Outline

- Data security
- Cryptography basics
- Cryptographic systems
- DES
- RSA
Cryptography

- Cryptography is the science of secret writing.
  - A cipher is a secret method of writing, where by plaintext (cleartext) is transformed into a ciphertext.
  - The process of transforming plaintext into ciphertext is called encipherment or encryption.
  - The reverse process of transforming ciphertext into plaintext is called decipherment or decryption.
  - Encryption and decryption are controlled by cryptographic keys.
Secret Writing

Encryption

Decryption

Plaintext

Key

Ciphertext
Attacks against Ciphers

- Cryptanalysis is the science and study of methods of breaking ciphers.
- A cipher is breakable if it is possible to determine the plaintext or key from the ciphertext, or to determine the key from plaintext-ciphertext pairs.
- Attacks
  - Ciphertext-only attack
  - Known-plaintext attack
  - Chosen-plaintext attack
A cryptographic system has five components:
- A plaintext message space, $M$
- A ciphertext message space, $C$
- A key space, $K$
- A family of enciphering transformations $E_k : M \rightarrow C$
- A family of deciphering transformations $D_k : C \rightarrow M$
Cryptographic Systems (cont.)

\[ D_k(E_k(m)) = m \text{, for a key } k \]

- Cryptosystem requirements:
  - Efficient enciphering/deciphering
  - Systems must be easy to use
  - The security of the system depends only on the keys, not the secrecy of E or D
Secure Cipher

- **Unconditionally secure**
  - A cipher is unconditionally secure if no matter how much ciphertext is intercepted, there is not enough information in the ciphertext to determine the plaintext uniquely.

- **Computationally secure**
  - A cipher is computationally infeasible to break.
Secrecy Requirements

- It should be computationally infeasible to systematically determine the deciphering transformation $D_k$ from intercepted $c$, even if corresponding $m$ is known.

- It should be computationally infeasible to systematically determine $m$ from intercepted $c$. 

```
M → $E_k$ → C → $D_k$ → M
```

- Disallowed
- Protected

C. H. HUANG IN CML
Authenticity requirements

- It should be computationally infeasible to systematically determine the enciphering transformation given $c$, even if corresponding $m$ is known.
- It should be computationally infeasible to systematically find $c'$ such that $D_k(c')$ is a valid plaintext in $M$. 

$$M \xrightarrow{E_k} C \xrightarrow{D_k} M$$

- protected
- disallowed
Key-distribution cryptosystem

- Encrypting & decrypting are closely tied together.
- The sender and the receiver must agree on the use of a common key before any message transmission takes place.
- A safe communication channel must exist between sender and receiver.
In a public key cryptosystem, each participant is assigned a pair of inverse keys $E$ and $D$.

- Different functions are used for enciphering and deciphering, one of the two keys can be made public, provided that it is impossible to generate one key from the other.
- $E$ can be made public, but $D$ is kept secret.
- The normal key transmission between senders and receivers can be replaced by an open directory of enciphering keys, containing the keys $E$ for all participants.
Using Public-Key Cryptosystem to Transfer Messages Secretly

- When a person A wishes to send a message to a person B, the receiver’s enciphering key $E_B$ is used to generate the ciphertext $E_B(m)$. Since the key $E_B$ is freely available, anyone can then encipher a message destined for B. However, only the receivers B with access to the decipher key $D_B$ can regenerate the original text by performing the inverse transform $D_B(E_B(m))$. 
Digital Signature

- Guaranteeing authenticity.
- Let B be the recipient of a message m signed by A. Then A’s signature must satisfy:
  1. B must be able to validate A’s signature on m.
  2. It must be impossible to forge A’s signature.
  3. If A disavows signing a message, a third party must be able to resolve the dispute.
Using Public-key Systems to Implement Digital Signatures

1. A signs m by computing $c = D_A(m)$
2. B validates A’s signature by checking $E_A(c) = m$
3. A dispute can be judged by checking whether $E_A(c)$ restores M in the same ways as B.

