Introduction to Cryptology

6.1 - How to construct MACs

Federico Pintore
Mathematical Institute, University of Oxford (UK)

UNIVERSITY OF OXFORD

Michaelmas term 2020

A Fixed-length MAC from a PRF

Let F be a length-preserving pseudorandom function F. Define a fixed-length MAC

$$S = (\mathrm{KeyGen}, \mathrm{Mac}, \mathrm{Verify})$$

for messages of length n, as follows:

- $k \leftarrow \mathrm{KeyGen}(n)$: it takes the security parameter n and outputs a uniformly random key $k \in \{0,1\}^n$.
- $t \leftarrow \mathrm{Mac}(k, m)$: given a key k and a message m, the tag $t := F_k(m)$ is returned.
- $1/0 \leftarrow \mathrm{Verify}(k, m, t)$: it is the canonical verification.

If $|m| \neq |k|$, then Mac outputs $\perp$ and Verify outputs 0.

A fixed-Length MAC from a PRF

Theorem

The fixed-length MAC S for messages of length n is secure.

A fixed-Length MAC from a PRF

Theorem

The fixed-length MAC S for messages of length n is secure.

Proof

An adversary $\mathcal{A}$ against S is exploited as a subroutine to construct a distinguisher D for F:

A fixed-Length MAC from a PRF

Theorem

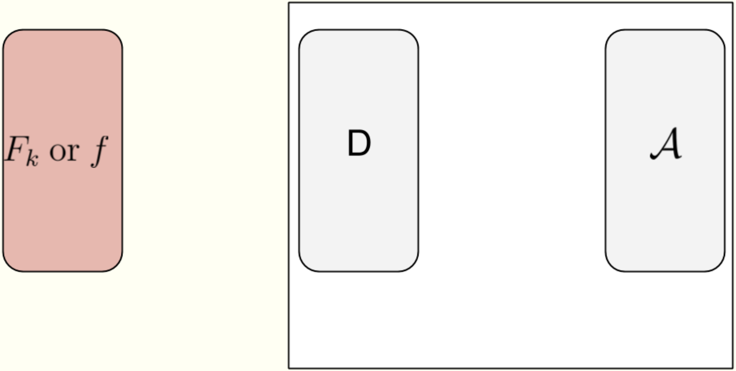
The fixed-length MAC S for messages of length n is secure.

Proof

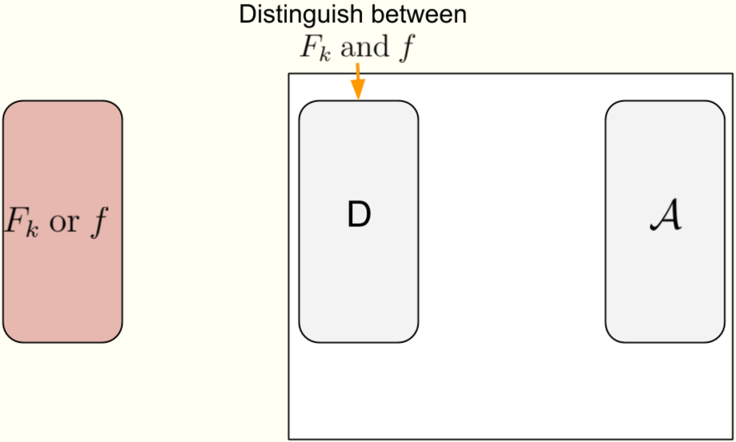
An adversary $\mathcal{A}$ against S is exploited as a subroutine to construct a distinguisher D for F:

- given a query $m_i \in \{0,1\}^n$ from $\mathcal{A}$, D updates the set Q, queries its oracle (F_k or a truly random function f) and returns the answer t;
- to check the validity of $\mathcal{A}$'s forgery (m, t), D queries its oracle as well;
- if the forgery is valid, D outputs 1, otherwise it outputs 0.

