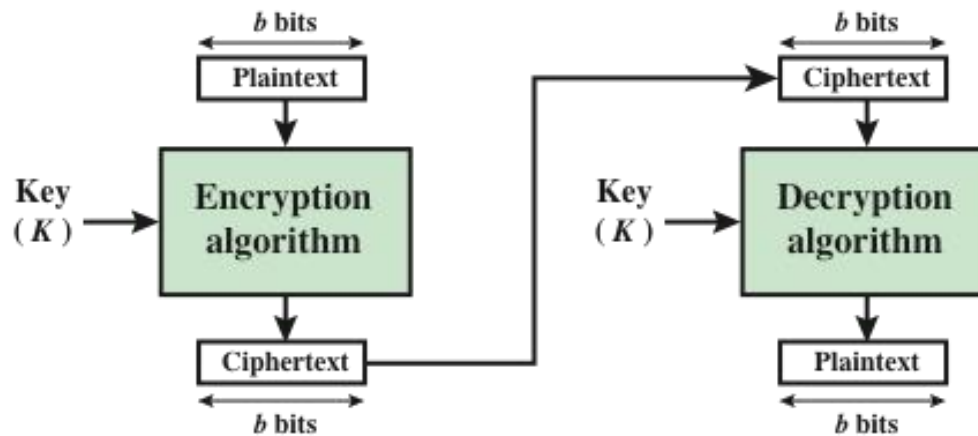
The two users need only share the generating key and each can produce the keystream

Block Cipher





(a) Stream Cipher Using Algorithmic Bit Stream Generator



(b) Block Cipher

Figure 4.1 Stream Cipher and Block Cipher

Block Ciphers

- So many choices!

v • d • e	Block ciphers (security summary)
Common algorithms	AES • Blowfish • DES • Triple DES • Serpent • Twofish
Less common algorithms	Camellia • CAST-128 • IDEA • RC2 • RC5 • SEED • Skipjack • TEA • XTEA
Other algorithms	3-Way • ABC • Akelarre • Anubis • ARIA • BaseKing • BassOmatic • BATON • BEAR and LION • CAST-256 • CICS-1 • CIPHERUNICORN-A • CIPHERUNICORN-E • CLEFIA • CMEA • Cobra • COCONUT98 • Crab • Cryptomeria/C2 • CRYPTON • CS-Cipher • DEAL • DES-X • DFC • E2 • FEAL • FEA-M • FROG • G-DES • GOST • Grand Cru • Hasty Pudding cipher • Hierocrypt • ICE • IDEA NXT • Intel Cascade Cipher • Iraqi • KASUMI • KeeLoq • KHAZAD • Khufu and Khafre • KN-Cipher • Ladder-DES • Libelle • LOKI97 • LOKI89/91 • Lucifer • M6 • M8 • MacGuffin • Madryga • MAGENTA • MARS • Mercy • MESH • MISTY1 • MMB • MULTI2 • MultiSwap • New Data Seal • NewDES • Nimbus • NOEKEON • NUSH • Q • RC6 • REDOC • Red Pike • S-1 • SAFER • SAVILLE • SC2000 • SHACAL • SHARK • SMS4 • Spectr-H64 • Square • SXAL/MBAL • Threefish • Treyfer • UES • Xenon • xmx • XXTEA • Zodiac
Design	Feistel network • Key schedule • Product cipher • S-box • P-box • SPN
Attack (Cryptanalysis)	Brute force • Linear • Differential (Impossible • Truncated) • Integral • Boomerang • Mod n • Related-key • Slide • Rotational • Timing • XSL
Standardization	AES process • CRYPTREC • NESSIE
Misc	Avalanche effect • Block size • Initialization vector • Key size • Modes of operation • Piling-up lemma • Weak key • EFF DES cracker • Key whitening

