# Introduction to cryptology (GBIN8U16)

✦

# Authentication & hashing

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

2024–02–21

# The goal: authentication

Current context:

- Two persons $\mathscr{A}$ & $\mathscr{B}$ wish to communicate (possibly *non-confidentially*) over a reliable channel
- Facing *active* adversaries

$\rightsquigarrow$ *one* option: using *message authentication codes* (MACs), to be evaluated w.r.t. UP or PRF

How to do it if $\mathscr{A}$ & $\mathscr{B}$ :

- Know a small *shared secret*?

Authentication; small shared secret

Cryptographic hash functions

Hash function applications

# Authentication: the idea

An active adversary over a channel may in all generality:

1. Block messages
2. Send messages
   - So modify messages

Defending with crypto:

1. Impossible?
2. *Detect* the messages coming from the adversary (and reject them)
   - The non-rejected messages (should) "emulate" a channel w/ an (only) passive adversary
   - $\rightsquigarrow$ Easy PoID (just say it)
   - $\rightsquigarrow$ Confidentiality against an active adversary (just use passive techno)

# Authentication with MACs

Using a (small) shared secret:

1. $\mathscr{A}$ and $\mathscr{B}$ *publicly* agree on a MAC $\mathsf{F} : \{0,1\}^\kappa \times \{0,1\}^* \to \{0,1\}^\tau$
2. $\mathscr{A}$ draws a uniform $K \in \{0,1\}^\kappa$ and shares it with $\mathscr{B}$
3. Every time $\mathscr{A}$ (resp. $\mathscr{B}$) wishes to send a message $m$, he sends $(m, t := \mathsf{F}(K, m))$ to $\mathscr{B}$ (resp. $\mathscr{A}$)
4. Upon reception of a message $(m, t)$, $\mathscr{A}$ (resp. $\mathscr{B}$) computes $t' := \mathsf{F}(K, m)$ and rejects the message if $t' \neq t$

Vocabulary: $t$ is a *tag*

Remarks:

- Doesn't need to be randomised (but mind the interactions w/ encryption, cf. TD)
- Possible parameters: $\kappa = \tau = 128$ (may vary)

# MACs: security

What security properties for F?

Informally, we want that:

- The adversary cannot send a message that passes verification
- Even after having seen many valid $(m, t)$ pairs

But:

- Not necessary for F to "hide" anything (if needed in the overall system, must be done before on $m$)
- Not needed to detect "replay" (if needed —)

# MACs: security (bis)

Typically, F is analysed w.r.t.:

- With oracle access to $\mathsf{F}(K \twoheadleftarrow \{0,1\}^\kappa, \cdot)$
  - *Universal unforgeability* (UUF): given an arbitrary challenge $m$, the adversary wins by returning a pair $(m, t)$ s.t. $t = \mathsf{F}(K, m)$
  - *Existential unforgeability* (EUF) (= UP): the adversary wins by returning an unqueried pair $(m, t)$ s.t. $t = \mathsf{F}(K, m)$
- PRF security

One may show (cf. TD) that a UUF attack $\Rightarrow$ EUF attack $\Rightarrow$ PRF attack

$\rightsquigarrow$ strongest *assumption* (most demanding): F is a PRF

# MACs: what constructions?

*Many* options:

- With a mode for block cipher (PRP), e.g. CBC-MAC (mind the details!)
- — (UP)
- With a statistical family of hash function + PRP/PRF
- With a mode for *cryptographic hash functions* $\leftarrow$ let's have a look

Authentication; small shared secret

Cryptographic hash functions

Hash function applications

# First definition

## Hash function

A hash function is a mapping $\mathcal{H} : \mathcal{M} \to \mathcal{D}$

So it really is just a function...

Usually:

- $\mathcal{M} = \bigcup_{\ell < N} \{0,1\}^\ell$, $\mathcal{D} = \{0,1\}^n$, $N \gg n$
- $N$ is typically $\geq 2^{64}$, $n \in \{\cancel{128}, \cancel{160}, \cancel{224}, 256, 384, 512\}$

Also popular now: extendable-output functions (XOFs): $\mathcal{D} = \bigcup_{\ell < N'} \{0,1\}^\ell$

N.B.: Hash functions are *keyless*

# Security of hash functions

*Ideal* (non-standard) model:

**Random oracle**

A function $\mathcal{H} : \mathcal{M} \to \mathcal{D}$ s.t. $\forall x \in \mathcal{M}$, $\mathcal{H}(x) \twoheadleftarrow \mathcal{D}$

- Difficult/impossible to get in real life
- But a useful concept; what is "difficult" to do with a random oracle should be difficult to do for a concrete hash function (not always the case!)
- Equivalent to an *Ideal block cipher* (Coron et al., 2008; + later patches)