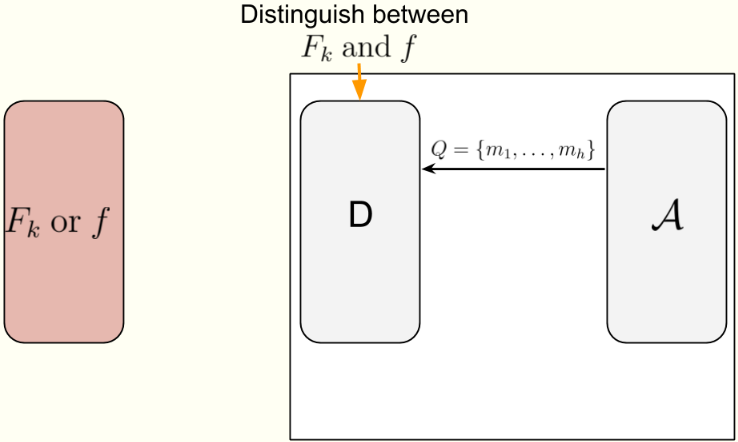
A fixed-Length MAC from a PRF



A fixed-Length MAC from a PRF

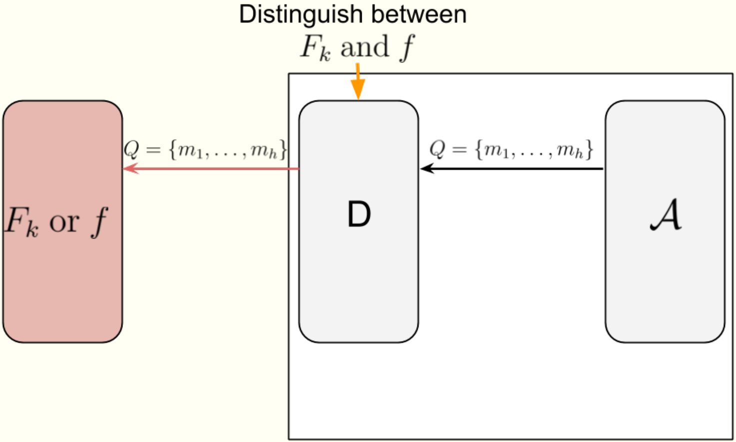


A fixed-Length MAC from a PRF

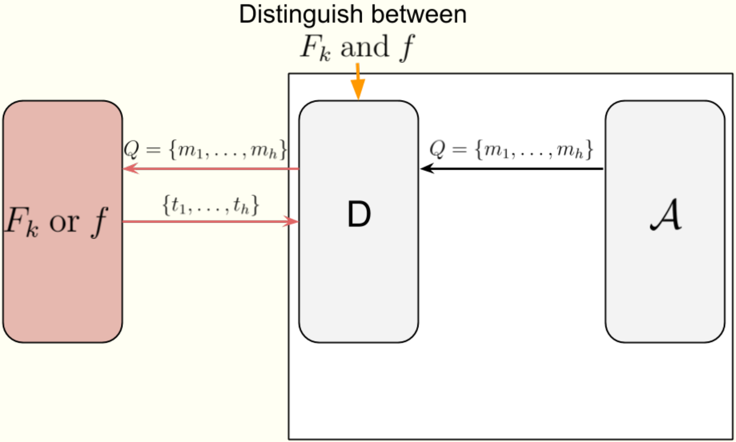


Remark: messages are sent separately!