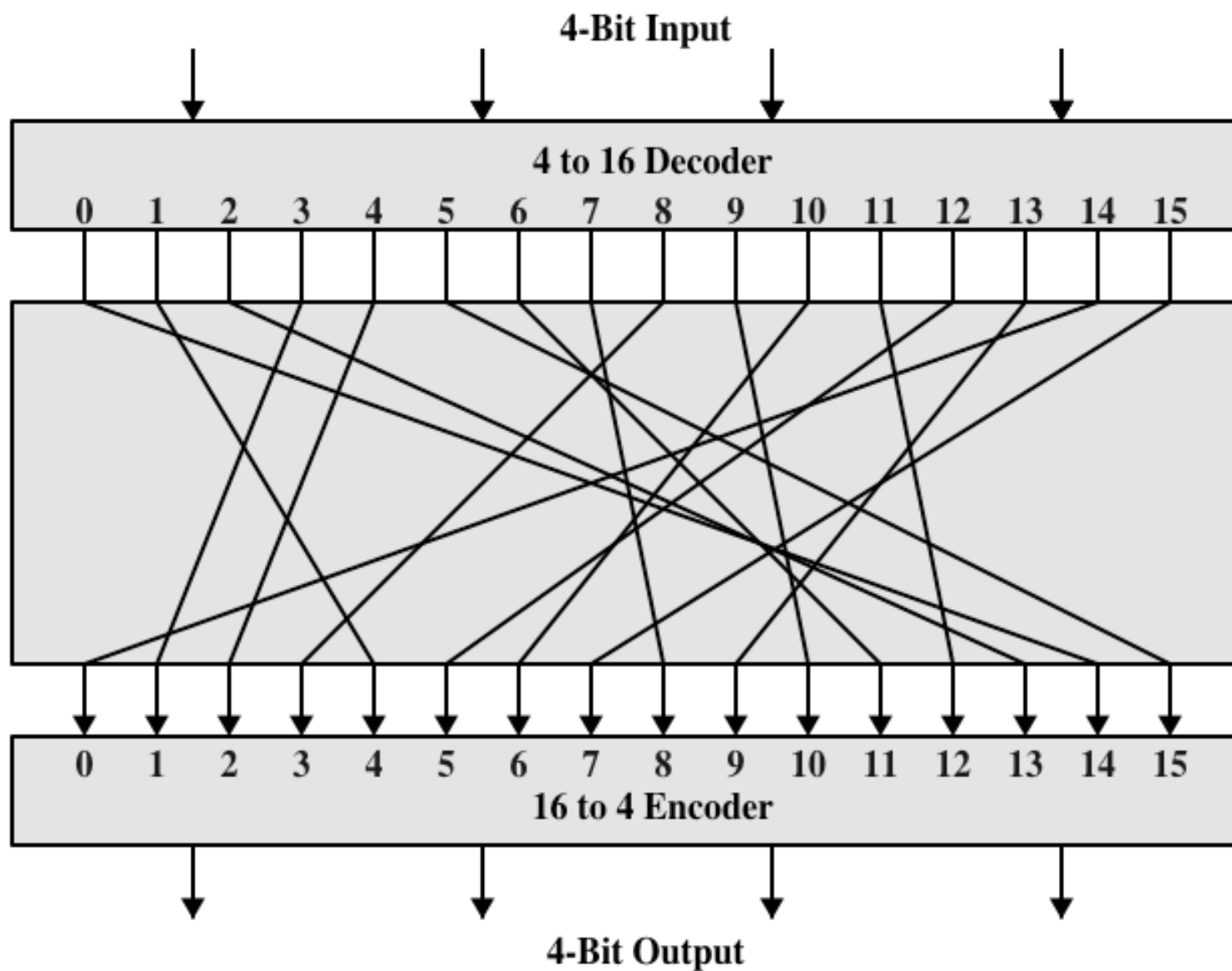


Figure 4.2 General n -bit- n -bit Block Substitution (shown with $n = 4$)

Encryption and Decryption Tables for Substitution Cipher

Plaintext	Ciphertext
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

Ciphertext	Plaintext
0000	1110
0001	0011
0010	0100
0011	1000
0100	0001
0101	1100
0110	1010
0111	1111
1000	0111
1001	1101
1010	1001
1011	0110
1100	1011
1101	0010
1110	0000
1111	0101

Feistel Cipher

- Feistel proposed the use of a cipher that alternates substitutions and permutations

Substitutions

- Each plaintext element or group of elements is uniquely replaced by a corresponding ciphertext element or group of elements

Permutation

- No elements are added or deleted or replaced in the sequence, rather the order in which the elements appear in the sequence is changed

- Is a practical application of a proposal by Claude Shannon to develop a product cipher that alternates confusion and diffusion functions
- Is the structure used by many significant symmetric block ciphers currently in use

Feistel Cipher Structure

- Use **substitutions** and **permutations** (or transposition) approaches similar to the classical cryptography
 - The idea rooted from Shannon's paper "*Communication Theory of Secrecy Systems*" in 1949
 - <http://www3.alcatel-lucent.com/bstj/vol28-1949/articles/bstj28-4-656.pdf> or <http://netlab.cs.ucla.edu/wiki/files/shannon1949.pdf>
- The process will be repeated or iterated for a number of rounds, e.g. 16 rounds
- **Reading Exercise: Shannon's paper**

Diffusion and Confusion

- Terms introduced by Claude Shannon to capture the two basic building blocks for any cryptographic system
 - Shannon's concern was to thwart cryptanalysis based on statistical analysis

Diffusion

- The statistical structure of the plaintext is dissipated into long-range statistics of the ciphertext
- This is achieved by having each plaintext digit affect the value of many ciphertext digits

Confusion

- Seeks to make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible
- Even if the attacker can get some handle on the statistics of the ciphertext, the way in which the key was used to produce that ciphertext is so complex as to make it difficult to deduce the key