# Intermission: Ideal block ciphers

**Ideal block cipher**

Let $\mathrm{Perms}(\mathcal{M})$ be the set of the $(\#\mathcal{M})!$ permutations of $\mathcal{M}$; an *ideal block cipher* $\mathsf{E} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ is s.t. $\forall k \in \mathcal{K}$, $\mathsf{E}(k, \cdot) \twoheadleftarrow \mathrm{Perms}(\mathcal{M})$

- All keys yield independent uniform permutations
- ($\infty$ PRP security, if computed over the sampling of the IBC)
- Difficult/impossible to get in real life

# Security of hash functions: classical definitions

One-way function:

- *First preimage*: given $t = \mathcal{H}(\$ \twoheadleftarrow \mathcal{M})$, find $m$ s.t. $\mathcal{H}(m) = t$
- *Second preimage*: given $m$, find $m' \neq m$ s.t. $\mathcal{H}(m) = \mathcal{H}(m')$
- *Collision*: given $\varnothing$, find $(m, m' \neq m)$ s.t. $\mathcal{H}(m) = \mathcal{H}(m')$

Success probability for a "generic" algorithm making $q$ queries to a random oracle over a (co-)domain of size $N$:
preimages: $\approx q/N$
collisions: $\approx q^2/N \leftarrow$ birthday paradox, again

# Classical definitions (bis)

Reductions between definitions:

- (resisting) 2PRE $\Rightarrow$ (resisting) PRE (Q: how to prove it?)
- (resisting) COL $\Rightarrow$ (resisting) (2)PRE (**WARNING:** exponential reduction!)
- Most (but not all!) applications of hash functions reduce to resistance w.r.t. one or several of those definitions

# Birthday paradox (again...)

Birthday paradox: collisions happen "quickly"
*One* way to phrase it:

**Expected number of collisions in a list of uniform & independent elements**

Let $L$ be a list of $q$ elements $x_i \twoheadleftarrow \mathcal{S}$, $\#\mathcal{S} =: N$,
$C := \#\{(i, j > i) : x_i = x_j\}$ the random variable that counts the number of collisions in $L$, then $\mathbb{E}[C] = q(q-1)/2N$

Proof: cf. TD

- Need $q \approx \sqrt{N}$ to get $\mathbb{E}[C] = 1$
- cf. TD for variants

# Hash function design

Bottom-up approach similar to encryption scheme design:

1. Design a “small” function (often a *compression functions*) of fixed-size input (“like” a block cipher)
2. Design a “mode of operation” to handle arbitrary-size inputs
3. Reduce the security of the thusly-obtained function to the one of the small function

# Compression functions

## Compression function

A compression function is a mapping
$f : \{0,1\}^n \times \{0,1\}^b \to \{0,1\}^n$

- A family of functions from $n$ to $n$ bits
- Not unlike a block cipher, only not (necessarily) invertible

Security defs. for compression functions:

- The same as for "full" hash functions, but with some additional freedom from the "index" parameter
- Keyed definitions (again), e.g. PRF, with either input treated as a key

# Compression function design

Can do it from scratch, or as a "mode" for block ciphers:

1. Take a block cipher, decide what goes where
2. Optionally add feedforward to prevent invertibility

Examples:
"Davies-Meyer": $f(h, m) = \mathrm{E}(m, h) + h$ (the "message" of $f$ becomes the "key" of E)
"BRSS/PGV–13": $f(h, m) = \mathrm{E}(m, h)$
"Matyas-Meyer-Oseas": $f(h, m) = \mathrm{E}(h, m) + m$

- Systematic analysis of simple BC-based constructions by Preneel, Govaerts and Vandewalle (1993). "PGV" constructions
- Then rigorous proofs **in the ideal cipher model** (Black et al., 2002), (Black et al., 2010)

# ICM PROOFS ARE NOT STANDARD

- Proofs in the ICM are **NOT REDUCTION PROOFS** $\leftarrow$ unlike e.g. reducing the IND-CPA security of CTR mode to the PRF security of the primitive
- Only rule-out "generic" attacks that don't exploit structural properties of the BC
- $\rightsquigarrow$ Don't give much guarantee about black-box instantiation

What does a security proof in the ICM say?

- Possibly a good basis for a construction
- But any instantiation needs a dedicated security analysis (e.g. through *cryptanalysis*) $\rightsquigarrow$ same as for a primitive

# Idealised models $\neq$ standard model

Microsoft needed a hash function for ROM integrity check of the XBOX

- Used TEA (Wheeler and Needham, 1994) in DM mode (Steil, 2005)
- Because of an earlier break of their RC4-CBC-MAC scheme (ibid.)
- Terrible idea, because of existence of equivalent keys for TEA (Kelsey et al., 1996)!
    - Keyspace is partitioned into (easy-to-define) classes of size 4
- For every $k$, it is easy to compute $\hat{k}$ s.t. $\mathrm{TEA}(k, m) = \mathrm{TEA}(\hat{k}, m) \Rightarrow \text{DM-TEA}(h, k) = \text{DM-TEA}(h, \hat{k}) \Rightarrow$ trivial collisions!

