Fundamentals of Cryptography: Algorithms, and Security Services

Professor Guevara Noubir
Northeastern University
noubir@ccs.neu.edu

Cryptography: Theory and Practice, Douglas Stinson, Chapman & Hall/CRC

Network Security: Private Communication in a Public World [Chap. 2-8]
Charles Kaufman, Mike Speciner, Radia Perlman, Prentice-Hall

Outline

- Introduction to security/cryptography
- Secret Key Cryptography
  - DES, IDEA, AES
- Modes of Operation
  - ECB, CBC, OFB, CFB, CTR
  - Message Authentication Code (MAC)
- Hashes and Message Digest
- Public Key Algorithms
Why/How?

Why security?

- Internet, E-commerce, Digi-Cash, disclosure of private information ...

Security services:

- Authentication, Confidentiality, Integrity, Access control, Non-repudiation, availability

Cryptographic algorithms:

- Symmetric encryption (DES, IDEA, AES)
- Hashing functions
- Symmetric MAC (HMAC)
- Asymmetric (RSA, El-Gamal)
Terminology

Security services:
- Authentication, confidentiality, integrity, access control, non-repudiation, availability, key management

Security attacks:
- Passive, active

Cryptography models:
- Symmetric (secret key), asymmetric (public key)

Cryptanalysis:
- Ciphertext only, known plaintext, chosen plaintext, chosen ciphertext, chosen text
Security services

- **Authentication:**
  - assures the recipient of a message the authenticity of the claimed source

- **Access control:**
  - limits the access to authorized users

- **Confidentiality:**
  - protects against unauthorized release of message content

- **Integrity:**
  - guarantees that a message is received as sent

- **Non-repudiation:**
  - protects against sender/receiver denying sending/receiving a message

- **Availability:**
  - guarantees that the system services are always available when needed

- **Security audit:**
  - keeps track of transactions for later use (diagnostic, alarms...)

- **Key management:**
  - allows to negotiate, setup and maintain keys between communicating entities
Security Attacks

Security attacks:
- Interception (confidentiality)
- Interruption (availability)
- Modification (integrity)
- Fabrication (authenticity)

Kent’s classification
- Passive attacks:
  - Release of message content
  - Traffic analysis
- Active attacks:
  - Masquerade
  - Replay
  - Modification of message
  - Denial of service
Kerchoff’s Principle

- The cipher should be secure when the intruder knows all the details of the encryption process except for the secret key

  “No security by obscurity”

  Examples of system that did not follow this rule and failed?
Attacks on Encrypted Messages

- Ciphertext only:
  - encryption algorithm, ciphertext to be decoded

- Known plaintext:
  - encryption algorithm, ciphertext to be decoded, pairs of (plaintext, ciphertext)

- Chosen plaintext:
  - encryption algorithm, ciphertext to be decoded, plaintext (chosen by cryptanalyst) + corresponding ciphertext

- Chosen ciphertext:
  - encryption algorithm, ciphertext to be decoded, ciphertext (chosen by cryptanalyst) + corresponding plaintext

- Chosen text:
  - encryption algorithm, ciphertext to be decoded, plaintext + corresponding ciphertext (both can be chosen by attacker)
Encryption Models

- Symmetric encryption (conventional encryption)
  - Encryption Key = Decryption Key
  - E.g., AES, DES, FEAL, IDEA, BLOWFISH
- Asymmetric encryption
  - Encryption Key ≠ Decryption key
  - E.g., RSA, Diffie-Hellman, ElGamal

Message source → Plaintext → Encryption Algorithm → Ciphertext → Decryption Algorithm → Plaintext → Message Destination
Encryption Models

**Symmetric encryption:**
- Shared key

**Asymmetric encryption:**
- **Public key**
- **Private key**

**Message source** → **Plaintext** → **Encryption Algorithm** → **Ciphertext** → **Decryption Algorithm** → **Plaintext** → **Message Destination**
Some Building Blocks of Cryptography/Security

- Encryption algorithms

- One-way hashing functions (= message digest, cryptographic checksum, message integrity check, etc.)
  - Input: variable length string
  - Output: fixed length (generally smaller) string
  - Desired properties:
    - Hard to generate a pre-image (input) string that hashes to a given string,
    - second preimage, and collisions

- One-way functions
  - $y = f(x)$: easy to compute
  - $x = f^{-1}(y)$: much harder to reverse (it would take millions of years)
  - Example:
    - multiplication of 2 large prime number versus factoring
    - discrete exponentiation/discrete logarithms

- Protocols
  - authentication, key management, etc.
Securing Networks

Where to put the security in a protocol stack?