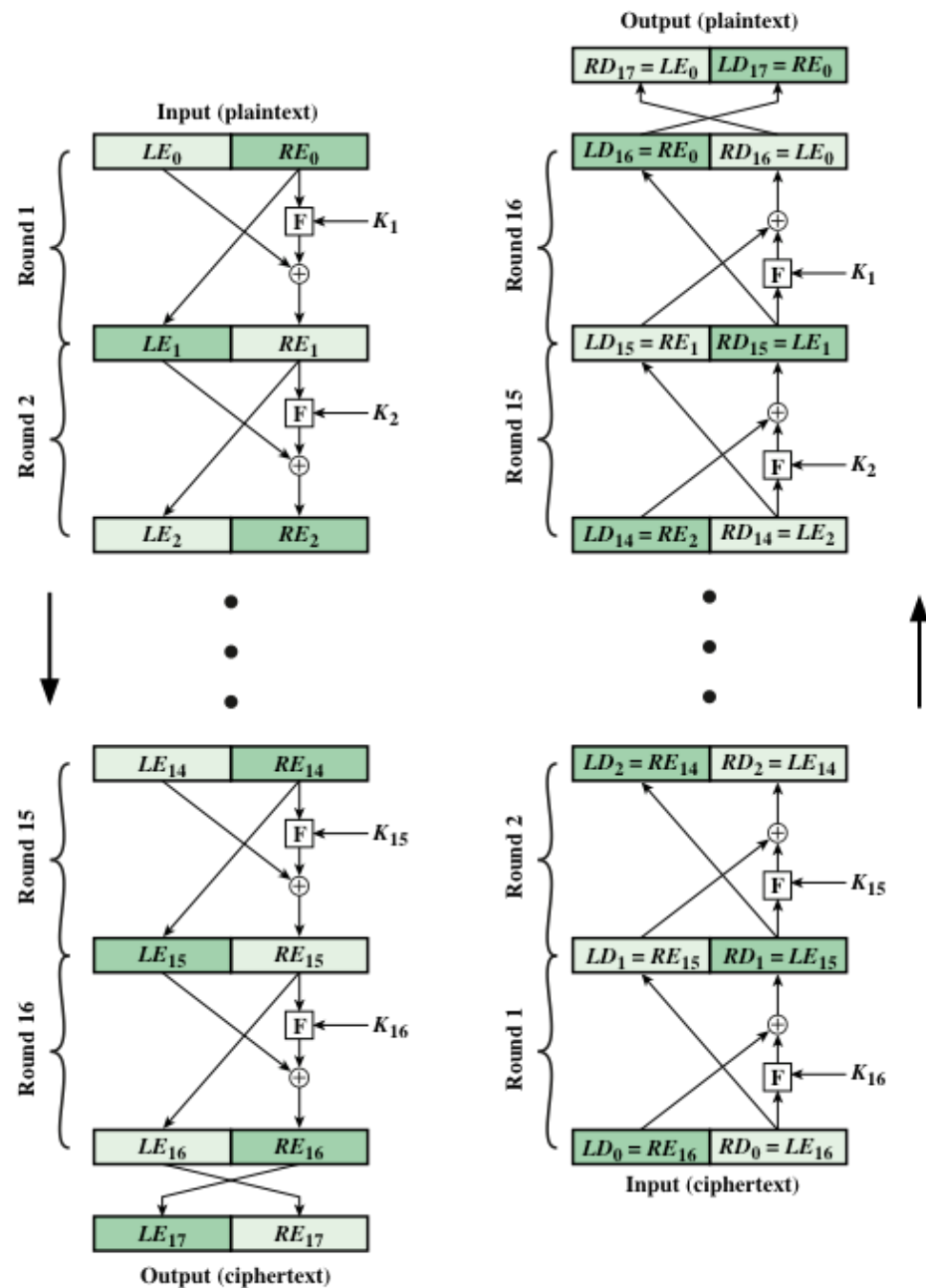


Figure 4.3 Feistel Encryption and Decryption (16 rounds)

Feistel Cipher Structure

- A **block** cipher is one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length
 - e.g. block sizes of 64 or 128 bits are commonly used
- Virtually all conventional **block** encryption algorithms have a structure first described by Horst Feistel of IBM in 1973
- The realization of a Feistel Network depends on the choice of a number of parameters and design features (see next slide)

Feistel Cipher Design Features

- Block size
 - Larger block sizes mean greater security but reduced encryption/decryption speed for a given algorithm
- Key size
 - Larger key size means greater security but may decrease encryption/decryption speeds
- Number of rounds
 - The essence of the Feistel cipher is that a single round offers inadequate security but that multiple rounds offer increasing security
- Subkey generation algorithm
 - Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis
- Round function F
 - Greater complexity generally means greater resistance to cryptanalysis
- Fast software encryption/decryption
 - In many cases, encrypting is embedded in applications or utility functions in such a way as to preclude a hardware implementation; accordingly, the speed of execution of the algorithm becomes a concern
- Ease of analysis
 - If the algorithm can be concisely and clearly explained, it is easier to analyze that algorithm for cryptanalytic vulnerabilities and therefore develop a higher level of assurance as to its strength

Feistel Cipher Structure

- What we need to prepare ourselves?
 - Be prepared to handle binary number
 - E.g. 'A' is 1000001 (in ASCII representation)
 - Get familiar with the operation XOR (or $\oplus$)

Exclusive-Or		
a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

$$a \oplus a = 0$$

$$a \oplus 0 = a$$

$$a \oplus 1 = \sim a, \text{ where } \sim \text{ is bit complement.}$$

$$a \oplus b = b \oplus a \text{ (commutativity)}$$

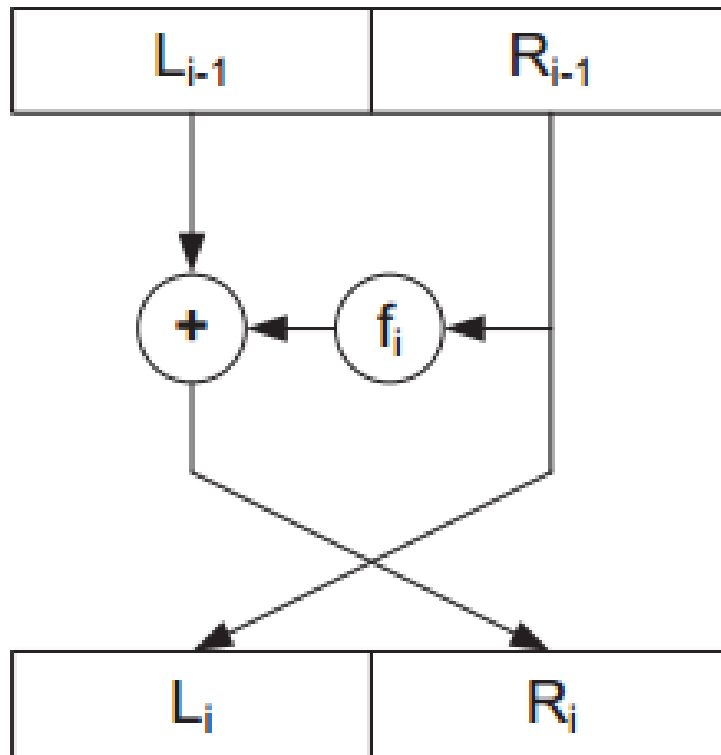
$$a \oplus (b \oplus c) = (a \oplus b) \oplus c \text{ (associativity)}$$

$$a \oplus a \oplus a = a$$

$$\text{if } a \oplus b = c, \text{ then } c \oplus b = a \text{ and } c \oplus a = b.$$

Feistel Cipher Structure

- Representative of one round:
 - The block will be split into two portions: Left (L) and Right (R)