The XBOX got hacked...

And yet, TEA is a "good" PRP (as far as we know)!

# It doesn't *have* to be bad, tho

- AES several PGV construction so far unbroken (see e.g. Sasaki (2011))
  - But small parameters‽
- Ditto, SHA-256's compression function as a block cipher: "SHACAL-2" (Handschuh and Naccache, 2001)
  - Enormous keys, 512 bit!

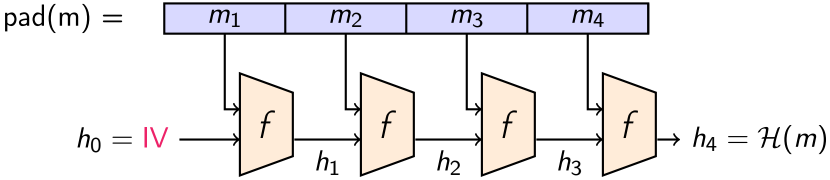
# *Domain extension* of compression functions

Assume a good $f$

- Main problem: fixed-size domain $\{0,1\}^n \times \{0,1\}^b$
- Objective: *domain extension* to $\bigcup_{\ell<N}\{0,1\}^\ell$

The classical answer: the Merkle-Damgård domain extender (1989)

# MD: with a picture




That is: $\mathcal{H}(m_1||m_2||m_3||\ldots) = f(\ldots f(f(f(\mathsf{IV}, m_1), m_2), m_3), \ldots)$

$\mathsf{pad}(m) \approx m||1000\ldots00\langle\text{length of } m\rangle \leftarrow$ *strengthening*

# MD security proof (sketch)

Method: simple contrapositive arguments

- Attack {PRE, COL} on $\mathcal{H} \Rightarrow$ attack {PRE, COL} on $f$

**First preimage case**

If $\mathcal{H}(m_1||m_2||\ldots||m_\ell) = t$, then $f(\mathcal{H}(m_1||m_2||\ldots||m_{\ell-1}), m_\ell) = t$

**Collision case (sketch)**

If $\mathcal{H}(m_1||m_2||\ldots||m_\ell) = \mathcal{H}(m'_1||m'_2||\ldots||m'_\ell)$, show that $\exists i$ s.t. $(h_i := \mathcal{H}(m_1||m_2||\ldots||m_{i-1}), m_i) \neq (h'_i := \mathcal{H}(m'_1||m'_2||\ldots||m'_{i-1}), m'_i)$ and $f(h_i, m_i) = f(h'_i, m'_i)$

- Proper message *padding* (such as (strengthening) necessary to make it work!

# What about 2PRE?

No proof (with optimal resistance), can't have one:

- Generic attack on messages of $2^k$ blocks for a cost $\approx k2^{n/2+1} + 2^{n-k+1}$ (Kelsey and Schneier, 2005)
- Idea: exploit internal collisions in the $h_i$s

This is not nice, but:

- Requires (very) long messages to gain something
- At least as expensive as collision search
  - Always going to be the case for a generic attack, since 2PRE attack $\Rightarrow$ COL attack (for which there *is* a reduction)
- If $n$ is chosen s.t. generic collisions are out of reach, we're somewhat fine

$\rightsquigarrow$ Didn't make people give up MD hash functions (MD5, SHA-1, SHA-2 family)

# MD variants w/ optimal 2PRE resistance

Simple MD variants: Chop-MD/Wide-pipe MD (Coron et al., 2005) and (Lucks, 2005)

- Build $\mathcal{H}$ from $f : \{0,1\}^{2n} \times \{0,1\}^{b} \to \{0,1\}^{2n}$, truncate output to $n$ bits (say)
- Collision in the output $\nRightarrow$ collision in the internal state
- Very strong provable guarantees (in an ideal model) (Coron et al.)
  - Secure domain extender for fixed-size RO (*ideal* compression function)
- Concrete instantiations: SHA-512/224, SHA-512/256 (2015)

# Careful with models (again)!

- Coron et al. prove very strong *indifferentiability* properties for Chop-MD w/ an ideal CF
- But this in fact doesn't guarantee things such as preservation of collision-resistance (Bellare & Ristenpart, 2006)!
  - One can do "stupid things" with a non-ideal compression function
  - $\rightsquigarrow$ Chop-MD with a (real) CR c.f. is not (necessarily) CR!
  - (In essence, one needs strengthening in the padding)

