

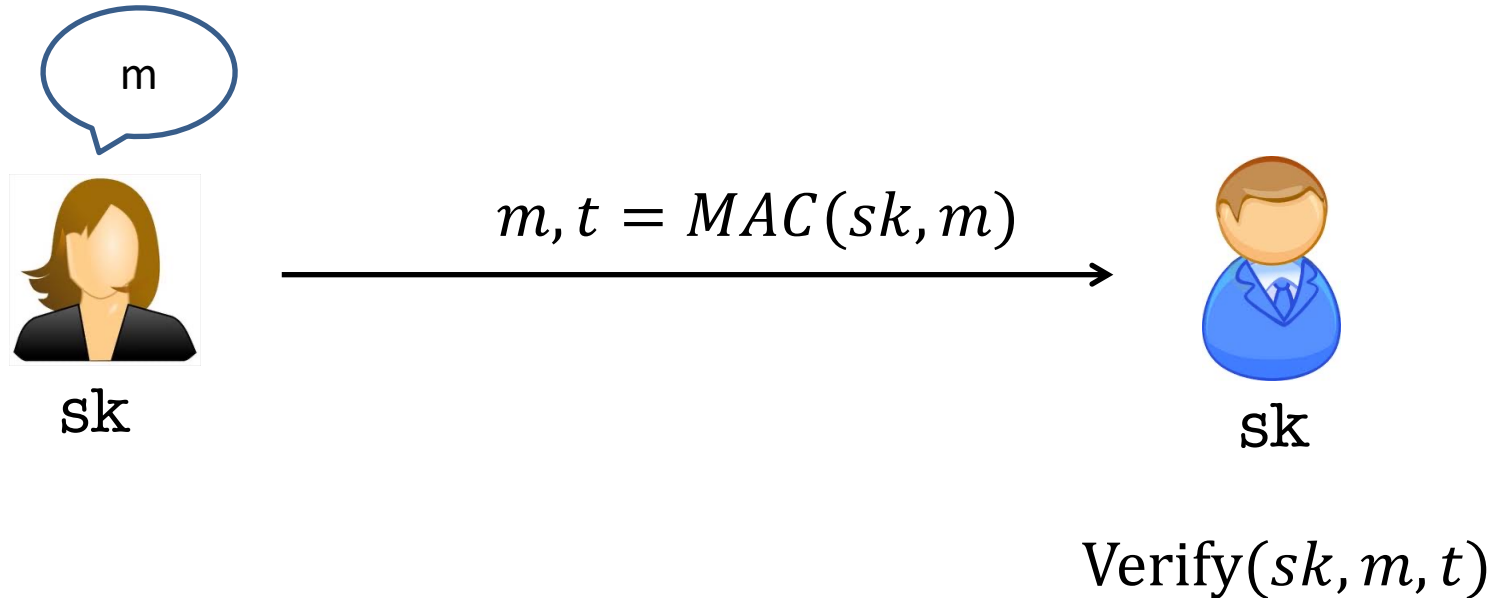
MIT 6.875

Foundations of Cryptography

Lecture 11

TODAY: Digital Signatures

Message Authentication Codes

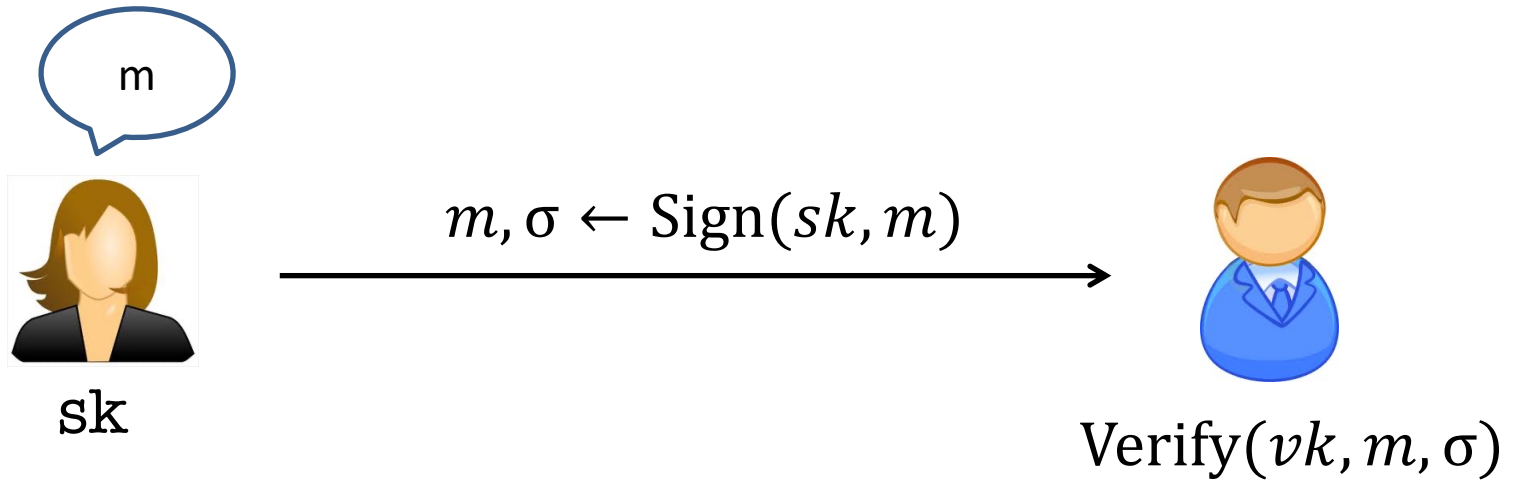


Authenticity: Bob wants to ensure that the message came from Alice.