Feistel Example

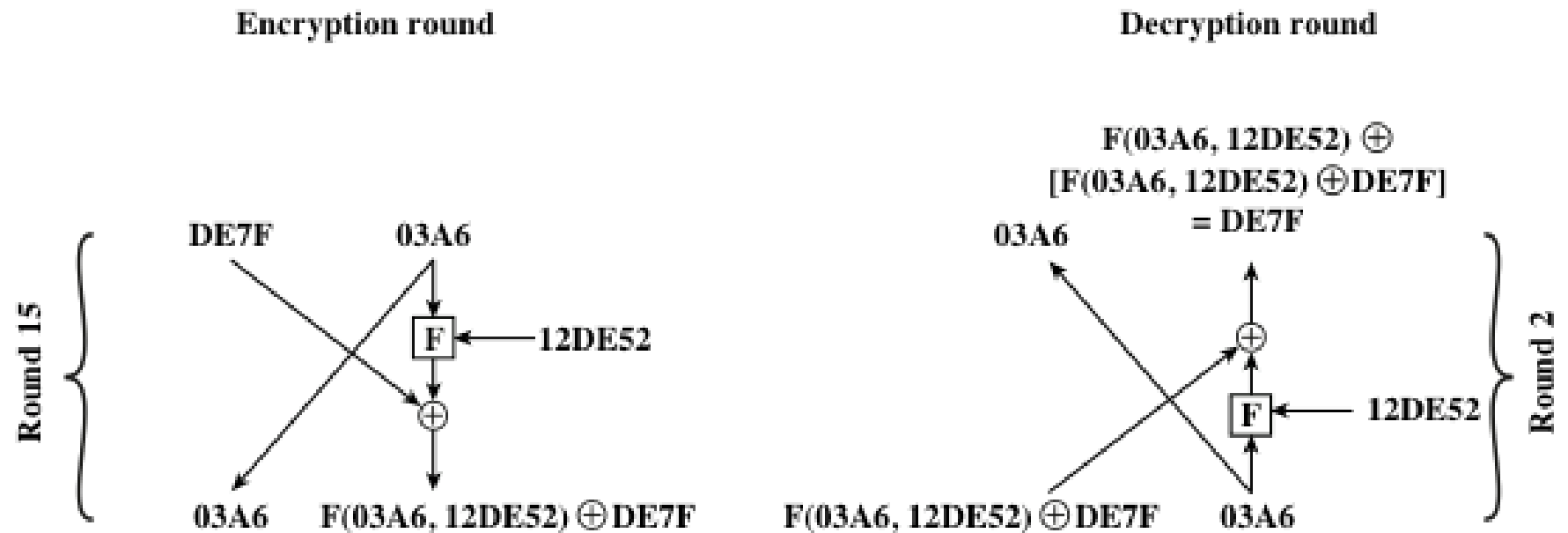


Figure 4.4 Feistel Example

Block Cipher Design Principles: Number of Rounds

The greater the number of rounds, the more difficult it is to perform cryptanalysis

In general, the criterion should be that the number of rounds is chosen so that known cryptanalytic efforts require greater effort than a simple brute-force key search attack

If DES had 15 or fewer rounds, differential cryptanalysis would require less effort than a brute-force key search

Block Cipher Design Principles:

Design of Function F

- The heart of a Feistel block cipher is the function F
- The more nonlinear F, the more difficult any type of cryptanalysis will be
- The SAC and BIC criteria appear to strengthen the effectiveness of the confusion function

The algorithm should have good avalanche properties

Strict avalanche criterion (SAC)

States that any output bit j of an S-box should change with probability $1/2$ when any single input bit i is inverted for all i, j

Bit independence criterion (BIC)

States that output bits j and k should change independently when any single input bit i is inverted for all i, j , and k

Block Cipher Design Principles: Key Schedule Algorithm

- With any Feistel block cipher, the key is used to generate one subkey for each round
- In general, we would like to select subkeys to maximize the difficulty of deducing individual subkeys and the difficulty of working back to the main key
- It is suggested that, at a minimum, the key schedule should guarantee key/ciphertext Strict Avalanche Criterion and Bit Independence Criterion

