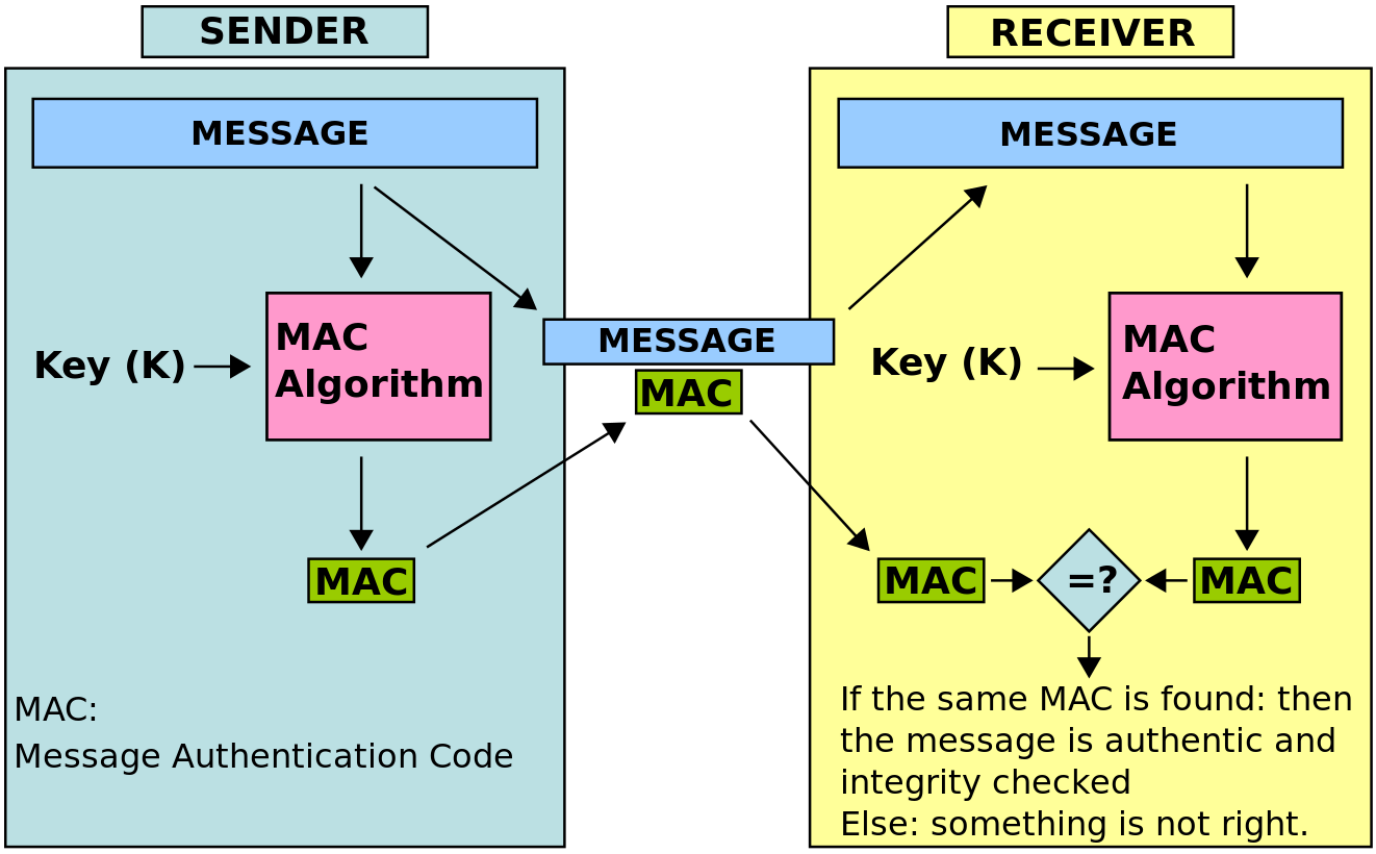
Message Authentication Code

MAC (Message authentication codes)

- https://en.wikipedia.org/wiki/Message_authentication_code
- https://www.tutorialspoint.com/cryptography/message_authentication.htm

Usually we deal with *secrecy* but that is not all. Secrecy does not cover the attacks of an active adversary which may tinker with the message before sending it to another party

Now we will look at methods to prove a message *intergity* (authenticity) which will assure the receiver that the message he receives comes from the real sender.



