Last class:

1. Defined MACs
2. Gave an intuitive argument for why MACs require some sort of magical "randomness generation".
3. Defined a pseudo-random function (PRF)
4. Showed that a PRF is a MAC with msg space being the domain of the PRF.
5. In practice AES is used as a PRF (PRP) with the domain and range being 128 bit strings.

Def: A MAC is secure (existentially unforgeable against adaptive chosen msg attack) if for every efficient adv A

$\Pr[\underset{\uparrow}{A}^{S(K,\cdot)}$ outputs (m^*, σ^*) s.t. $V(K,m^*,\sigma^*)=1$, and m^* was not sent as an oracle query$] =$ negl

oracle access

Going from small MAC to big MAC:

This problem should sound familiar:

Recall the construction of a collision resistant hash function (CRHF),

where we started with SHA2 and extended it using a tree construction. $\longleftarrow$ small CRHF to big CRHF

One approach: Use CRHF!

$S(K,M) = F(K,H(M))$

where F is a PRF (such as AES) and H is a CRHF

This is secure!

Problem: We need a hash function that outputs 128 bits, but hash functions in practice outputs 256 bits.

This is due to the birthday paradox, which allows finding collisions in time sqrt of the range size.

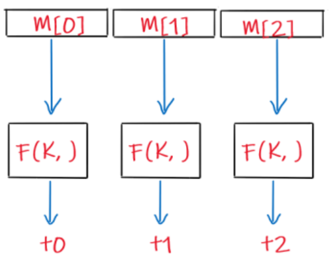
Two MAC constructions standardized by NIST: One based on AES (CMAC) and one based on SHA2 (HMAC).

$\uparrow$ Today!

Later in the course we will see a different construction of authenticated encryption AES-GCM (Gallois Counter Mode)

AES-based MAC (There is also a hash-based MAC called HMAC)

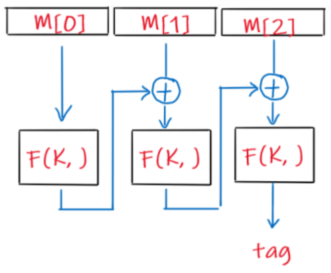
Try 1:



Insecure!

Adversary can use the tag for message (M[0],M[1]) to tag the message (M[1],M[0]).

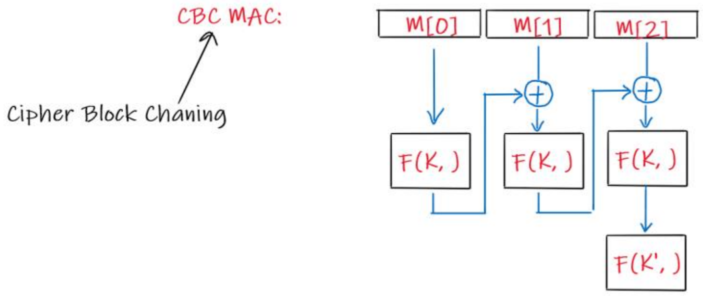
Try 2:



Insecure! Adversary can use the tag for message (M[0],M[1],m[2]) and tag' for message m'[0],

to tag the message (M[0].M[1],M[2],tag xor M'[0]).

Final try:



The secret key is (K,K')

This additional secret key prohibits these "extension attacks"

Similar to why adding the message length in the construction of a collision resistant hash function is needed to make it secure.

Drawback: MACs assume Alice and Bob share a secret key.

Question: What is Alice and Bob are far away and cannot meet to "whisper to each other" the secret key?

Seems impossible without sharing a secret!

Alice must know a secret that the adv does not know, and Bob needs to be able to verify this secret!

$$\not\Downarrow$$

know the secret

Authentication without shared secrets:

Alice has a secret key sk,

Bob has a corresponding public verification key vk=vk(sk), which he used to verify the authenticity of Alice's messages

Digital Signature Scheme:

Alice has a secret key sk, she published a corresponding verification key vk.

She "signs" each of her messages using sk,

anyone can verify her signatures using vk.

Definition:

A digital signature scheme is associated with three PPT (probabilistic poly time)

algorithms: Gen, Sign, Ver, and a message space $\mathcal{M}$

* Gen takes as input a security parameter k, and outputs a pair (sk,vk). $\longleftarrow$ [In theory, Gen takes as input 1^k to allow Gen to run in itme poly(k)]

* Sign takes as input asecret key sk, a message m in $\mathcal{M}$, and outputs a signature σ.

* Ver takes as input a message m in $\mathcal{M}$, a signature σ, and a verification key vk, and outputs 0/1.

Correctness: For a randomly chosen (sk,vk) from Gen, and any messages m in $\mathcal{M}$,

Pr[Ver(vk,m, sign(sk,m))=1]=1

Security: ??

Attacker's power: Chosen message attack.