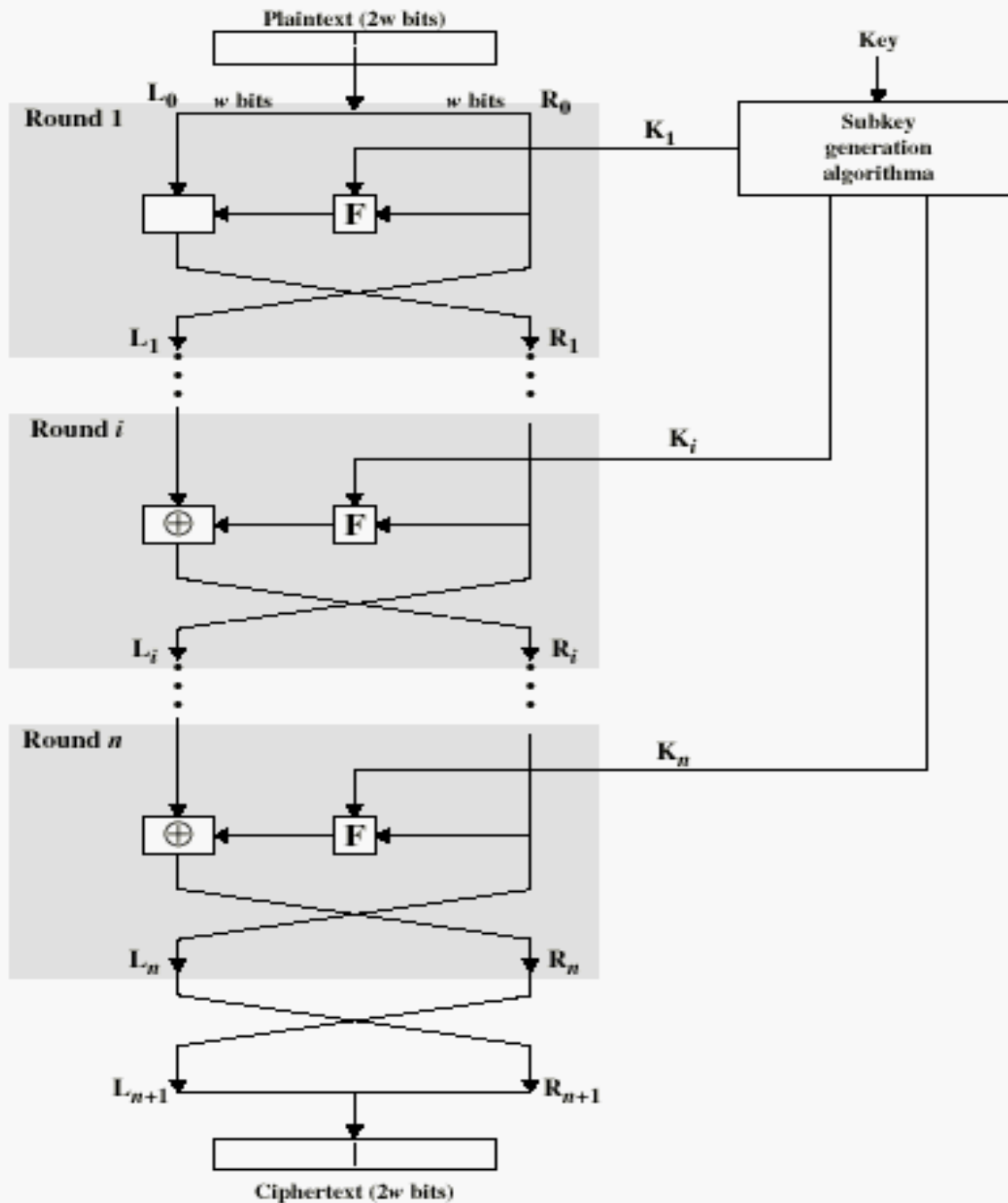


Figure 2.2 Classical Feistel Network

Lecture 3: Number Theory

- Number Theory
- Feistel Cipher Structure
- DES

Encryption Algorithms

- **Data Encryption Standard (DES)**

- The most widely used encryption scheme until early 2000's
- The algorithm is referred to the Data Encryption Algorithm (DEA)
- DES is a block cipher
- The plaintext is processed in 64-bit blocks
- The key is 56-bits in length

