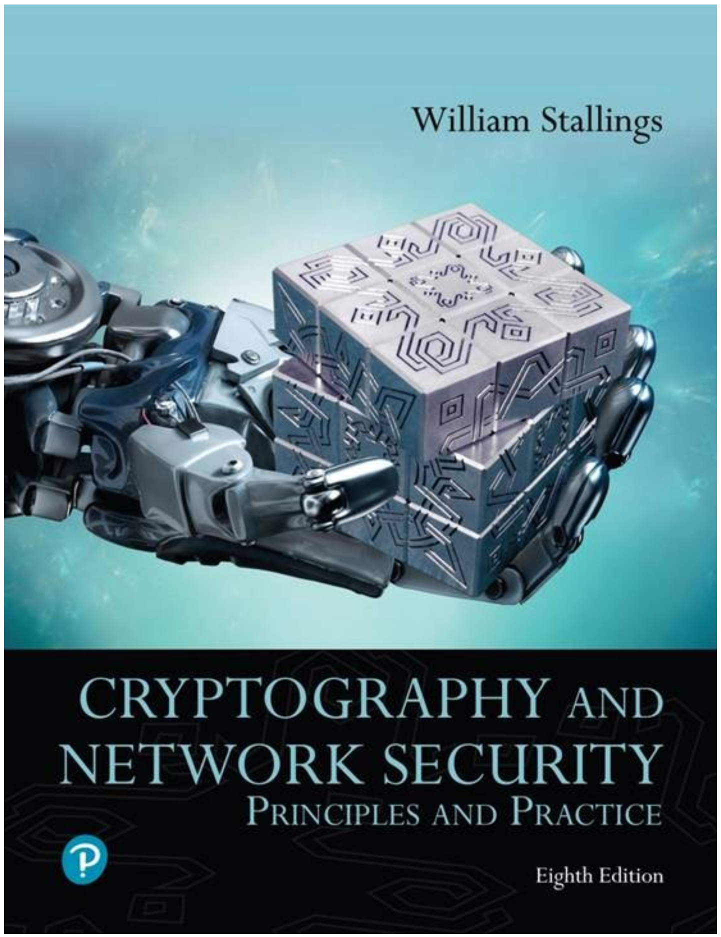
Cryptography and Network Security: Principles and Practice

Eighth Edition



Chapter 2

Introduction to Number Theory

Pearson

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

$$a^{p} = a \pmod{p}$$

Euler's Phi-Function

Euler's phi-function, ϕ (n), which is sometimes called the Euler's totient function plays a very important role in cryptography.

ϕ (n) denotes the number of integers that are both smaller than n and relatively prime to n.

1. $\phi(1) = 0$.
2. $\phi(p) = p - 1$ if p is a prime.
3. $\phi(m \times n) = \phi(m) \times \phi(n)$ if m and n are relatively prime.
4. $\phi(p^e) = p^e - p^{e-1}$ if p is a prime.

Continued

We can combine the above four rules to find the value of $\phi(n)$. For example, if n can be factored as

$$n = p_1^{e_1} \times p_2^{e_2} \times \ldots \times p_k^{e_k}$$

then we combine the third and the fourth rule to find

$$\boxed{\phi(n) = (p_1^{e_1} - p_1^{e_1-1}) \times (p_2^{e_2} - p_2^{e_2-1}) \times \cdots \times (p_k^{e_k} - p_k^{e_k-1})}$$

Note

The difficulty of finding $\phi(n)$ depends on the difficulty of finding the factorization of *n*.

Continued

Example 9.7

What is the value of $\phi(13)$?

Solution

Because 13 is a prime, $\phi(13) = (13 - 1) = 12$.

Example 9.8

What is the value of $\phi(10)$?

Solution

We can use the third rule: $\phi(10) = \phi(2) \times \phi(5) = 1 \times 4 = 4$, because 2 and 5 are primes.

Continued

Example 9.9

What is the value of $\phi(240)$?

Solution

We can write $240 = 2^4 \times 3^1 \times 5^1$. Then

$$\phi(240) = (2^4 - 2^3) \times (3^1 - 3^0) \times (5^1 - 5^0) = 64$$

Example 9.10

Can we say that $\phi(49) = \phi(7) \times \phi(7) = 6 \times 6 = 36$?

