

Lecture 2. Block ciphers

Passive adversaries, small messages and small shared secret

Bruno Grenet



<https://membres-ljk.imag.fr/Bruno.Grenet/IntroCrypto.html>

Introduction to cryptology
Université Grenoble Alpes – IM²AG
M1 INFO, MOSIG & AM

Block ciphers: what do we want to achieve?

Goal: Symmetric Encryption

- ▶ Encryption: from a message and a key $\rightarrow$ ciphertexts
- ▶ Decryption: from a ciphertext and the key $\rightarrow$ message
- ▶ Security: a ciphertext alone should not give much information

non-determinism

Objects

- ▶ Message: any string $\in \{0, 1\}^*$.
- ▶ Ciphertext: string $\in \{0, 1\}^*$, not much larger than the message
- ▶ Key: string $\in \{0, 1\}^*$ not too large, not too small

efficiency

Block cipher

- ▶ Message / ciphertext: fixed-length
- ▶ One-to-one *correspondence* for each key $\rightarrow$ deterministic!
(bijective)

block size

Block ciphers are (mainly) a tool to build higher-level schemes

Contents

1. Definitions and security

2. Construction of block ciphers

3. Another (generic) attack: Meet in the middle

Block cipher: definition

Definition

A **block cipher** is a mapping $E : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}$ such that for all $k \in \mathcal{K}$, $E(k, \cdot)$ is one-to-one, with

- ▶ $\mathcal{K} = \{0, 1\}^\kappa$: the *key space*
- ▶ $\mathcal{M} = \{0, 1\}^n$: the *message space*

$$\kappa \in \{\cancel{64}, \cancel{80}, \cancel{96}, \cancel{112}, 128, 192, 256\}$$
$$n \in \{64, 128, 256\}$$

- ▶ A block cipher is a family of permutations, indexed by the keys
- ▶ Examples:
 - ▶ One-time pad: $\text{OTP}(k, m) = k \oplus m$
 - ▶ Identity: $\text{ID}(k, m) = m$

Notation

- ▶ We write interchangeably $E_k(m)$ or $E(k, m)$
- ▶ For a fixed k , we write E_k or $E(k, \cdot) : \mathcal{M} \rightarrow \mathcal{M}$

What are *good* block ciphers?

Efficiency

- ▶ Fast: e.g. *few cycles per byte* on modern CPUs
- ▶ Compact: small code / small circuit size
- ▶ Easy to implement → avoid side-channel attacks, etc.
- ▶ ...

Security

- ▶ Given $c = E(k, m)$, *hard* to find m without knowing k (1)
- ▶ Given m , *hard* to compute c without knowing k (2)
- ▶ Given *oracle* access to $E(k, \cdot)$, *hard* to find k (3)
- ▶ Given *oracle* access to $E^\pm(k, \cdot)$, *hard* to find k (4)

$E^\pm$: both E and E^{-1}

Example: One-time pad

- (1) ✓
- (2) ✓
- (3) ✗
- (4) ✗

if you know m and c , you can compute $k = m \oplus c$

What are *good* block ciphers?

Efficiency

- ▶ Fast: e.g. *few cycles per byte* on modern CPUs
- ▶ Compact: small code / small circuit size
- ▶ Easy to implement → avoid side-channel attacks, etc.
- ▶ ...

Security

- ▶ Given $c = E(k, m)$, *hard* to find m without knowing k
- ▶ Given m , *hard* to compute c without knowing k
- ▶ Given *oracle access* to $E(k, \cdot)$, *hard* to find k
- ▶ Given *oracle access* to $E^\pm(k, \cdot)$, *hard* to find k

$E^\pm$: both E and E^{-1}

→ Not enough! Ex.: given E , define $E'(k, m_L \| m_R) = m_L \| E(k, m_R)$

Need a *more general security definition*, that encompasses all of the above (and other)

In an ideal world

Definition

- ▶ Let Perm_n the set of all the $(2^n)!$ permutations of $\mathcal{M} = \{0, 1\}^n$
 - ▶ $E : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}$ is an **ideal block cipher** if for all $k \in \mathcal{K}$, $E_k \leftarrow \text{Perm}_n$.
 - ▶ All keys provide perfectly random and independent permutations
 - ▶ Non-realistic world:
 - ▶ $(2^n)^{2^{n-1}} < (2^n)! < (2^n)^{2^n}$
 - ▶ Key size $\simeq \log(2^n!) \simeq n \cdot 2^n$ bits
- $n = 32 \Rightarrow 2^{37}$ -bit keys!

