DD2448 Foundations of Cryptography
Lecture 3

Douglas Wikström
KTH Royal Institute of Technology
dog@kth.se

February 20, 2020

Last lecture: Double DES

We have seen that the key space of DES is too small. One way to increase it is to use DES twice, so called "double DES".

$$\mathrm{2DES}_{k_1,k_2}(x) = \mathrm{DES}_{k_2}(\mathrm{DES}_{k_1}(x))$$

Is this more secure than DES?

This question is valid for any cipher.

Meet-In-the-Middle Attack

- Get hold of a plaintext-ciphertext pair (m, c)
- Compute $X = \{x \mid k_1 \in \mathcal{K}_{\mathrm{DES}} \wedge x = \mathsf{E}_{k_1}(m)\}$.
- For $k_2 \in \mathcal{K}_{\mathrm{DES}}$ check if $\mathsf{E}_{k_2}^{-1}(c) = \mathsf{E}_{k_1}(m)$ for some k_1 using the table X. If so, then (k_1, k_2) is a good candidate.
- Repeat with (m', c'), starting from the set of candidate keys to identify correct key.

Triple DES

What about triple DES?

$$\mathrm{3DES}_{k_1,k_2,k_3}(x) = \mathrm{DES}_{k_3}(\mathrm{DES}_{k_2}(\mathrm{DES}_{k_1}(x)))$$

- Seemingly 112 bit "effective" key size.
- 3 times as slow as DES. DES is slow in software, and this is even worse. One of the motivations of AES.
- Triple DES is still considered to be secure.

Modes of Operation

Modes of Operation

- Electronic codebook mode (ECB mode).
- Cipher feedback mode (CFB mode).
- Cipher block chaining mode (CBC mode).
- Output feedback mode (OFB mode).
- Counter mode (CTR mode).

ECB Mode

Electronic codebook mode

Encrypt each block independently:

$$c_i = \mathsf{E}_k(m_i)$$

ECB Mode

Electronic codebook mode

Encrypt each block independently:

$$c_i = \mathsf{E}_k(m_i)$$

- Identical plaintext blocks give identical ciphertext blocks.

ECB Mode

Electronic codebook mode

Encrypt each block independently:

$$c_i = \mathsf{E}_k(m_i)$$

- Identical plaintext blocks give identical ciphertext blocks.
- How can we avoid this?

CFB Mode

Cipher feedback mode

xor plaintext block with previous ciphertext block **after** encryption:

$$
\begin{aligned}
c_0 &= \text{initialization vector} \\
c_i &= m_i \oplus \mathsf{E}_k(c_{i-1})
\end{aligned}
$$

CFB Mode

Cipher feedback mode

xor plaintext block with previous ciphertext block **after** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = m_i \oplus \mathsf{E}_k(c_{i-1})$$

- Sequential encryption and parallel decryption.

CFB Mode

Cipher feedback mode

xor plaintext block with previous ciphertext block **after** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = m_i \oplus \mathsf{E}_k(c_{i-1})$$

- Sequential encryption and parallel decryption.
- Self-synchronizing and unidirectional.

CFB Mode

Cipher feedback mode

xor plaintext block with previous ciphertext block **after** encryption:

$$\begin{aligned} c_0 &= \text{initialization vector} \\ c_i &= m_i \oplus \mathsf{E}_k(c_{i-1}) \end{aligned}$$

- Sequential encryption and parallel decryption.
- Self-synchronizing and unidirectional.
- How do we pick the initialization vector?

CBC Mode

Cipher block chaining mode

xor plaintext block with previous ciphertext block **before** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = \mathsf{E}_k\left(c_{i-1} \oplus m_i\right)$$

CBC Mode

Cipher block chaining mode

xor plaintext block with previous ciphertext block **before** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = \mathsf{E}_k\left(c_{i-1} \oplus m_i\right)$$

- Sequential encryption and parallel decryption

CBC Mode

Cipher block chaining mode

xor plaintext block with previous ciphertext block **before** encryption:

$$c_0 = \text{initialization vector}$$
$$c_i = \mathsf{E}_k(c_{i-1} \oplus m_i)$$

- Sequential encryption and parallel decryption
- Self-synchronizing.

OFB Mode

Output feedback mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$s_0 = \text{initialization vector}$$
$$s_i = \mathsf{E}_k(s_{i-1})$$
$$c_i = s_i \oplus m_i$$

OFB Mode

Output feedback mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$
\begin{aligned}
s_0 &= \text{initialization vector} \\
s_i &= \mathsf{E}_k(s_{i-1}) \\
c_i &= s_i \oplus m_i
\end{aligned}
$$

- Sequential.

OFB Mode

Output feedback mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$
\begin{aligned}
s_0 &= \text{initialization vector} \\
s_i &= \mathrm{E}_k(s_{i-1}) \\
c_i &= s_i \oplus m_i
\end{aligned}
$$

- Sequential.
- Synchronous.