Solution

No. The third rule applies when m and n are relatively prime. Here $49 = 7^2$. We need to use the fourth rule: $\phi(49) = 7^2 - 7^1 = 42$.

Continued

Example 9.11

What is the number of elements in Z_{14}*?

Solution

The answer is $\phi(14) = \phi(7) \times \phi(2) = 6 \times 1 = 6$. The members are 1, 3, 5, 9, 11, and 13.

Note

Interesting point: If $n > 2$, the value of $\phi(n)$ is even.

Table 2.6 Some Values of Euler's Totient Function $ø(n)$

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

Euler's Theorem

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

$$a^{\text{ø}(n)} = 1(\text{mod } n)$$

- An alternate form is:

$$a^{\text{ø}(n)+1} = a(\text{mod } n)$$

Euler's Theorem

If a and n are co-prime (relatively prime)

First Version

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

Second Version

$$\boxed{a^{k \times \phi(n) + 1} \equiv a \pmod{n}}$$

Note

The second version of Euler's theorem is used in the RSA cryptosystem in Chapter 10.

10.2.2 Continued

Two Algebraic Structures

Encryption/Decryption Ring: $R = \langle Z_n, +, \times \rangle$

Key-Generation Group: $G = \langle Z_{\phi(n)}^*, \times \rangle$

RSA uses two algebraic structures:
a public ring $R = \langle Z_n, +, \times \rangle$ and a private group $G = \langle Z_{\phi(n)}^*, \times \rangle$.

In RSA, the tuple (e, n) is the public key; the integer d is the private key.

10.2.2 Continued

Algorithm 10.2 *RSA Key Generation*

RSA_Key_Generation

{

Select two large primes p and q such that $p \neq q$.

$n \leftarrow p \times q$

$\phi(n) \leftarrow (p-1) \times (q-1)$

Select e such that $1 < e < \phi(n)$ and e is coprime to $\phi(n)$

$d \leftarrow e^{-1} \bmod \phi(n)$ // d is inverse of e modulo $\phi(n)$

Public_key $\leftarrow$ (e, n) // To be announced publicly

Private_key $\leftarrow d$ // To be kept secret

return Public_key and Private_key

}

Key Generation by Alice	
Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption by Bob with Alice's Public Key	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption by Alice with Alice's Private Key	
Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

Figure 9.5 The RSA Algorithm

10.2.2 Continued

Proof of RSA

If $n = p \times q$, $a < n$, and k is an integer, then $a^{k \times \phi(n)+1} \equiv a \pmod{n}$.

$P_1 = C^d \bmod n = (P^e \bmod n)^d \bmod n = P^{ed} \bmod n$

$ed = k\phi(n) + 1$ // d and e are inverses modulo $\phi(n)$

$P_1 = P^{ed} \bmod n \rightarrow P_1 = P^{k\phi(n)+1} \bmod n$

$P_1 = P^{k\phi(n)+1} \bmod n = P \bmod n$ // Euler's theorem (second version)

PRIMALITY TESTING

Finding an algorithm to correctly and efficiently test a very large integer and output a prime or a composite has always been a challenge in number theory, and consequently in cryptography. However, recent developments look very promising.

<u>Topics discussed in this section:</u>

Deterministic Algorithms
Probabilistic Algorithms
Recommended Primality Test

Deterministic Algorithms

Divisibility Algorithm

Algorithm 9.1 *Pseudocode for the divisibility test*

```
Divisibility_Test (n)                    // n is the number to test for primality
{
  r ← 2
  while (r < √n )
  {
   if (r | n) return "a composite"
   r ← r + 1
  }
  return "a prime"
}
```

Note

The bit-operation complexity of the divisibility test is exponential.

Continued

Example 9.18

Assume *n* has 200 bits. What is the number of bit operations needed to run the divisibility-test algorithm?