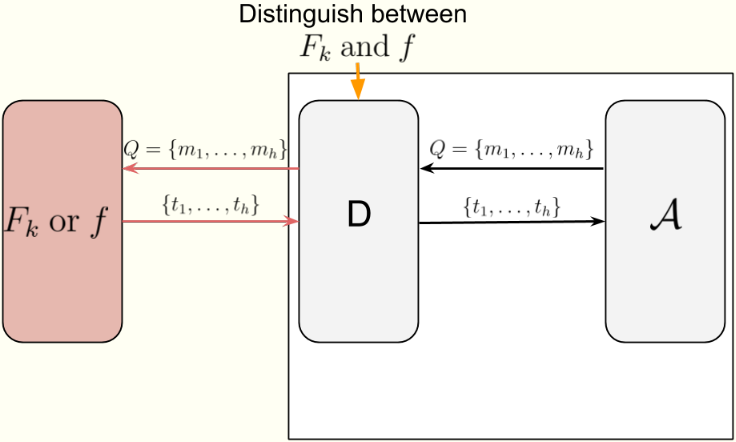
A fixed-Length MAC from a PRF



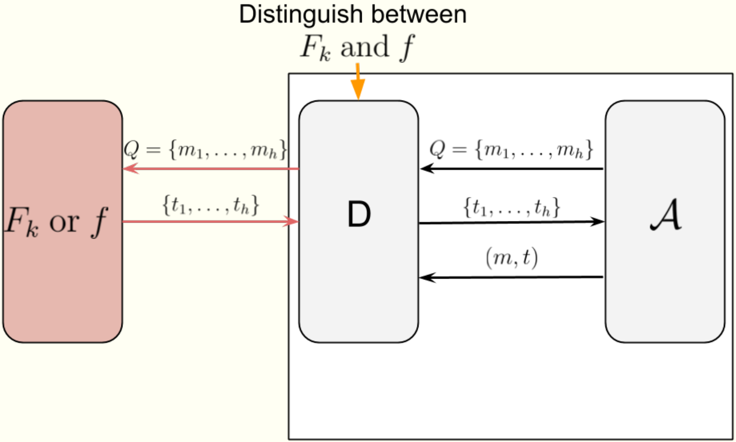
A fixed-Length MAC from a PRF



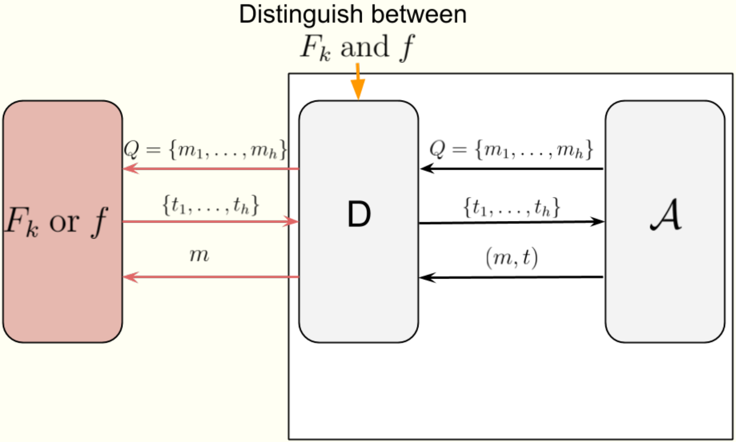
A fixed-Length MAC from a PRF



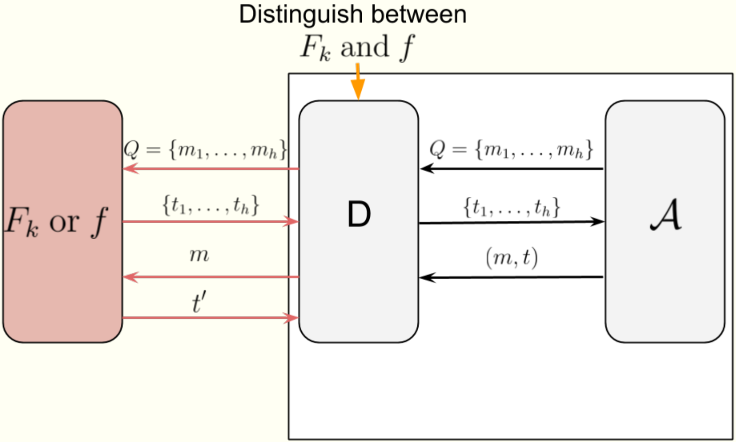
A fixed-Length MAC from a PRF



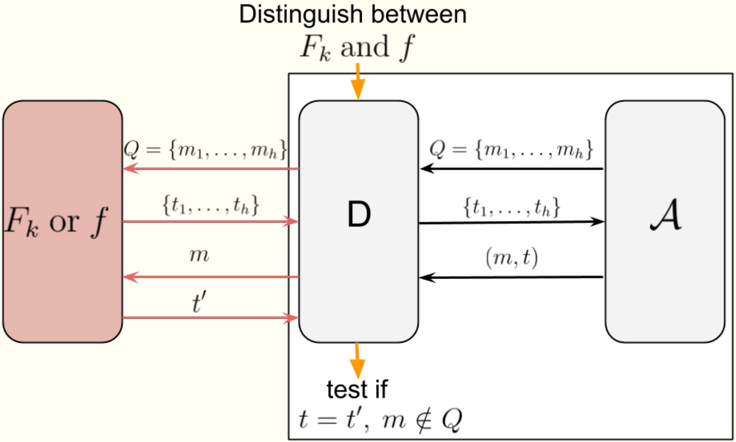
A fixed-Length MAC from a PRF



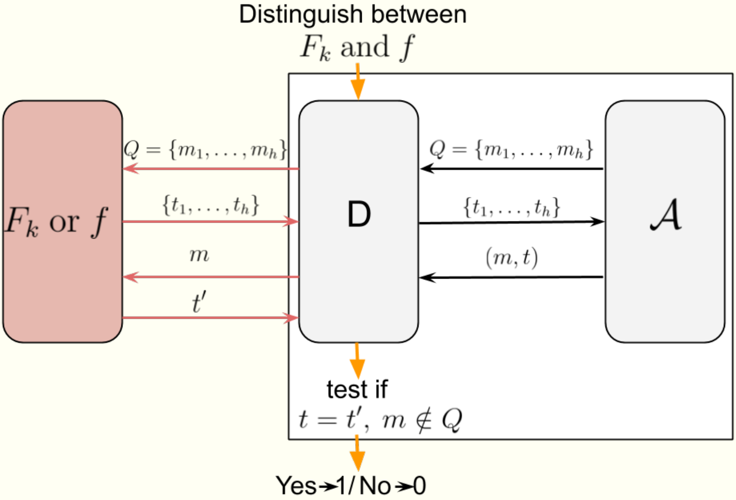
A fixed-Length MAC from a PRF



