

Cryptography CS 555

Lecture 11



Department of Computer Sciences
Purdue University

Lecture Outline

- Number theory:
 - Extended Euclidian Algorithm
 - Euclid's Theorem
 - Chinese remainder theorem
- Public-key encryption
- RSA

Euclidian Algorithm (Review)

Euclidian Algorithm

gcd (b, a)

d=a; t=b

while t \neq 0 {

w \leftarrow d mod t

d \leftarrow t

t \leftarrow w

}

return d

Towards Extended Euclidian Algorithm

- **Theorem:** Given integers $a, b > 0$ and $a > b$, then $d = \gcd(a,b)$ is the least positive integer that can be represented as $ax + by$, x, y integer numbers.
- Corollary: if a and b are relative prime, then there exist x and y such that $ax + by = 1$.
 - In other words, $ax \bmod b = 1$.
- How to find such x and y ?

Euclidian Algorithm Example

Find $\text{gcd}(143, 111)$

$$143 = 1 \times 111 + 32$$

$$111 = 3 \times 32 + 15$$

$$32 = 2 \times 15 + 2$$

$$15 = 7 \times 2 + 1$$

$$\text{gcd}(143, 111) = 1$$

$$32 = 143 - 1 \times 111$$

$$\begin{aligned} 15 &= 111 - 3 \times 32 \\ &= 4 \times 111 - 3 \times 143 \end{aligned}$$

$$\begin{aligned} 2 &= 32 - 2 \times 15 \\ &= 7 \times 143 - 9 \times 111 \end{aligned}$$

$$\begin{aligned} 1 &= 15 - 7 \times 2 \\ &= 67 \times 111 - 52 \times 143 \end{aligned}$$

Extended Euclidian Algorithm

```
x=1; y=0; d=a; r=0; s=1; t=b;
while (t>0) {
    q = ⌊d/t⌋
    u=x-qr; v=y-qs; w=d-qt
    x=r;   y=s;   d=t
    r=u;   s=v;   t=w
}
return (d, x, y)
```

Invariants:

$$ax + by = d$$

$$ar + bs = t$$

Review of Groups

Definition

A *group* (G, \times) is a set G on which a binary operation is defined which satisfies the following axioms:

Closure: For all $a, b \in G$, $a \times b \in G$.

Associative: For all $a, b, c \in G$, $(a \times b) \times c = a \times (b \times c)$.

Identity: $\exists e \in G$ s.t. for all $a \in G$, $a \times e = a = e \times a$.

Inverse: For all $a \in G$, $\exists a^{-1} \in G$ s. t. $a \times a^{-1} = a^{-1} \times a = e$.

Example

$(\mathbb{Z}, +)$ is a group

$\mathbb{Z}_n = \{0, 1, \dots, n - 1\}$, $(\mathbb{Z}_n, + \text{ modulo } n)$ is a group

$(\mathbb{Z}_n, \times \text{ modulo } n)$ NOT a group Why?

The group (\mathbb{Z}_p^*, \times) and (\mathbb{Z}_n^*, \times)

- $\mathbb{Z}_p^* = \{1, 2, \dots, p-1\}$
 - \mathbb{Z}_p^* is a group
 - $|\mathbb{Z}_p^*| = p-1$
 - For any $a \in \mathbb{Z}_p^*$, a^{-1} exists in \mathbb{Z}_p^* . Why? How to compute a^{-1} ?
- \mathbb{Z}_n^* consists of all integers in $[1..n-1]$ that are relative prime to n
 - $\mathbb{Z}_n^* = \{ a \mid 1 \leq a \leq n \text{ and } \gcd(a, n) = 1 \}$
 - (\mathbb{Z}_n^*, \times) is a group
 - $\gcd(a, n) = 1$ and $\gcd(b, n) = 1 \Rightarrow \gcd(ab, n) = 1$

The Euler Phi Function

Definition

Given an integer n , $\Phi(n) = |Z_n^*|$ is the number of all numbers a such that $0 < a < n$ and a is relatively prime to n (i.e., $\gcd(a, n) = 1$).

