Cryptography 2017
Lecture 4

- Attacks against Block Ciphers
- Introduction to Public Key Cryptography
What have seen? What are we discussing today? What is coming later?

**Lecture 3**

- PRGs, PRFs and PRPs
- Block Ciphers (definition)
- Block Ciphers (examples: DES, AES)
- Block Ciphers (modes of operation: ECB, CBC, CTR)

**Lecture 4**

- Attacks against Block Cipher Modes
- Intro to Public Key Cryptography
- Public Key Infrastructures - Certification Authorities
- Textbook RSA

**Lecture 5**

- PKE - Digital Signatures
- Prime number - Primality Tests
Reminder Block ciphers

Definition

A **block cipher** is a cipher \((E, D)\) where:

\[ E : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n \]

and for each \( K \in \mathcal{K} = \{0, 1\}^k \) and \( m \in \mathcal{M} = \{0, 1\}^n \),

\( E(K, m) \) is invertible and it holds:

\[ D(K, c) = E^{-1}(K, c) \]

Definition

A PRF \( \mathcal{F} \) is called **Pseudo Random Permutation (PRP)** if it holds:

1. \( \mathcal{M} = \mathcal{C} \) (i.e., the sets of the plaintext and the ciphertext are the same)
2. the function \( \mathcal{F}(K, m) \) is **one-to-one**
3. there exists an efficient, **deterministic** algorithm to compute \( \mathcal{F}(K, m) = E(K, m) \) for any message \( m \in \mathcal{M} \).
4. there exists an efficient, **deterministic** algorithm to compute:

\[ \mathcal{F}^{-1}(K, c) = D(K, c) \], for any ciphertext \( c \in \mathcal{C} \).
Performance - How fast are Block Ciphers?

Run in an AMD Opteron, 2.2 GHz (Linux) Using Crypto++ library

<table>
<thead>
<tr>
<th>Cipher</th>
<th>Block/key size</th>
<th>Speed (MB/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stream Ciphers</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RC4</td>
<td>-</td>
<td>126</td>
</tr>
<tr>
<td>Salsa20/12</td>
<td>-</td>
<td>643</td>
</tr>
<tr>
<td><strong>Block Ciphers</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3DES</td>
<td>64/168</td>
<td>13</td>
</tr>
<tr>
<td>AES-128</td>
<td>128/168</td>
<td>109</td>
</tr>
</tbody>
</table>

Info from Dan Boneh’s online course

Comparative execution times (in seconds) of encryption algorithms in ECB mode on a P-4 2.4 GHz machine.

How to get a PRG from a PRF?

- Let $F : \mathcal{K} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ be a secure PRF.
- Let us define $G : \mathcal{K} \rightarrow \{0, 1\}^{nt}$ as:

$$G(k) = F(k, 0) || F(k, 1) || F(k, 2) || \ldots || F(k, t)$$

- Then, $G$ is a secure PRG and it is parallelizable!

Why?

- If an adversary was able to distinguish the output of $G(k)$ from a truly random string $r \in \{0, 1\}^{nt}$, then the same adversary would be able to distinguish $F$ from a truly random function.
- But this contradicts the assumption that $F$ is a secure PRF!

Remember: Used in Counter Mode in Block Ciphers
Reminder: Modes of operation - CTR (deterministic counter mode)

Let $F$ be a secure PRF. $F : \mathcal{K} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$

The CTR block cipher is defined as follows: pick a random $IV \in \{0, 1\}^{nt}$ and do

<table>
<thead>
<tr>
<th>IV</th>
<th>m[0]</th>
<th>m[1]</th>
<th>...</th>
<th>m[L]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F(k, IV)$</td>
<td>$F(k, IV+1)$</td>
<td>...</td>
<td>$F(k, IV+L)$</td>
<td></td>
</tr>
</tbody>
</table>

| IV | c[0] | c[1] | ... | c[L] |

Remember: The IV is chosen at random for every message!

Even if I encrypt the same message twice I will get different ciphertexts!