A fixed-Length MAC from a PRF



A fixed-Length MAC from a PRF



A fixed-Length MAC from a PRF

Let S' be a variation of S, where F_k is replaced by a truly random function $f : \{0,1\}^n \to \{0,1\}^n$.

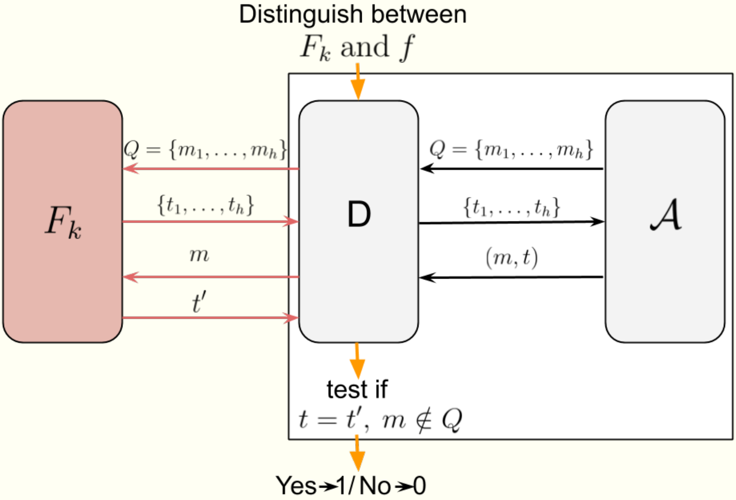
- D has access to F_k: in this case, $\mathcal{A}$ is in the message authentication experiment for S, and

$$\Pr(\mathrm{D}^{F_k()}(n) = 1) = \Pr(\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S}(n) = 1).$$

