Introduction to cryptology (GBIN8U16)

✧

Finite fields, block ciphers

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

2018–01–31

Bits as field elements

- Digital processing of information $\rightsquigarrow$ dealing with bits
- Error-correcting codes, crypto $\rightsquigarrow$ need analysis $\rightsquigarrow$ maths
- $\Rightarrow$ bits (no structure) $\mapsto$ field elements (math object)

- "Natural" match: $\{0, 1\} \cong \mathbb{F}_2 \equiv \mathbb{Z}/2\mathbb{Z} \equiv$ "(natural) integers modulo 2"
- $\mathbb{F}_2$: two elements (0, 1), two operations ($+$, $\times$)

What's $\mathbb{F}_2$ like?

- Addition $\equiv$ exclusive or (XOR ($\oplus$))
- Multiplication $\equiv$ logical and ($\wedge$)
- $\Rightarrow$ "Boolean" arithmetic

- Fact: any Boolean function $f : \{0, 1\}^n \to \{0, 1\}$ can be computed using only $\oplus$ and $\wedge$
- Fact 2: ditto, $g : \{0, 1\}^n \to \{0, 1\}^m$
- Fact 3: ditto, using NAND ($\neg \circ \wedge$)

One bit is nice, but...

- We rather need bit strings $\{0,1\}^n$ than single bits
- Now two "natural" matches:

- $\mathbb{F}_2^n$ (vectors over $\mathbb{F}_2$)
  - Can add two vectors
  - Cannot multiply "internally" (but there's a dot/scalar product)

- $\mathbb{Z}/2^n\mathbb{Z}$ (natural integers modulo 2^n)
  - Can add, multiply
  - Not all elements are invertible (e.g. 2) $\Rightarrow$ only a ring

A third way

- Also possible: $\mathbb{F}_{2^n}$: an *extension* field
  - Addition "like in $\mathbb{F}_2^n$"
  - Plus an internal multiplication
  - All elements (except zero) are invertible
- Not for today!

Why are these useful?

- Allows to perform operations on inputs
  - E.g. adding two messages together
- Vector spaces ⇒ linear algebra (matrices)
  - Powerful tools to solve "easy" problems
  - (Intuition: crypto shouldn't be linear)
- Fields ⇒ polynomials
  - With one or more variable
  - ⇒ Can compute differentials
- Can mix $\mathbb{F}_2^n$, $\mathbb{Z}/2^n\mathbb{Z}$ to make things "hard"
  - Popular "ARX" strategy in symmetric cryptography (FEAL/MD5/SHA-1/Chacha/Speck/...)

Block ciphers: "simple" binary mappings

Block ciphers

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 $\kappa \in \{64, 80, 96, 112, 128, 192, 256\}$ (but e.g. 64's too short)
- Plaintexts/ciphertexts $\mathcal{M} = \mathcal{M}' = \{0,1\}^n$, with $n \in \{64, 128, 256\}$

Note

Block cipher inputs are *bits*, not vectors; field, ring elements

Block ciphers: for what?

Ultimate goal: symmetric encryption

- plaintext + key $\mapsto$ ciphertext
- ciphertext + key $\mapsto$ plaintext
- ciphertext $\mapsto$???

With *arbitrary* plaintexts $\in \{0,1\}^*$

Block ciphers: do that for plaintexts $\in \{0,1\}^n$

- (Very) small example: 32 randomly shuffled cards = 5-bit block cipher
- Typical block sizes n = “what’s easy to implement”

Block ciphers: only a building block

A "vanilla" block cipher is useless

- Only works on fixed-size inputs
- Is not randomized (remember?)
  - Fix k, $x \Rightarrow \mathcal{E}(k, x)$ always the same
  - $\Rightarrow$ leaks information about repeated messages
- (Does not authenticate coms)

$\Rightarrow$ Use block ciphers with a *mode of operation*

Encryption modes

Randomized encryption scheme

An encryption scheme is a mapping $\mathfrak{E} : \mathcal{K} \times \mathcal{R} \times \mathcal{M} \to \mathcal{M}'$ s.t. $\forall k \in \mathcal{K}$, $\forall r \in \mathcal{R}$, $\mathfrak{E}(k, r, \cdot)$ is invertible

With e.g.

- plaintexts/ciphertexts $\mathcal{M} = \mathcal{M}' = \bigcup_{\ell \leq 2^{40}} \{0, 1\}^{128\ell}$
- keys $\mathcal{K}$
- public randomness $\mathcal{R}$

- Encryption scheme $\approx$ block cipher + mode of operation

Criteria for a mode

Not all modes are equivalent

- How much can you encrypt? (In function of $\{0,1\}^n$)
- With what security?
- With what performance?
- (Do you get auth?)