Practical considerations:
- End to end security
- No modification to OS

Applications Layer
  telnet/ftp, http: sfttp, mail: PGP

(SSL/TLS, ssh)

Transport Layer (TCP)

(IPSec, IKE)

Network Layer (IP)

Link Layer
  (IEEE802.1x/IEEE802.10)

Physical Layer
  (spread-Spectrum, quantum crypto, etc.)

Control/Management (configuration)

Network Security Tools:
  Monitoring/Logging/Intrusion Detection
Secret Key Cryptography

=  

Symmetric Cryptography

=  

Conventional Cryptography
Symmetric cryptosystems (conventional cryptosystems)

- **Substitution techniques:**
  - **Caesar cipher**
    - Replace each letter with the letter standing $x$ places further
    - Example: $(x = 3)$
      - plain: meet me after the toga party
      - cipher: phhw ph diwhu wkh wrjd sduwb
    - Key space: 25
  - Brut force attack: try 25 possibilities

  - **Monoalphabetic ciphers**
    - Arbitrary substitution of alphabet letters
    - Key space: $26! > 4 \times 10^{26} > \text{key-space(DES)}$
    - Attack if the nature of the plaintext is known (e.g., English text):
      - compute the relative frequency of letters and compare it to standard distribution for English (e.g., E:12.7, T:9, etc.)
      - compute the relative frequency of 2-letter combinations (e.g., TH)
English Letters Frequencies

CSU610: SWARM

Cryptography
Symmetric cryptosystems (Continued)

- Multiple-Letter Encryption (Playfair cipher)
  - Plaintext is encrypted two-letters at a time
  - Based on a 5x5 matrix
  - Identification of individual diagraphs is more difficult (26x26 possibilities)
  - A few hundred letters of ciphertext allow to recover the structure of plaintext (and break the system)
  - Used during World War I & II

- Polyalphabetic Ciphers (Vigenère cipher)
  - 26 Caesar ciphers, each one denoted by a key letter
    - key: deceptive deceptive deceptive
    - plain: wearediscovered saveyourself
    - cipher: ZICVTWQNGRZGVTWAVZHCQYGLMGJ
  - Enhancement: auto-key (key = initial||plaintext)

- Rotor machines: multi-round monoalphabetic substitution
  - Used during WWII by Germany (ENIGMA) and Japan (Purple)
One-Time Pad

Introduced by G. Vernam (AT&T, 1918), improved by J. Mauborgne

Scheme:
- Encryption: $c_i = p_i \oplus k_i$
- $c_i$: $i$th binary digit of plaintext, $p_i$: plaintext, $k_i$: key
- Decryption: $p_i = c_i \oplus k_i$
- Key is a random sequence of bits as long as the plaintext

One-Time Pad is unbreakable
- No statistical relationship between ciphertext and plaintext
- Example (Vigenère One-Time Pad):
  - Cipher: ANKYODKYUREPBFJBYOJDSPREYIUN
  - Plain-1 (with k1): MR MUSTARD WITH THE CANDLE
  - Plain-2 (with k2): MISS SCARLET WITH THE KNIFE

Share the same long key between the sender & receiver
Transposition/Permutation Techniques

- Based on permuting the plaintext letters
- Example: rail fence technique
  ```
  mematrhtgpwy
  etefeteoaat
  ```
- A more complex transposition scheme
  - Key: 4312567
  - Plain: attackp
    - ospone
    - duntilt
    - woamxyz
  - Cipher: TTNAAPTMTSUOAOODWCOIXKNNLYPETZ
- Attack: letter/diagraph frequency
- Improvement: multiple-stage transposition
Today’s Block Encryption Algorithms