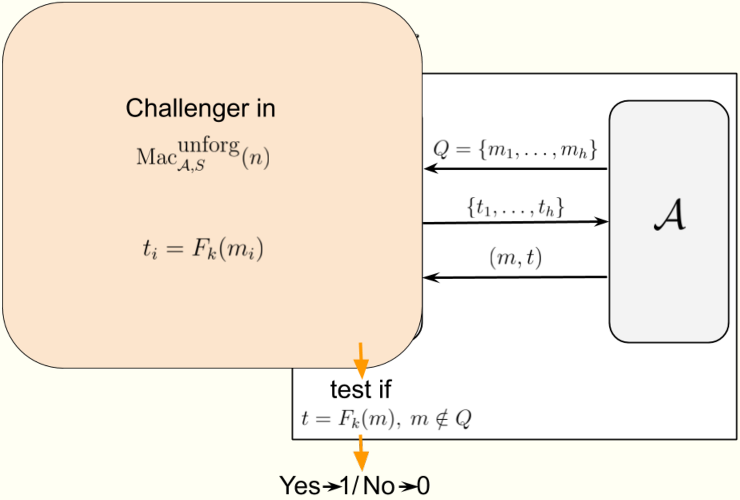
- D has access to f: in this case, $\mathcal{A}$ is in the message authentication experiment for S', and

$$\Pr(\mathrm{D}^{f()}(n) = 1) = \Pr(\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S'}(n) = 1).$$

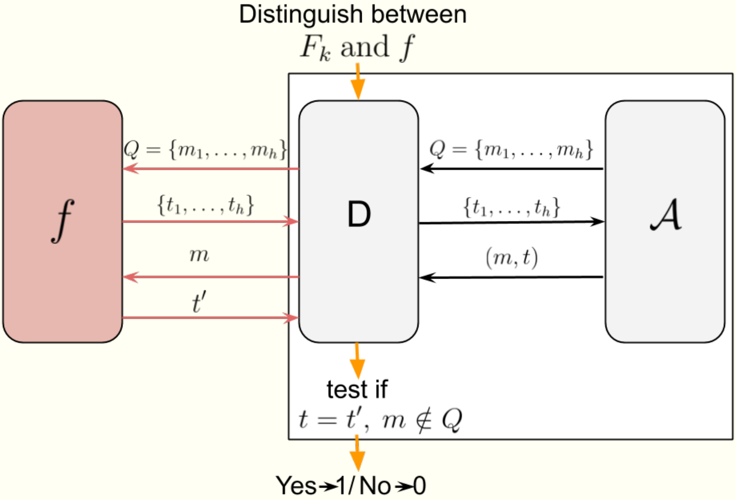
A fixed-Length MAC from a PRF



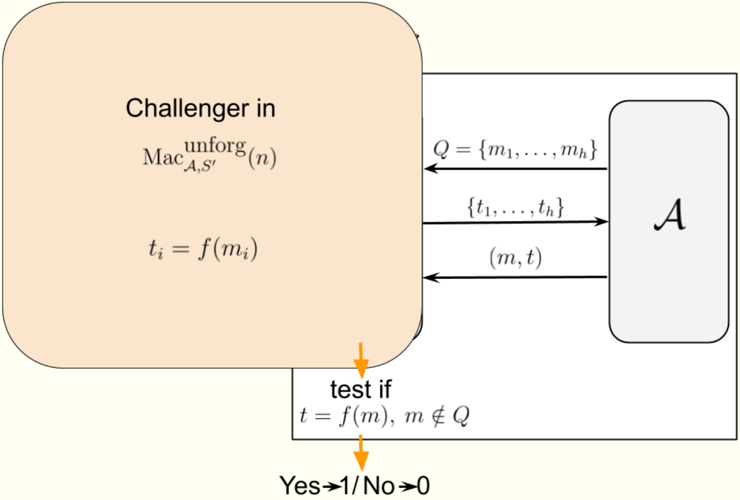
A fixed-Length MAC from a PRF



A fixed-Length MAC from a PRF



A fixed-Length MAC from a PRF



A fixed-Length MAC from a PRF

Since F is a PRF, it holds:

$$|\Pr(D^{f()}(n) = 1) - \Pr(D^{F_k()}(n) = 1)| =$$

$$= |\Pr(\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S'}(n) = 1) - \Pr(\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S}(n) = 1)| \leq \mathrm{negl}(n).$$

For any message $m \notin Q$, the value $t = f(m)$ is uniformly distributed in $\{0,1\}^n$ from the point of view of $\mathcal{A}$. So:

$$\Pr(\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S'}(n) = 1) \leq 2^{-n}.$$

Putting all together we conclude:

$$\Pr(\mathrm{Mac}^{\mathrm{unforg}}_{\mathcal{A},S}(n) = 1) \leq 2^{-n} + \mathrm{negl}(n)\,.$$

$\square$

From a fixed-length MAC to a general MAC

Pseudorandom functions used in practice
(block ciphers) only take short fixed-length inputs.