Solution

The bit-operation complexity of this algorithm is $2^{n_b/2}$. This means that the algorithm needs 2^{100} bit operations. On a computer capable of doing 2^{30} bit operations per second, the algorithm needs 2^{70} seconds to do the testing (forever).

Probabilistic Algorithms

Fermat Test

$\boxed{\text{If } n \text{ is a prime, then } a^{n-1} \equiv 1 \bmod n.}$

If n is a prime, $a^{n-1} \equiv 1 \bmod n$
If n is a composite, it is possible that $a^{n-1} \equiv 1 \bmod n$

Example 9.20

Does the number 561 pass the Fermat test?

Solution

Use base 2

The number passes the Fermat test, but it is not a prime, because 561 = 33 × 17.

Continued

Square Root Test

If n is a prime, $\sqrt{1} \bmod n = \pm 1$.
If n is a composite, $\sqrt{1} \bmod n = \pm 1$ and possibly other values.

Example 9.21

What are the square roots of 1 mod *n* if *n* is 7 (a prime)?

Solution

The only square roots are 1 and −1. We can see that

$$
\begin{array}{ll}
1^2 = 1 \bmod 7 & (-1)^2 = 1 \bmod 7 \\
2^2 = 4 \bmod 7 & (-2)^2 = 4 \bmod 7 \\
3^2 = 2 \bmod 7 & (-3)^2 = 2 \bmod 7
\end{array}
$$

Continued

Example 9.22

What are the square roots of 1 mod n if n is 8 (a composite)?

Solution

There are four solutions: 1, 3, 5, and 7 (which is −1). We can see that

$$
\begin{array}{ll}
1^2 = 1 \bmod 8 & (-1)^2 = 1 \bmod 8 \\
3^2 = 1 \bmod 8 & 5^2 \quad = 1 \bmod 8
\end{array}
$$

Continued

Example 9.23

What are the square roots of 1 mod *n* if *n* is 17 (a prime)?

Solution

There are only two solutions: 1 and −1

$$
\begin{array}{ll}
1^2 = 1 \bmod 17 & (-1)^2 = 1 \bmod 17 \\
2^2 = 4 \bmod 17 & (-2)^2 = 4 \bmod 17 \\
3^2 = 9 \bmod 17 & (-3)^2 = 9 \bmod 17 \\
4^2 = 16 \bmod 17 & (-4)^2 = 16 \bmod 17 \\
5^2 = 8 \bmod 17 & (-5)^2 = 8 \bmod 17 \\
6^2 = 2 \bmod 17 & (-6)^2 = 2 \bmod 17 \\
(7)^2 = 15 \bmod 17 & (-7)^2 = 15 \bmod 17 \\
(8)^2 = 13 \bmod 17 & (-8)^2 = 13 \bmod 17
\end{array}
$$

Continued

Example 9.24

What are the square roots of 1 mod n if n is 22 (a composite)?

Solution

Surprisingly, there are only two solutions, +1 and −1, although 22 is a composite.

$$1^2 = 1 \bmod 22$$
$$(-1)^2 = 1 \bmod 22$$

Continued

Miller-Rabin Test

$$n - 1 = m \times 2^k$$

Figure 9.2 ***Idea behind Fermat primality test***

$$a^{n-1} = a^{m \times 2^k} = [a^m]^{2^k} = [a^m]^{\underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}_{k \text{ times}}}$$

Note

The Miller-Rabin test needs from step 0 to step $k - 1$.

Miller-Rabin Algorithm

- Typically used to test a large number for primality
- Algorithm is:

  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") ;

Continued

Algorithm 9.2 *Pseudocode for Miller-Rabin test*

```
Miller_Rabin_Test (n, a)                          // n is the number; a is the base.
{
    Find m and k such that n − 1 = m × 2^k
    T ← a^m mod n
    if (T = ± 1)   return "a prime"
    for (i ← 1 to k − 1)                          // k − 1 is the maximum number of steps.
    {
        T ← T^2 mod n
        if (T = +1) return "a composite"
        if (T = −1) return "a prime"
    }
    return "a composite"
}
```

Continued

Example 9.25

Does the number 561 pass the Miller-Rabin test?