➤ Requirements:

- $D_k(E_k(m)) = E_k(D_k(m)) = m$
Secrecy and Authenticity in A Public-Key System

\[ E_A(D_B(C)) = E_A(D_B(E_B(D_A(M))))) = E_A(D_A(M)) = M \]
Reference

- Cryptography and Data Security, D. Elizabeth and R. Denning, Purdue University, 1998
- FAQ about Today’s Cryptography, RSA Laboratory, (found in www.rsa.com)
- The reference listed in course handout.
Conventional Cryptosystems

- Using substitution transform and permutation transform
  - Substitution Ciphers
  - Running Key Ciphers
  - Transposition Ciphers
    - (Permutation ciphers)
  - Stream Ciphers
Substitution Ciphers

- Replace bits, characters, or blocks of characters with substitutes.
  - Example: Caesar cipher
    - which shift each letter in the English forward by K positions (shifts past Z cycle back to A)
- A simple substitution cipher is easy to solve by performing a frequency analysis.
Running Key Ciphers

- The security of a substitution cipher generally increases with the key length. In a running key cipher, the key length is equal to the plaintext message. (not using a fixed key alphabet)
  - E.g. use the text in a book as the key sequence.
- The cipher may be breakable by Friedman’s method based on the observation that both plaintext and key letters are high frequency ones in natural language.
Permutation Ciphers

- Rearrange bits or characters in the data.

INFORMATION TECHNIQUES FOR IPR

IRITNERENOMTOEHIUSOIR

FANCQFP

IRITNERENOMTOEHIUSOIRFANCQFP

- What is the key?

- Attacks: frequency analysis of characters.
A product cipher is the composition of functions $F_1, \ldots, F_t$, where each $F_i$ may be a substitution or permutation.

Examples of product ciphers
- DES
Data Encryption Standard (DES)

- The National Bureau of Standards announced DES to be used in unclassified U.S. Government applications.
- DES enciphers 64-bit blocks with a 56-bit key.
An input block $T$ is first transposed under an initial permutation $IP$, giving $T_0 = IP(T)$.

- E.g. $t_1 t_2 \ldots t_{64} \rightarrow t_{58} t_{50} \ldots t_7$

Then $T_0$ is passed through 16 iterations of function $f$.

Finally, it is transposed under the inverse permutation $IP^{-1}$ to give the final result.
DES (cont.)

- Let $T_i$ denote the result of the $i$th iteration, and let $L_i$ and $R_i$ denote the left and right halves of $T_i$. Then

  \[ L_i = R_{i-1} \]
  \[ R_i = L_{i-1} \oplus f(R_{i-1}, K_i) \]

  where $\oplus$ is the exclusive-or operation and $K$ is a 48-bit key.

- After the last iteration, the left and right halves are not changed, but instead passed to $IP^{-1}$. 
Calculate the function $F(R_{i-1}, K_i)$:

1. Using bit-selection Table E to expand 32-bit $R_{i-1}$ to a 48-bit block $E(R_{i-1})$. (Similar to permutation)
2. Calculate the exclusive-or of $E(R_{i-1})$ and $K_i$. Then break the result into 8 6-bit blocks $B_1, \ldots, B_8$.
3. Use each 6-bit $B_j b_1b_2b_3b_4b_5b_6$ as input to a selection (substitution) and return a 4-bit block $S_j(B_j)$.

\[
\begin{align*}
  b_1b_6 & \rightarrow \text{row} \\
  b_2b_3b_4b_5 & \rightarrow \text{column}
\end{align*}
\]
Key calculation

- Each iteration \( i \) uses a different 48-bit key \( K_i \) derived from the initial key \( K \), which is input as a 64-bit block with 8 parity bits in positions 8, 16, ..., 64.
- PC-1 discards the parity bits and transposes the remaining 56-bit bits to obtain PC-1(\( K \)).
- PC-1(\( K \)) is then split to \( C \) and \( D \) of 28-bits each, and circular shifted by LS.
- \( C_i = LS_i(C_{i-1}) \), \( D_i = LS_i(D_{i-1}) \)
- \( K_i = PC-2(C_iD_i) \).
DES (cont.)

- Deciphering
  - The same algorithm is used, except that the order of key for each iteration is reversed. E.g. $K_{16}$ is used in 1st iteration, $K_{15}$ is used in 2nd iteration....
Disputes about DES

- 56-bit key length should be doubled?
  - A special purpose machine containing a million LSI chips could try $2^{56}$ keys in 1 day. The cost of this machine is about $20$ million. Amortized over 5 years, the cost per day would be $10,000$.
  - The same level of security could be obtained using multiple encryption scheme.