- **Key size:**
  - Too short => easy to guess
- **Block size:**
  - Too short easy to build a table by the attacker: (plaintext, ciphertext)
  - Minimal size: 64 bits
- **Properties:**
  - One-to-one mapping
  - Mapping should look random to someone who doesn’t have the key
  - Efficient to compute/reverse
- **How:**
  - Substitution (small chunks) & permutation (long chunks)
  - Multiple rounds
  ⇒ SPN (Substitution and Permutation Networks) and variants
Data Encryption Standard (DES)

- Developed by IBM for the US government
- Based on Lucifer (64-bits, 128-bits key in 1971)
- To respond to the National Bureau of Standards CFP
  - Modified characteristics (with help of the NSA):
    - 64-bits block size, 56 bits key length
    - Concerns about trapdoors, key size, sbox structure
- Adopted in 1977 as the DES (FIPS PUB 46, ANSI X3.92) and reaffirmed in 1994 for 5 more years

- Replaced by AES
DES is based on Feistel Structure

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

\[ L_i = R_{i-1} \]

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

Key (56 bits)

Expansion Permutation

S-Box Substitution

P-Box Permutation

Shift

Compression Permutation

Key (56 bits)
S-Box Substitution

- S-Box heart of DES security
- S-Box: 4x16 entry table
  - Input 6 bits:
    - 2 bits: determine the table (1/4)
    - 4 bits: determine the table entry
  - Output: 4 bits
- S-Boxes are optimized against Differential cryptanalysis
Double/Triple DES

Double DES
- Vulnerable to Meet-in-the-Middle Attack [DH77]

Triple DES
- Used two keys $K_1$ and $K_2$
- Compatible with simple DES ($K_1=K_2$)
- Used in ISO 8732, PEM, ANS X9.17

CSU610: SWARM

Cryptography 24
Linear/Differential Cryptanalysis

Differential cryptanalysis
- “Rediscovered” by E. Biham & A. Shamir in 1990
- Based on a chosen-plaintext attack:
  - Analyze the difference between the ciphertexts of two plaintexts which have a known fixed difference
  - The analysis provides information on the key
- 8-round DES broken with $2^{14}$ chosen plaintext
- 16-round DES requires $2^{47}$ chosen plaintext
- DES design took into account this kind of attacks

Linear cryptanalysis
- Uses linear approximations of the DES cipher (M. Matsui 1993)
- IDEA first proposal (PES) was modified to resist to this kind of attacks
- GSM A3 algorithm is sensitive to this kind of attacks
  - SIM card secret key can be recovered $\Rightarrow$ GSM cloning
Breaking DES

Electronic Frontier Foundation built a “DES Cracking Machine” [1998]

- Attack: brute force
- Inputs: two ciphertext
- Architecture:
  - PC
  - array of custom chips that can compute DES
    - 24 search units/chip x 64 chips/board x 27 boards
- Power:
  - searches 92 billion keys per second
  - takes 4.5 days for half the key space
- Cost:
  - $130’000 (all the material: chips, boards, cooling, PC etc.)
  - $80’000 (development from scratch)
International Data Encryption Algorithm (IDEA)

Developed by Xu Lai & James Massey (ETH Zurich, Switzerland)

Characteristics:
- 64-bits block cipher
- 128-bits key length
- Uses three algebraic groups: XOR, + mod $2^{16}$, x mod $2^{16}+1$
- 17 rounds (or 8 rounds according to the description)

Speed: software: 2 times faster than DES

Used in PGP

Patented (expires in 2011)
The Advanced Encryption Standard (AES) Cipher - Rijndael

- Designed by Rijmen-Daemen (Belgium)
- Key size: 128/192/256 bit
- Block size: 128 bit data
- Properties: **iterative** rather than **Feistel** cipher
  - Treats data in 4 groups of 4 bytes
  - Operates on an entire block in every round
- Designed to be:
  - Resistant against known attacks
  - Speed and code compactness on many CPUs
  - Design simplicity
AES