Types

- Keyless
  - Not designed for security (Ethernet protocol uses CRC32)
  - Detects random transmissions errors
  - Can be bypassed
- with a secret key
  - This is the only way to provide message integrity

Desired proprieties

- Takes in arbitrary length inputs
- Outputs a fixed output
- **Authentication**: Receiver is certain the sender send that message
- **Integrity**: Manipulations will be detected by the receiver

Definition

> A MAC system $I = (S, V)$ is a pair of efficient algorithms,
>
> - S = **probabilistic algorithm** that generates a tag $t \xleftarrow{\text{random}} S(k, m)$
> - V = **deterministic algorithm** $r \leftarrow V(k, m, t)$ -> where r is either `accept` or `reject`.

Idea

- The symmetric key version of public key signatures

Correctness property

- All tags generated by S are verified by V
  $Pr[V(k, m, S(k, m)) = accept] = 1$

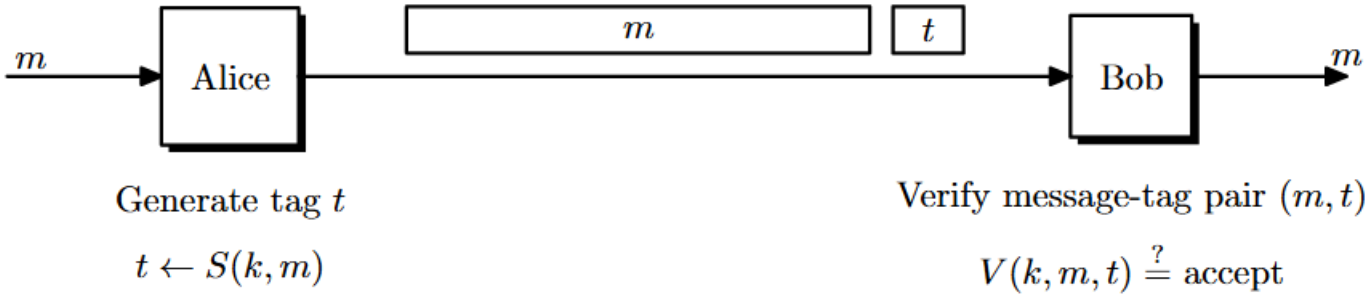


Figure 6.1: Short message integrity tag added to messages

Security

We allow the attacker to request tags $t = S(k, m)$ for messages m of his choice -> (m, t) = **Signed pairs**

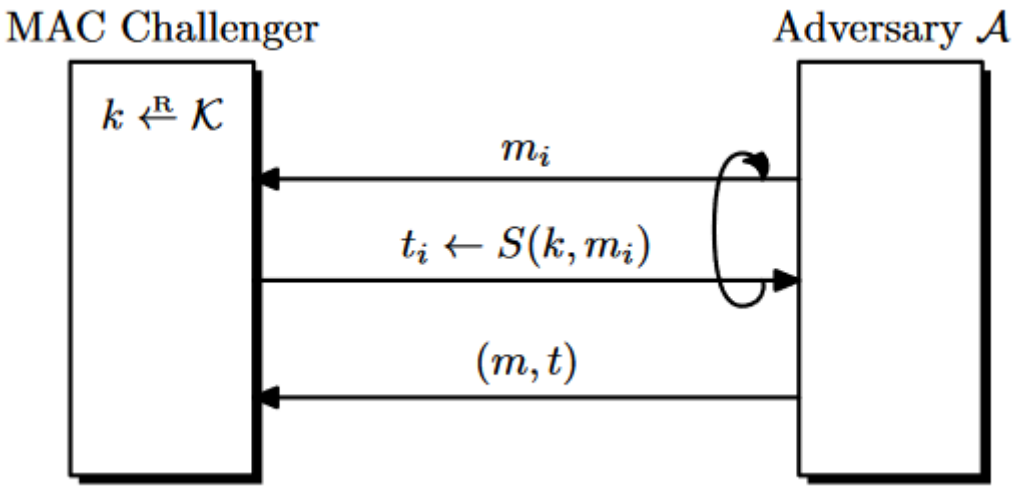
Existential MAC forgery

> The attacker can create a new signed pair (m, t) (different from the others)

Attack game

- challenger picks random $k \in \mathcal{K}$
- Attacker queries $(m_1, ..., m_i)$ and gets the corresponding tags $(t_1, ..., t_i)$
- Attacker can compute (m, t) not seen before