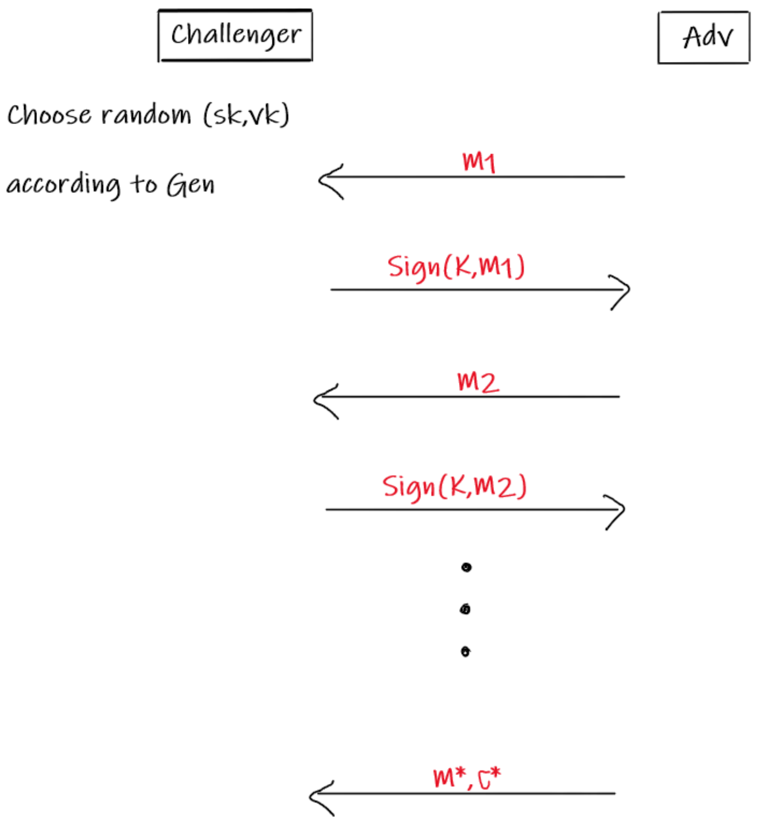
Attacker can obtain valid signatures for any messages of his choice: m1, m2,...

Eg. Alice may sign each email she recieves, thus, an attacker can email her any message of his choice and she will sign it!

Attacker's goal: Existential forgery.

An attacker who is given signatures Sign(sk,m1), Sign(sk,m2),... for messages m1, m2,... of his choice, cannot produce a valid signature for any new message m*.

Security as a game:



Choose random (sk,vk) according to Gen

Adv wins if m* is different from m1,m2,... and V(K,m*,σ*)=1

Def: The signature scheme (Gen, Sign, Ver) is secure (existentially secure against adaptive chosen message attacks) if any efficient adversary wins in this game with negligible probability.

How do we construct a signature scheme??

Step 1: Construct a signature scheme that is one-time secure for msgs of bounded length.

Step 2: From bounded length msgs to unbounded length msgs.

Step 3: From one-time security for unbounded length msgs to standard (many-time) security.

One-time security:

Def: A signature scheme is one-time secure if any efficient adversary given vk (generated by Gen) and given a single msg-signature pair (m,σ) for any msg m of its choice, outputs a pair (m^*,σ^*) s.t. $\mathrm{Ver}(vk, m^*, \sigma^*)=1$ and $m^* \neq m$, only with negligible prob.

Step 1:

Lamport signature scheme: One-time secure signature scheme for bounded length msgs:

Assumes the existence of a one-way function f:

$$\begin{tikzcd} f : X \arrow[r] \arrow[r, bend left=60, "\text{easy}"] & Y \arrow[l, bend left=60, "\text{hard}"] \end{tikzcd}$$

Message space $\{0,1\}^n$

Gen: Chooses at random $x_{10}, x_{11}, \dots, x_{n0}, x_{n1} \in X$

$$sk = \begin{pmatrix} x_{10} & x_{20} & \cdots & x_{n0} \\ x_{11} & x_{21} & & x_{n1} \end{pmatrix} \qquad vk = \begin{pmatrix} f(x_{10}) & f(x_{20}) & \cdots & f(x_{n0}) \\ f(x_{11}) & f(x_{21}) & & f(x_{n1}) \end{pmatrix}$$

Sign: The signature for a msg $m=(m_1,\dots,m_n)$ is $(x_{1,m_1},\dots,x_{n,m_n})$

Ver: Given a msg $m=(m_1,\dots,m_n)$ and a signature $(x'_1,\dots,x'_n)$ output 1 if and only if for every $i \in [n]$, $f(x'_i) = f(x_{i,m_i})$

Correctness: ✓

One-time security:

An adv. breaks the one-time security can be used to break (i.e., invert) the one-way function.

The adv. asks for a signature for a msg m, and produces a signature for a new msg m'.

Suppose $m_i = 0$, $m'_i = 1$. $\longleftarrow$ (simplifying assumption since m and m' can depend on vk.)

Then to invert the OWF, given a random y=f(x), run Gen to obtain (sk,vk) as above, and

replace $f(x_{i,1})$ with y. Denote the resulting verification key by vk'.

Emulate adv. by giving it a signature for m of his choice (as long as $m_i = 0$).

Obtain a msg for m', which includes x' s.t. h(x')=y, thus inverting the OWF, contradiction. ■

Note: The size of the keys grow with the length of the msgs in the msg space.

As we will see, this dependency will cause an exponential blowup when going from

one-time security to full (many-time) security.

Thus, we want to key size to be independent fo the msg length.