How to build a MAC for arbitrary-length messages?

From a fixed-length MAC to a general MAC

Natural approach: process each block of the message separately.

From a fixed-length MAC to a general MAC

Natural approach: process each block of the message separately.

Block re-ordering attack: if (t_1, t_2) is a valid tag on (m_1, m_2) where $m_1 \neq m_2$, then (t_2, t_1) is a valid tag on (m_2, m_1).

From a fixed-length MAC to a general MAC

Natural approach: process each block of the message separately.

Block re-ordering attack: if (t_1, t_2) is a valid tag on (m_1, m_2) where $m_1 \neq m_2$, then (t_2, t_1) is a valid tag on (m_2, m_1).

Solution: authenticate a block index with each block.

From a fixed-length MAC to a general MAC

Natural approach: process each block of the message separately.

Block re-ordering attack: if (t_1, t_2) is a valid tag on (m_1, m_2) where $m_1 \neq m_2$, then (t_2, t_1) is a valid tag on (m_2, m_1).

Solution: authenticate a block index with each block.

Truncation attack: the attacker removes blocks from the end of the message and the corresponding blocks from the tag.

From a fixed-length MAC to a general MAC

Natural approach: process each block of the message separately.

Block re-ordering attack: if (t_1, t_2) is a valid tag on (m_1, m_2) where $m_1 \neq m_2$, then (t_2, t_1) is a valid tag on (m_2, m_1).

Solution: authenticate a block index with each block.

Truncation attack: the attacker removes blocks from the end of the message and the corresponding blocks from the tag.

Solution: authenticate the message length with each block.

From a fixed-length MAC to a general MAC

Mix-and-match attack: given the valid tags (t_1, t_2, t_3) and (t'_1, t'_2, t'_3) on the distinct messages (m_1, m_2, m_3) and (m'_1, m'_2, m'_3), output (t_1, t'_2, t_3) on the message (m_1, m'_2, m_3).

From a fixed-length MAC to a general MAC

Mix-and-match attack: given the valid tags (t_1, t_2, t_3) and (t'_1, t'_2, t'_3) on the distinct messages (m_1, m_2, m_3) and (m'_1, m'_2, m'_3), output (t_1, t'_2, t_3) on the message (m_1, m'_2, m_3).

Solution: authenticate a random message identifier along with each block.

From a fixed-length MAC to a general MAC

Mix-and-match attack: given the valid tags (t_1, t_2, t_3) and (t'_1, t'_2, t'_3) on the distinct messages (m_1, m_2, m_3) and (m'_1, m'_2, m'_3), output (t_1, t'_2, t_3) on the message (m_1, m'_2, m_3).

Solution: authenticate a random message identifier along with each block.

Lessons learnt!

From a fixed-length MAC to a general MAC

Let $S_1 = (\mathrm{KeyGen}_1, \mathrm{Mac}_1, \mathrm{Verify}_1)$ be a fixed-length MAC for messages of length n. Define a MAC S for arbitrary-length messages as follows:

- $k \leftarrow \mathrm{KeyGen}(n)$: given the security parameter n, it runs $\mathrm{KeyGen}_1(n)$ and returns its output.
- $t \leftarrow \mathrm{Mac}(k, m)$: given a key k and a message m with $|m| = \ell < 2^{n/4}$, the algorithm
  - parses m into d blocks of length $n/4$, i.e. $m_1, \cdots, m_d$;
  - if the last block is not of size $n/4$, it is padded with $0s$;
  - uniformly chooses $r \in \{0,1\}^{n/4}$;
  - for $i = 1, \cdots, d$, computes $t_i = \mathrm{Mac}_1(k, r||\ell||i||m_i)$, where i and ℓ are encoded as strings of length $n/4$;
  - output $t = (r, t_1, \cdots, t_d)$.
- $1/0 \leftarrow \mathrm{Verify}(k, m, (r, t_1, \cdots, t_d))$: it parses m into d' blocks and returns 1 iff $d' = d$ AND $\mathrm{Verify}_1(k, r||\ell||i||m_i, t_i) = 1\ \forall i$

From a fixed-length MAC to a general MAC

Theorem

If S_1 is a secure fixed-length MAC for messages of length n, then S as defined above is a secure MAC.

