Topics

1. Number Theory Review
2. Public Key Cryptography
3. One-Way Trapdoor Functions
4. Diffie-Hellman Key Exchange
5. RSA Cipher
6. Modern Steganography
Number Theory Review

- Prime Numbers
- Fundamental Theorem of Arithmetic
- Greatest Common Divisors
- Relatively Prime Numbers
- Modular Inverses
- Euler’s Totient Function
Definition: An integer \( n > 1 \) is **prime** if its only factors are 1 and \( n \).

Definition: An integer \( n > 1 \) that is not prime is **composite**.

**Theorem:** An integer \( n > 1 \) can be written as a product of prime numbers.

\[
n = p_1^a p_2^b p_3^c p_4^d \ldots
\]
Greatest Common Divisor

**Definition:** The greatest common divisor of integers $a$ and $b > 0$, $gcd(a, b)$ is the largest number that divides both $a$ and $b$.

**Definition:** Two integers $a$ and $b > 0$ are *relatively prime* if $gcd(a, b) = 1$.

**Theorem:** If $n$ is prime, then every integer from 1 to $n-1$ is relatively prime to $n$. 
Modular Arithmetic

The operation $x \mod n$, where $x$ and $n$ are integers, is taking the remainder of $x$ divided by $n$, which is one of $n$ possible values:

$$\{0, 1, 2, \ldots, (n - 1)\}$$

Congruence is equality of two numbers modulo $n$

$$a = b \mod n \text{ iff } a = b + kn$$

Equivalent to performing arithmetic in $Z_n$, which

$$Z_n = \{0, 1, 2, \ldots, (n - 1)\}$$
Laws of Modular Arithmetic

\[(a + b) \mod N = (a \mod N + b \mod N) \mod N\]

\[(a - b) \mod N = (a \mod N - b \mod N) \mod N\]

\[ab \mod N = (a \mod N)(b \mod N) \mod N\]

\[a(b+c) \mod N = ((ab \mod N) + (ac \mod N)) \mod N\]
Modular Inverses

**Multiplicative Inverse:** The inverse of a number $x$ is a number $x^{-1}$ such that $xx^{-1} = 1$.

**Modular Inverse:** An integer $x^{-1}$ such that $1 = (x x^{-1}) \mod n$

For example, $4 \times 3 \mod 11 = 12 \mod 11 = 1$

1 and $n-1$ are their own modular inverses.

However, there is not always a solution.

2 has no inverse mod 16

If $n$ is prime, all elements but 0 have an inverse.
## Multiplication Table for $\mathbb{Z}_{10}$

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>2</td>
<td>5</td>
<td>8</td>
<td>1</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>4</td>
<td>8</td>
<td>2</td>
<td>6</td>
<td>0</td>
<td>4</td>
<td>8</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>6</td>
<td>2</td>
<td>8</td>
<td>4</td>
<td>0</td>
<td>6</td>
<td>2</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>7</td>
<td>4</td>
<td>1</td>
<td>8</td>
<td>5</td>
<td>2</td>
<td>9</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>8</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>8</td>
<td>6</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
Modular Exponentiation

Identities
\[ a^b \cdot a^c = a^{(b+c)} \mod n \]
\[ (a^b)^c = a^{bc} \mod n \]

Can take modulo operator inside
\[ a^2 \mod n = (a \mod n)(a \mod n) \mod n \]
\[ a^2 \mod n = (a \mod n)^2 \mod n \]
Euler Totient Function

Euler’s totient function $\phi(n)$

- Number of positive integers less than $n$ that are relatively prime to $n$.
- Or equivalently, the number of integers $k$ in the range $1 \leq k \leq n$ for which $\gcd(n,k)=1$.