Note: parallelizable (unlike CBC)

To guarantee $F(k, x)$ is never used more than once, choose $IV$ as:

IV + counter: 128 bits

\begin{array}{ll}
\text{nonce} & 64 \text{ bits} \\
\text{counter} & 64 \text{ bits} \\
\end{array}

starts at 0 for every msg

Picture from Dan Boneh’s online course.
Nonce based CTR
Let us consider that we use CTR to encrypt a message and the IV has length 128 bits while the nonce has length 64 bits. How many blocks can you encrypt with one nonce to guarantee security (i.e., a message will not be encrypted using the same IV)??

A $2^{32}$
B $2^{64}$
C $2^{128}$
D Unlimited!

If it is greater than $2^{64}$ then two blocks will be encrypted with the same one time pad since the counter of CTR will be reset after $2^{64}$ blocks.
Reminder: ECB (Electronic Code Book) mode

**Attention:** The ECB block cipher is *not semantically secure* for messages longer than one block.

Two equal blocks of plaintext produce two equal blocks of ciphertext!

![Diagram of ECB mode](image)

long message

<table>
<thead>
<tr>
<th>$M_4$</th>
<th>$M_3$</th>
<th>$M_2$</th>
<th>$M_1$</th>
<th>$M_0$</th>
</tr>
</thead>
</table>

Block Cipher

<table>
<thead>
<tr>
<th>$C_4$</th>
<th>$C_3$</th>
<th>$C_2$</th>
<th>$C_1$</th>
<th>$C_0$</th>
</tr>
</thead>
</table>

ciphertext as long as the message
Reminder: The Semantic Security Game

Attacker $A$

$\begin{align*}
 m_0, m_1 &\overset{D}{\leftarrow} \mathcal{M} \\
 \text{len}(m_0) &= \text{len}(m_1) \\
\end{align*}$

\[ k \in \mathcal{K} \]

Challenger $C$

$\begin{align*}
 c &= \mathbf{E}(k, m_b) \\
 b &\overset{\text{chosen uniformly at random}}{\leftarrow} \{0, 1\} \\
\end{align*}$

if $b=0$, $c$ is the encryption of $m_0$
if $b=1$, $c$ is the encryption of $m_1$

Output $b' \in \{0, 1\}$
a guess for $b$

Useful for the exercises/exams!!

Definition

A cipher $(\mathbf{E}, \mathbf{D})$ is **semantically secure** (with one time key) if for any ‘efficient’ adversary, it holds:

\[ P(b' = b) < \frac{1}{2} + \text{negligible} \]
Reminder: The Semantic Security Game

Attacker $A$

$\begin{array}{l}
m_0, m_1 \overset{\text{P}}{\leftarrow} \mathcal{M} \\
\text{len}(m_0) = \text{len}(m_1)
\end{array}$

$\begin{array}{l}
\text{Output } b' \in \{0, 1\} \\
a \text{ guess for } b
\end{array}$

Useful for the exercises/exams!!

Challenger $C$

$\begin{array}{l}
k \in \mathcal{K} \\
\text{chosen uniformly at random}
\end{array}$

$\begin{array}{l}
c = \mathbf{E}(k, m_b)
\end{array}$

if $b=0$, $c$ is the encryption of $m_0$
if $b=1$, $c$ is the encryption of $m_1$

Let $W_0$ be the event that $C$ chooses $b = 0$, and $A$ outputs $b' = 0$.

Let $W_1$ be the event that $C$ chooses $b = 1$, and $A$ outputs $b' = 0$.