Attacker wins if $V(k, m, t) = accept$



> A MAC system is secure if for all efficient adversaries the probability that the adversary wins(advantage) is neglijable

Note

- Notice that our definition says that there might be multiple tags for the same message $(m, t_1), (m, t_2), ...$
- Our security definition covers that case
  - In order for a MAC system to be secure an attacker must be unable to generate a new tag for a previously signed message
- In the above definition the attacker has no way to verify if a signed pair is accepted or not => He can't even know if he won the game!
  - If we have a secure mac system then it's secure in the presence of verification queries

MACs from PRFs

Reminder

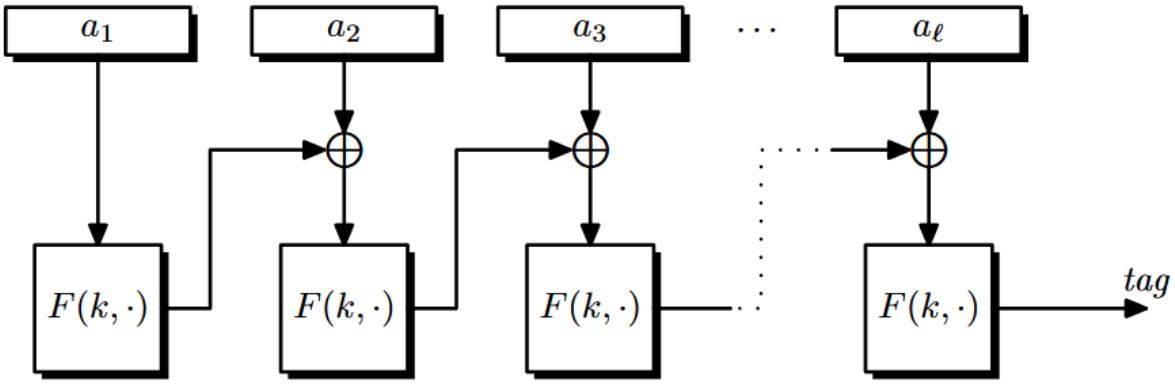
- PRF - $F : \mathcal{K} \times \mathcal{X} \longrightarrow \mathcal{Y}$
- A PRF is unpredictable if the output space is super-poly => A MAC built on it should be secure
- Even if you've seen the output of the PRF on several chosen inputs, all other outputs look independently & uniformly random.

We define the deterministic MAC

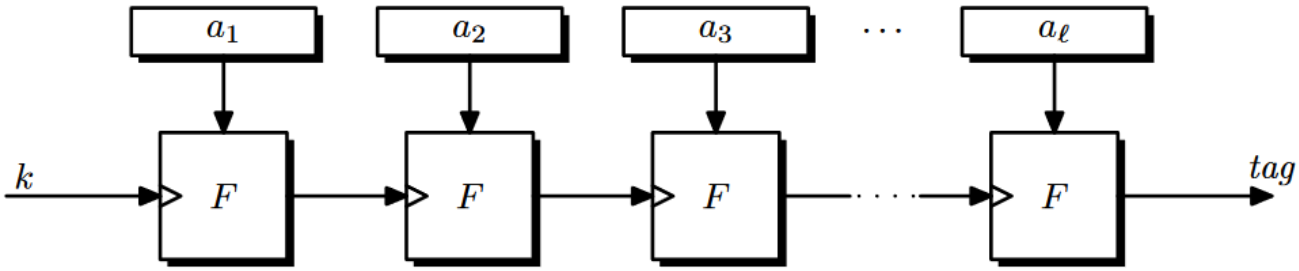
- $S(k, m) := F(k, m)$;
- $V(k, m, t) =$
  - $accept$ if $F(k, m) == t$
  - $reject$ otherwise
    Example:
- AES128

- Also a truncated MAC works (take only a part of the output)

