Module 5: Pseudo Random Permutations and Block Ciphers

Disclaimer: large parts from Mark Manulis and Dan Boneh

Dresden, WS 14/15
You know *CIA*, perfect secrecy and semantic security

You know different classes of cryptographic algorithms

You can explain (and show) CTO, KPA, IND-CPA and IND-CCA *adversary models*

You can prove that the OTP has perfect secrecy

You understand when PRGs are secure, and you can explain stream ciphers

You can explain how semantic security of stream ciphers is proven
Module Outline

Mini function theory refresher

(Trapdoor) One-way functions

Pseudo Random Functions

Pseudo Random Permutations

Building PRPs:
Confusion – Diffusion Paradigm / Subst-Perm Networks
Feistel Networks and DES / 3DES
AES

Making it work: Modes of operation
A little refresher on functions...

\[ f: X \rightarrow Y \]

\[ X = \{a,b,c\} \quad Y = \{1,2,3,4\} \]

\[ y = f(x) \]

\[ \text{Im}(f) = \{1,2,4\} \]
$X = \{1, 2, 3, \ldots, 10\}$  
$f(x) = x^2 \mod 11$

$f: X \rightarrow Y$

$Y = \{1, 3, 4, 5, 9\}$

$f$ is called "onto" (surjective): $Y = \text{Im}(f)$ or: $\forall y \in Y \; \exists x \in X: y = f(x)$

"one-to-one" (injective): $\forall x_1, x_2 \in X: f(x_1) = f(x_2) \Rightarrow x_1 = x_2$

bijection: $f(x)$ is 1–1 and $\text{Im}(f) = Y$

For bijection $f$ there is an inverse: $g = f^{-1}: g(y) = x \; (= f(g(x)) \;)$
Finding the inverse $f^{-1}$ is not always „easy“

**One way functions:**
A function $f: X \to Y$ is called a one-way-function, if $f(x)$ is „easy“ to compute for all $x \in X$, but for “essentially all” elements $y \in \text{Im}(f)$ it is computationally infeasible to find the preimage $x$.

**Trapdoor one-way functions:**
A trapdoor one-way function is a one-way function that, given some additional trapdoor information, is feasible to invert.
Permutations and Involutions:

A permutation $\pi$ is a bijective function from a domain to itself:

$$\pi: X \rightarrow X \quad \text{Im}(f) = X$$

A permutation $\pi$ with: $\pi = \pi^{-1}$ (or: $\pi(\pi(x)) = x$) is called an involution.

Pseudo Random Functions (PRF):

$$F: K \times X \rightarrow Y$$
on "domain" $X$ and "range" $K$, with "efficient" algorithm to evaluate $F(k,x)$

Pseudo Random Permutation (PRP):

Permutation $E: K \times X \rightarrow X$

has efficient deterministic algorithm to evaluate $E(k,x)$ \textit{and} efficient inversion algorithm $D = E^{-1}$
A PRF is secure, if it is indistinguishable from a random function:

Consider

\[ \text{Funs}[X,Y]: \text{the set of all functions from } X \text{ to } Y \]

\[ \text{PRF } F_k = \{ F(k, \cdot) \text{ s.t. } k \in K \} \subseteq \text{Funs}[X,Y] \]
A PRP is secure, if it is indistinguishable from a random permutation:

Consider

Perms[X]: the set of all one-to-one functions from X to X

PRP $E_k = \{E(k, \cdot) \text{ s.t. } k \in K \} \subseteq \text{Perms}[X]$
**Goal:**

Build a secure PRP for b-bit blocks

**Examples:**

3DES:  $n = 64, k = 168$

AES:  $n = 128, k = 128, 192, 256$
**Confusion – Diffusion Paradigm**

**Original idea:**
Construct a random-looking permutation \( F \) with large block size using random-looking permutations \( \{f_i\}_i \) with smaller block sizes

Shannon, 1949

Create product cipher with confusion step (hide relation between CT and \( k \)) and diffusion step (distribute redundancy of PT)

**Construction:**
Let's construct \( F_k : \{0,1\}^{128} \rightarrow \{0,1\}^{128} \):

Combine \( f_1,...,f_{16} \) random-looking permutations \( f_i : \{0,1\}^8 \rightarrow \{0,1\}^8 \), defined by random keys \( k_i \) derived from \( k \)

(confusion)

(repeat rounds)

(bitwise diffusion)
SPN implement the Confusion – Diffusion Paradigm:

- Round keys $k_i$ are derived from $k$, then usually $\oplus$-ed with intermediate round output
- round functions $f_i$ are fixed, invertible substitution boxes (S-Box)
Rounds and Round Keys: Key Expansion

Recall from stream ciphers:
Short key expanded to encrypt bitstream

**Idea:**
Perform several keyed permutations in rounds
Expand key to round keys as parameters for random permutations
**Goal:**
Create a PRP from arbitrary (non-invertible) functions

**Idea:**
\[ R_i = f_i(R_{i-1}) \oplus L_{i-1} \quad \text{L}_i = R_{i-1} \]
with round function \( f_i \) (possibly non-invertible), keyed with round key \( k_i \)