State: 16 bytes structured in a array

<table>
<thead>
<tr>
<th>$S_{0,0}$</th>
<th>$S_{0,1}$</th>
<th>$S_{0,2}$</th>
<th>$S_{0,3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{1,0}$</td>
<td>$S_{1,1}$</td>
<td>$S_{1,2}$</td>
<td>$S_{1,3}$</td>
</tr>
<tr>
<td>$S_{2,0}$</td>
<td>$S_{2,1}$</td>
<td>$S_{2,2}$</td>
<td>$S_{2,3}$</td>
</tr>
<tr>
<td>$S_{3,0}$</td>
<td>$S_{3,1}$</td>
<td>$S_{3,2}$</td>
<td>$S_{3,3}$</td>
</tr>
</tbody>
</table>

Each byte is seen as an element of $\mathbf{F}_{2^8} = \text{GF}(2^8)$

- $\mathbf{F}_{2^8}$ finite field of 256 elements
- Operations
  - Elements of $\mathbf{F}_{2^8}$ are viewed as polynomials of degree 7 with coefficients $\{0, 1\}$
  - Addition: polynomials addition $\Rightarrow$ XOR
  - Multiplication: polynomials multiplication modulo $x^8 + x^4 + x^3 + x + 1$
AES Outline

1. **Initialize** \( \text{State} \leftarrow x \oplus \text{RoundKey}; \)

2. **For each of the Nr-1 rounds:**
   1. \text{SubBytes}(\text{State});
   2. \text{ShiftRows}(\text{State});
   3. \text{MixColumns}(\text{State});
   4. \text{AddRoundKey}(\text{State});

3. **Last round:**
   1. \text{SubBytes}(\text{State});
   2. \text{ShiftRows}(\text{State});
   3. \text{AddRoundKey}(\text{State});

4. **Output** \( y \leftarrow \text{State} \)
Implementation Aspects

- Can be efficiently implemented on 8-bit CPU
  - byte substitution works on bytes using a table of 256 entries
  - shift rows is a simple byte shifting
  - add round key works on byte XORs
  - mix columns requires matrix multiply in GF($2^8$) which works on byte values, can be simplified to use a table lookup
Implementation Aspects

Can be efficiently implemented on 32-bit CPU
  - redefine steps to use 32-bit words
  - can pre-compute 4 tables of 256-words
  - then each column in each round can be computed using 4 table lookups + 4 XORs
  - at a cost of 16Kb to store tables

Designers believe this very efficient implementation was a key factor in its selection as the AES cipher
Encryption Modes: Electronic Codebook (ECB)
Encryption Modes: Cipher Block Chaining (CBC)
Encryption Modes: Cipher Feedback (CFB)
Encryption Modes: Output Feedback (OFB)
Counter (CTR)

- Similar to OFB but encrypts counter value rather than any feedback value
- Must have a different key & counter value for every plaintext block (never reused)
  
  \[ C_i = P_i \oplus O_i \]
  
  \[ O_i = \text{DES}_{k_1}(i) \]

- Uses: high-speed network encryptions, random access to files
Inside vs. Outside CBC-3DES

What is the impact of using 3DES with CBC on the outside vs. inside?
Message Authentication Code (MAC) Using an Encryption Algorithm

Also called Message Integrity Code (MIC)

Goal:

- Detect any modification of the content by an attacker

Some techniques:

- Use CBC mode, send only the last block (residue) along with the plaintext message
- For confidentiality + integrity:
  - Use two keys (one for CBC encryption and one for CBC residue computation)
  - Append a cryptographic hash to the message before CBC encryption
- New technique: use a Nested MAC technique such as HMAC
Hashes and MessageDigests

Goal:
- Input: long message
- Output: short block (called hash or message digest)
- Property: given a hash $h$ it is computationally infeasible to find a message that produces $h$

Examples: http://www.slavasoft.com/quickhash/links.htm
- Secure Hash Algorithm (SHA-1, SHA-2) by NIST
- MD2, MD4, and MD5 by Ron Rivest [RFC1319, 1320, 1321]
- SHA-1: output 160 bits
- SHA-2: output 256-384-512 believed to be more secure than others