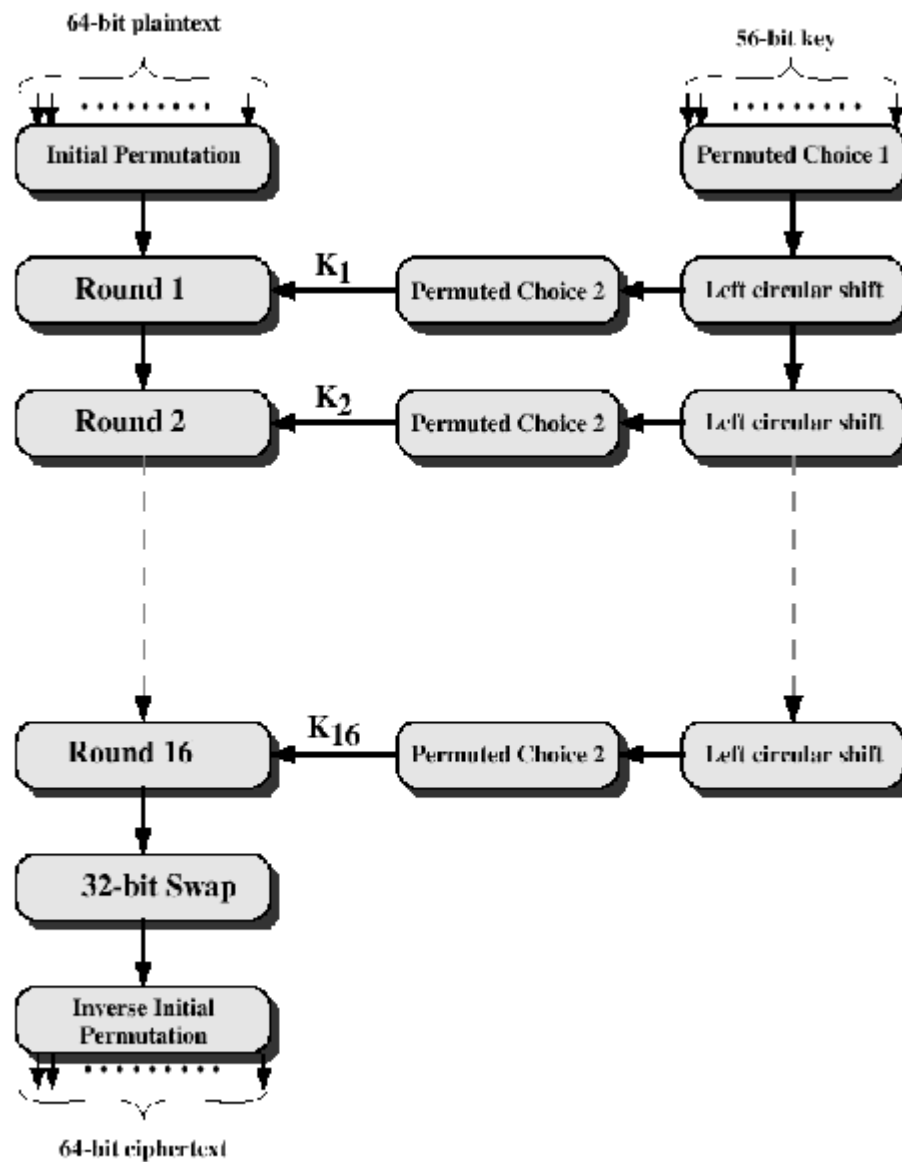


Figure 2.3 General Depiction of DES Encryption Algorithm

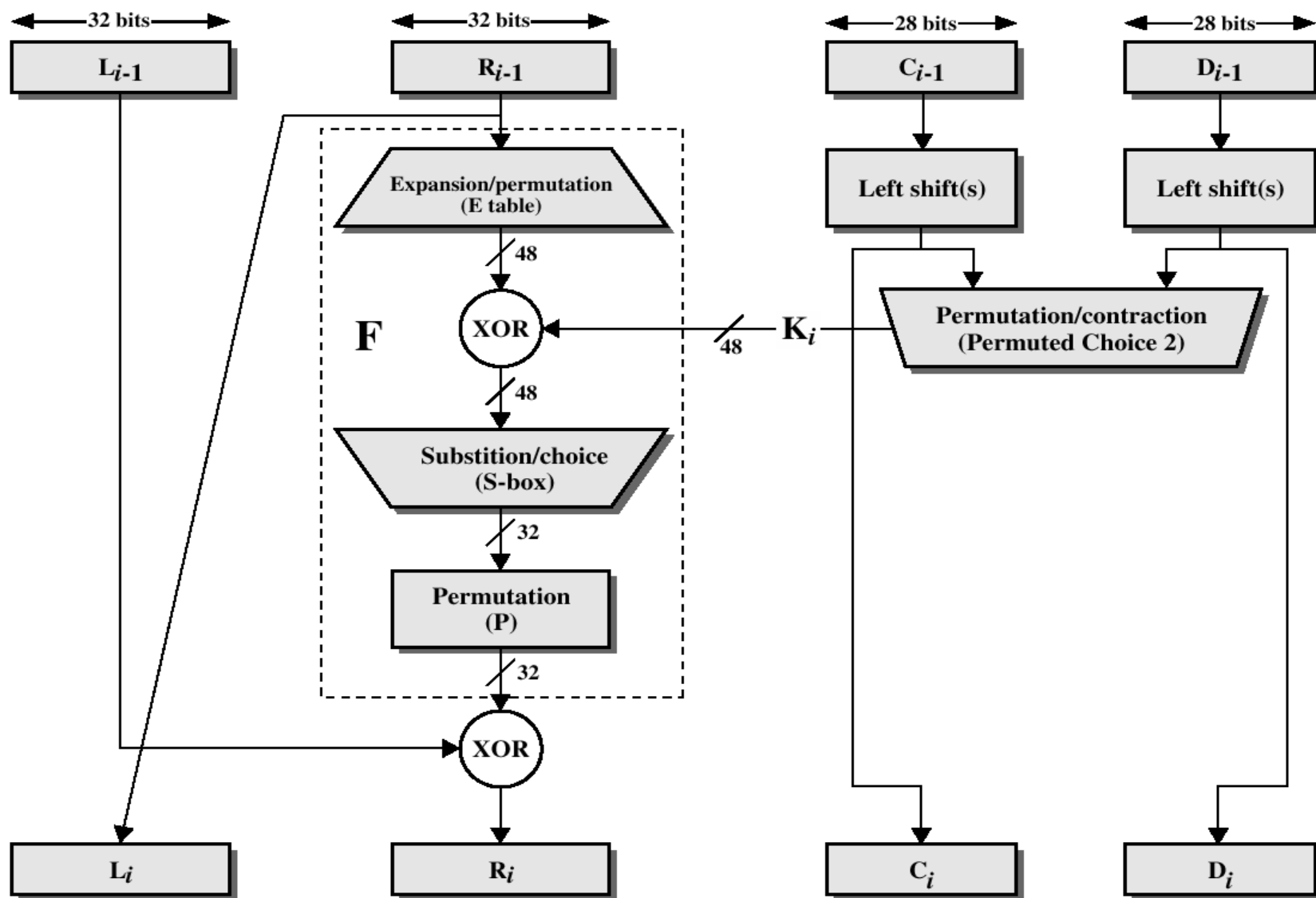


Figure 2.4 Single Round of DES Algorithm

DES

- The overall processing at each iteration:
 - $L_i = R_{i-1}$
 - $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$
- Concerns about:
 - The algorithm and the key length (56-bits)
- Considered broken

Data Encryption Standard (DES)

- Issued in 1977 by the National Bureau of Standards (now NIST) as Federal Information Processing Standard 46
- Was the most widely used encryption scheme until the introduction of the Advanced Encryption Standard (AES) in 2001
- Algorithm itself is referred to as the Data Encryption Algorithm (DEA)
 - Data are encrypted in 64-bit blocks using a 56-bit key
 - The algorithm transforms 64-bit input in a series of steps into a 64-bit output
 - The same steps, with the same key, are used to reverse the encryption