*Inverting is easy (basically identical, \( f_1 \) to \( f_d \) reversed):*
\[ R_{i-1} = L_i \]
\[ L_{i-1} = R_i \oplus f_i(L_i) \]

Luby-Rackoff ‘85: a 3 round Feistel-Network

\[ F: K^3 \times \{0,1\}^{2n} \rightarrow \{0,1\}^{2n} \], built using PRF, is a PRP
„Lucifer“ at DES challenge (16 rounds; b,k = 128 bit FN, IBM)
Standardized as DES after adaptation (b=64, k = 56,..., due to NSA)

---

**Diagram:**

- **Expansion via Replication:**
  - 32 bits → 48 bits
  - S-Boxes (substitution)
  - each $s_i: \{0,1\}^6 \rightarrow \{0,1\}^4$
  - Mixing round permutation (RP)
  - $16, 7, 20, 21, ...$

- **Initial bit Permutation:**
  - $R_{i-1}$ → E → $k_i$
  - $E$ → 48 bits → $\oplus$ → 48 bits
  - $48$ bits → 48 bits
  - 48 bits → 48 bits
  - 48 bits → 48 bits

- **Final bit Permutation:**
  - $R_i$ → $L_i$ → $f_i$ → $k_i$
  - $L_i$ → $R_i$
  - $L_{15}$ → $R_{15}$ → $f_{16}$ → $k_{16}$
  - $R_{15}$ → $L_{15}$
  - $FP(block)$

---

**Table:**

<table>
<thead>
<tr>
<th>$S_5$</th>
<th>Middle 4 bits of input</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0000 0001 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111</td>
</tr>
<tr>
<td>Outer bits</td>
<td>00 01 10 11 00 01 10 11 00 01 10 11 00 01</td>
</tr>
</tbody>
</table>

---
Exhaustive Search

Given a few input output pairs \( (m_i, c_i = E(k, m_i)) \) \( i=1,..,3 \), find key \( k \).

**DES challenge:**

\[
\begin{align*}
\text{msg} &= \text{"The unknown message is: XXXX ... "} \\
\text{CT} &= c_1 \quad c_2 \quad c_3 \quad c_4 \quad ...
\end{align*}
\]

DES broken by exhaustive search (DESCHALL) in 96 days in 1997

„The unknown message is: It's time to move to a longer key length.”

distributed.net: 39 days in 1998

„The secret message is: Many hands make light work.”

EFF „deep crack“ (250k$) breaks DES in 56h in 1998

„The secret message is: It's time for those 128-, 192-, and 256-bit keys.”

Combined search: 22h in 1999

„See you in Rome (second AES Conference, March 22-23, 1999)”
Goal:

Strengthen DES by increasing key length

Let $E : K \times M \rightarrow M$ be a block cipher (DES)

Define $3E : K^3 \times M \rightarrow M$ as

$$3E((k_1,k_2,k_3), m) = E(k_1, D(k_2, E(k_3, m)))$$

For 3DES: key-size = $3 \times 56 = 168$ bits. 3×slower than DES.

Why not $E(E(E(m)))$? ... What if: $k_1 = k_2 = k_3$?

Simple attack feasible in time $\approx 2^{118}$
Meet-in-the-middle attack (no double DES?)

Define \[2E\left((k_1, k_2), m\right) = E(k_1, E(k_2, m))\]

**Idea:** test if \(E(m) = D(c)\)

Step 1: build table of encryptions \(E(k, m)\)
Step 2: for all \(k \in \{0,1\}^{56}\) do:
- test if \(D(k, c)\) is in 2\(^{nd}\) column.

### Example

- \(k_0 = 00\ldots00\)
- \(k_1 = 00\ldots01\)
- \(k_2 = 00\ldots10\)
  - \(\vdots\)
- \(k_N = 11\ldots11\)

\(2^{56}\) entries
Complexity of Meet-in-the-Middle

Time = \(2^{56}\log(2^{56}) + 2^{56}\log(2^{56}) < 2^{63} \ll 2^{112}\), space \(\approx 2^{56}\)

Same attack on 3DES: Time = \(2^{118}\), space \(\approx 2^{56}\)
1. Side channel attacks:
   - Measure **time** to do enc/dec, measure **power** for enc/dec

2. Fault attacks:
   - Computing errors in the last round expose the secret key $k$

[Kocher, Jaffe, Jun, 1998]
Quantum Attacks on Block Ciphers

**Generic search problem:**
Let \( f: X \rightarrow \{0,1\} \) be a function.

Goal: find \( x \in X \) s.t. \( f(x) = 1 \).

**Classical computer:** best generic algorithm time = \( O(|X|) \)

**Quantum Algorithm (Grover):**
Given \( m, c = E(k, m) \) define
\[
    f(k) = \begin{cases} 
        1 & \text{if } E(k, m) = c \\
        0 & \text{otherwise}
    \end{cases}
\]

Quantum computer can find \( k \) in time \( O(|K|^{1/2}) \)

DES: time \( \approx 2^{28} \) (btw: **AES-128**: time \( \approx 2^{64} \))

**Quantum adversary:** 256-bits key ciphers (e.g. AES-256)