Solution

Using base 2, let $561 - 1 = 35 \times 2^4$, which means $m = 35$, $k = 4$, and $a = 2$.

Initialization:	$T = 2^{35} \bmod 561 = 263 \bmod 561$		
$k = 1$:	$T = 263^2 \bmod 561 = 166 \bmod 561$		
$k = 2$:	$T = 166^2 \bmod 561 = 67 \bmod 561$		
$k = 3$:	$T = 67^2 \bmod 561 = +1 \bmod 561$	$\rightarrow$	**a composite**

Continued

Example 9.26

We already know that 27 is not a prime. Let us apply the Miller-Rabin test.

Solution

With base 2, let $27 - 1 = 13 \times 2^1$, which means that $m = 13$, $k = 1$, and $a = 2$. In this case, because $k - 1 = 0$, we should do only the initialization step: $T = 2^{13} \bmod 27 = 11 \bmod 27$. However, because the algorithm never enters the loop, it returns a composite.

Continued

Example 9.27

We know that 61 is a prime, let us see if it passes the Miller-Rabin test.

Solution

We use base 2.

$$
\begin{array}{lll}
61 - 1 = 15 \times 2^2 \;\rightarrow\; m = 15 \quad k = 2 \quad a = 2 & & \\
\textit{Initialization:} & T = 2^{15} \bmod 61 = 11 \bmod 61 & \\
k = 1 & T = 11^2 \bmod 61 = -1 \bmod 61 & \rightarrow \textbf{a prime}
\end{array}
$$

Today, one of the most popular primality test is a combination of the divisibility test and the Miller-Rabin test.

Continued

Example 9.28

The number 4033 is a composite (37 × 109). Does it pass the recommended primality test?

Solution

1. Perform the divisibility tests first. The numbers 2, 3, 5, 7, 11, 17, and 23 are not divisors of 4033.
2. Perform the Miller-Rabin test with a base of 2, $4033 - 1 = 63 \times 2^6$, which means m is 63 and k is 6.

Initialization: $T \equiv 2^{63} \pmod{4033} \equiv 3521 \pmod{4033}$		
$k = 1$	$T \equiv T^2 \equiv 3521^2 \pmod{4033} \equiv -1 \pmod{4033}$	$\rightarrow$ **Passes**

Continued

Example 9.28 Continued

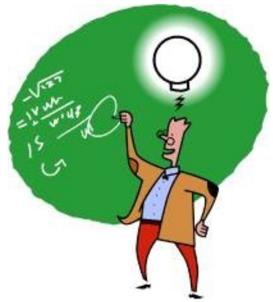
3. **But we are not satisfied. We continue with another base, 3.**

Initialization: $T \equiv 3^{63} \pmod{4033} \equiv 3551 \pmod{4033}$

$k = 1$	$T \equiv T^2 \equiv 3551^2$ (mod 4033 $\equiv 2443 \pmod{4033}$
$k = 2$	$T \equiv T^2 \equiv 2443^2$ (mod 4033 $\equiv 3442 \pmod{4033}$
$k = 3$	$T \equiv T^2 \equiv 3442^2$ (mod 4033 $\equiv 2443 \pmod{4033}$
$k = 4$	$T \equiv T^2 \equiv 2443^2$ (mod 4033 $\equiv 3442 \pmod{4033}$
$k = 5$	$T \equiv T^2 \equiv 3442^2$ (mod 4033 $\equiv 2443 \pmod{4033}$ $\rightarrow$ **Failed (composite)**

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



CHINESE REMAINDER THEOREM

The Chinese remainder theorem (CRT) is used to solve a set of congruent equations with one variable but different moduli, which are relatively prime, as shown below:

$$x \equiv a_1 \ (\text{mod } m_1)$$
$$x \equiv a_2 \ (\text{mod } m_2)$$
$$\cdots$$
$$x \equiv a_k \ (\text{mod } m_k)$$