Why *ideal*?

Fix a key k and a subset $\mathcal{S} \subset \mathcal{M}$ of messages

Assume an adversary knows:

- ▶ $E(k', m)$ for all $k' \in \mathcal{K} \setminus k$ and $m \in \mathcal{M}$
- ▶ $E(k, m)$ for all $m \in \mathcal{M} \setminus \mathcal{S}$

Perfect secrecy: The adversary has no information about $E(k, m)$ for m in $\mathcal{S}$

(Strong) PRP security: informal presentation

Informally, a block cipher is secure if its behavior is *close enough* to the ideal world

PRP Game

- ▶ Challenger gives the Adversary access to an *oracle* $\mathcal{O}$
 - ▶ The adversary can *query* $\mathcal{O}(m)$ for any $m \in \mathcal{M}$
 - ▶ $\mathcal{O}$ is either a random permutation, or a block cipher E_k with $k \leftarrow \mathcal{K}$
- ▶ The adversary must distinguish between the two *worlds*
- ▶ Strong version: access to $\mathcal{O}^\pm$

Why does it encompass previous tentative definitions?

- ▶ If m can be found from $c = E(k, m)$ without k
 - ▶ Take any c and compute the corresponding m
 - ▶ Query the oracle on m and compare the result with c
- ▶ *Other definitions: exercise!*

(Strong) PRP game

PRP game for a block cipher E

Challenger chooses $b \in \{0, 1\}$ and defines an oracle $\mathcal{O}$:

- ▶ if $b = 0$: $\mathcal{O} \leftarrow \text{Perm}_n$
- ▶ if $b = 1$: $\mathcal{O} \leftarrow E_k$ where $k \leftarrow \mathcal{K}$

Adversary submits queries m_i and gets $c_i = \mathcal{O}(m_i)$

Adversary outputs a bit $\hat{b}$

Strong PRP game for E

Adversary also submits queries c_j and gets $m_j = \mathcal{O}^{-1}(c_j)$

Remark

- ▶ The adversary *knows* $E \rightarrow$ can compute $E(k', m)$, given k' and m

(Strong) PRP advantage

PRP advantage of $\mathcal{A}$

$$\text{Adv}_E^{\text{PRP}}(\mathcal{A}) = \left| \Pr[\mathcal{A}^{\mathcal{O}} \rightarrow 1 : \mathcal{O} = E_k, k \leftarrow \mathcal{K}] - \Pr[\mathcal{A}^{\mathcal{O}} \rightarrow 1 : \mathcal{O} \leftarrow \text{Perm}_n] \right|$$

- Advantage strongly related to the probability of success of $\mathcal{A}$

exercise

(Strong) PRP advantage

PRP advantage of $\mathcal{A}$

$$\text{Adv}_E^{\text{PRP}}(\mathcal{A}) = \left| \Pr[\mathcal{A}^{\mathcal{O}} \rightarrow 1 : \mathcal{O} = E_k, k \leftarrow \mathcal{K}] - \Pr[\mathcal{A}^{\mathcal{O}} \rightarrow 1 : \mathcal{O} \leftarrow \text{Perm}_n] \right|$$

- Advantage strongly related to the probability of success of $\mathcal{A}$ *exercise*