Needs Bob and Alice to share a secret key beforehand.

Alice	vk

Digital Signatures: Public-key Analog of MACs



(Public) verification keys are stored in a “directory”.

Only Alice can produce signatures; but Bob (or indeed, anyone else) can verify them.

Digital Signatures vs. MACs

Signatures

n users require n key-pairs

Publicly Verifiable

Transferable

Provides Non-Repudiation

(is this a good thing or a bad thing?)

MACs

n users require n^2 keys

Privately Verifiable

Not Transferable

Does not provide Non-Rep.

Other Applications

Alice	pk, vk

1. *Certificates*, or a public-key directory in practice:

Trusted Certificate Authority, e.g. Verisign, Let's Encrypt.

When Alice (= *www.google.com*) wants to register her public (encryption and signing) keys pk and vk , first check that she *is* Alice.

Issue a “certificate” $\sigma \leftarrow \text{Sign}(SK_{\text{Verisign}}, \text{Alice} || pk || vk)$

Alice can later produce this certificate to prove that she “owns” pk and vk .

Browsers store VK_{Verisign} and check the certificate.

Other Applications




2. Bitcoin and other cryptocurrencies:

I am identified by my verification key vk .

When I pay you ($= vk'$), I sign “\$x paid to vk' ” with my sk .

Digital Signatures: Definition

A triple of PPT algorithms $(Gen, Sign, Verify)$ s.t.

- $(vk, sk) \leftarrow Gen(1^n)$. 
PPT Key generation algorithm generates a public-private key pair.
- $\sigma \leftarrow Sign(sk, m)$. 
(possibly probabilistic) Signing algorithm uses the secret signing key to produce a signature σ .
- $Acc(1)/Rej(0) \leftarrow Verify$  (vk, m, σ) .
Verification algorithm uses the public verification key to check the signature σ against a message m .

Correctness: For all vk, sk, m :

$$Verify(vk, m, Sign(sk, m)) = \text{accept.}$$

Digital Signatures: Security

“The adversary after seeing signatures of many msgs, should not be able to produce a signature of any new msg.”

1. What are the adversary’s powers? Request for, and obtain, signatures of (poly many) messages m_1, m_2, \dots

Chosen-message attack

2. What is her goal? She wins if she produces a signature of any message $m^* \notin \{m_1, m_2, \dots\}$.

Existential Forgery

EUF-CMA Security

(Existentially Unforgeable against a Chosen Message Attack)



Challenger



Eve

$(vk, sk) \leftarrow Gen(1^n)$

vk



m_i



$\sigma_i \leftarrow Sign(sk, m_i)$

σ_i



poly many times

m^*, σ^*



Eve wins if $Verify(vk, m^*, \sigma^*) = 1$ and $m^* \notin \{m_1, m_2, \dots\}$.

The signature scheme is EUF-CMA-secure if no PPT Eve can win with probability better than $\text{negl}(n)$.

Lamport (One-time) Signatures

How to sign a bit

Signing Key SK : $[x_0, x_1]$

Verification Key VK : $[y_0 = f(x_0), y_1 = f(x_1)]$

Signing a bit b : The signature is $\sigma = x_b$

Verifying (b, σ) : Check if $f(\sigma) \stackrel{?}{=} y_b$

Claim: Assuming f is a OWF, no PPT adversary can produce a signature of \bar{b} given a signature of b .

Lamport (One-time) Signatures

How to sign n bits

Signing Key SK : $\begin{bmatrix} x_{1,0} & x_{2,0} & \dots & x_{n,0} \\ x_{1,1} & x_{2,1} & \dots & x_{n,1} \end{bmatrix}$

Verification Key VK : $\begin{bmatrix} y_{1,0} & y_{2,0} & \dots & y_{n,0} \\ y_{1,1} & y_{2,1} & \dots & y_{n,1} \end{bmatrix}$

where $y_{i,c} = f(x_{i,c})$.

Signing an n -bit message (m_1, \dots, m_n) :

The signature is $(x_{1,m_1}, \dots, x_{n,m_n})$.

Verifying $(\vec{m}, \vec{\sigma})$: Check if $\forall i: f(\sigma_i) \stackrel{?}{=} y_{i,m_i}$