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

Continued

Solution To Chinese Remainder Theorem

1. Find $M = m_1 \times m_2 \times \ldots \times m_k$. This is the common modulus.
2. Find $M_1 = M/m_1$, $M_2 = M/m_2$, …, $M_k = M/m_k$.
3. Find the multiplicative inverse of M_1, M_2, …, M_k using the corresponding moduli (m_1, m_2, …, m_k). Call the inverses M_1^{-1}, M_2^{-1}, …, M_k^{-1}.
4. The solution to the simultaneous equations is

$$x = (a_1 \times M_1 \times M_1^{-1} + a_2 \times M_2 \times M_2^{-1} + \cdots + a_k \times M_k \times M_k^{-1}) \bmod M$$

Continued

Example 9.36

Find the solution to the simultaneous equations:

$$x \equiv 2 \pmod 3$$
$$x \equiv 3 \pmod 5$$
$$x \equiv 2 \pmod 7$$

Solution

We follow the four steps.

1. $M = 3 \times 5 \times 7 = 105$

2. $M_1 = 105 / 3 = 35$, $M_2 = 105 / 5 = 21$, $M_3 = 105 / 7 = 15$

3. The inverses are $M_1^{-1} = 2 \bmod 3$,
   $M_2^{-1} = 1 \bmod 5$,
   $M_3^{-1} = 1 \bmod 7$

4. $x = (2 \times 35 \times 2 + 3 \times 21 \times 1 + 2 \times 15 \times 1) \bmod 105 = 23 \bmod 105$

Continued

Example 9.37

Find an integer that has a remainder of 3 when divided by 7 and 13, but is divisible by 12.

Solution

This is a CRT problem. We can form three equations and solve them to find the value of x.

$$x = 3 \bmod 7$$
$$x = 3 \bmod 13$$
$$x = 0 \bmod 12$$

If we follow the four steps, we find x = 276. We can check that 276 = 3 mod 7, 276 = 3 mod 13 and 276 is divisible by 12 (the quotient is 23 and the remainder is zero).

Continued

Example 9.38

Assume we need to calculate $z = x + y$ where $x = 123$ and $y = 334$, but our system accepts only numbers less than 100. These numbers can be represented as follows:

$$
\begin{array}{ll}
x \equiv 24 \pmod{99} & y \equiv 37 \pmod{99} \\
x \equiv 25 \pmod{98} & y \equiv 40 \pmod{98} \\
x \equiv 26 \pmod{97} & y \equiv 43 \pmod{97}
\end{array}
$$

Adding each congruence in x with the corresponding congruence in y gives

$$
\begin{array}{lcl}
x + y \equiv 61 \pmod{99} & \rightarrow & z \equiv 61 \pmod{99} \\
x + y \equiv 65 \pmod{98} & \rightarrow & z \equiv 65 \pmod{98} \\
x + y \equiv 69 \pmod{97} & \rightarrow & z \equiv 69 \pmod{97}
\end{array}
$$

Now three equations can be solved using the Chinese remainder theorem to find z. One of the acceptable answers is $z = 457$.

EXPONENTIATION AND LOGARITHM

Exponentiation: $y = a^x$ $\rightarrow$ **Logarithm:** $x = \log_a y$

<u>Topics discussed in this section:</u>

9.6.1 Exponentiation
9.6.2 Logarithm

Exponentiation

Fast Exponentiation

Figure 9.6 ***The idea behind the square-and-multiply method***

$$y = a^{\boxed{x_{n_b-1} \times 2^{n_b-1} + x_{n_b-2} \times 2^{n_b-2} + \cdots + x_1 \times 2^1 + x_0 \times 2^0}}$$

in which x_i is 0 or 1

$$\downarrow$$

$$y = \boxed{\boxed{a^{2^{n_b-1}}} \text{ or } \boxed{1}} \times \boxed{\boxed{a^{2^{n_b-2}}} \text{ or } \boxed{1}} \times \cdots \times \boxed{\boxed{a^2} \text{ or } \boxed{1}} \times \boxed{\boxed{a} \text{ or } \boxed{1}}$$