OFB Mode

Output feedback mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$\begin{aligned} s_0 &= \text{initialization vector} \\ s_i &= \mathsf{E}_k(s_{i-1}) \\ c_i &= s_i \oplus m_i \end{aligned}$$

- Sequential.
- Synchronous.
- Allows batch processing.

OFB Mode

Output feedback mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$\begin{aligned} s_0 &= \text{initialization vector} \\ s_i &= \mathsf{E}_k(s_{i-1}) \\ c_i &= s_i \oplus m_i \end{aligned}$$

- Sequential.
- Synchronous.
- Allows batch processing.
- Malleable!

CTR Mode

Counter mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$
\begin{aligned}
s_0 &= \text{initialization vector} \\
s_i &= \mathsf{E}_k(s_0 \| i) \\
c_i &= s_i \oplus m_i
\end{aligned}
$$

CTR Mode

Counter mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$
\begin{aligned}
s_0 &= \text{initialization vector} \\
s_i &= \mathsf{E}_k(s_0 \| i) \\
c_i &= s_i \oplus m_i
\end{aligned}
$$

- Parallel.

CTR Mode

Counter mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$
\begin{aligned}
s_0 &= \text{initialization vector} \\
s_i &= \mathsf{E}_k(s_0 \| i) \\
c_i &= s_i \oplus m_i
\end{aligned}
$$

- Parallel.
- Synchronous.

CTR Mode

Counter mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$
\begin{aligned}
s_0 &= \text{initialization vector} \\
s_i &= \mathrm{E}_k(s_0 \| i) \\
c_i &= s_i \oplus m_i
\end{aligned}
$$

- Parallel.
- Synchronous.
- Allows batch processing.

CTR Mode

Counter mode

Generate stream, xor plaintexts with stream (emulate "one-time pad"):

$$
\begin{aligned}
s_0 &= \text{initialization vector} \\
s_i &= \mathsf{E}_k(s_0 \| i) \\
c_i &= s_i \oplus m_i
\end{aligned}
$$

- Parallel.
- Synchronous.
- Allows batch processing.
- Malleable!

Linear Cryptanalysis of the SPN

Basic Idea – Linearize

Find an expression of the following form with a high probability of occurrence.

$$P_{i_1} \oplus \cdots \oplus P_{i_p} \oplus C_{j_1} \oplus \cdots \oplus C_{j_c} = K_{\ell_1, s_1} \oplus \cdots \oplus K_{\ell_k, s_k}$$

Each random plaintext/ciphertext pair gives an estimate of

$$K_{\ell_1, s_1} \oplus \cdots \oplus K_{\ell_k, s_k}$$

Collect many pairs and make a better estimate based on the majority vote.

How do we come up with the desired expression?

How do we compute the required number of samples?

Bias

Definition. The bias $\epsilon(X)$ of a binary random variable X is defined by

$$\epsilon(X) = \Pr\left[X = 0\right] - \frac{1}{2} \ .$$

Bias

Definition. The bias $\epsilon(X)$ of a binary random variable X is defined by

$$\epsilon(X) = \Pr[X = 0] - \frac{1}{2} \ .$$

$\approx 1/\epsilon^2(X)$ samples are required to estimate X (Matsui)

Linear Approximation of S-Box (1/3)

Let X and Y be the input and output of an S-box, i.e.

$$Y = S(X) \ .$$

We consider the bias of linear combinations of the form

$$a \cdot X \oplus b \cdot Y = \left(\bigoplus_i a_i X_i \right) \oplus \left(\bigoplus_i b_i Y_i \right) \ .$$

Linear Approximation of S-Box (1/3)

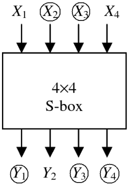
Let X and Y be the input and output of an S-box, i.e.

$$Y = S(X) \ .$$

We consider the bias of linear combinations of the form

$$a \cdot X \oplus b \cdot Y = \left(\bigoplus_i a_i X_i \right) \oplus \left(\bigoplus_i b_i Y_i \right) \ .$$

Example: $X_2 \oplus X_3 = Y_1 \oplus Y_3 \oplus Y_4$
The expression holds in 12 out of the 16 cases. Hence, it has a bias of $(12-8)/16 = 4/16 = 1/4$.