Definition

A cipher $(\mathbf{E}, \mathbf{D})$ is **semantically secure** (with one time key) if for any ‘efficient’ adversary, it holds:

$$|P(W_0) - P(W_1)| \text{ is negligible}$$
ECB is not semantically secure

**Useful for the exercises/exam!**

**Exercise:** Prove that the ECB block cipher is not semantically secure when a message is longer than one block.

**Solution:** Describe how ECB works and then the strategy of the attacker:

Attacker $A$  
Challenger $C$

\[
\begin{align*}
&\text{two blocks} \\
&m_0 = \text{Hello World} \\
&m_1 = \text{Hello Hello} \\
&c_1 || c_2 = E_{\text{ECB}}(k, m_b) \\
&\text{len}(m_0) = \text{len}(m_1) \\
&\text{Output } b' \in \{0, 1\} \\
&\text{a guess for } b
\end{align*}
\]

Let $W_0$ be the event that $C$ chooses $b = 0$, and $A$ outputs $b' = 0$.

Let $W_1$ be the event that $C$ chooses $b = 1$, and $A$ outputs $b' = 0$.

When $C$ chooses $b = 0$, since $m_0 = \text{Hello World}$ it holds $c_1 \neq c_2$ then $b' = 0$

When $C$ chooses $b = 1$, since $m_1 = \text{Hello Hello}$ it holds $c_1 = c_2$ then $b' = 1$

$A$ can output $b' = 0$ when $c_1 \neq c_2$ and $b' = 1$ when $c_1 = c_2$.

Then, we have: $|P(W_0) - P(W_1)| = |1 - 0| = 1$
The CBC block cipher is not secure against \textit{chosen plaintext attacks (CPA)} if the initialisation vector (IV) is \textit{predictable} (e.g., a counter).
Chosen Plaintext Attack (CPA) - Secret Key Crypto

**Attacker \( \mathcal{A} \)**

- for \( i = 1, \ldots q \) do
  - choose \( \bar{m}_i \in \mathcal{M} \)
  - learn \( \bar{c}_i \)

**Challenge Phase**

- \( m_0, m_1 \leftarrow \mathcal{D}_\mathcal{M} \)
- \( \text{len}(m_0) = \text{len}(m_1) \)

**Output**

- Output \( b' \in \{0, 1\} \)
- a guess for \( b \)

**Definition**

A cipher \((E,D)\) is secure under CPA if for any PPT adversary, it holds that:

\[
P(b' = b) < \frac{1}{2} + \text{negligible}
\]
Exercise: Prove that the CBC block cipher with predictable IV is not secure against CPA.

Solution: Describe how CBC works and then the strategy of the attacker.

Attacker $\mathcal{A}$

Choose $m = (0, \ldots, 0)$

learn $\bar{c}$

Predict $\text{IV}(\text{IV}_1) = \text{IV}_2$

$m_0 = \text{IV}_1 \oplus \text{IV}_2$

pick $m_1 \neq m_0$ with $\text{len}(m_1) = \text{len}(m_0)$

If $c = [\text{IV}_2, \bar{c}_2]$ output $b' = 0$

else, output $b' = 0$

Challenge Phase

$m_0, m_1$

$c = E_{\text{CBC}}(k, m_b)$

Challenger $\mathcal{C}$

$k \in \mathcal{K}$

$b \leftarrow \{0, 1\}$

$c = [\text{IV}_1, E(k, (0, \ldots, 0) \oplus \text{IV}_1)]$

$= [\text{IV}_1, E(k, \text{IV}_1)] = [\bar{c}_1, \bar{c}_2]$

Let $W_0$ be the event that $\mathcal{C}$ chooses $b = 0$, and $\mathcal{A}$ outputs $b' = 0$.

Let $W_1$ be the event that $\mathcal{C}$ chooses $b = 1$, and $\mathcal{A}$ outputs $b' = 0$.

Then, it holds: $|\mathbb{P}(W_0) - \mathbb{P}(W_1)| = |1 - 0| = 1$
Overview - What have we covered in Symmetric Crypto?