Example:

$$\boxed{y = a^9 = a^{1001_2} = a^8 \times 1 \times 1 \times a}$$

Continued

Algorithm 9.7 *Pseudocode for square-and-multiply algorithm*

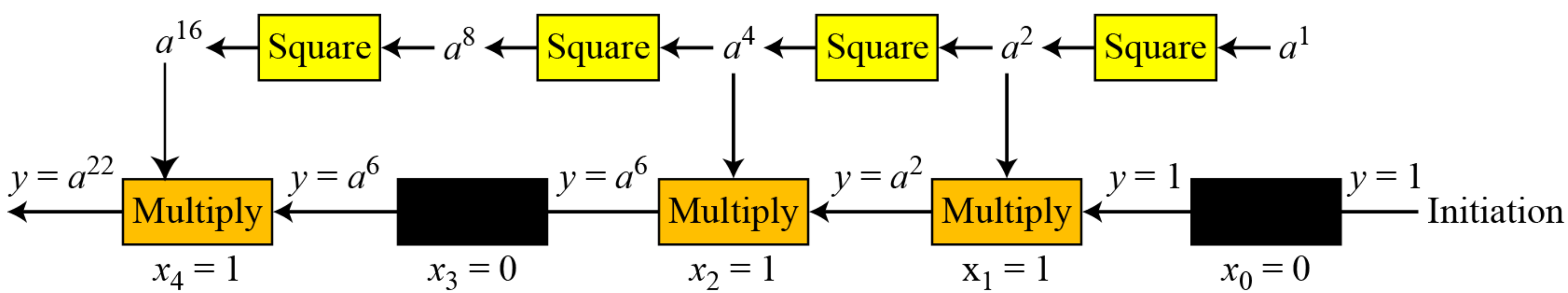
```
Square_and_Multiply (a, x, n)
{
    y ← 1
    for (i ← 0 to n_b − 1)                     // n_b is the number of bits in x
    {
        if (x_i = 1)   y ← a × y  mod n        // multiply only if the bit is 1

        a ← a^2 mod n                          // squaring is not needed in the last iteration
    }
    return y
}
```

Continued

Example 9.45

Figure 9.7 shows the process for calculating $y = a^x$ using the Algorithm 9.7 (for simplicity, the modulus is not shown). In this case, $x = 22 = (10110)2$ in binary. The exponent has five bits.

Figure 9.7 ***Demonstration of calculation of a^{22} using square-and-multiply method***



Continued

Table 9.3 *Calculation of 17^{22} mod 21*

i	x_i	Multiplication (Initialization: $y = 1$)		Squaring (Initialization: $a = 17$)
0	0		$\rightarrow$	$a = 17^2 \bmod 21 = 16$
1	1	$y = 1 \times 16 \bmod 21 = 16$	$\rightarrow$	$a = 16^2 \bmod 21 = 4$
2	1	$y = 16 \times 4 \bmod 21 = 1$	$\rightarrow$	$a = 4^2 \bmod 21 = 16$
3	0		$\rightarrow$	$a = 16^2 \bmod 21 = 4$
4	1	$y = 1 \times 4 \bmod 21 = 4$	$\rightarrow$	

Note

The bit-operation complexity of the fast exponential algorithm is polynomial.

Logarithm

In cryptography, we also need to discuss modular logarithm.

Exhaustive Search

Algorithm 9.8 *Exhaustive search for modular logarithm*

```
Modular_Logarithm (a, y, n)
{
    for (x = 1 to n −1)                              // k is the number of bits in x
    {
        if (y ≡ a^x mod n) return x
    }
    return failure
}
```

Continued

Order of the Group.

Example 9.46

What is the order of group $G = \langle Z_{21}*, \times\rangle$? $|G| = \phi(21) = \phi(3) \times \phi(7) = 2 \times 6 = 12$. There are 12 elements in this group: 1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, and 20. All are relatively prime with 21.

Continued

Order of an Element

Example 9.47

Find the order of all elements in $G = \langle Z_{10}*, \times \rangle$.

Solution

This group has only $\phi(10) = 4$ elements: 1, 3, 7, 9. We can find the order of each element by trial and error.

a. $1^1 \equiv 1 \bmod (10) \rightarrow \text{ord}(1) = 1$.

b. $3^4 \equiv 1 \bmod (10) \rightarrow \text{ord}(3) = 4$.

c. $7^4 \equiv 1 \bmod (10) \rightarrow \text{ord}(7) = 4$.

d. $9^2 \equiv 1 \bmod (10) \rightarrow \text{ord}(9) = 2$.