Linear Approximation of S-Box (2/3)

- Let $N_L(a, b)$ be the number of zero-outcomes of $a \cdot X \oplus b \cdot Y$.
- The bias is then

$$\epsilon(a \cdot X \oplus b \cdot Y) = \frac{N_L(a, b) - 8}{16} \ ,$$

since there are four bits in X, and Y is determined by X.

$$N_L(a, b) - 8$$

	Output Sum															
Input Sum	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	–2	–2	0	0	–2	+6	+2	+2	0	0	+2	+2	0	0
2	0	0	–2	–2	0	0	–2	–2	0	0	+2	+2	0	0	–6	+2
3	0	0	0	0	0	0	0	0	+2	–6	–2	–2	+2	+2	–2	–2
4	0	+2	0	–2	–2	–4	–2	0	0	–2	0	+2	+2	–4	+2	0
5	0	–2	–2	0	–2	0	+4	+2	–2	0	–4	+2	0	–2	–2	0
6	0	+2	–2	+4	+2	0	0	+2	0	–2	+2	+4	–2	0	0	–2
7	0	–2	0	+2	+2	–4	+2	0	–2	0	+2	0	+4	+2	0	+2
8	0	0	0	0	0	0	0	0	–2	+2	+2	–2	+2	–2	–2	–6
9	0	0	–2	–2	0	0	–2	–2	–4	0	–2	+2	0	+4	+2	–2
A	0	+4	–2	+2	–4	0	+2	–2	+2	+2	0	0	+2	+2	0	0
B	0	+4	0	–4	+4	0	+4	0	0	0	0	0	0	0	0	0
C	0	–2	+4	–2	–2	0	+2	0	+2	0	+2	+4	0	+2	0	–2
D	0	+2	+2	0	–2	+4	0	+2	–4	–2	+2	0	+2	0	0	+2
E	0	+2	+2	0	–2	–4	0	+2	–2	0	0	–2	–4	+2	–2	0
F	0	–2	–4	–2	–2	0	+2	0	0	–2	+4	–2	–2	0	+2	0

This gives linear approximation for one round.

How do we come up with linear approximation for more rounds?

Piling-Up Lemma

Lemma. Let $X_1, \ldots, X_t$ be independent binary random variables and let $\epsilon_i = \epsilon(X_i)$. Then

$$\epsilon\left(\bigoplus_i X_i\right) = 2^{t-1} \prod_i \epsilon_i \quad .$$

Proof. Case $t = 2$:

$$\begin{aligned}
\Pr[X_1 \oplus X_2 = 0] &= \Pr[(X_1 = 0 \wedge X_1 = 0) \vee (X_1 = 1 \wedge X_1 = 1)] \\
&= (\frac{1}{2} + \epsilon_1)(\frac{1}{2} + \epsilon_2) + (\frac{1}{2} - \epsilon_1)(\frac{1}{2} - \epsilon_2) \\
&= \frac{1}{2} + 2\epsilon_1\epsilon_2 \quad .
\end{aligned}$$

By induction $\Pr[X_1 \oplus \cdots \oplus X_t = 0] = \frac{1}{2} + 2^{t-1} \prod_i \epsilon_i$

Linear Trail

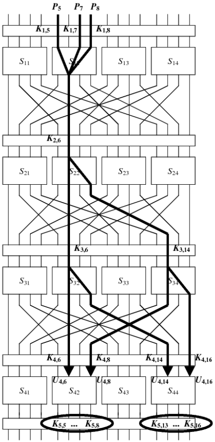
Four linear approximations with $|\epsilon_i| = 1/4$

$$\begin{aligned} S_{12}: &\quad X_1 \oplus X_3 \oplus X_4 = Y_2 \\ S_{22}: &\quad X_2 = Y_2 \oplus Y_4 \\ S_{32}: &\quad X_2 = Y_2 \oplus Y_4 \\ S_{34}: &\quad X_2 = Y_2 \oplus Y_4 \end{aligned}$$

Combine them to get:

$$U_{4,6} \oplus U_{4,8} \oplus U_{4,14} \oplus U_{4,16} \oplus P_5 \oplus P_7 \oplus P_8 = \bigoplus K_{i,j}$$

with bias $|\epsilon| = 2^{4-1}(\frac{1}{4})^4 = 2^{-5}$



Attack Idea

- Our expression (with bias 2^{-5}) links plaintext bits to input bits to the 4th round
- Partially undo the last round by guessing the last key. Only 2 S-Boxes are involved, i.e., $2^8 = 256$ guesses
- For a correct guess, the equation holds with bias 2^{-5}. For a wrong guess, it holds with bias zero[1].

[1] Why is this a harmless little lie for didactic reasons?

Attack Idea

- Our expression (with bias 2^{-5}) links plaintext bits to input bits to the 4th round
- Partially undo the last round by guessing the last key. Only 2 S-Boxes are involved, i.e., $2^8 = 256$ guesses
- For a correct guess, the equation holds with bias 2^{-5}. For a wrong guess, it holds with bias zero[1].

Required pairs $2^{10} \approx 1000$
Attack complexity $2^{18} \ll 2^{32}$ operations

[1] Why is this a harmless little lie for didactic reasons?

Linear Cryptanalysis Summary

1. Find linear approximation of S-Boxes.
2. Compute bias of each approximation.
3. Find linear trails.
4. Compute bias of linear trails.
5. Compute data and time complexity.
6. Estimate key bits from many plaintext-ciphertexts pairs.

Linear cryptanalysis is a **known plaintext attack**.