Alice

Plaintext

See you today at 5pm

Encryption

Secret key

Ciphertext

Bob

Decryption

Secret key

Ciphertext

See you today at 5pm

Encryption Schemes

- Substitution ciphers
- One Time Pad (OTP)
- Block Ciphers (AES, DES)
- Modes of operation (ECB, CBC, CTR)

Useful tools/primitives

- Distributions/Probabilities
- Pseudorandom Generators (PRG)
- Pseudorandom Functions (PRF)
- Pseudorandom Permutations (PRP)

Provable Security

- Semantic Security
- Chosen Plaintext Attacks (CPA)
- Meet in the Middle Attack
Secret Key Cryptography - Pros & Cons

**Symmetric key encryption** (also known as secret key crypto) is using the **same key** to encrypt and decrypt data.

**Pros**

- It can be very **secure** (if the key remains secret) and
- **Very fast**!

**Cons**

- Need for a secure channel to transfer the secret key
- **Does not scale well**: I have to keep track of all secret keys for all parties I want to communicate with!

For secure communication among **5 parties** we need **10 keys**!
Quiz question!

Go to: http://socrative.com/
and use the code below or scan the QR code.
Student Login
Classroom: CRYPTOCHALMERS

Number of Secret Keys! - Secret Key Crypto

How many secret keys do we need for secure communication among $n$ parties in secret key cryptography?

1. $n^2$
2. $2n$
3. $\frac{n(n-1)}{2}$
4. $\frac{n-1}{2}$
New Directions in Cryptography

Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER,

Abstract—Two kinds of contemporary developments in cryptog- 
rophy are examined. Widening applications of teleprocessing 
have given rise to a need for new types of cryptographic systems, 
which minimize the need for secure key distribution channels and 
supply the equivalent of a written signature. This paper suggests 
ways to solve these currently open problems. It also discusses how 
the theories of communication and computation are beginning to 
provide the tools to solve cryptographic problems of long stand-

I. INTRODUCTION

We stand today on the brink of a revolution in cryptography. The development of cheap digital 
hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade 
cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers 
and computer terminals. In turn, such applications create a need for new types of cryptographic systems which

The best known cryptographic method for the protection of pri-

vacy: preventing the unauthorized interception of information from communications over an insecure channel. In 
order to use cryptography to insure privacy, however, it is 
currently necessary for the communicating parties to share 
a key which is known to no one else. This is done by send-
ing the key in advance over some secure channel such as 
private courier or registered mail. A private conversation 
between two people with no prior acquaintance is a com-
mon occurrence in business, however, and it is unrealistic 
to expect initial business contacts to be postponed long 

The cost and delay imposed by this key distribution 
problem is a major barrier to the transfer of business 
communications to large teleprocessing networks.

Section III proposes two approaches to transmitting 
keying information over public (i.e., insecure) channels 
without compromising the security of the system. In a 
public key cryptosystem enciphering and deciphering are 
governed by distinct keys, E and D, such that computing
Introduction to Public Key Encryption

Alice

Plaintext

See you today at 5pm

Encryption

Ciphertext

Bob's Public Key

Secret key

Bob

Decryption

Ciphertext

See you today at 5pm

Bob's Private Key

Secret key

Bob's Private Key

Secret key

Bob

◮ **Intuition:** **Anyone** can encrypt a message for Bob (using Bob’s **public key**).
◮ Only Bob with his **private (secret) key**, can decrypt correctly!
◮ The public key is **known** to everyone!
◮ The private (secret) key must remain secret!
◮ The **public key** depends on the **secret** key!
Public Key Encryption

Alice needs **one pair of public** and **secret keys** to securely communicate with **multiple parties**!

Alice needs one pair of public and secret keys to securely communicate with multiple parties!
Public Key Encryption

Bob’s key **will not work** to decrypt Charlie’s message.