PRP advantage function of the block cipher E

$$\text{Adv}_E^{\text{PRP}}(q, t) = \max_{\mathcal{A}_{q,t}} \text{Adv}_E^{\text{PRP}}(\mathcal{A}_{q,t})$$

where $\mathcal{A}_{q,t}$ denotes an algorithm that runs in time $\leq t$ and makes $\leq q$ queries to $\mathcal{O}$

- The PRP advantage provides a *measure* on the quality of a PRP, hence a block cipher
- The PRP advantage does *not* define when it is *good*
- Strong PRP advantage: same with the *strong* PRP game queries to $\mathcal{O}^{\pm}$

The generic attack

Generic adversary $\mathcal{A}_{\text{GEN}}$:

Input: Oracle access to either $\mathcal{O} \leftarrow \text{Perm}_n$ or $\mathcal{O} = E_k$ with $k \leftarrow \mathcal{K}$

1. $m_1, \dots, m_q \leftarrow \mathcal{M}$
2. $k_1, \dots, k_{t/q} \leftarrow \mathcal{K}$
3. $C_i \leftarrow [E(k_i, m_1), \dots, E(k_i, m_q)]$ for $1 \leq i \leq t/q$
4. $C \leftarrow [\mathcal{O}(m_1), \dots, \mathcal{O}(m_q)]$
5. Return 1 if there exists i s.t. $C = C_i$, 0 otherwise

computations
oracle queries

Complexity analysis

- ▶ Number of queries: q
- ▶ Running time: $O(t)$ calls to $E(\cdot, \cdot)$

Probability analysis for the generic attack

Random permutation world

$$\Pr[\mathcal{A}_{\text{GEN}}^{\mathcal{O}} \rightarrow 1 : \mathcal{O} \leftarrow \text{Perm}_n] \stackrel{(\ominus)}{=} \Pr[\exists k_i, \forall m_j, \mathcal{O}(m_j) = E(k_i, m_j)] \stackrel{(\leq)}{\leq} t/q \cdot 2^{(n-2)q}$$

Proof.

$$\stackrel{(\ominus)}{=} \Pr[\mathcal{A}_{\text{GEN}}^{\mathcal{O}} \rightarrow 1] = \Pr[\exists i, C = c_i] = \Pr[\exists i \forall j, \mathcal{O}(m_j) = E(k_i, m_j)]$$

$\stackrel{(\leq)}{\leq}$ Fix the key k_i . Then $E(k_i, m_j)$ is fixed $\leadsto$ not a random variable

$$\text{For a fixed } c, \Pr[\mathcal{O}(m_j) = c] = 1/2^n$$

$$\text{For fixed } c_1, c_2 \Pr[\mathcal{O}(m_{j_1}) = c_1 \wedge \mathcal{O}(m_{j_2}) = c_2] = \frac{1}{2^n} \cdot \frac{1}{2^n - 1}$$

$$\hookrightarrow \Pr[\forall j, E(k_i, m_j) = \mathcal{O}(m_j)] = \frac{1}{2^n} \cdot \frac{1}{2^n - 1} \cdots \frac{1}{2^n - q + 1} = \frac{(2^n - q)!}{(2^n)!} \leq 1/2^{(n-2)q}$$

$$\Rightarrow \Pr[\exists i \forall j, E(k_i, m_j) = \mathcal{O}(m_j)] \leq t/q \cdot 1/2^{(n-2)q}$$

Probability analysis for the generic attack

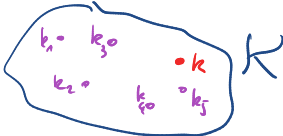
Random permutation world

$$\Pr[\mathcal{A}_{\text{GEN}}^{\mathcal{O}} \rightarrow 1 : \mathcal{O} \leftarrow \text{Perm}_n] = \Pr[\exists k_i, \forall m_j, \mathcal{O}(m_j) = E(k_i, m_j)] \leq t/q \cdot 2^{(n-2)q}$$

Block cipher world

$$\Pr[\mathcal{A}_{\text{GEN}}^{\mathcal{O}} \rightarrow 1 : \mathcal{O} = E_k, k \leftarrow \mathcal{K}] \geq \Pr[\exists k_i, k = k_i] = t/(q \cdot 2^K)$$