Uses:
- MAC: How? Problems? ... HMAC
- Authentication: how?
- Encryption: how?
HMAC

\[ \text{HMAC}_K(x) = \text{SHA-1}((K \oplus \text{opad}) \mid \text{SHA-1}((K \oplus \text{ipad}) \mid x)) \]

\[ \text{ipad} = 3636...36; \text{opad} = 5C5C...5C \]

Assumption:

\[ \text{SHA-1 restricted to one application is a secure MAC} \]
Message Digest 5 (MD5) by R. Rivest [RFC1321]

- Input: message of arbitrary length
- Output: 128-bit hash
- Message is processed in blocks of 512 bits (padding if necessary)
- Security:
  - Designed to resist to the Birthday attack
  - Collisions where found in MD5, SHA-0, and almost found for SHA-1
  - Near-Collisions of SHA-0, Eli Biham, Rafi Chen, Proceedings of Crypto 2004
  - http://www.cs.technion.ac.il/~biham/publications.html
  - Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD
  - Xiaoyun Wang and Dengguo Feng and Xuejia Lai and Hongbo Yu
Birthday Attacks

- Is a 64-bit hash secure?
  - Brute force: 1ns per hash $=> 10^{13}$ seconds over 300 thousand years
  - But by Birthday Paradox it is not
- Example: what is the probability that at least two people out of 23 have the same birthday? $P > 0.5$
- **Birthday attack technique**
  - opponent generates $2^{m/2}$ variations of a valid message all with essentially the same meaning
  - opponent also generates $2^{m/2}$ variations of a desired fraudulent message
  - two sets of messages are compared to find pair with same hash (probability $> 0.5$ by birthday paradox)
  - have user sign the valid message, then substitute the forgery which will have a valid signature
- Need to use larger MACs
Public Key Systems
Asymmetric cryptosystems

- Invented by Diffie and Hellman [DH76]
  - When DES was proposed for standardization
- Asymmetric systems are much slower than the symmetric ones (~1000 times)
- Advantages:
  - does not require a shared key
  - simpler security architecture (no-need to a trusted third party)
Modular Arithmetic

- Modular addition:
  - E.g., $3 + 5 = 1 \mod 7$

- Modular multiplication:
  - E.g., $3 \times 4 = 5 \mod 7$

- Modular exponentiation:
  - E.g., $3^3 = 6 \mod 7$

- Group, Rings, Finite/Galois Fields ...
**RSA Cryptosystem [RSA78]**

- \( E(M) = M^e \mod n = C \) (Encryption)
- \( D(C) = C^d \mod n = M \) (Decryption)

**RSA parameters:**

- \( p, q, \text{ two big prime numbers} \) (private, chosen)
- \( n = pq, \phi(n) = (p-1)(q-1) \) (public, calculated)
- \( e, \text{ with } \gcd(\phi(n), e) = 1, \ 1 < e < \phi(n) \) (public, chosen)
- \( d = e^{-1} \mod \phi(n) \) (private, calculated)

- \( D(E(M)) = M^d \mod n = M^\phi(n)+1 = M \) (Euler’s theorem)
Prime Numbers Generation

Density of primes (prime number theorem):

\[ \pi(x) \sim x/\ln(x) \]

Sieve of Erathostène

Try if any number less than SQRT(n) divides n

Fermat’s Little Theorem does not detect Carmichael numbers

\[ b^{n-1} = 1 \mod n \]

Solovay-Strassen primality test

If \( n \) is not prime at least 50\% of \( b \) fail to satisfy the following:

\[ b^{(n-1)/2} = J(b, n) \mod n \]

Rabin-Miller primality test

If \( n \) is not prime then it is not pseudoprime to at least 75\% of \( b < n \):

Pseudoprime: \( n-1 = 2^st, \ b^s = \pm1 \mod n \ \text{OR} \ b^{2^r} = -1 \mod n \) for some \( r < s \)

Probabilistic test, deterministic if the Generalized Riemann Hypothesis is true

Deterministic polynomial time primality test [Agrawal, Kayal, Saxena’2002]
Use of RSA

