Cryptology complementary

✧

Message Authentication Codes, Authenticated Encryption

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

2019–04

Authentication (in crypto)

Crypto is not all about encrypting. One may also want to:

- Get access to a building/car/spaceship
- Electronically sign a contract/software/Git repository
- Detect tampering on a message
- Detect "identity theft"
- Etc.

$\Rightarrow$ domain of digital signatures and/or message authentication codes (MACs)

A major rule

In the case of a symmetric channel (e.g. on a network):

- It may be fine to only authenticate
- It is *never okay* to only encrypt

$\Rightarrow$ "Authenticated encryption" (This is hard to do properly.)

Today: MACs (symmetric authentication)

Message authentication code (MAC)

A MAC is a mapping $\mathcal{M} : \mathcal{K}(\times\mathcal{N}) \times \mathcal{X} \to \mathcal{T}$ that maps a key, message (and possibly a (random) nonce) to a *tag*.

- $\mathcal{K}$ is for instance $\{0,1\}^{128}$ (key space, secret)
- $\mathcal{N}$ is for instance $\{0,1\}^{64}$ ("nonce" space, public, either "random" or not)
- $\mathcal{X}$ is for instance $\bigcup_{\ell<2^{64}}\{0,1\}^{\ell}$ (message space)
- $\mathcal{T}$ is for instance $\{0,1\}^{256}$ ("tag" space)

$\Rightarrow$ The tag is a "link" between a message and a key

- Note: MACs are not the *only* way to provide authentication

MACs: what do we want?

Given a MAC $\mathcal{M}(k, \cdot)$ with an unknown key, it should be hard to:

- Given m, find t s.t. $\mathcal{M}(k, m) = t$ (*Universal forgery*)
- Find m, t s.t. $\mathcal{M}(k, m) = t$ (*Existential forgery*)
- (Of course, retrieving k leads to those)

UF: ability to forge a tag for **any** message

EF: ability to forge a tag for **some** messages

UF $\Rightarrow$ EF

MACs: really, what do we want?

More generally, we want $\mathcal{M}(k, \cdot)$ to be like a "variable input-length (pseudo-) random function"
$\rightsquigarrow$ (VIL-) PRF security:

- An adversary has access to an oracle $\mathbb{O}$
- In one world, $\mathbb{O} \xleftarrow{\$} \mathsf{Func}(\mathcal{X}, \mathcal{T})$
- In another, $k \xleftarrow{\$} \mathcal{K}$, $\mathbb{O} = \mathcal{M}(k, \cdot)$
- The adversary cannot tell in which world he lives

Where $\mathsf{Func}(\mathcal{X}, \mathcal{T})$ are the functions from the message to the tag space
$\rightsquigarrow$ Define $\mathbf{Adv}^{\mathsf{PRF}}$ in the same way as $\mathbf{Adv}^{\mathsf{PRP}}$

So, how to build a MAC?

- From scratch
- Using a block cipher in a “MAC mode”
- Ditto, with a hash function
- Using a “polynomial” hash function
- Etc.

MACs from block ciphers: CBC-MAC example

Observation:

- The last block of CBC-ENC(m) “strongly depends” on the entire message
- $\Rightarrow$ Take MAC(m) = LastBlockOf(CBC-ENC(m))
- Not quite secure as is, but overall a sound idea

Advantage:

- “Only” need a block cipher

Disadvantage:

- Not the fastest approach

$\Rightarrow$ Alternative: polynomial MACs

Polynomials

"Polynomials = vectors"

Let $m = \begin{pmatrix} m_1 & m_2 & \dots & m_n \end{pmatrix}$ be a vector of k^n, one can interpret it as $M = m_1 X + m_2 X^2 + \ldots + m_n X^n$, a degree-$n$ polynomial of $k[X]$

Polynomial evaluation

Let $M \in k[X]$ be a degree-n polynomial, the *evaluation* of M on an element of k is given by the map $\mathrm{eval}(M, \cdot) : x \mapsto m_1 x + \ldots + m_n x^n$

Polynomial hash functions

Polynomial hash function

Let $m \in k^n$ be a "message". The "hash" of $m \equiv M \in k[X]$ for the function $\mathcal{H}_x$ is given by $\mathrm{eval}(M, x)$.

Some properties:

$\mathcal{H}_x$ is linear (over k)

- $\mathcal{H}_x(a + b) = \mathcal{H}_x(a) + \mathcal{H}_x(b)$

$\forall n \in \mathbb{N}^*$, $\forall x \in k$, $\forall a \in k^n$,

- $\Pr[\mathcal{H}_x(b) = \mathcal{H}_x(a) : b \xleftarrow{\$} k^n]$
- $= \Pr[\mathcal{H}_x(b - a) = 0 : b \xleftarrow{\$} k^n]$
- $= \Pr[\mathrm{eval}(B - A, x) = 0 : B \xleftarrow{\$} k[X], \deg(B) \le n] \le n/\#k$

How's that useful?