Constructions



(a) The CBC construction $F_{\text{CBC}}(k, m)$



(b) The cascade construction $F^*(k, m)$

Security

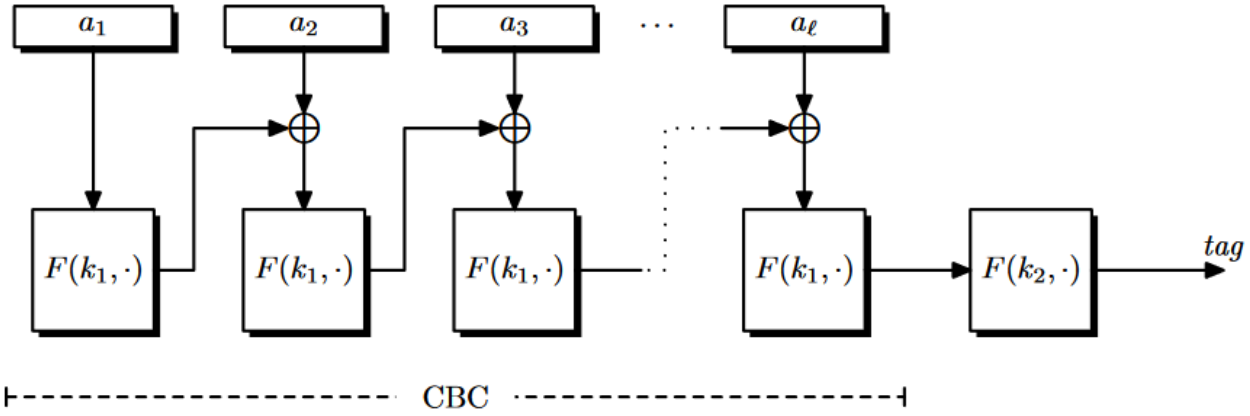
- Those constructions are insecure to **Length-extension** attacks
- Example Cascade construction
  - Let $t' = F(k, m||m')$
  - Knowing $F(k, m)$ you can continue evaluating the chain
- Example CBC construction
  - Pick $a_1 \in \mathcal{X}$
  - Request tag t on one block msg a_1 => $t = F(k, a_1)$
  - Let $a_2 = a_1 \oplus t => a_1 = F(k, a_1) \oplus a_2$ =>
  - $F(k, (a_1, a_2)) = F(k, F(k, a_1) \oplus a_2) = F(k, a_1) = t$ => t is the output for the 2 block msh (a_1, a_2)

Streaming MACs

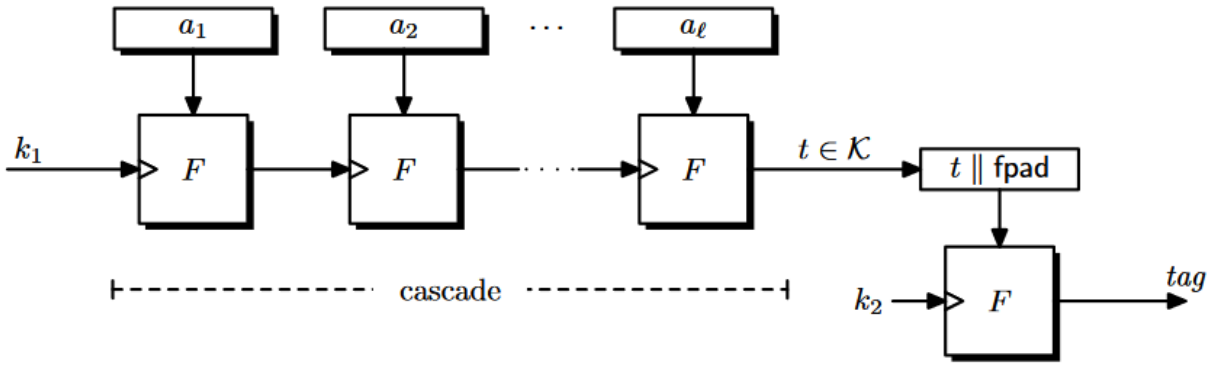
> The length need not to be known ahead of time

- Used for streaming videos
- There are MAC systems that need the message length to be known in advance => harder to use in practice

More secure constructions



(a) The ECBC construction ECBC(k,m) (encrypted CBC)



(b) The NMAC construction NMAC(k,m) (encrypted cascade)

Resources

- https://www.youtube.com/watch?v=wlSG3pEiQdc&t - Computerphile
- https://www.youtube.com/watch?v=Q4EmXJTwcdo
- https://en.wikipedia.org/wiki/Message_authentication_code
- https://cryptobook.nakov.com/mac-and-key-derivation
- http://www.crypto-it.net/eng/theory/mac.html
- - https://www.youtube.com/watch?v=DiLPn_IdAAQ&t- Lecture