Proof.

$$\begin{aligned} \bullet \Pr[\mathcal{A}_{\text{GEN}}^{\mathcal{O}} \rightarrow 1] &= \Pr[\exists i, c_i = c] \geq \Pr[\exists k_i, k = k_i] \\ &\quad \uparrow \text{we ignore the cases where } c_i = c \text{ while } k_i \neq k \\ \bullet \Pr[\exists k_i, k = k_i] &= \frac{\#\{k_i\}}{\#\text{possible keys}} = \frac{t/q}{2^K} \end{aligned}$$


Probability analysis for the generic attack

Random permutation world

$$\Pr [\mathcal{A}_{\text{GEN}}^{\mathcal{O}} \rightarrow 1 : \mathcal{O} \leftarrow \text{Perm}_n] = \Pr [\exists k_i, \forall m_j, \mathcal{O}(m_j) = E(k_i, m_j)] \leq t/q \cdot 2^{(n-2)q}$$

Block cipher world

$$\Pr [\mathcal{A}_{\text{GEN}}^{\mathcal{O}} \rightarrow 1 : \mathcal{O} = E_k, k \leftarrow \mathcal{K}] \geq \Pr [\exists k_i, k = k_i] = t/q \cdot 2^{\kappa}$$

Conclusion

$$\text{Adv}_E^{\text{PRP}}(q, t) \geq \text{Adv}_E^{\text{PRP}}(\mathcal{A}_{\text{GEN}}) \geq \frac{t}{q \cdot 2^{\kappa}} - \frac{t}{q \cdot 2^{(n-2)q}} \simeq \frac{t}{q \cdot 2^{\kappa}}$$

So, what are *good* PRPs or block ciphers?

In this course, no **formal** definition of a good PRP

Informal (equivalent) definitions

- ▶ The advantage is the same as for an ideal block cipher
- ▶ The generic attack is almost the best possible
- ▶ $\text{Adv}_E^{\text{PRP}}(q, t) \simeq t/(q \cdot 2^\kappa)$

Remarks

- ▶ A good PRP is useless if κ is small *brute force attack*
 - ▶ $\kappa \simeq 40$ on a laptop, $\kappa \simeq 60$ on a CPU/GPU cluster, $\kappa \simeq 80$ on an ASIC cluster
- ▶ In *asymptotic security*, $\text{good} \simeq \text{Adv}_E^{\text{PRP}}(\text{poly}(n), \text{poly}(n)) \ll 1/\text{poly}(n)$

Some final remarks

Block cipher: $E : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}$ s.t. for all $k \in \mathcal{K}$, E_k is a permutation

- ▶ functional definition

what does it do?

Pseudo-random permutation: $\sigma : \mathcal{M} \rightarrow \mathcal{M}$ *indistinguishable* from a random permutation

- ▶ security definition

how does it behave?

Models of security to use a block cipher E in a more general construction

Random oracle model: Consider E as a *random permutation*

- ▶ Shows resistance against *generic* attacks
- ▶ Not sufficient!

(S)PRP model: Consider E as a *good* (S)PRP

- ▶ Stronger guarantee
- ▶ Still need to be careful

Contents

1. Definitions and security

2. Construction of block ciphers

3. Another (generic) attack: Meet in the middle

Generalities

How to build a block cipher?

- ▶ Several families of construction
 - ▶ Substitution-permutation network (SPN) *e.g. AES*
 - ▶ Feistel network *e.g. DES*
- ▶ (Non exhaustive) security goals: prevent the known attacks
 - ▶ Brute force
 - ▶ Linear cryptanalysis
 - ▶ Differential cryptanalysis

Some known block cipher(s families)

- ▶ Lucifer / DES:
 - ▶ 56-bit key; 64-bit block length
 - ▶ Variants (3-DES & DES-X) with larger key length
- ▶ Rijndael / AES
 - ▶ 128, 192 or 256-bit key; 128-bit block length
 - ▶ Current standard
- ▶ Others: Blowfish, Twofish, Camellia, TEA, ...

