Crypto Engineering '23

✧

Symmetric encryption (1)

Pierre Karpman
pierre.karpman@univ-grenoble-alpes.fr
https://membres-ljk.imag.fr/Pierre.Karpman/tea.html

2023–09–26

Symmetric encryption: context

For now assume:

- A shared secret ("symmetric")
- Passive adversaries (wholly unrealistic??)
- Blackbox adversaries

$\rightsquigarrow$ (binary) *(Symmetric) encryption scheme*:

$$\mathsf{Enc} : \{0,1\}^{\kappa} \times \{0,1\}^{*} \to \{0,1\}^{*}$$

s.t. $\forall k \in \{0,1\}^{\kappa}$, $\mathsf{Enc}(k,\cdot)$ is invertible

N.B. Such schemes usually take additional parameters, hidden here

Perfect one-time one-bit encryption

Informal minimal security requirement: "Enc must be able to hide one bit, once"

Possible formalisation: require:

$$\mathsf{Enc}(\$, 0) \approx \mathsf{Enc}(\$, 1)$$

where $\mathsf{Enc}(\$, b)$ is the distribution of encryptions of b over uniform keys

But what if we:

- Only care about computationally-bounded adversaries?
- Want to encrypt more than one bit?

Computational indistinguishability: from **Adv**

Let $\mathfrak{D}_b = \mathsf{Enc}(\$, b)$, *computational indistinguishability* of $\mathfrak{D}_0$ and $\mathfrak{D}_1$ may be expressed from:

$$\mathbf{Adv}^{\mathfrak{D}_0,\mathfrak{D}_1}(1, t)$$

by requiring for instance that for "small" t, $\mathbf{Adv}^{\mathfrak{D}_0,\mathfrak{D}_1}(1, t)$ is "small" (cf. previous discussion on orders of magnitude)
$\rightsquigarrow$ it's all (somewhat) relative, no definitive meaning

More than once: let's play a game!

The idea:

- give knowledge of prior encryptions of 0's and 1's
- $\mathsf{Enc}(\$, 0)$ and $\mathsf{Enc}(\$, 1)$ must still be indist. conditioned on this knowledge
- (For instance, this completely fails if Enc with a fixed key is deterministic)

More generally:

- encrypt more than one bit
- let the adversary *choose* (adaptively) the messages encrypted before
- look at the advantage in function of #known encryption

$\rightsquigarrow$ IND(istinguishability)-C(hosen)P(laintext)A(ttack) security

IND-CPA game (for symmetric encryption)

IND-CPA security for Enc: try to distinguish $\mathsf{Enc}(k, m_0)$ from $\mathsf{Enc}(k, m_1)$ for chosen equal-length messages m_0, m_1 when given oracle access to an oracle for $\mathsf{Enc}(k, \cdot)$, with unknown $k \twoheadleftarrow \{0,1\}^\kappa$:

1. The "Challenger" chooses a key $k \twoheadleftarrow \{0,1\}^\kappa$
2. The Adversary may repeatedly submit queries x_i to the Challenger
3. The Challenger answers a query with $\mathsf{Enc}(k, x_i)$
4. The Adversary now submits m_0, m_1 of equal length
5. The Challenger draws $b \twoheadleftarrow \{0,1\}$, answers with $\mathsf{Enc}(k, m_b)$
6. The Adversary may again submit queries, and tries to guess b

$\rightsquigarrow$ $\mathbf{Adv}_{\mathsf{Enc}}^{\mathsf{IND\text{-}CPA}}(q, t)$: the advantage associated to the winning probability for adversaries running in time t, making q queries

Immediate implications from IND-CPA security

Exercise:
Let Enc be a deterministic encryption scheme. Give a very efficient attack against Enc w.r.t. IND-CPA security.