From a fixed-length MAC to a general MAC

Theorem

If S_1 is a secure fixed-length MAC for messages of length n, then S as defined above is a secure MAC.

Unfortunately, S is rather inefficient.

From a fixed-length MAC to a general MAC

Theorem

If S_1 is a secure fixed-length MAC for messages of length n, then S as defined above is a secure MAC.

Unfortunately, S is rather inefficient.

More efficient constructions:

- CBC-MAC
- MACs from hash functions (will be covered soon!)

Basic CBC-MAC for fixed-length messages

Let F be a length-preserving pseudorandom function. The basic fixed-length CBC-MAC for messages of length $\ell(n) \cdot n$ is defined as follows:

- $k \leftarrow \mathrm{KeyGen}(n)$: given the security parameter n, it returns a uniform $k \in \{0,1\}^n$.
- $t \leftarrow \mathrm{Mac}(k, m)$: it takes a key k and a message m and
  - parses m as $m_1, \cdots, m_{\ell(n)}$, where $|m_i| = n$;
  - initialises $t_0 \leftarrow 0^n$ and, for $i = 1, \cdots, \ell(n)$, it computes

$$t_i \leftarrow F_k(t_{i-1} \oplus m_i)$$

  - outputs the tag $t_{\ell(n)}$.
- $1/0 \leftarrow \mathrm{Verify}(k, m, t)$: it is the canonical verification with the extra check that $|m|$ is $n \cdot \ell(n)$.

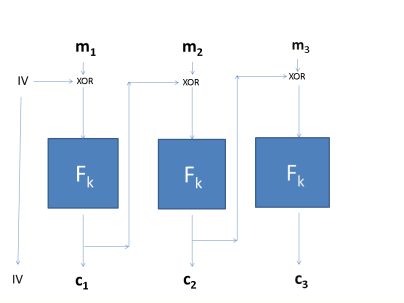
CBC-MAC

The previous construction is secure, but for fixed-length messages.

It is possible to modify the construction in order to handle arbitrary-length messages.

Example: the key generation chooses two uniformly independent keys, $k_1, k_2 \in \{0,1\}^n$. The tagging algorithm obtains t_1 using the CBC-MAC on k_1 and m, and outputs the tag $t = F_{k_2}(t_1)$.

CBC-MAC and CBC-mode encryption



- The CBC-mode encryption takes a random IV, whereas CBC-MAC takes a fixed string (i.e. 0^n). They are only secure under these conditions.
- The CBC-mode encryption outputs all the intermediate values c_i, as they form the ciphertext; CBC-MAC only outputs the final tag $t_{\ell n}$.

Further Reading I

N.J. Al Fardan and K.G. Paterson.
Lucky thirteen: Breaking the TLS and DTLS record protocols.
In Security and Privacy (SP), 2013 IEEE Symposium on, pages 526–540, May 2013.

J Lawrence Carter and Mark N Wegman.
Universal classes of hash functions.
In Proceedings of the ninth annual ACM symposium on Theory of computing, pages 106–112. ACM, 1977.

Jean Paul Degabriele and Kenneth G Paterson.
On the (in) security of IPsec in MAC-then-Encrypt configurations.
In Proceedings of the 17th ACM conference on Computer and communications security, pages 493–504. ACM, 2010.

Further Reading II

Ted Krovetz and Phillip Rogaway.
The software performance of authenticated-encryption modes.
In Fast Software Encryption, pages 306–327. Springer, 2011.

Douglas R. Stinson.
Universal hashing and authentication codes.
Designs, Codes and Cryptography, 4(3):369–380, 1994.