Classical examples: ECB (not a mode), CBC, CTR

ECB (not a mode, for reference)

Electronic Code Book mode

$m_0 || m_1 || \ldots \mapsto \mathcal{E}(k, m_0) || \mathcal{E}(k, m_1) || \ldots$

- Vanilla use of the block cipher
- Efficient
- No security

CBC (classical, not so great)

Cipher Block Chaining mode

$r \times m_0||m_1||\ldots \mapsto c_0 := \mathcal{E}(k, m_0 \oplus r)||c1 := \mathcal{E}(k, m_1 \oplus c_0)||\ldots$

- Chain blocks together (duh)
  - Output block i (ciphtertext) added (XORed) w/ input block $i+1$ (plaintext)
  - For first (m_0) block: use random IV r
- Sequential $\rightsquigarrow$ not so efficient
- Need $\mathcal{E}^{-1}$ to decrypt
- Security in the square root of the block size = "birthday bound" (no details for now)
  - E.g., 128-bit blocks $\Rightarrow$ change key before encrypting $\ll 2^{64}$ blocks

CBC: need random IVs

CBC is not IND-CPA if the IVs are not random

- Attacker asks $\mathcal{E}$ –CBC(m), gets $r, c = \mathcal{E}(k, m \oplus r)$
- Knows that next IV $= x$
- Sends two challenges $m_0 = m \oplus r \oplus x$, $m_1 \xleftarrow{\$} \mathcal{M}$
- Gets $c_b = \mathcal{E}$ –CBC(m_b), $b \xleftarrow{\$} \{0, 1\}$
- If $c_b = c$, guess $b = 0$, else $b = 1$

CTR mode (classical, better)

Counter mode

$r \times m_0 || m_1 || \ldots \mapsto \mathcal{E}(k, r) \oplus m_0 || \mathcal{E}(k, r+1) \oplus m_1 || \ldots$

- Like a stream cipher
  - Encrypt a public counter $\Rightarrow$ pseudo-random keystream
  - Add (XOR) the keystream and the message
- Parallel $\rightsquigarrow$ efficient (multi-core, pipelining & all)
- "Inverse-free": don't need $\mathcal{E}^{-1}$ to decrypt
- Security up to the birthday bound (like CBC)
- This time, r can be known in advance (but cannot repeat!)

Other nice modes

- CENC (CTR-like, "beyond birthday")
- OCB (Authenticated-Encryption (AE) mode)
- GCM (ditto)
- TAE (OCB-like w/ *tweakable* block ciphers)
- OTR (OCB-like, inverse-free)

Maybe for another day...

Back to BCs: how do you build one?

- Many design strategies
- Different choices possible at
  - high level (main structure)
  - low level (tiny building blocks)
- Two brief examples today: Feistel, SPN
  - In both cases: define a *round function*, iterate it many times

Feistel ladder/networks

- A framework to extend the domain of a function (not necessarily invertible)
- Very versatile, can be used to build
  - Block ciphers (obvs.) / Hash functions
  - Modes of operation (e.g. OTR)
  - Padding schemes (e.g. OAEP)
  - S-boxes (part of block ciphers)
  - Etc.

Feistels: main idea

Basic equations (two-branch Feistel):

- $(L, R) \mapsto (L' = R, R' = L \oplus F(R))$ (forward)
- $(L', R') \mapsto (L = R' \oplus F(L'), R = L')$ (backward)
- $\Rightarrow$ Don't need F^{-1} to invert (does not need to be defined!)
- Can iterate to many rounds, with possibly different Fs

Then, can extend (in many ways) to more than two branches!

A Feistel, in picture

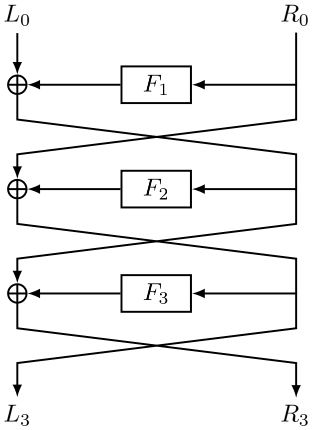

Figure: 3-Round Feistel (`https://www.iacr.org/authors/tikz/`)

We're not done, tho

- Q.1 How to build F?
- Q.2 How to add a key?

$\Rightarrow$ No single answer, but for instance

- A.1.1 Use random-looking small tables (S-boxes)
- A.1.2 Mix operations in $\mathbb{F}_2^n$, $\mathbb{Z}/2^n\mathbb{Z}$, Boolean functions (ARX)
- A.2.1 Add a key before/after F
- A.2.2 Use key-dependent F
- Etc.