# Reduction proofs: what do they tell us? (MD case study)

- If one doesn't have "efficient" attacks for COL/PRE security of $f$ underlying $\mathcal{H}$, all is well
- Else, ...???
- $\rightsquigarrow$ Attacking the assumption (i.e. $f$) is a meaningful goal for cryptographers ($\rightsquigarrow$ *(semi-)freestart attacks*)
- Don't use an $\mathcal{H}$ with broken $f$
  - Same as not using CTR[E] w/ a broken E (w.r.t. PRP security)

# The MD5 failure

- MD5: designed by Rivest (1992)
- 1993: very efficient collision attack on the compression function (den Boer and Bosselaers); mean time of 4 minutes on a 33 MHz 80386
- MD5 still massively used...
- 2005: very efficient collision attack on the *hash function* (Wang and Yu)
- Still used...
- 2007: practically threatening collisions (Stevens et al.)
- Still used...
- 2009: even worse practical collision attacks (Stevens et al.)
- Hmm, maybe we should move on?

# Was this avoidable?

Yes!

- Early signs of weaknesses $\rightsquigarrow$ move to alternatives ASAP!
- What were they (among others)?
  - 1992: RIPEMD (RIPE); practically broken (collisions) 2005 (Wang et al.)
  - 1993: SHA-0 (NSA); broken (collisions) 1998 (Chabaud and Joux); practically broken 2005 (Biham et al.)
  - 1995: SHA-1 (NSA); broken (collisions) 2005 (Wang et al.); practically broken 2017 (Stevens et al. (and me!))
  - 1996: RIPEMD-128 (Dobbertin et al.); broken (collisions) 2013 (Landelle and Peyrin)
  - 1996: RIPEMD-160 (Dobbertin et al.); unbroken so far
  - 2001: SHA-2 (NSA); —
  - 2008: SHA-3 (Keccak team); —

# Some remarks

- CRHF are (were) hard to design!
- "Theoretical" attacks (that are too expensive to run) are still worrisome
- An attack that's "too expensive" may become practical in the future
- Don't use broken algorithms
  - Don't start using SHA-1 in 2005, like Git did...
- Much "easier" to be secure w.r.t. (2)PRE, but COL resistance usually needed
  - Don't use a hash function without understanding why!

Authentication; small shared secret

Cryptographic hash functions

Hash function applications

# Hash functions: application to authentication

Let $\mathcal{H} : \mathcal{M} \to \mathcal{D}$, one may define:

- $\texttt{PrefixMAC}_{\mathcal{H}} : \{0,1\}^{\kappa} \times \mathcal{M} \to \mathcal{D}$ as $\texttt{PrefixMAC}_{\mathcal{H}}(k, m) = \mathcal{H}(k||m)$
- $\texttt{SuffixMAC}_{\mathcal{H}} : \{0,1\}^{\kappa} \times \mathcal{M} \to \mathcal{D}$ as $\texttt{SuffixMAC}_{\mathcal{H}}(k, m) = \mathcal{H}(m||k)$

Remark : $\texttt{PrefixMAC}_{\mathcal{H}} \approx \texttt{SuffixMAC}_{\mathcal{H}^{\triangleleft}}$ with $\mathcal{H}^{\triangleleft}$ like $\mathcal{H}$ that reads its input "backwards"

- Constructions generically secure (for a random oracle) but **without reduction** to the traditional standard security properties COL/PRE
- $\rightsquigarrow$ subtle to instantiate
- (For instance, okay with the "wide-pipe" SHA-512/256, but not with the "narrow-pipe" SHA-512 or SHA-256!)

# Authentication (bis)

Several hash-based MACs reduce their security to standard properties of sub-components:

- $\texttt{SandwichMAC}_{\mathcal{H}} : \{0,1\}^{\kappa} \times \mathcal{M} \to \mathcal{D} \approx \texttt{SandwichMAC}_{\mathcal{H}}(k, m) = \mathcal{H}(k||p||m||p'||k)$ (for appropriate paddings $p$ et $p'$) reduces its PRF security to PRF security of compression functions *of the MD hash function* $\mathcal{H}$
  - Also probably reduces to COL resistance of $\mathcal{H}$ generically (but TBC)
- Other popular MACs with the same kind of reduction: NMAC, HMAC

# Other applications

Hash functions (-based constructions) are useful in many settings, e.g. for

- "Hash-and-sign" signatures (RSA signatures, (EC-)DSA...)
- Derandomization (e.g. FS transform)
- RSA *padding* (OAEP, PSS...)
- Hash-based signatures (rather expensive put PQ!)
- Key derivation
- Password hashing

**WARNING:** These require various security properties, sometimes never met by "classical" hash functions (e.g. for password-hashing)
$\rightsquigarrow$ sometimes hard to navigate