Encryption (A wants to send a message to B):
- A uses the public key of B and encrypts \( M \) (i.e., \( E_B(M) \))
- Since only B has the private key, only B can decrypt \( M \) (i.e., \( M = D_B(M) \))

Digital signature (A want to send a signed message to B):
- Based on the fact that \( E_A(D_A(M)) = D_A(E_A(M)) \)
- A encrypts \( M \) using its private key (i.e., \( D_A(M) \)) and sends it to B
- B can check that \( E_A(D_A(M)) = M \)
- Since only A has the decryption key, only can generate this message
Diffie-Hellman Key Exchange

<table>
<thead>
<tr>
<th>Private: A</th>
<th>Public</th>
<th>Private: B</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$p$: prime number, $a$: primitive element of $GF(p)$</td>
<td>$y$</td>
</tr>
<tr>
<td>compute:</td>
<td>compute:</td>
<td>receive:</td>
</tr>
<tr>
<td>$a^x \mod p$</td>
<td>$a^y \mod p$</td>
<td>$a^y \mod p$</td>
</tr>
<tr>
<td>receive:</td>
<td>receive:</td>
<td>Compute shared key:</td>
</tr>
<tr>
<td>$a^y \mod p$</td>
<td>$a^x \mod p$</td>
<td>$(a^y)^x \mod p$</td>
</tr>
</tbody>
</table>

- Based on the difficulty of computing discrete logarithms
- Works also in extension Galois fields: $GF(p^q)$
Attack on Diffie-Hellman Scheme: Public Key Integrity

**Man-in-the-Middle Attack**

A

\[ x \]

\[ a^x \]

I (intruder)

\[ z \]

\[ a^z \]

B

\[ y \]

\[ a^y \]

\[ a^z \]

Shared key: \( K_{AI} = a^{xz} \)

Shared key: \( K_{BI} = a^{yz} \)

Message encrypted using \( K_{AI} \)

Decrypt using \( K_{AI} + K_{BI} \)

---

- Need for a mean to verify the public information: certification
- Another solution: the Interlock Protocol (Rivest & Shamir 1984)
El Gamal Scheme

**Parameters:**
- \( p \): prime number (public, chosen)
- \( g < p \): random number (public, chosen)
- \( x < p \): random number (private, chosen)
- \( y = g^x \mod p \) (public, computed)

**Encryption of message \( M \):**
- choose random \( k < p-1 \)
- \( a = g^k \mod p \)
- \( b = y^k M \mod p \)

**Decryption:**
- \( M = b / y^k \mod p = b / g^{xk} \mod p = b / a^x \)

**Message signature**
- choose random \( k \) relatively prime with \( p-1 \)
- find \( b \): \( M = (xa + kb) \mod (p-1) \) (extended Euclid algorithm)
- signature\( (M) = (a, b) \)
- verify signature: \( y^a a^b \mod p = g^M \mod p \)
Knapsack

- Introduced by R. Merkle
- Based on the difficulty of solving the Knapsack problem in polynomial time (Knapsack is an NP-complete problem)
  - cargo vector: \( a = (a_1, a_2, ..., a_n) \) (seq. Int)
  - plaintext msg: \( x = (x_1, x_2, ..., x_n) \) (seq. Bits)
  - ciphertext:
    \[ S = a_1x_1 + a_2x_2 + ... + a_nx_n \]
  - \( a_i = wa'_i \) such that \( a'_1 + a'_2 + ... + a'_{i-1}, m > a'_1 + ... + a'_n \)
  - \( w \) is relatively prime with \( m \)
- One-round Knapsack was broken by A. Shamir in 1982
- Several variations of Knapsack were broken
Others

- Elliptic Curve Cryptography (ECC)
- Zero Knowledge Proof Systems
Security Services

Confidentiality:
- Use an encryption algorithm
- Generally a symmetric algorithm

Integrity:
- MAC algorithm

Access control:
- Use access control tables

Authentication
- Use authentication protocols

Non-repudiation
Questions

- How many keys are derived in DES?
- How do rounds relate to the key size in AES?
- Is the decryption process exactly the same as the encryption process for DES? AES?
- If a bit error occurs in the transmission of a ciphertext character in 8-bit CFB mode how far does it propagate?