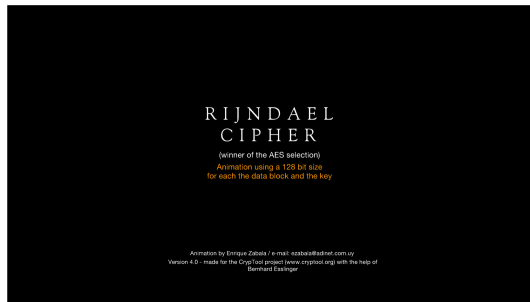
Data Encryption Standard

broken using brute force
quite slow

Advanced Encryption Standard

Example : AES

- ▶ NIST Competition (1997-2000)
- ▶ Winner: Rijndael, due to V. Rijmen & J. Daemen
- ▶ 128-bit block length; Key length 128, 192 or 256 (3 versions)
- ▶ Substitution-Permutation Network



Some algebraic considerations

Bit strings, bytes and finite field

- ▶ Input: 128-bit string $\rightarrow$ 16-byte string
- ▶ One byte $\simeq$ element of $\mathbb{F}_{2^8}$
- ▶ $\mathbb{F}_{2^8} \simeq \mathbb{F}_2[x] / \langle x^8 + x^4 + x^3 + x + 1 \rangle$

finite field with 2^8 elements
Degree-7 polynomials

SubBytes

- ▶ Inverse in $\mathbb{F}_{2^8}$ (with $0 \mapsto 0$)
- ▶ Composed with an invertible affine transformation

MixColumns

- ▶ Column $\rightarrow$ vector in $\mathbb{F}_{2^8}^4$
- ▶ Matrix multiplication by an *MDS circulant matrix*

coding theory

$\rightarrow$ Algebraic considerations to avoid known attacks

Contents

1. Definitions and security
2. Construction of block ciphers
3. Another (generic) attack: Meet in the middle

The context

Increase key length

Given a block cipher with (small) key length κ

e.g. DES

Build a block cipher with larger key length $\lambda = 2\kappa$ or 3κ , etc.

Rationale: a block cipher can be *very good* except its key length

The simple idea

► Double encryption: $EE_2(k_1 \| k_2, m) = E(k_2, E(k_1, m))$

► Triple encryption:

► $EEE_3(k_1 \| k_2 \| k_3, m) = E(k_3, E(k_2, E(k_1, m)))$

► $EEE_2(k_1 \| k_2, m) = E(k_1, E(k_2, E(k_1, m)))$

► $EDE_3(k_1 \| k_2 \| k_3, m) = E(k_3, E^{-1}(k_2, E(k_1, m)))$

► $EDE_2(k_1 \| k_2, m) = E(k_1, E^{-1}(k_2, E(k_1, m)))$

3-DES

Are these constructions safe?

► For instance, 3-DES *is* safe

► Exhaustive search: $O(2^{2\kappa})$ or $O(2^{3\kappa})$

Attack on double encryption

$$EE_2(k_1 \| k_2, m) = E(k_2, E(k_1, m)), \text{ with } k_1, k_2 \in \{0, 1\}^\kappa.$$