Euler's Theorem

Example 9.48

Table 9.4 *Finding the orders of elements in Example 9.48*

	$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$	$i = 6$	$i = 7$
$a = 1$	**x: 1**	x: 1	x: 1	x: 1	x: 1	x: 1	x: 1
$a = 3$	x: 3	**x: 1**	x: 3	x: 1	x: 3	x: 1	x: 3
$a = 5$	x: 5	**x: 1**	x: 5	x: 1	x: 5	x: 1	x: 5
$a = 7$	x: 7	**x: 1**	x: 7	x: 1	x: 7	x: 1	x: 7

Continued

Primitive Roots ***In the group $G = <Z_n*, \times>$, when the order of an element is the same as $\phi(n)$, that element is called the primitive root of the group.***

Example 9.49

Table 9.4 shows that there are no primitive roots in $G = <Z_8*, \times>$ because no element has the order equal to $\phi(8) = 4$. The order of elements are all smaller than 4.

Continued

Example 9.50

Table 9.5 shows the result of $a^i \equiv x \pmod 7$ for the group $G = \langle Z_7{*}, \times \rangle$. In this group, $\phi(7) = 6$.

Table 9.5 *Example 9.50*

		$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$	$i = 6$
	$a = 1$	$\boxed{x: 1}$	$x: 1$	$x: 1$	$x: 1$	$x: 1$	$x: 1$
	$a = 2$	$x: 2$	$x: 4$	$\boxed{x: 1}$	$x: 2$	$x: 4$	$x: 1$
Primitive root →	$a = 3$	$x: 3$	$x: 2$	$x: 6$	$x: 4$	$x: 5$	$\boxed{x: 1}$
	$a = 4$	$x: 4$	$x: 2$	$\boxed{x: 1}$	$x: 4$	$x: 2$	$x: 1$
Primitive root →	$a = 5$	$x: 5$	$x: 4$	$x: 6$	$x: 2$	$x: 3$	$\boxed{x: 1}$
	$a = 6$	$x: 6$	$\boxed{x: 1}$	$x: 6$	$x: 1$	$x: 6$	$x: 1$

Table 2.7 Powers of Integers, Modulo 19

a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1
6	17	7	4	5	11	9	16	1	6	17	7	4	5	11	9	16	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
8	7	18	11	12	1	8	7	18	11	12	1	8	7	18	11	12	1
9	5	7	6	16	11	4	17	1	9	5	7	6	16	11	4	17	1
10	5	12	6	3	11	15	17	18	9	14	7	13	16	8	4	2	1
11	7	1	11	7	1	11	7	1	11	7	1	11	7	1	11	7	1
12	11	18	7	8	1	12	11	18	7	8	1	12	11	18	7	8	1
13	17	12	4	14	11	10	16	18	6	2	7	15	5	8	9	3	1
14	6	8	17	10	7	3	4	18	5	13	11	2	9	12	16	15	1
15	16	12	9	2	11	13	5	18	4	3	7	10	17	8	6	14	1
16	9	11	5	4	7	17	6	1	16	9	11	5	4	7	17	6	1
17	4	11	16	6	7	5	9	1	17	4	11	16	6	7	5	9	1
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

Table 2.8 Tables of Discrete Logarithms, Modulo 19 (1 of 2)

(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

Continued

Note

The group $G = <Z_n^*, \times>$ has primitive roots only if n is 2, 4, p^t, or $2p^t$.

Example 9.51

For which value of n, does the group $G = <Z_n*, \times>$ have primitive roots: 17, 20, 38, and 50?

Solution

a. $G = <Z_{17}*, \times>$ has primitive roots, 17 is a prime.

b. $G = <Z_{20}*, \times>$ has no primitive roots.

c. $G = <Z_{38}*, \times>$ has primitive roots, $38 = 2 \times 19$ prime.

d. $G = <Z_{50}*, \times>$ has primitive roots, $50 = 2 \times 5^2$ and 5 is a prime.

