CSE508  Network Security

2/24/2016  Encrypted Communication  (Part 1)

Michalis Polychronakis
Stony Brook University
Cryptography
Goals

Confidentiality
Keep content secret from all but authorized entities

Integrity
Protect content from unauthorized alteration

Authentication
Confirm the identity of communicating entities or data

Non-repudiation
Prevent entities from denying previous commitments or actions
Basic Terminology

**Plaintext:** top secret message

**Ciphertext:** eza dpncpe xpddIrp

**Cipher:** algorithm for transforming plaintext to ciphertext (*encryption*) and back (*decryption*)

**Key:** (usually secret) information used in a cipher, known to sender, receiver, or both

**Cryptanalysis (codebreaking):** the study of methods of deciphering ciphertext without knowing the key

**Cryptology:** the broader field of “information hiding” cryptography, cryptanalysis, steganography, …
Plaintext vs. Ciphertext

Plaintext → Encryption algorithm → Ciphertext

Decryption algorithm ← Ciphertext

Plaintext ← Decryption algorithm ← Ciphertext

Key_E

Key_D
Cryptographic Function Types

**Hash functions:** no key

Input of arbitrary length is transformed to a fixed-length value
One-way function: hard to reverse

**Secret (symmetric) key functions:** one key

Shared secret key is used for both encryption and decryption

**Public (asymmetric) key functions:** two keys

*Key pair:* public key is known, private key is kept secret
Encrypt with public key and decrypt with private key
Encrypt with private key and decrypt with public key
Kerckhoffs's Principle

A cryptosystem should be secure even if everything about the system, except the key, is public knowledge

The security of the system must rest entirely on the secrecy of the key

- Only brute force attacks are possible
- Otherwise the algorithm is broken

Contrast with security by obscurity: every secret creates a potential failure point

- Widely used secret algorithms would be eventually reverse engineered
- Difficult to deploy a new algorithm if an old one is compromised

A public implementation enables scrutiny by experts
Caesar Cipher

Ciphertext: WKH TXLFN EURZQ IRA MXPSV RYHU WKH ODCB GRJ
Plaintext:  the quick brown fox jumps over the lazy dog

Shift by X (e.g., ROT-13)

Monoalphabetic substitution
Easy to break using frequency analysis

Distribution of letters in a typical sample of English language text
Vigenère Cipher

Plaintext: ATTACKATDAWN
Key: LEMONLEMONLE
Ciphertext: LXFOPVEFRNHR

Successive Caesar ciphers with different shift values depending on a key

Defeats simple frequency analysis, but still breakable

Polyalphabetic substitution
Properties of a Good Cryptosystem

Given the ciphertext, an adversary should not be able to recover the original message

- Enumerating all possible keys must be infeasible
- There should be no way to produce plaintext from ciphertext without the key

The ciphertext must be indistinguishable from true random values

- Given a ciphertext, the probability of any possible plaintext being encrypted should be the same

Cryptographic algorithms should be computationally efficient for practical use

- Fast encryption/decryption/hashing
- There are exceptions (e.g., deliberately slow password-based key derivation functions to hinder brute force/dictionary attacks)
Computational Difficulty

Modern cryptography: seek guarantees about the “strength” of encryption schemes

Codes, secret writing, and other older encryption schemes were ad hoc and eventually broken

Information-theoretic security

Cannot be broken even with unlimited computing power: there is simply not enough information

Not possible if the key is shorter than the message size ➔ impractical

Computational security

Can be broken with enough computation, but not in a reasonable amount of time

Rely on computationally hard problems: easy to compute but hard to invert in polynomial time (integer factorization, discrete logarithm, …)

Assume computationally limited adversaries ➔ frustrate exhaustive enumeration
One-time Pad

XOR plaintext with a keystream

1882 Frank Miller [Bellovin ’11]
1917 Vernam/Mauborgne cipher

Information-theoretically secure against ciphertext-only attacks (Shannon 1949)