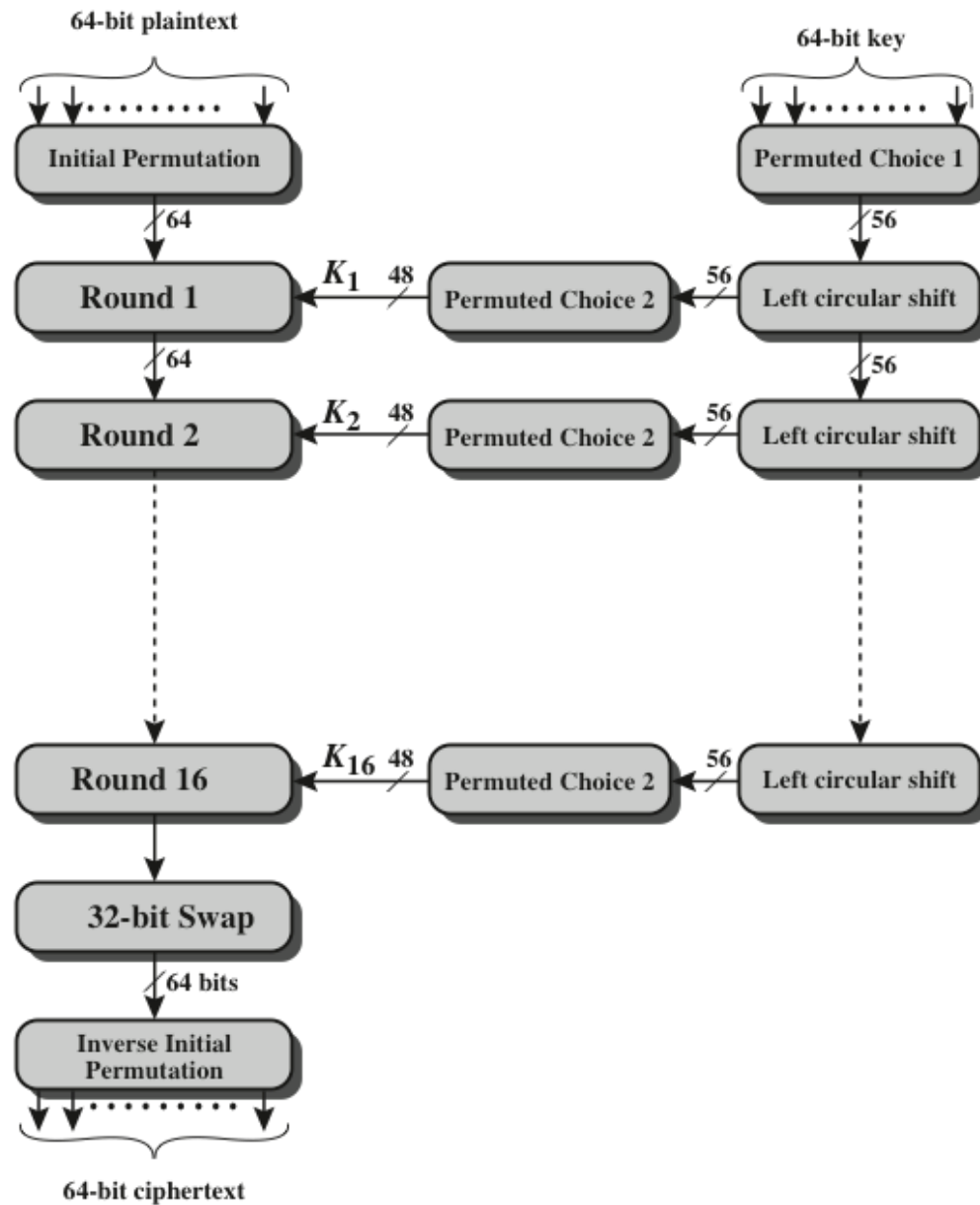


Figure 4.5 General Depiction of DES Encryption Algorithm

DES Example

Round	K_i	L_i	R_i
IP		5a005a00	3cf03c0f
1	1e030f03080d2930	3cf03c0f	bad22845
2	0a31293432242318	bad22845	99e9b723
3	23072318201d0c1d	99e9b723	0bae3b9e
4	05261d3824311a20	0bae3b9e	42415649
5	3325340136002c25	42415649	18b3fa41
6	123a2d0d04262a1c	18b3fa41	9616fe23
7	021f120b1c130611	9616fe23	67117cf2
8	1c10372a2832002b	67117cf2	c11bfc09
9	04292a380c341f03	c11bfc09	887fbc6c
10	2703212607280403	887fbc6c	600f7e8b
11	2826390c31261504	600f7e8b	f596506e
12	12071c241a0a0f08	f596506e	738538b8
13	300935393c0d100b	738538b8	c6a62c4e
14	311e09231321182a	c6a62c4e	56b0bd75
15	283d3e0227072528	56b0bd75	75e8fd8f
16	2921080b13143025	75e8fd8f	25896490
IP-1		da02ce3a	89ecac3b

Note: DES subkeys are shown as eight 6-bit values in hex format

Round		δ
	02468aceeca86420 12468aceeca86420	1
1	3cf03c0fbad22845 3cf03c0fbad32845	1
2	bad2284599e9b723 bad3284539a9b7a3	5
3	99e9b7230bae3b9e 39a9b7a3171cb8b3	18
4	0bae3b9e42415649 171cb8b3ccaca55e	34
5	4241564918b3fa41 ccaca55ed16c3653	37
6	18b3fa419616fe23 d16c3653cf402c68	33
7	9616fe2367117cf2 cf402c682b2cefbcb	32
8	67117cf2c11bfc09 2b2cefbcb99f91153	33

Round		δ
9	c11bfc09887fbc6c 99f911532eed7d94	32
10	887fbc6c600f7e8b 2eed7d94d0f23094	34
11	600f7e8bf596506e d0f23094455da9c4	37
12	f596506e738538b8 455da9c47f6e3cf3	31
13	738538b8c6a62c4e 7f6e3cf34bc1a8d9	29
14	c6a62c4e56b0bd75 4bc1a8d91e07d409	33
15	56b0bd7575e8fd8f 1e07d4091ce2e6dc	31
16	75e8fd8f25896490 1ce2e6dc365e5f59	32
IP-1	da02ce3a89ecac3b 057cde97d7683f2a	32

Avalanche Effect in DES: Change in Plaintext

Round		δ
	02468aceeca86420 02468aceeca86420	0
1	3cf03c0fbad22845 3cf03c0f9ad628c5	3
2	bad2284599e9b723 9ad628c59939136b	11
3	99e9b7230bae3b9e 9939136b768067b7	25
4	0bae3b9e42415649 768067b75a8807c5	29
5	4241564918b3fa41 5a8807c5488dbe94	26
6	18b3fa419616fe23 488dbe94aba7fe53	26
7	9616fe2367117cf2 aba7fe53177d21e4	27
8	67117cf2c11bfc09 177d21e4548f1de4	32

Round		δ
9	c11bfc09887fbc6c 548f1de471f64dfd	34
10	887fbc6c600f7e8b 71f64dfd4279876c	36
11	600f7e8bf596506e 4279876c399fdc0d	32
12	f596506e738538b8 399fdc0d6d208dbb	28
13	738538b8c6a62c4e 6d208dbbb9bdeea	33
14	c6a62c4e56b0bd75 b9bdeeaad2c3a56f	30
15	56b0bd7575e8fd8f d2c3a56f2765c1fb	33
16	75e8fd8f25896490 2765c1fb01263dc4	30
IP-1	da02ce3a89ecac3b ee92b50606b62b0b	30

Avalanche Effect in DES: Change in Key

Average Time Required for Exhaustive Key Search

Key size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 decryptions/s	Time Required at 10^{13} decryptions/s
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	2^{55} ns = 1.125 years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	2^{127} ns = 5.3×10^{21} years	5.3×10^{17} years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	2^{167} ns = 5.8×10^{33} years	5.8×10^{29} years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	2^{191} ns = 9.8×10^{40} years	9.8×10^{36} years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	2^{255} ns = 1.8×10^{60} years	1.8×10^{56} years
26 characters (permutation)	Monoalphabetic	$26! = 4 \times 10^{26}$	2×10^{26} ns = 6.3×10^9 years	6.3×10^6 years