W.h.p., $\neq m \Rightarrow \neq \mathcal{H}_x(m)$

- E.g. take $\#k \approx 2^{128}$, $n = 2^{32}$, the "collision probability" between two messages is $\leq 2^{-96=32-128}$
- This is "optimum"

Problem: for a MAC, linearity is a weakness! (cf. TD)

- One way to solve this: encrypt the result of the hash with a block cipher!

Polynomial MACs

Toy polynomial MAC

Let $\mathcal{H} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a polynomial hash function family, $\mathcal{E} : \mathcal{K}' \times \mathcal{Y} \to \mathcal{Y}$ be a block cipher. The MAC $\mathcal{M} : \mathcal{K} \times \mathcal{K}' \times \mathcal{X} \to \mathcal{Y}$ is defined as $\mathcal{M}(k, k', m) = \mathcal{E}(k', \mathcal{H}_k(m))$.

(Remark: not randomized)

Advantage of polynomial MACs:

- Fast
- Good and "simple" security
  - But still rely on block ciphers and friends!

Examples: UMAC; VMAC; Poly1305-AES; NaT (more sophisticated variant of the above), NaK, HaT, HaK

MACs from hash functions 1

If $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$ is a hash function, one may define:

- $\texttt{PrefixMAC}_{\mathcal{H}} : \{0,1\}^\kappa \times \{0,1\}^* \to \{0,1\}^n$ as $\texttt{PrefixMAC}_{\mathcal{H}}(k,m) = \mathcal{H}(k||m)$
- $\texttt{SuffixMAC}_{\mathcal{H}} : \{0,1\}^\kappa \times \{0,1\}^* \to \{0,1\}^n$ as $\texttt{SuffixMAC}_{\mathcal{H}}(k,m) = \mathcal{H}(m||k)$
- (Note that $\texttt{PrefixMAC}_{\mathcal{H}} \approx \texttt{SuffixMAC}_{\mathcal{H}^\triangleleft}$, where $\mathcal{H}^\triangleleft$ is $\mathcal{H}$ "reversed")

These constructions are fine *generically* but may be weak for some specific hash functions

Length-extension attack for `PrefixMAC`

Let $\mathcal{H}$ be a (narrow-pipe) Merkle-Damgård hash function

- Let $h = \mathcal{H}(m)$ for some m
- Then $\mathcal{H}(m \| \operatorname{pad}(m) \| m') = \mathcal{H}_h(m')$

What consequence for the security of $\texttt{PrefixMAC}_{\mathcal{H}}$?

- Assume an adversary knows m, $t = \texttt{PrefixMAC}_{\mathcal{H}}(k, m)$ and $\kappa = |k|$
- Then $t' = \mathcal{H}'_t(m')$ is a valid tag under k for $m \| \operatorname{pad}(m) \| m'$
  - ($\mathcal{H}'$ is $\mathcal{H}$ with an appropriately modified padding)

$\Rightarrow$ Existential forgeries are trivial!
(NB: Problems also exist for $\texttt{SuffixMAC}_{\mathcal{H}}$)
(NB: Similar attacks apply to raw CBC-MAC from two slides ago)

MACs from hash functions 2

How to defend against the previous attack?

- Use a better $\mathcal{H}$ framework, e.g. a *wide-pipe* Merkle-Damgård hash function (e.g. SHA-512/256) or a sponge (e.g. SHA-3)
- Use a Sandwich MAC construction (e.g. `HMAC`, SandwichMAC, ...)

`HMAC` (Bellare et al., 1996):

- Let $\mathcal{H}$ be a hash function with b-bit blocks, pad a function that pads to b bits with zeroes, $\texttt{opad} = \texttt{0x36}^{b/8}$, $\texttt{ipad} = \texttt{0x5C}^{b/8}$
- Then $\texttt{HMAC}_{\mathcal{H}}(k, m) = \mathcal{H}(\mathrm{pad}(k) \oplus \texttt{opad} \| \mathcal{H}(\mathrm{pad}(k) \oplus \texttt{ipad} \| m))$

HMAC facts

- HMAC is secure up to the birthday bound (of its hash function)
- It only needs black-box calls to a hash function $\Rightarrow$ simple to implement
- It is popular (widespread use in e.g. TLS)
- It is overkill if $\mathcal{H}$ is e.g. wide-pipe
- Some variants exist, some being more efficient

Block cipher v. Hash-based MACs

Block cipher and Hash-based MACs both use a black box to build a MAC, but

- Block cipher block sizes are usually "small" (e.g. 64/128 bits) $\rightsquigarrow$ somewhat limited generic security
- Hash functions are more efficient at processing large amounts of data