Meet-in-the-middle

Input: (m, c) where $c = EE_2(k_1^* \| k_2^*, m)$ for *unknown* k_1^*, k_2^*

Output: a (small) set of keys that contains $k_1^* \| k_2^*$

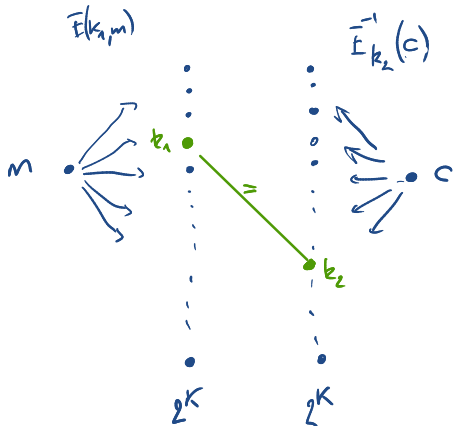
1. Compute each $y_{k_1} = E(k_1, m)$ for $k_1 \in \{0, 1\}^\kappa$
2. Compute each $z_{k_2} = E^{-1}(k_2, c)$ for $k_2 \in \{0, 1\}^\kappa$
3. For each *match* $y_{k_1} = z_{k_2}$, add $k_1 \| k_2$ to the set of keys
 - $EE_2(k_1 \| k_2, m) = E(k_2, E(k_1, m)) = E(k_2, y_{k_1}) = E(k_2, z_{k_2}) = c$

Analysis

- Time: twice $O(2^\kappa)$ calls to $E^\pm$ + the matches $\rightarrow$ roughly $O(2^\kappa)$
 - Space: two lists of 2^κ ciphertexts and keys $\rightarrow O((n + \kappa) \cdot 2^\kappa)$
- $\rightarrow$ Same time as brute force attack with key length 2^κ !

Attack on double encryption

$$EE_2(k_1 \| k_2, m) = E(k_2, E(k_1, m)) = c$$



$$\begin{aligned} E(k_2, E(k_1, m)) \\ &= E(k_2, E^{-1}(k_2, c)) \\ &= c \end{aligned}$$

$$\Rightarrow \text{I found } k_1 \| k_2 \text{ s.t.} \\ EE_2(k_1 \| k_2, m) = c$$

→ Same time as brute force attack with key length $2 \cdot$

Attack on double encryption

$$EE_2(k_1 \| k_2, m) = E(k_2, E(k_1, m)), \text{ with } k_1, k_2 \in \{0, 1\}^\kappa.$$

Meet-in-the-middle

Input: (m, c) where $c = EE_2(k_1^* \| k_2^*, m)$ for *unknown* k_1^*, k_2^*

Output: a (small) set of keys that contains $k_1^* \| k_2^*$

1. Compute each $y_{k_1} = E(k_1, m)$ for $k_1 \in \{0, 1\}^\kappa$
2. Compute each $z_{k_2} = E^{-1}(k_2, c)$ for $k_2 \in \{0, 1\}^\kappa$
3. For each *match* $y_{k_1} = z_{k_2}$, add $k_1 \| k_2$ to the set of keys
 - $EE_2(k_1 \| k_2, m) = E(k_2, E(k_1, m)) = E(k_2, y_{k_1}) = E(k_2, z_{k_2}) = c$

Analysis

- Time: twice $O(2^\kappa)$ calls to $E^\pm$ + the matches $\rightarrow$ roughly $O(2^\kappa)$
 - Space: two lists of 2^κ ciphertexts and keys $\rightarrow O((n + \kappa) \cdot 2^\kappa)$
- $\rightarrow$ Same time as brute force attack with key length 2^κ !

Conclusion

Definitions and security

Block cipher: $E : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}$ such that each $E(k, \cdot)$ is a permutation

Pseudo-random permutation: $\sigma : \mathcal{M} \rightarrow \mathcal{M}$ *indistinguishable* from a random permutation using the (S)PRP game and advantage

Ideal block cipher: each $E(k, \cdot)$ is a random permutation

In practice

- ▶ AES / Rijndael:
 - ▶ Most used block cipher nowadays, standardized by the NIST, replacement of DES
 - ▶ Block size $n = 128$ bits; Key size $\kappa = 128, 196$ or 256 bits
- ▶ Some other (less used) possibilities:
 - ▶ PRESENT: $n = 64, \kappa = 80$ or 128 *lightweight*
 - ▶ SHACAL-2: $n = 256, \kappa = 512$ *large parameters*
 - ▶ ...

Next lecture

- ▶ Symmetric encryption: from fixed-length to variable-length encryption