Continued

Note

If the group G = <Z_n*, ×> has any primitive root, the number of primitive roots is $\phi(\phi(n))$.

Continued

Cyclic Group ***If g is a primitive root in the group, we can generate the set Z_n* as $Z_n* = \{g^1, g^2, g^3, \ldots, g^{\phi(n)}\}$***

Example 9.52

The group G = <Z_{10}*, ×> has two primitive roots because $\phi(10) = 4$ and $\phi(\phi(10)) = 2$. It can be found that the primitive roots are 3 and 7. The following shows how we can create the whole set Z_{10}* using each primitive root.

$g = 3 \rightarrow$	$g^1 \bmod 10 = 3$	$g^2 \bmod 10 = 9$	$g^3 \bmod 10 = 7$	$g^4 \bmod 10 = 1$
$g = 7 \rightarrow$	$g^1 \bmod 10 = 7$	$g^2 \bmod 10 = 9$	$g^3 \bmod 10 = 3$	$g^4 \bmod 10 = 1$

The group G = <Z_n*, ×> is a cyclic group if it has primitive roots.
The group G = <Z_p*, ×> is always cyclic.

Continued

The idea of Discrete Logarithm

Properties of $G = \langle Z_p{}^*, \times \rangle$:

1. Its elements include all integers from 1 to $p - 1$.

2. It always has primitive roots.

3. It is cyclic. The elements can be created using g^x where x is an integer from 1 to $\phi(p) = p - 1$.

4. The primitive roots can be thought as the base of logarithm.

Continued

Solution to Modular Logarithm Using Discrete Logs

Tabulation of Discrete Logarithms

Table 9.6 *Discrete logarithm for* $\mathbf{G} = \langle \mathbf{Z}_7{}^*, \times \rangle$

y	1	2	3	4	5	6
$x = L_3\, y$	6	2	1	4	5	3
$x = L_5\, y$	6	4	5	2	1	3

Table 2.8 Tables of Discrete Logarithms, Modulo 19 (2 of 2)

(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

Continued

Example 9.53

Find x in each of the following cases:

a. $4 \equiv 3^x \pmod 7$.

b. $6 \equiv 5^x \pmod 7$.

Solution

We can easily use the tabulation of the discrete logarithm in Table 9.6.

a. $4 \equiv 3^x \bmod 7 \rightarrow x = L_3 4 \bmod 7 = 4 \bmod 7$

b. $6 \equiv 5^x \bmod 7 \rightarrow x = L_5 6 \bmod 7 = 3 \bmod 7$

Continued

Using Properties of Discrete Logarithms

Table 9.7 *Comparison of traditional and discrete logarithms*

Traditional Logarithm	*Discrete Logarithms*
$\log_a 1 = 0$	$L_g 1 \equiv 0 \pmod{\phi(n)}$
$\log_a (x \times y) = \log_a x + \log_a y$	$L_g(x \times y) \equiv (L_g x + L_g y) \pmod{\phi(n)}$
$\log_a x^k = k \times \log_a x$	$L_g x^k \equiv k \times L_g x \pmod{\phi(n)}$

Using Algorithms Based on Discrete

Note

The discrete logarithm problem has the same complexity as the factorization problem.

Summary

- Understand the concept of divisibility and the division algorithm
- Understand how to use the Euclidean algorithm to find the greatest common divisor
- Present an overview of the concepts of modular arithmetic
- Explain the operation of the extended Euclidean algorithm
- Discuss key concepts relating to prime numbers
- Understand Fermat's theorem
- Understand Euler's theorem
- Define Euler's totient function
- Make a presentation on the topic of testing for primality
- Explain the Chinese remainder theorem
- Define discrete logarithms

Copyright

This work is protected by United States copyright laws and is provided solely for the use of instructors in teaching their courses and assessing student learning. Dissemination or sale of any part of this work (including on the World Wide Web) will destroy the integrity of the work and is not permitted. The work and materials from it should never be made available to students except by instructors using the accompanying text in their classes. All recipients of this work are expected to abide by these restrictions and to honor the intended pedagogical purposes and the needs of other instructors who rely on these materials.