Engineering "good" IND-CPA schemes

- Very easy to build very inefficient "perfect" IND-CPA encryption from a uniform random source (cf. TD)
- Very easy to build very efficient ""anti-perfect"" IND-CPA encryption from nothing (cf. here)
- Not easy to build "efficient" "good" IND-CPA encryption

Possible ways to build efficient good Enc:

- From scratch
- From a smaller *primitive*, used appropriately ← the most common approach; let's have a closer look

Modular IND-CPA encryption: a roadmap

- The "I want something that works" part
  - Define a primitive that you know how to build (e.g. *block ciphers*)
  - Find ways to build encryption schemes from (**any black-box instance** of) this primitive
- The "I want some proofs" part
  - Find expressive security definitions syntactically compatible with this primitive (e.g. PRP, PRF security)
  - Prove appropriate security reductions ("good PRP-security of the block cipher $\Rightarrow$ good IND-CPA security of the derived encryption scheme")

Block ciphers: what

Block cipher

A block cipher is a mapping $\mathcal{E} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}'$ s.t. $\forall k \in \mathcal{K}$, $\mathcal{E}(k, \cdot)$ is invertible

In practice, most of the time:

- Keys $\mathcal{K} = \{0,1\}^\kappa$, with $\kappa \in \{\cancel{64}, \cancel{80}, \cancel{96}, \sout{112}, 128, 192, 256\}$
- Plaintexts/ciphertexts $\mathcal{M} = \mathcal{M}' = \{0,1\}^n$, with $n \in \{64, 128, 256\}$

$\Rightarrow$ BCs are *families of permutations* over binary domains

- Exception (non-binary): *Format Preserving Encryption* (FPE)

Block ciphers: why

Block ciphers are:

- “Natural”; “simple”
- “Easy” to design
- Expressive (can be used to build many things)
- The weight of history

(Nonetheless, alternatives exist)

Block cipher security

How to define "security" of a BC ? (Intuition: it should "hide stuff")

- *ideal* definition?
- search-based definition?
- decision-based definition?

Ideal block ciphers

Ideal block cipher

Let $\mathrm{Perm}(\mathcal{M})$ be the set of the $(\#\mathcal{M})!$ permutations of $\mathcal{M}$; an *ideal block cipher* $\mathcal{E} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ is s.t. $\mathcal{E}(\$, \cdot) \approx \mathfrak{P}$

$\mathfrak{P}$: shorthand for $\mathfrak{U}(\mathrm{Perm}(\mathcal{M}))$, itself the the uniform distribution over $\mathrm{Perm}(\mathcal{M})$

- "Maximally random"
- All keys yield truly random and independent permutations
- Quite costly to implement
  - Say $\mathcal{M} = \{0, 1\}^{32} \rightsquigarrow (2^{32})^{2^{31}} < 2^{32}! < (2^{32})^{2^{32}}$ permutations
  - So about $32 \times 2^{32} = 2^{37}$ *bits* to describe one ($\leftsquigarrow$ key size)

$\rightsquigarrow$ Not very practical

IBC (cont.)

Why is an ideal block cipher ideal?

- The idea: for all fixed k the full knowledge of $\mathcal{E}(\mathcal{K} \backslash k, \mathcal{M})$ and $\mathcal{E}(k, \mathcal{S})$ gives *no information* on $\mathcal{E}(k, \overline{\mathcal{S}})$ except that it is disjoint from $\overline{\mathcal{E}(k, \mathcal{S})}$ (as functionally required)
- ¿Being an ideal cipher is a postulate, not (really) something measurable (tho some things still are possible) $\rightsquigarrow$ can't *reduce* to it ‽
- ¿(Proofs in the *ideal cipher model* are a bit tricky to use (cf. the hash function lecture). Not in the *standard model*) ‽
- ¿(You can't readily instantiate an ideal cipher) ‽

From ideal defs. to standard model ones

Two approaches:

- "Search based": look at things hard to do for an IBC, ask the same, *in some context*
- "Decision based": measure how close you're from ideal, *in some context*

A search-based def.: unpredictability

To attack the *unpredictability* of a BC $\mathcal{E} : \{0,1\}^\kappa \to \mathcal{M}$, define:

Game Forge$^{\mathcal{E}}$

Give the adversary oracle access to $\mathbb{O} = \mathcal{E}(k, \cdot)$ for $k \twoheadleftarrow \{0,1\}^\kappa$
The adversary wins iff. it returns a couple (x, y) s.t.:

1. x was not queried to $\mathbb{O}$
2. $\mathcal{E}(k, x) = y$

$\rightsquigarrow$

InSec$^{\mathsf{UP}}$

$\mathbf{InSec}^{\mathsf{UP}}_{\mathcal{E}}(q, t) = \max_{A_{q,t}} \Pr[A^{\mathbb{O}}_{q,t}() \text{ wins Forge}^{\mathcal{E}}]$

Where $A_{q,t}$ run in time t and make q queries to its oracle

UP comments

- The *full* $\mathcal{E}(k, x)$ needs to be predicted; predicting all bits minus one is not enough
  - "good" UP doesn't guarantee unpredictability of individual bits
  - (Security notion appropriate for e.g. authentication, not so much for encryption)
- For an IBC, $\mathbf{InSec}^{\mathsf{UP}}(q, \infty) = 1/(\#\mathcal{M} - q)$

A decision-based def.: pseudorandomness

To attack the *pseudorandomness* of a BC $\mathcal{E} : \{0,1\}^\kappa \to \mathcal{M}$, define:

Game $\mathrm{PRP}^{\mathcal{E}}$

Pick the *real* or *ideal* world, w/ equal prob.
Give the adversary oracle access to $\mathbb{O}$ where:

- in the ideal world, $\mathbb{O} \twoheadleftarrow \mathsf{Perm}(\mathcal{M})$ (or $\mathbb{O} \sim \mathfrak{P}$))
- in the real world, $\mathbb{O} = \mathcal{E}(k, \cdot)$ for $k \twoheadleftarrow \{0,1\}^\kappa$ (or $\mathbb{O} \sim \mathcal{E}(\$, \cdot)$)

The adversary wins iff. it correctly decides which world it lives in

PRP (cont.)

$\rightsquigarrow$

$\mathbf{Adv}^{\mathsf{PRP}}$

$\mathbf{Adv}_{\mathcal{E}}^{\mathsf{PRP}}(q, t) =$

$$\max_{A_{q,t}} \left| \Pr[A_{q,t}^{\mathbb{O}}() = 1 : \mathbb{O} \sim \mathfrak{P}] - \Pr[A_{q,t}^{\mathbb{O}}() = 1 : \mathbb{O} \sim \mathcal{E}(\$, \cdot)] \right|$$

PRP comments

- It's fair to rely one only one bit to distinguish
  - "good" PRP guarantees indistinguishability of individual bits
  - (Security notion appropriate for e.g. encryption)
- PRP $\Rightarrow$ UP (cf. TD), but not the converse
- For an IBC, $\mathbf{Adv}^{\mathsf{PRP}}(\infty, \infty) = 0$ (given how we've defined IBCs; for some variant definitions, this isn't true any more)

Super variants; general comments

- Both UP and PRP admit *super* (or *strong*) variants where the adversary is also given oracle access to $\mathbb{O}^{-1}$
- Both UP and PRP (in the real world) pick a *uniform*, *secret* member of the family defined by $\mathcal{E}$ (i.e. sample $\mathcal{E}(\$, \cdot)$) $\rightsquigarrow$ definitions *not appropriate* for different contexts (e.g. block cipher-based hash function design)

Pseudorandomness for more than permutations

- Block ciphers are (families of) permutations $\rightsquigarrow$ natural to compare them to random permutations
- ... But not the only way; anything that's syntactically similar could make sense
- ... For instance random functions (not necessarily invertible)
  - sometimes the definition you actually want to use (even if you yourself happen to be invertible)

Pseudorandom functions

For $\mathcal{F} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}'$ a family of functions:

Adv$^{\mathsf{PRF}}$

$\mathbf{Adv}^{\mathsf{PRF}}_{\mathcal{F}}(q, t) =$

$$\max_{A_{q,t}} \left| \Pr[A^{\mathbb{O}}_{q,t}() = 1 : \mathbb{O} \sim \mathfrak{F}] - \Pr[A^{\mathbb{O}}_{q,t}() = 1 : \mathbb{O} \sim \mathcal{F}(\$, \cdot)] \right|$$