**Example:** $\phi(10) = 4$

- $1, 3, 7, 9$ are relatively prime to $10$

**Theorem:** If $\gcd(a,b)=1$, $\phi(ab) = \phi(a) \phi(b)$

**Note:** If $n$ is prime, $\phi(n) = n - 1$

**Result:** If $a, b$ prime, $\phi(ab) = (a-1)(b-1)$
Euler’s Totient Theorem

Theorem: If $n > 1$ and $\gcd(a,n) = 1$, then
$$a^{\phi(n)} \mod n = 1.$$ 

Corollary: If $n > 1$ and $\gcd(a,n) = 1$, then
$$a^{\phi(n)-1} \mod n$$
is the modular inverse of $a \mod n$.

Ex: $7^4 \equiv 1 \pmod{10}$, which we use to compute $7^{222} \mod 10$:
$$7^{222} \equiv 7^{4 \times 55 + 2} \equiv (7^4)^{55} \times 7^2 \equiv 1^{55} \times 7^2 \equiv 49 \equiv 9 \pmod{10}.$$
Public Key Cryptography

Two keys

*Private key* known only to owner.
*Public key* available to anyone.

Applications

**Confidentiality:**
- Sender enciphers using recipient’s public key,
- Receiver deciphers using their private key.

**Integrity/authentication:**
- Sender enciphers using own private key,
- Recipient deciphers using sender’s public key.
Requirements

Based on mathematical problems such that

1. It must be computationally easy to encipher or decipher a message given the appropriate key.
2. It must be computationally infeasible to derive the private key from the public key.
3. It must be computationally infeasible to determine the private key from a chosen plaintext attack.

Computationally infeasible means exponential time complexity, so that encryption with an n+1 bit key is 2X harder to crack than an n-bit key based encryption.

- It is faster to solve a computationally infeasible problem than to do a brute force exhaustive key search.
- Therefore keys for public key encryption must be longer than keys for symmetric encryption algorithms.
Diffie-Hellman Key Exchange

Compute a common, shared key
- Called a symmetric key exchange protocol

Based on discrete logarithm problem
- One way function: \( f(k) = g^k \mod p \).
- Invert function by computing logarithm to find \( k \).
- Solutions known for small \( p \).
- Computationally infeasible for large \( p \).

Shared Constants
- prime modulus \( p \),
- integer base \( g \neq \{0, 1, p-1\} \)
Algorithm

1. Alice chooses a random private key $k$
2. Computes public key $A = g^k \mod p$ and sends to Bob.
3. Bob chooses a random private key $k'$
4. Computes public key $B = g^{k'} \mod p$ and sends to Alice.
5. Alice computes secret key
   1. $K_1 = B^k \mod p$
6. Bob computes secret key
   1. $K_2 = A^{k'} \mod p$
7. Encrypt messages w/ symmetric cipher using $K_1=K_2$ as the secret key.
How do we know $K_1 = K_2$?

Let $K_1 = B^k \mod p$

$$= (g^{k'} \mod p)^k \mod p$$

$$= g^{k'k} \mod p$$

$$= g^{kk'} \mod p$$

$$= (g^k \mod p)^{k'} \mod p$$

$$= A^{k'} \mod p$$

$$= K_2$$
Example

Assume $p = 53$ and $g = 17$

Alice chooses $k_{Alice} = 5$

Then $K_{Alice} = 17^5 \mod 53 = 40$

Bob chooses $k_{Bob} = 7$

Then $K_{Bob} = 17^7 \mod 53 = 6$

Shared key:

$K_{Bob}^{{k_{Alice}}} \mod p = 6^5 \mod 53 = 38$

$K_{Alice}^{{k_{Bob}}} \mod p = 40^7 \mod 53 = 38$
Activity: DH with a Partner

Use an arbitrary precision calculator like python/ruby interpreter or https://apfloat.appspot.com/ for activity.