Theorem:

If $\gcd(m, n) = 1$, $\Phi(mn) = \Phi(m) \Phi(n)$

The Euler Phi Function

Theorem: Formula for $\Phi(n)$

Let p be prime, e, m, n be positive integers

1) $\Phi(p) = p-1$

2) $\Phi(p^e) = p^e - p^{e-1}$

3) If $n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$ then

$$\Phi(n) = n \left(1 - \frac{1}{p_1}\right) \left(1 - \frac{1}{p_2}\right) \dots \left(1 - \frac{1}{p_k}\right)$$

Euler's Theorem

Euler's Theorem

Given integer $n > 1$, such that $\gcd(a, n) = 1$ then
$$a^{\Phi(n)} \equiv 1 \pmod{n}$$

Corollary

Given integer $n > 1$, such that $\gcd(a, n) = 1$ then
 $a^{\Phi(n)-1} \pmod{n}$ is a multiplicative inverse of $a \pmod{n}$.

Corollary

Given integer $n > 1$, x , y , and a positive integers with
 $\gcd(a, n) = 1$. If $x \equiv y \pmod{\Phi(n)}$, then
$$a^x \equiv a^y \pmod{n}.$$

Linear Equation Modulo

Theorem

If $\gcd(a, n) = 1$, the equation $ax \equiv 1 \pmod{n}$

has a unique solution, $0 < x < n$

Proof Idea:

if $ax_1 \equiv 1 \pmod{n}$ and $ax_2 \equiv 1 \pmod{n}$, then $a(x_1 - x_2) \equiv 0 \pmod{n}$

Linear Equation Modulo (cont.)

Theorem

If $\gcd(a, n) = 1$, the equation

$$ax \equiv b \pmod{n}$$

has a solution.

Proof Idea:

$$x = a^{-1} b \pmod{n}$$

Chinese Remainder Theorem (CRT)

Theorem

Let n_1, n_2, \dots, n_k be integers s.t. $\gcd(n_i, n_j) = 1$,
 $i \neq j$.

$$x \equiv a_1 \pmod{n_1}$$

$$x \equiv a_2 \pmod{n_2}$$

...

$$x \equiv a_k \pmod{n_k}$$

There exists a unique solution modulo

$$n = n_1 n_2 \dots n_k$$

Proof of CMT

- Consider the function $\chi: \mathbb{Z}_n \rightarrow \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times \dots \times \mathbb{Z}_{n_k}$
$$\chi(x) = (x \bmod n_1, \dots, x \bmod n_k)$$
- We need to prove that χ is a bijection.
- For $1 \leq i \leq k$, define $m_i = n / n_i$, then $\gcd(m_i, n_i) = 1$
- For $1 \leq i \leq k$, define $y_i = m_i^{-1} \bmod n_i$
- Define function $\rho(a_1, a_2, \dots, a_k) = \sum a_i m_i y_i \bmod n$
 - $a_i m_i y_i \equiv a_i \pmod{n_i}$
 - $a_i m_i y_i \equiv 0 \pmod{n_j}$ where $i \neq j$

Example:

$$\begin{aligned}x &\equiv 5 \pmod{7} \\x &\equiv 3 \pmod{11} \\x &\equiv 10 \pmod{13}\end{aligned}$$

- $n_1=7, n_2=11, n_3=13, n=1001$
- $m_1=143, m_2=91, m_3=77$
- $y_1=143^{-1} \pmod{7} = 3^{-1} \pmod{7} = 5$
- $y_2=91^{-1} \pmod{11} = 3^{-1} \pmod{11} = 4$
- $y_3=77^{-1} \pmod{13} = 12^{-1} \pmod{13} = 12$

- $x=(5 \times 143 \times 5 + 3 \times 91 \times 4 + 10 \times 77 \times 12) \pmod{1001}$
 $= 13907 \pmod{1001} = 894$

The Euler Phi Function Revisited

Theorem:

If $\gcd(m,n) = 1$, $\Phi(mn) = \Phi(m) \Phi(n)$

Proof: for every $a \in \mathbb{Z}_m^*$ and $b \in \mathbb{Z}_n^*$

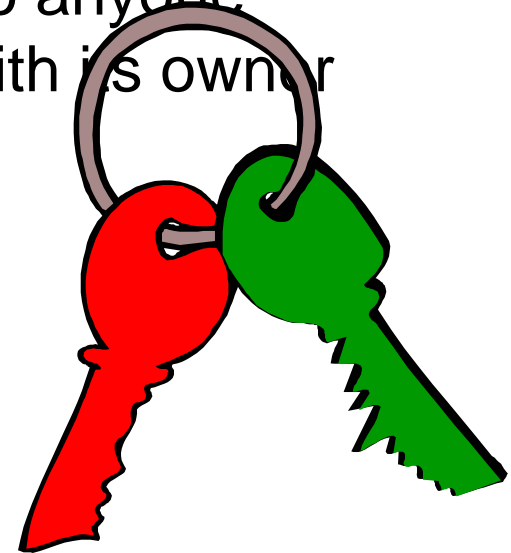
solve $x = a \pmod{m}$

$x = b \pmod{n}$

$\gcd(x, mn) = 1$

Public Key Cryptography

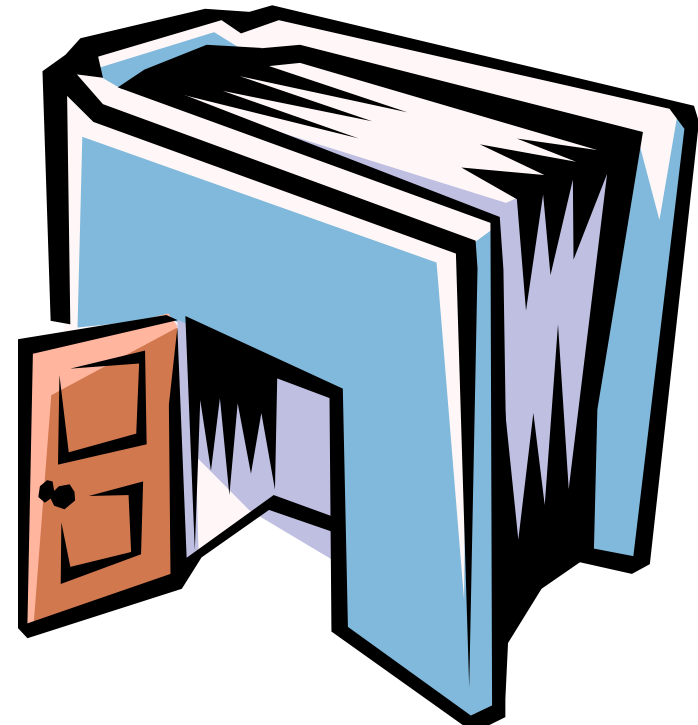
- Each party has a PAIR (P, S) of keys: P is the **public** key and S is the **secret** key.
- Knowing the public-key and the cipher, it is still computationally infeasible to compute the private key (an NP-class problem).
- The public-key may be distributed to anyone wishing to communicate securely with its owner
- $\text{Dec}_S(\text{Enc}_P(M)) = M$



Trapdoor One-way Functions

Definition:

A function $f: \{0,1\}^* \rightarrow \{0,1\}^*$ is a trapdoor one-way function iff $f(x)$ is a one-way function; however, given some extra information it becomes feasible to compute f^{-1} : given y , find x s.t. $y = f(x)$



Public key cryptography
relies on trapdoor one-way
functions

RSA Algorithm

- Invented in **1978** by Ron **R**ivest, Adi **S**hamir and Leonard **A**dleman
- Security relies on the difficulty of factoring large composite numbers
- Published as R L Rivest, A Shamir, L Adleman, "*On Digital Signatures and Public Key Cryptosystems*", Communications of the ACM, vol 21 no 2, pp120-126, Feb 1978

RSA Description

Key generation:

Select 2 large prime numbers of about the same size, p and q

Compute $n = pq$, and $\Phi = (q-1)(p-1)$

Select a random integer e , $1 < e < \Phi(n)$, s.t. $\gcd(e, \Phi(n)) = 1$

Compute d , $1 < d < \Phi$ s.t. $ed \equiv 1 \pmod{\Phi(n)}$

Public key: (e, n)

Secret key: d

Note: $\Phi(n)$ is Euler's Phi function

How to compute d ?