$\mathfrak{F}$: uniform distribution over all functions $\mathcal{M} \to \mathcal{M}'$

PRP/PRF switching

"Every good PRP is a good PRF" (over the same function space), up to the birthday bound

Let $\mathcal{E}$ be a BC over a domain of size N:

$$\mathbf{Adv}_{\mathcal{E}}^{\mathsf{PRF}}(q,t) \leqslant \mathbf{Adv}_{\mathcal{E}}^{\mathsf{PRP}}(q,t) + q(q-1)/2N$$

Proof: cf. "advanced crypto" course

Building encryption schemes from BCs

- A *mode of operation* transforms a block cipher into a *symmetric encryption scheme*
- $\approx \mathcal{E} \rightsquigarrow \mathsf{Enc} : \{0,1\}^\kappa \times \{0,1\}^r \times \{0,1\}^* \to \{0,1\}^*$
- For all $k \in \{0,1\}^\kappa$, $r \in \{0,1\}^r$, $\mathsf{Enc}(k, r, \cdot)$ is invertible
- ($\{0,1\}^r$, $r \geqslant 0$ is used to make encryption non-deterministic; made explicit here for emphasis)

First (non-) mode example: ECB

- ECB: just concatenate independent calls to $\mathcal{E}$

Electronic Code Book mode

$m_1||m_2||\ldots \mapsto \mathcal{E}(k, m_1)||\,\mathcal{E}(k, m_2)||\ldots$

- No IND-CPA security
- (Even worse than "just" being deterministic)
  - Exercise: give a simple attack on ECB for the IND-CPA security notion w/ $q = 0$ and advantage 1

Second (actual) mode example: CBC

- Cipher Block Chaining: Chain blocks together (duh)

Cipher Block Chaining mode

$r \times m_1||m_2||\ldots \mapsto$
$c_0 := r||c_1 := \mathcal{E}(k, m_1 \oplus c_0)||c2 := \mathcal{E}(k, m_2 \oplus c_1)||\ldots$

- Output block i (ciphertext) added (XORed) w/ input block $i+1$ (plaintext)
- For first (m_1) block: use random IV r
- Okay security in theory $\rightsquigarrow$ okay security in practice *if used properly*

CBC IVs

CBC has bad IND-CPA security if the IVs are not random

- Consider an IND-CPA adversary who asks an oracle query CBC-ENC(m), gets $r, c = \mathcal{E}(k, m \oplus r)$ (where $\mathcal{E}$ is the cipher used in CBC-ENC)
- Assume the adversary knows that for the next IV r', $\Pr[r' = x]$ is "large"
- Sends two challenges $m_0 = m \oplus r \oplus x$, $m_1 = m_0 \oplus 1$
- Gets $c_b = \text{CBC-ENC}(m_b)$, $b \twoheadleftarrow \{0, 1\}$
- If $c_b = c$, guess $b = 0$, else $b = 1$

Generic CBC collision attack

Even with random IVs, CBC's security degrades with # encryptions
An observation:

- For a fixed k, $\mathcal{E}(k, \cdot)$ is a permutation so $\mathcal{E}(k, x) = \mathcal{E}(k, y) \Leftrightarrow x = y$
- In CBC, inputs to $\mathcal{E}$ are of the form $x \oplus y$ where x is a message block and y an IV or a ciphertext block
- So $\mathcal{E}(k, x \oplus y) = \mathcal{E}(k, x' \oplus y') \Leftrightarrow x \oplus y = x' \oplus y'$

A consequence:

- If $c_i = \mathcal{E}(k, m_i \oplus c_{i-1}) = c'_j = \mathcal{E}(k, m'_j \oplus c'_{j-1})$, then $m_i \oplus c_{i-1} = m'_j \oplus c'_{j-1}$, and then $c_{i-1} \oplus c'_{j-1} = m_i \oplus m'_j$
- $\rightsquigarrow$ knowing identical ciphertext blocks reveals information about the message blocks
- $\Rightarrow$ breaks IND-CPA security
- Regardless of the security of $\mathcal{E}$ (i.e. even if it is ideal)!

CBC collisions: how likely?