1. Together agree on a prime modulus > 100 for p.
2. Together select \( g \neq \{0, 1, p-1\} \).
3. Individually, each person selects a private key.
4. Individually, each person computes public key.
5. Share public keys with your partner.
6. Individually, each person computes shared key.
7. Compare shared keys and verify they’re identical.
Elliptic Curve Cryptography

ECC is based on the algebra of elliptic curves (like $y^2 = x^3 + ax + b$) over finite fields instead of being based on simple finite groups like modular arithmetic.

- Analogs to discrete logarithm and other one-way functions suitable for public key crypto exist in EC.
- Patents may prevent use of some ECC algorithms.

Suite B is NIST-recommended crypto including ECC

- Includes Elliptic Curve Diffie-Hellman (ECDH)
- NSA recommended until August 2015, when it announced a rapid transition away from Suite B to a new set of quantum-resistant algorithms to be published.
One-Way Trapdoor Functions

*Trapdoor one-way Function*: One-way function whose inverse is easy to calculate only if given a special piece of information.

**Example**: Prime factoring

- Easy to calculate product.
- Difficult to calculate prime factors from product.
- Easy to calculate one prime factor, given others.
RSA

1. Modular exponentiation cipher.
2. Treat message blocks as large numbers.
3. Relies on the difficulty of determining the number of integers relatively prime to a large integer $n$. 
Algorithm

Choose two large prime numbers $p$, $q$
- Let $n = pq$; then $\phi(n) = (p-1)(q-1)$
- Choose $e < n$ such that $e$ relatively prime to $\phi(n)$.
- Compute inverse of $e$, $d$
  - $ed \mod \phi(n) = 1$
  - $d = e^{-1} \mod \phi(n)$

Public key: $(e, n)$
Private key: $d$
Encipher: $c = m^e \mod n$
Decipher: $m = c^d \mod n$
Example: Confidentiality

Take $p = 7$, $q = 11$, so $n = 77$ and $\phi(n) = 60$

Alice chooses $e = 17$, making $d = 53$

Bob wants to send Alice secret message HELLO, which we represent by the numbers 07 04 11 11 14

$07^{17} \mod 77 = 28$
$04^{17} \mod 77 = 16$
$11^{17} \mod 77 = 44$
$11^{17} \mod 77 = 44$
$14^{17} \mod 77 = 42$

Bob sends 28 16 44 44 42
Example

Alice receives 28 16 44 44 42
Alice uses private key, $d = 53$, to decrypt message:

\[
\begin{align*}
28^{53} \mod 77 &= 07 \\
16^{53} \mod 77 &= 04 \\
44^{53} \mod 77 &= 11 \\
44^{53} \mod 77 &= 11 \\
42^{53} \mod 77 &= 14
\end{align*}
\]

Alice translates message to letters to read HELLO

No one else could read it, as only Alice knows her private key and that is needed for decryption.
Ex: Integrity/Authentication

Take \( p = 7, q = 11 \), so \( n = 77 \) and \( \phi(n) = 60 \)

Alice chooses \( e = 17 \), making \( d = 53 \)

Alice wants to send Bob message HELLO (07 04 11 11 14) so Bob knows it is what Alice sent (no changes in transit, and authenticated)

\[
\begin{align*}
07^{53} \mod 77 &= 35 \\
04^{53} \mod 77 &= 09 \\
11^{53} \mod 77 &= 44 \\
11^{53} \mod 77 &= 44 \\
14^{53} \mod 77 &= 49
\end{align*}
\]

Alice sends 35 09 44 44 49
Example

Bob receives 35 09 44 44 49
Bob uses Alice’s public key, $e = 17$, $n = 77$, to decrypt message:

\[
\begin{align*}
35^{17} \mod 77 &= 07 \\
09^{17} \mod 77 &= 04 \\
44^{17} \mod 77 &= 11 \\
44^{17} \mod 77 &= 11 \\
49^{17} \mod 77 &= 14
\end{align*}
\]

Bob translates message to letters to read HELLO

Alice sent it as only she knows her private key

If (enciphered) message’s blocks (letters) altered in transit, would not decrypt properly
Example: Both