The keystream must be

Truly random
As long as the plaintext
Kept completely secret
Used only once…
SEND CASH + K₁ = E₁

Smiley + K₁ = E₂

E₁ + E₂ = SEND CASH
Basic Attack Models

**Known Ciphertext:** attacker has access to only a set of ciphertexts

In practice some information about the plaintext might be available: language, character distribution, protocol fields, …
Brute force frequency analysis, …

**Known Plaintext:** attacker has access to both the plaintext and its corresponding ciphertext

*Passive attacker:* has at least one sample of both
Even partial mappings can be enough

**Chosen Plaintext:** attacker can obtain the ciphertexts of arbitrary plaintexts

*Active attacker:* has access to an encryption oracle
Symmetric Key Cryptography

**Pros:**
- Fast
- Short keys
- Well known
- Simple key generation

**Cons:**
- Secrecy of keys
- Number of keys
- Management of keys
Block Ciphers

Process one block at a time
Substitution and transposition (permutation) techniques
Examples: DES, AES, …

Stream Ciphers

Process one bit or byte at a time
Plaintext is combined (XOR) with a pseudorandom keystream (NOT the same as one-time pad)
Synchronous vs. asynchronous (self-synchronizing)
Examples: RC4, any block cipher in OFB or CTR mode, …
Block Ciphers

Multiple rounds of substitution, permutation, …

*Confusion:* each character of the ciphertext should depend on several parts of the key

*Diffusion:* changing a plaintext character should result in several changed ciphertext characters

<table>
<thead>
<tr>
<th></th>
<th>DES</th>
<th>AES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key length</td>
<td>56 bits</td>
<td>128, 192, 256 bits</td>
</tr>
<tr>
<td>Block size</td>
<td>64 bits</td>
<td>128 bits</td>
</tr>
<tr>
<td>Rounds</td>
<td>16</td>
<td>10, 12, 14</td>
</tr>
<tr>
<td>Construction</td>
<td>Substitution, permutation</td>
<td>Substitution, permutation, mixing, addition</td>
</tr>
<tr>
<td>Developed</td>
<td>1977</td>
<td>1998</td>
</tr>
<tr>
<td>Status</td>
<td>Broken!</td>
<td>OK (for now)</td>
</tr>
</tbody>
</table>
Modes of Operation

Direct use of block ciphers is not very useful

Enemy can build a “code book” of plaintext/ciphertext equivalents

Message length should be multiple of the cipher block size

How to repeatedly apply a block cipher to securely encrypt/decrypt arbitrary inputs?

Five standard modes

ECB: Electronic Code Book
CBC: Cipher Block Chaining
CFB: Cipher Feedback
OFB: Output Feedback
CTR: Counter
ECB: Electronic Code Book Mode

Direct use of the block cipher

Each block is encrypted independently -> parallelizable
No chaining, no error propagation

\[
\begin{align*}
& m_{i-1} \\
& \downarrow \\
& E_k \\
& \downarrow \\
& c_{i-1} \\

& m_i \\
& \downarrow \\
& E_k \\
& \downarrow \\
& c_i \\

& m_{i+1} \\
& \downarrow \\
& E_k \\
& \downarrow \\
& c_{i+1} \\

& \ldots \\
& m_i = m_j \text{ then } c_i = c_j
\end{align*}
\]
ECB: Electronic Code Book Mode

Data patterns may remain visible
Susceptible to replay attacks, block insertion/deletion

Plaintext  ECB Mode Encryption  CBC/Other Modes
CBC: Cipher Block Chaining Mode

Each plaintext block is XORed with the previous ciphertext block before being encrypted -> obscures any output patterns

Sequential process (non-parallelizable)

Ensures that no messages have the same beginning

Must be random! Must never be reused!
CBC: Decryption

An error in a transmitted ciphertext block also affects its following block

Both parties must use the same IV: can be transmitted with the message
CTR: Counter Mode

Turns a block cipher into a stream cipher

Next keystream block is generated by encrypting successive values of a counter combined with a nonce (IV)