- The S-box may have hidden trapdoors.
  - The analysis is still classified.
Stream Ciphers

- A random number generator (typically LFSR) may be used to generate a stream of key characters, each character of the key being added to a character of the input stream to produce an output character.
Cipher Based on Computationally Difficult Problems

- One-way function: $C = f(P)$
  
  $f$: computationally simple
  
  $f^{-1}$: computationally difficult except in special cases when supplementary information (keys) is available
  
  - exponentiation and logarithm
  - multiplication/factoring
  - review of number theory

- NP-complete problems
  
  - A systematic deterministic solution is likely to require exponential time in the number of inputs.
Diffie Hellman’s public-key cryptosystem

- Each user i in the system has a pair of keys $X_i$ and $Y_i$, where
  $Y_i = \alpha^{X_i} \mod q$, $1 \leq X_i \leq q-1$, $1 \leq \alpha \leq q-1$, $q$: prime number
  $X_i$ is kept secret, but $Y_i$ is made public.

- Sender i generates the key
  $K_{ij} = Y_j^{X_i} \mod q = \alpha^{X_iX_j} \mod q$
  from receiver j’s public key $Y_j$ and his own private key $X_i$.

- Receiver j obtains $K_{ij}$ similarly from $Y_i$ and $X_j$. 
Security of Diffie Hellman’s System

- To generate the key $K_{ij}$, one of the private keys $X_i$ or $X_j$ must be known.

- To generate the $K_{ij}$ from $Y_i$ and $Y_j$, a form of logarithm below must be computed:
  
  $$K_{ij} = Y_i^{(log Y_j)} \mod q$$

  which is computationally difficult.
The RSA Algorithm

- Each user selects two large prime numbers P and Q at random, and multiplies them to obtain \( N = P \times Q \).
  - \( N \) should be about 200 digits long and can be made public.
  - \( P \) and \( Q \) are kept secret.
Using $P$ and $Q$, the user computes the Euler totient function $\Phi(N)$, representing the number of positive integers relatively prime to $N$.

\[ \Phi(N) = \Phi(P) \Phi(Q) = (P-1)(Q-1) \]

The user then chooses a quantity $E$ less than $N$ and relatively prime to $\Phi(N)$. The quantity $E$ is made public.
The RSA Algorithm (cont.)

- Given a message $M$ to be enciphered, $M$ is broken down into a sequence of quantities $M_1, M_2, \ldots, M_p$, where each component $M_i$ is represented by an integer between 0 and $N-1$. The enciphering is now done separately on each block $M_i$ using the public information $E$ and $N$ to generate a cryptogram $C_i$ as
  - $C_i = M_i^E \mod N$
  - at most $2 \cdot \log_2(N)$ multiplications are required
Using the secret information $\Phi(N)$, the user can easily compute a quantity $D$ such that $E \cdot D = 1 \mod \Phi(N)$ (deciphering key). I

- $E \cdot D = 1 \mod \Phi(N) = K \Phi(N) + 1$
- $D = K \Phi(N) + 1/E$. 
The RSA Algorithm (cont.)

- By Fermat’s theorem: \( M^{\Phi(N)} \mod N = 1 \mod N \), or \( M^{K\Phi(N)+1} \mod N = M \mod N \).

- Deciphering procedure:
  \[
  C_i^D \mod N \\
  = M_i^{ED} \mod N \\
  = M_i^{K\Phi(N)+1} \mod N \\
  = M_i \mod N \\
  = M_i
  \]
Using RSA

- Suppose user A want to send a message $m$ to user B. User A creates the ciphertext $c$ by $c = m^E \mod N$, where $E$ and $N$ are user B’s public key.
- User A sends $c$ to user B.
- User B decrypts $c$ by calculate $m = c^D \mod N$. The relation between $D$ and $E$ ensures that B correctly recovers $m$.
- Since only B knows $D$, only B can decrypt the message.
Attacks against RSA

- Attacks to recover all messages for a given key
  - Factor the public modulus N to P and Q. With P, Q, and E, the attacker can easily compute D.

- Attacks to recover a message
  - Guessed-plaintext attacks.
  - This attacks can be defeated by appending random bits.
Security of RSA

- The size of a key in the RSA algorithm typically refers to the size of the modulus N. The two primes P and Q should be roughly equal length.
- The longer the key size, the greater the security, but also the slower the RSA algorithm.
- The 512-bit RSA-155 was factored in seven month during 1999.
- The RSA lab currently recommends key sizes of 1024 bits for corporate use.