Alice wants to send Bob message HELLO both enciphered and authenticated (integrity-checked)

Alice’s keys: public (17, 77); private: 53
Bob’s keys: public: (37, 77); private: 13

Alice enciphers HELLO (07 04 11 11 14):

\[(07^{53} \mod 77)^{37} \mod 77 = 07\]
\[(04^{53} \mod 77)^{37} \mod 77 = 37\]
\[(11^{53} \mod 77)^{37} \mod 77 = 44\]
\[(14^{53} \mod 77)^{37} \mod 77 = 14\]

Alice sends 07 37 44 44 14
Activity: RSA with a Partner

Use an arbitrary precision calculator and an online Euler’s totient calculator at the following URL:
http://www.javascripter.net/math/calculators/eulertotientfunction.htm

1. Together agree on primes p,q < 100.
2. Together compute n and \( \phi(n) = (p-1)(q-1) \)
3. Individually choose \( e < n \), s.t. \( e \) relatively prime to \( \phi(n) \).
4. Individually compute \( d = e^{-1} \mod \phi(n) \).
5. Share public keys \( e \) with each other.
6. Individually encrypt “HELLO” with partner’s public key.
7. Share ciphertext with partner.
8. Individually decrypt message with your private key.
Security Services

Confidentiality

- Only the owner of the private key knows it, so text enciphered with public key cannot be read by anyone except the owner of the private key.

Authentication

- Only the owner of the private key knows it, so text enciphered with private key must have been generated by the owner.
More Security Services

Integrity

- Enciphered letters cannot be changed undetectably without knowing private key.

Non-Repudiation

- Message enciphered with private key came from someone who knew it.
Warnings

Use large blocks for encryption:

- If 1 character per block, RSA can be broken using statistical attacks (just like classical cryptosystems.)
- Attacker cannot alter letters, but can rearrange them and alter message meaning.
  - Ex: reverse ciphertext of message ON to get NO
Steganography

Hiding messages in another text (the covertext) so that no one except intended recipient knows a message has been sent.

**Wax Tablets:** In ancient times, messages written in wax poured on top of a stone/wood tablet. Messages hidden by engraving in wood then pouring wax over them.

**Invisible Ink:** Write message using lemon juice on paper. Write covertext in regular ink after dries. Heat to view hidden message.

**Null Cipher:** Hide message in ordinary text, using $n^{th}$ letter of each word, or every $n^{th}$ word of the message.
Digital Steganography

1. Choose a cover medium file.
   JPEG, MP3, etc.
2. Identify redundant bits in cover medium.
   Low order bits in image and audio files.
3. Select and replace data.

Steganographic Image

Picture of cat hidden in the image above.
JSteg: JPEG Steganography

JPEG image format

- For each color component, a discrete cosine transform (DCT) turns 8x8 pixel blocks into 64 DCT coefficients.

Derek Upham’s JSteg algorithm

- LSBs of DCT coefficients are redundancy.
- Modification of a single DCT coef affects all 64 pixels.
Steganalysis

Compare stegographic file with original.
  • 100% effective at identifying presence.
  • Original file is “secret key” of steganography.

Statistical analysis
  • Inserting high entropy changes histogram of color frequencies in predictable ways.
  • Reduces frequency difference between adjacent colors.

Countermeasures
  • Insert less information to reduce impact.
  • Choose DCT coefficients to modify at random.
  • Alternate +/- DCT coefficient value to encode bits.
  • Use parity of groups of DCT LSBs to encode a message.
Key Points

1. Two types of cryptosystems:
   1. classical (symmetric)
   2. public key (asymmetric)

2. Public Key Cryptography
   1. Based on mathematical problems; solving the problem is exponentially hard but easier than brute force search of key space.
   2. Confidentiality: encipher with public, decipher with private
   3. Integrity: encipher with private, decipher with public

3. Steganography
   1. Hide messages in text or media files.
   2. Analyze using statistical techniques.