The TWINE round function

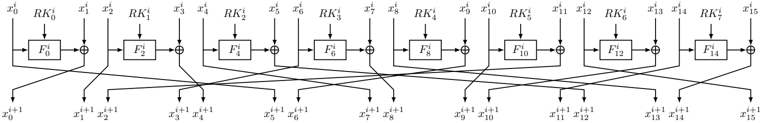

Figure: One round of TWINE
(https://www.iacr.org/authors/tikz/)

One SHA-1 step

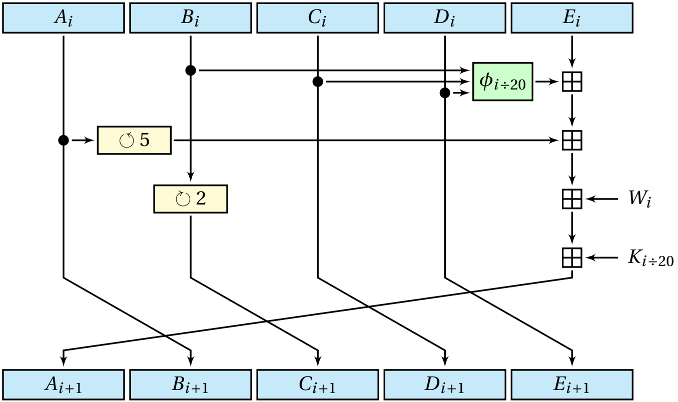

Figure: One SHA-1 step (compression function, ≈ block cipher)

Another way: Substitution Permutation Networks

One round: compose S and P where:

- P is an invertible matrix over $\mathbb{F}_2$ (i.e. $P \in \mathrm{GL}_n(\mathbb{F}_2)$)
- S is not $\mathbb{F}_2$-linear
- (Plus add a key at some point)

Often

- P is a permutation matrix
- Or a sparse matrix (e.g. composition of block diagonal and permutation)
- S is made of small invertible S-boxes

Small drawing: better than long description

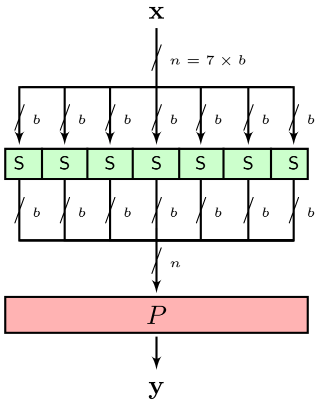

Figure: SPN, still quite abstract

Example: PRESENT

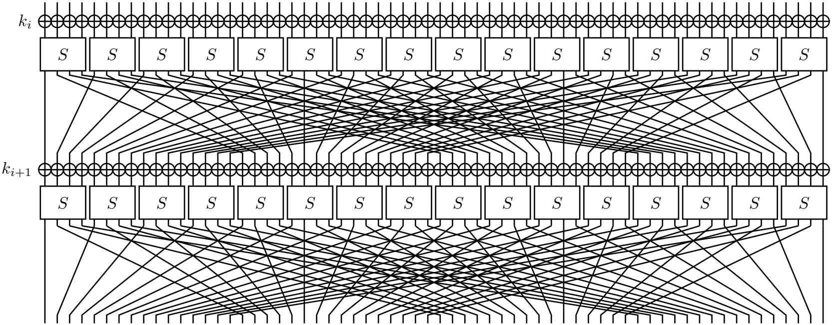

Figure: Two rounds of PRESENT
(`https://www.iacr.org/authors/tikz/`)

Example: AES

$\rightsquigarrow$ blackboard

Why not a single block cipher?

"It's all about context" ⇒ objectives?

- Fast?
- Small?
- Secure? (LOL)
- Versatile?
- Dedicated?
- Software/hardware?
- Etc.

We've barely scratched the surface

Anyways, what's a secure one?

- Let $\mathrm{Perm}(\mathcal{M})$ be the set of the $(\#\mathcal{M})!$ permutations of $\mathcal{M}$
- Ideally, $\forall k$, $\mathcal{E}(k,\cdot) \xleftarrow{\$} \mathrm{Perm}(\mathcal{M})$
- In practice, good enough if $\mathcal{E}$ is a "good" pseudo-random permutation (PRP):
  - An adversary has access to an oracle $\mathfrak{O}$
  - In one world, $\mathfrak{O} \xleftarrow{\$} \mathrm{Perm}(\mathcal{M})$
  - In another, $k \xleftarrow{\$} \mathcal{K}$, $\mathfrak{O} = \mathcal{E}(k,\cdot)$
  - The adversary cannot tell in which world he leaves
- Example: $\mathcal{E}$ cannot be $\mathbb{F}_2$-linear (or even "close to")

Next week

- Extensions of $\mathbb{F}_2$
- LFSRs
- MACs

References

- Knudsen & Robshaw, *The Block Cipher Companion*
- Daemen & Rijmen, *The Design of Rijndael*