How soon does a collision happen?

- Assumption: the distribution of the $(x \oplus y)$ is $\approx$ uniform
  - If y is an IV it has to be (close to) uniformly random, otherwise we have an attack (two slides ago)
  - If $y = \mathcal{E}(k, z)$ is a ciphertext block, ditto for y knowing z, otherwise we have an attack on $\mathcal{E}$
- $\Rightarrow$ A collision occurs w/ prob. $\approx q^2/2^n, q \leqslant 2^{n/2}$ (q: #blocks)

Some CBC recap

A decent mode, but

- Must use uniformly random IVs
- Must change key *much* before encrypting $2^{n/2}$ blocks when using an n-bit block cipher
- And this *regardless of the key size* κ
- Only birthday-bound security: this is a common restriction for modes of operation (cf. next slide)

Another classical mode: CTR

Counter mode

$m_1||m_2||\ldots \mapsto$
$c_0 := s||c_1 := \mathcal{E}(k, s) \oplus m_1||c_2 := \mathcal{E}(k, s+1) \oplus m_2||\ldots$

- The *counter* s may be (appropriately incremented and) kept from one message to another, or picked freshly (uniformly at random) every time (last option: not a significant security issue if $\mathcal{E}$ is a block cipher (why?))
- Encrypts a public counter $\rightsquigarrow$ pseudo-random keystream $\rightsquigarrow$ one-time-pad approximation (i.e. a *stream cipher*)
- Like CBC, must change key *much* before encrypting $2^{n/2}$ blocks when using an n-bit block cipher

PRP sec. of $\mathcal{E} \Rightarrow$ IND-CPA sec. of CTR[$\mathcal{E}$]

For $\mathcal{E}$ of domain $\mathcal{M}$ of size N:

$$\mathbf{Adv}^{\text{IND-CPA}}_{\text{CTR}[\mathcal{E}]}(q,t) \leqslant \mathbf{Adv}^{\text{PRP}}_{\mathcal{E}}(q',t) + q'(q'-1)/2N$$

where q' is the total number of queries to $\mathcal{E}$ implied by the q queries to CTR[$\mathcal{E}$]

Proof sketch:

1. For $\mathcal{F} \sim \mathfrak{F}$, $\mathbf{Adv}^{\text{IND-CPA}}_{\text{CTR}[\mathcal{F}]}(\leqslant N, \infty) = 0$ (for the stateful variant; cf. TD)
2. $\mathbf{Adv}^{\text{IND-CPA}}_{\text{CTR}[\mathcal{E}]}(q,t) \leqslant \mathbf{Adv}^{\text{PRF}}_{\mathcal{E}}(q',t)$ (any IND-CPA attack can be used as a PRF one)
3. Use PRP/PRF switching

Symmetric encryption: new context

Now assume:

- A shared secret (“symmetric”) (again)
- *Active* adversaries (much more realistic)
- Blackbox adversaries

Active adversaries $\approx$ may modify/inject messages over the channel

Q: Are active adversaries a threat for *confidentiality* (even if integrity is of no concern)?
A: Yes :(

Some real-life examples

Ciphertext-only decryption attacks!

- The *padding oracle* attacks on CBC (Vaudenay, 2002)
- $\rightsquigarrow$ (for instance) *Attacking the IPsec Standards in Encryption-only Configurations* (Degabriele & Paterson, 2007)

Typically, an active attack works when:

1. The adversary's actions have an impact on the future
2. The different future leaks information
3. The adversary can observe the difference

Toy active attack on raw CTR mode

- A target system sends control messages to a lo-power device with raw CTR mode
- Messages are all one-block 64-bit seven-letter ASCII-7 text, and use a byte-wise (modular) *sum complement* checksum for error detection
- If the checksum verification fails, the device sends a special "SENDAGN" code in clear

What could we do?? $\rightsquigarrow$ TD

Capturing confidentiality w/ active adv.: IND-CCA sec.