Strength of DES



- Timing attacks

- One in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts
- Exploits the fact that an encryption or decryption algorithm often takes slightly different amounts of time on different inputs
- So far it appears unlikely that this technique will ever be successful against DES or more powerful symmetric ciphers such as triple DES and AES

DES in Java

Encrypting a String with DES

```
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;

class DesEncrypter {
    Cipher ecipher;

    Cipher dcipher;

    DesEncrypter(SecretKey key) throws Exception {
        ecipher = Cipher.getInstance("DES");
        dcipher = Cipher.getInstance("DES");
        ecipher.init(Cipher.ENCRYPT_MODE, key);
        dcipher.init(Cipher.DECRYPT_MODE, key);
    }

    public String encrypt(String str) throws Exception {
        // Encode the string into bytes using utf-8
        byte[] utf8 = str.getBytes("UTF8");

        // Encrypt
        byte[] enc = ecipher.doFinal(utf8);

        // Encode bytes to base64 to get a string
        return new sun.misc.BASE64Encoder().encode(enc);
    }
}
```

```
public String decrypt(String str) throws Exception {
    // Decode base64 to get bytes
    byte[] dec = new sun.misc.BASE64Decoder().decodeBuffer(str);

    byte[] utf8 = dcipher.doFinal(dec);

    // Decode using utf-8
    return new String(utf8, "UTF8");
}

public class Main {
    public static void main(String[] argv) throws Exception {
        SecretKey key = KeyGenerator.getInstance("DES").generateKey();
        DesEncrypter encrypter = new DesEncrypter(key);
        String encrypted = encrypter.encrypt("Don't tell anybody!");
        String decrypted = encrypter.decrypt(encrypted);
    }
}
```

- <http://www.java2s.com/Code/Java/Security/EncryptingaStringwithDES.htm>

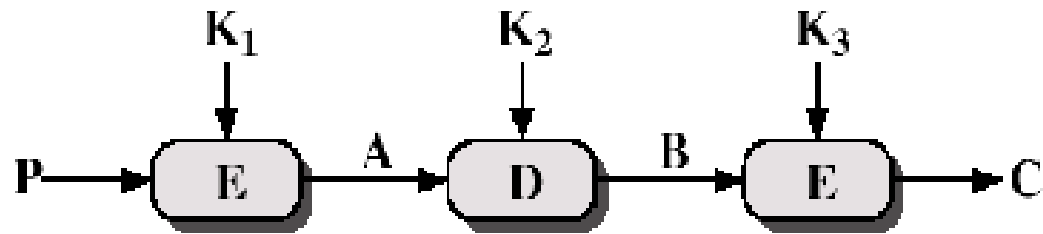
Triple DES

- Use three keys and three executions of the DES algorithm (encrypt-decrypt-encrypt)

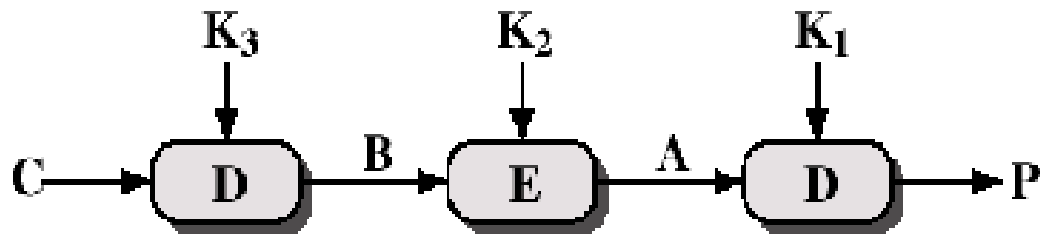
$$C = E_{K3}[D_{K2}[E_{K1}[P]]]$$

- C = ciphertext
 - P = Plaintext
 - $E_K[X]$ = encryption of X using key K
 - $D_K[Y]$ = decryption of Y using key K
-
- Effective key length of 168 bits

Triple DES



(a) Encryption



(b) Decryption

Figure 2.6 Triple DEA

Other Symmetric Block Ciphers

- International Data Encryption Algorithm (IDEA)
 - 128-bit key
 - Used in PGP
- Blowfish
 - Easy to implement
 - High execution speed
 - Run in less than 5K of memory

Other Symmetric Block Ciphers

- RC5
 - Suitable for hardware and software
 - Fast, simple
 - Adaptable to processors of different word lengths
 - Variable number of rounds
 - Variable-length key
 - Low memory requirement
 - High security
 - Data-dependent rotations

Summary

- Divisibility and the division algorithm
- The Euclidean algorithm
 - Greatest Common Divisor
 - Finding the Greatest Common Divisor
- Modular arithmetic
 - The modulus
 - Properties of congruence
 - Modular arithmetic operations
 - Properties of modular arithmetic
 - Euclidean algorithm revisited
 - The extended Euclidean algorithm
- Prime numbers



- Fermat's Theorem
- Euler's totient function
- Euler's Theorem
- Testing for primality
 - Miller-Rabin algorithm
 - A deterministic primality algorithm
 - Distribution of primes
- The Chinese Remainder Theorem
- Discrete logarithms
 - Powers of an integer, modulo n
 - Logarithms for modular arithmetic
 - Calculation of discrete logarithms

Summary

- Traditional Block Cipher Structure

- Stream ciphers
- Block ciphers
- Motivation for the Feistel cipher structure
- Feistel cipher

- The Data Encryption Standard (DES)

- Encryption
- Decryption
- Avalanche effect



- The strength of DES

- Use of 56-bit keys
- Nature of the DES algorithm
- Timing attacks

- Block cipher design principles

- Number of rounds
- Design of function F
- Key schedule algorithm

Lecture 3: What you have learnt?

- Feistel Cipher Structure
- DES
- Number Theory

Your mission



- Understand classical cryptography
- Build up the interests
- Get ready for upcoming Homework #1