RSA Description (cont.)

Encryption

For the message M , $0 < M < n$
use public key (e, n)
compute $C = M^e \bmod n$

Decryption

For a message C
use private key (d)
Compute $C^d \bmod n = (M^e \bmod n)^d \bmod n$
 $= M^{ed} \bmod n = M$

RSA Example

- $p = 11, q = 7, n = 77, \Phi(n) = 60$
- $d = 13, e = 37$ ($ed = 481; ed \bmod 60 = 1$)
- Let $M = 15$. Then $C \equiv M^e \pmod{n}$
 - $C \equiv 15^{37} \pmod{77} = 71$
- $M \equiv C^d \pmod{n}$
 - $M \equiv 71^{13} \pmod{77} = 15$

Why Does RSA Work?

Want to show that $(M^e)^d \pmod n = M$, $n = pq$

$ed \equiv 1 \pmod{\Phi(n)}$, so $ed = k*\Phi(n) + 1$, for some integer k .

Case one: $\gcd(M, n) = 1$

$$M^{ed} \equiv M^{k*\Phi(n)} M \equiv 1^k M \equiv M \pmod n$$

Case two: $\gcd(M, n) = p$

$$M^{ed} \pmod p = (M \pmod p)^{ed} \pmod p = 0 \quad \text{so } M^{ed} \equiv M \pmod p$$

$$M^{ed} \pmod q = (M^{k*\Phi(n)} \pmod q) (M \pmod q) = M \pmod q \quad \text{so } M^{ed} \equiv M \pmod q$$

Since p and q are distinct primes we obtain $M^{ed} \equiv M \pmod{pq}$

RSA Implementation

- Select p and q prime numbers
- In general, select numbers, then test for primality
- Many implementations use the Rabin-Miller test, (probabilistic test)



Efficiency of computation modulo n

- Suppose that n is a k -bit number, and $0 \leq x, y \leq n$
 - computing $(x+y) \bmod n$ takes time $O(k)$
 - computing $(x-y) \bmod n$ takes time $O(k)$
 - computing $(xy) \bmod n$ takes time $O(k^2)$
 - computing $(x^{-1}) \bmod n$ takes time $O(k^3)$
 - computing $(x)^c \bmod n$ takes time $O((\log c) k^2)$

Square and Multiply Algorithm for Exponentiation

- Computing $(x)^c \bmod n$
 - Example: suppose that $c=110101=53$
 - $x^c = (((x^2 \cdot x)^2)^2 \cdot x)^2 \cdot x \bmod n$

Alg: Square-and-multiply ($x, n, c = c_{k-1} c_{k-2} \dots c_1 c_0$)

$z=1$

 for $i \leftarrow k-1$ downto 0 {

$z \leftarrow z^2 \bmod n$

 if $c_i = 1$ then $z \leftarrow (z \times x) \bmod n$

 }

 return z

RSA Implementation

e

- e is usually chosen to be 3 or $2^{16} + 1 = 65537$
- Speed up the encryption, the algorithm is called square-and-multiply (computed exponentiation)
- The smaller the number of 1 bits, the better



RSA Implementation

n, p, q

- RSA is usually described as the number of bits for n. Current recommendation is 1024 bits for n.
- p and q must have the same bit length, so for 1024 bits RSA, p and q must be about 512 bits.
- p-q should not be small, otherwise $p \cong \sqrt{n}$

Summary

- Number theory:
 - Extended Euclidian Algorithm
 - Euclid's Theorem
 - Chinese remainder theorem
- Public-key encryption
- RSA



Next ...

- Testing prime numbers
- Attacks on RSA
- The Rabin cryptosystem