**Alice**

Charlie's Public Key

Bob's Public Key

**Bob**

Charlie's Public Key

Bob's Private Key

**Charlie**

Hi Charlie, ok see you today at 3pm

Hi Bob, let's have a tea next week

Charlie's Public Key

Bob's Private Key

**Charlie**
Uses of PKC

- Encrypted emails/messages (PGP - Assignment 1)
- Onion routing (TOR - anonymity of Web)

- Establishing secure Channels /connections (SSL/TLS) (covered in the network security course).
- Certificates/ digital signatures (for data authentication/integrity)

**Important:** Will be covered in detail in later lectures!
Randomised Algorithm

- Let $R(x, r)$ be an algorithm that takes...
- $x$ as input and $r$ are some random bits and produces an output.
- Every time we run $R(x, r)$ the randomness (random bits) change.
- Thus, for the same input $x$ we always have different output!
Public Key Encryption (PKE): Definition

A public-key encryption scheme is a triple of efficient algorithm \((\text{KeyGen}, \text{Enc}, \text{Dec})\) defined as follows:

- **KeyGen**\((\lambda) \rightarrow (pk, sk)\) : is a randomised key generation algorithm which given a security parameter outputs a public and a private key.

- **Enc**\((pk, m) \rightarrow c\) : is a randomised encryption algorithm that, given a public key \(pk\) and a message \(m\) it outputs a ciphertext \(c\).

- **Dec**\((sk, c) \rightarrow m\) : is a randomised encryption algorithm that, given the secret key \(sk\) and a ciphertext, it outputs the corresponding plaintext \(m\).

**Randomised Encryption**

- Encrypting the same message twice gives different ciphertexts (with high probability).

- This implies that the ciphertext will be longer than the plaintext i.e., roughly it holds: \(\text{size of ciphertext} = \text{size of plaintext} + \text{number of random bits}\)
PKE: basic properties

**Correctness Property:**
If $c = \text{Enc}(pk, m)$, then $\text{Dec}(sk, c) = m$ for all messages $m$ (defined in the message space) and all keys pairs $(pk, sk)$.

We can rewrite this property as follows: $\text{Dec}(sk, \text{Enc}(pk, m)) = m$

**Security:**
We will see Chosen Plaintext Attacks (CPA) and Chosen Ciphertext Attacks (CCA) in an upcoming lecture!
PKE: security intuition!

- Public key (pk) should be **easy to compute** from the secret key (sk)
- Secret key (sk) should be **hard to compute** from the public key (pk)
- Usually rely on some “**hard**” problem!
- Usually this is a **computationally hard** problem i.e., a very computationally powerful server could solve the problem.
Preview: Signatures in PKC

Intuition: Only Alice with her secret key can sign any message. Everyone can verify if a message is signed by Alice, using her public key. (You will sign Assignment 1 before submitting it!)
PKC: distribution of keys

Lets assume that everything works well with the keys!

- The public key is **easy** to compute from the secret key!
- The secret key is large enough and **hard** to find from the public key!

**Problem:** How does Alice know which is Bob’s public key?

**Option 1:**
Bob could give his public key to Alice in person.
**Problem:** Quite cumbersome method!

**Option 2:**
Use a secure channel to transfer the key.

**Problem:** How to establish a secure channel?

**Option 3:**
Get someone to check and make sure that a specific key is Bob’s ⇒ **Certification!**
Option 1: Certification Authorities (CA)

A certification authority issues certificates that certify that for example Alice indeed has a public key $PK$.  

**Problem:** A single certification authority can be a single point of failure if it is compromised!
PKI: Public Key Infrastructure

**Option 1: Certification Authorities (CA)**

**Option 2: Distributed Certification Authority (CA)**

Instead of one, multiple certification authorities exist. Each CA may trust or not the others.
**PKI: Public Key Infrastructure**

**Option 1:** Certification Authorities (CA)

**Option 2:** Distributed Certification Authority (CA)

**Option 3:** Web of Trust (Certificate Chains)

Anyone can issue certificates to anyone else. The users decide how much they trust the certificates issued by other users (e.g., PGP that you use in Assignment 1).

I trust the King and everyone he certifies. So I trust Charlie.
How does a certificate look like?

Here is my certificate!

Does the signature verify? If yes I will use it.
Modular Arithmetic

Let $a$ be an integer and $N$ be a positive integer, then we define: $a \pmod{N}$ to be the \textbf{remainder} when $a$ is divided by $N$!

Example

$a = 11, \ N = 7$.