$\Rightarrow$ Hash-based MACs tend to be used more than block cipher-based

- But both loose in speed against polynomial MACs (e.g. VMAC) or dedicated constructions (e.g. PelicanMAC)

Introducing Authenticated-Encryption

The “modern” view:
If you must never encrypt w/o authentication, why separating the two? $\Rightarrow$ Authenticated-Encryption

- Maybe more efficient (less redundancy)?
- Maybe more secure (no careless combinations)?
- Maybe more complex

$\rightsquigarrow$ AEAD (Authenticated-Encryption with Associated Data)

AEAD

AEAD

An AEAD scheme is a pair of mappings $(\mathcal{E}, \mathcal{D})$ with:
$\mathcal{E} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{X} \to \mathcal{C}$
$\mathcal{D} : \mathcal{K} \times \mathcal{C} \to \mathcal{X} \cup \{\perp\}$

- $\mathcal{E}$ encrypts a message from $\mathcal{X}$ with a key and a nonce, and authenticates it together with associated data from $\mathcal{A}$
- $\mathcal{D}$ decrypts a ciphertext and returns the message if authentication is successful, or $\perp$ ("bottom") otherwise
- Security is typically analysed w.r.t. IND-CPA (for confidentiality) and IND-CTXT (for integrity)

AEAD designs

An AEAD scheme can be built in many ways:

- By combining a BC mode w/ a MAC (e.g. GCM: CTR mode + a polynomial MAC)
- As a single BC mode (e.g. OCB)
- From a permutation/sponge consruction (e.g. Keyak)
- From a hash function (e.g. OMD)
- From a variable input-length wide-block block cipher (e.g. AEZ)
- Etc.

AEAD: A quick hash function example

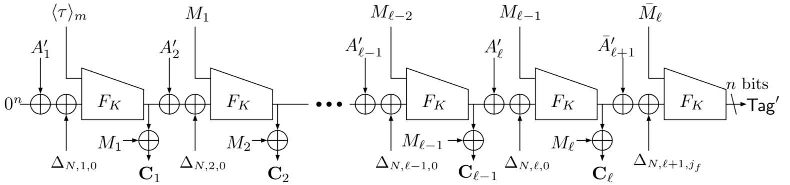

Figure: The p-OMD mode (excerpt; source: p-OMD specifications)

pure Offset Merkle-Damgård (Reyhanitabar et al., 2015), based on a keyed hash function (e.g. SHA-256 w/ semi-secret message)

AEAD: A quick VIL-WBC idea

If $\mathcal{E} : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is a block cipher, one can encrypt *and* authenticate any message m of fixed length $b < n$ by:

- Computing $c = \mathcal{E}(k, m\|0^{n-b}\|r)$
- Decrypting c to m iff. $\mathcal{E}^{-1}(k, c) = m\|0^{n-b}\|*$

If $\mathcal{E}$ is "good", it is "hard" for an adversary to forge $\hat{c}$ s.t. $\mathcal{E}^{-1}(\hat{c})$ has $n - b$ zeroes at specific positions (roughly: success prob. $\approx 2^{b-n}$)

$\rightsquigarrow$ Good paradigm, but very limited if $\mathcal{E}$ has typical block size $n \leq 256$

VIL-WBC

(VIL)-[W]BC

A *Variable input-length wide block cipher* is a family $\mathcal{W} = \{\mathcal{E}^\ell\}$ of mappings $\mathcal{E}^\ell : \mathcal{K} \times \mathcal{X}_\ell \to \mathcal{X}_\ell$ s.t. for all ℓ, $\mathcal{E}^\ell$ is a block cipher, where $\ell \in \mathcal{S} \subseteq \mathbb{N}$

- One can for instance take $\mathcal{X}_\ell = \{0,1\}^\ell$, $\ell \in [2^7, 2^{64}]$
- The PRP security of $\mathcal{W}$ is defined as the $\min_\ell$ PRP security of $\mathcal{E}^\ell$
- ‽The notion of VIL-WBC is stronger than IND-CPA/CCA symmetric encryption
  - Exercise: Why isn't encryption with CBC mode a VIL-WBC?

VIL-WBC constructions

Some various strategies have been proposed to build VIL-WBC

- Sequential two-pass (e.g. CBC-MAC feeding CTR, Bellare and Rogaway, 1999; CBC forward and backward, Houley; Matyas, 1999)
- Wide Feistel (e.g. Naor and Reingold, 1997 $\rightsquigarrow$ Mr Monster Burrito, Bertoni et al., 2014, and several others)
- Parallel Feistel (e.g. AEZ, Hoang et al., 2014)

Maybe not the easiest/fastest way, but conceptually beautiful

Conclusion

- Authentication is essential
- Most of the time, both encryption and authentication are needed
- The "modern" way: do both at the same time
- Still an active research topic (cf. the perpetual CAESAR competition $\rightsquigarrow$ `https://competitions.cr.yp.to/caesar.html`)