IND-CCA game:

- Same as the IND-CPA one, except that the adversary may now make oracle queries to $\mathrm{Dec}(k, \cdot)$
- But it looses if it queries $\mathrm{Dec}(k, \cdot)$ on answer to its challenge query

$\rightsquigarrow$ captures the ability of the adversary to modify & inject messages, and to "see what happens"

IND-CCA security is then defined from the IND-CCA advantage function

N.B. Here we have defined what is sometimes called IND-CCA*2* security, where the second 'A' emphasises the *adaptive* nature of the attacks

CTR[$\mathcal{E}$] is not IND-CCA

Exercise: show that $\mathbf{Adv}^{IND-CCA}_{CTR[\mathcal{E}]}(1,1) = 1$
(¿Like previous examples, this attacks the *mode*, regardless of how good $\mathcal{E}$ is‽)

CTR[$\mathcal{E}$] is not IND-CCA

Exercise: show that $\mathbf{Adv}^{IND-CCA}_{CTR[\mathcal{E}]}(1,1) = 1$
(¿Like previous examples, this attacks the *mode*, regardless of how good $\mathcal{E}$ is‽)

1. Make a challenge query (m_0, m_1), get c_b
2. Make a decryption query $c_b \oplus 1$, get m'_b
3. Return $[m'_b \oplus 1 = m_1]$

How to get IND-CCA security?

The idea:

- If IND-CPA $\not\Rightarrow$ IND-CCA because of active attacks, simply make those inoperative?
- ... by adding some detection mechanism?

$\rightsquigarrow$

$$\text{IND-CPA} + \text{INT-CTXT} \Rightarrow \text{IND-CCA}$$

$\rightsquigarrow$ "Modern" view: what you want isn't an encryption scheme, but an *Authenticated* Encryption scheme (with Additional Data) (cf. a next lecture)

Definition for passive confidentiality

Block ciphers

Modes of operation for block ciphers

Definition for active confidentiality

Appendix: BC evolution

Block cipher evolution's

Block ciphers are very versatile, $\rightsquigarrow$

- Symmetric encryption
- Authentication
- Hashing
- (More exotic constructions)

But not the only candidate primitives for the above

Two possible variations:

- Add one parameter (*tweakable* block ciphers)
- Remove one parameter (*permutations*)

Tweakable block ciphers

Tweakable block cipher

A tweakable block cipher is a mapping $\widetilde{\mathcal{E}} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}'$ s.t. $\forall k \in \mathcal{K},\ t \in \mathcal{T},\ \widetilde{\mathcal{E}}(k, t, \cdot)$ is invertible

The *tweak* t:

- Acts like a key in how it parameterizes a permutation
- Is *public* (known to any adversary)
- Could even be chosen by anyone (in the stronger security models)

Why TBCs?

Tweakable block ciphers are nice:

- Simplify the design/proofs of higher-level constructions: they're an expressive abstraction for when we add some non-determinism "close" to the BC
- Typically useful in authenticated-encryption modes (e.g. ΘCB)
- Help a lot in getting beyond-birthday-bound (BBB) security

An intuition of usefulness:

- Never reuse a tweak $\Rightarrow$ always use independent permutations
- Becomes quite harder to attack/distinguish

TBC constructions

Tweakable block ciphers may be built either:

- As high-level constructions, typically from a regular BC
  - Example: $\widetilde{\mathcal{E}}(k, t, \cdot) = \mathcal{E}(k \oplus t, \cdot)$ (adequate if $\mathcal{E}$ is secure against XOR related-key attacks)
- As dedicated designs (like a regular BC)
  - Example: KIASU-BC

Permutations

Permutation

A permutation is an invertible mapping $\mathcal{P} : \mathcal{M} \to \mathcal{M}$

- No key anymore!
  - One consequence: no notion similar to PRP to formalize sec.
- Easy to build as $\mathcal{E}(0, \cdot)$

Rationale:

- In BCs, it may be wasteful to process the key and plaintext separately
- Inverting a permutation is often not necessary in constructions; usages like $\mathcal{P}(k||m)$ are okay

Permutation uses

Hash functions:

- SHA-3 (Keccak)
- JH
- Grøstl
- Etc.

Authenticated encryption:

- River/Lake/Sea/Ocean/Lunar Keyak
- Ascon
- Etc.