How much is $11 \pmod{7} \equiv 4$

How to compute the “$\pmod{N}$”?

1. Find the largest integer $q$ s.t. $a > qN$.
2. Compute $r = a - qN$
3. Then, $r \equiv a \pmod{N}$

Why does this work?

Because $qN$ is a multiple of $N \Rightarrow$ If we divide $qN$ with $N$ the remainder is 0. This means that we have:

$r \pmod{N} \equiv (a - qN) \pmod{N}$

$\equiv a \pmod{N} - qN \pmod{N}$

$\equiv a \pmod{N} - 0 \equiv a \pmod{N}$
Quiz question!

Go to: http://socrative.com/
and use the code below or scan the QR code.
Student Login
Classroom: CRYPTOCHALMERS

Modular arithmetics
Calculate the following modular arithmetics:
- $23 \pmod{5}$
- $-4 \pmod{11}$
- $5 + (3 \pmod{8})$
- $(2 \times 100) \pmod{99}$
- $122 \pmod{11}$
An Intuition to Trapdoors & One-way function

**Easy**

It is easy to multiply prime numbers.

**Hard**

*(Factoring problem)* Hard to take the product number and decompose it to prime numbers.

**Open Problem:** To provide a method that can be used to factor large numbers!

---

**Trapdoor function**

- A one-way function function that can be computed easily **on one way** but difficult to compute of the **other (inverse) way** e.g., factoring large numbers
- Very useful for building public key encryption schemes!
- **Example:**
  - **Easy** to compute the public key from the secret key.
  - **Hard** to compute the secret key from the public key!
Definitions of basic concepts: prime numbers & factoring

**Definition**

A positive integer $p \in \mathbb{Z}$ is called a **prime number** if it is divisible only by itself and 1.

**Definition**

For integers $m, n$, we say that $m$ divides $n$ if $n$ is a multiple of $m$, i.e., if there exists an integer $k$, such that $n = km$.

**Fundamental Theorem of Arithmetics**

Every positive integer $n > 1$ can be represented in exactly one way as a product of powers of primes:

$$n = p_1^{a_1} p_2^{a_2} \cdots p_k^{a_k} = \prod_{i=1}^{k} p_i^{a_i}$$

where $p_1 < p_2 < \ldots < p_k$
Textbook RSA encryption Scheme (Rivest-Shamir-Adleman)

Useful for the exercises/exams!

**Textbook RSA**

\[ \text{KeyGen}(\lambda) \rightarrow (pk, sk) \]

1. generate two distinct $\lambda$-bit primes $p$ and $q$, compute $N = pq$ and $\Phi(N)$.
2. choose an integer $e \overset{R}{\leftarrow} \mathbb{Z}_{\Phi(N)}$ such that $\text{GCD}(e, \Phi(N)) = 1$ and compute its (modular) inverse $d = e^{-1} \mod \Phi(N)$.
3. set: $pk = (N, e)$ and $sk = (N, d)$

\[ \text{Enc}(pk, m) \rightarrow c : \text{compute} \quad c = m^e \mod N \]

\[ \text{Dec}(sk, m) \rightarrow m : \text{compute} \quad m = c^d \mod N \]

$\Phi$ denotes Euler’s phi (or Totient) function (more info about this in lec06). For now remember $\Phi(N) = \Phi(pq) = (p - 1)(q - 1)$. 
Things to Remember

- Prove that ECB is not semantically secure.
- How do we define CPA (in secret key crypto) (description of the game).
- Prove CBC with a predictable IV is not secure against CPA.
- How do we define a Public Key Encryption scheme?
- How may we achieve key distribution for PKE?
References:

- “Cryptography and Network Security: Principles and practice” (Chapters 8.1, 9.1, 9.2 and 14.5)
- “Introduction to Modern Cryptography”, Lindell and Katz (Chapter 3.4.2, 10.4, 8.1.1, 8.1.2, 11.1, 11.2.0, 11.5.1, 12.1)

Thank you for your attention!