

COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Spring 2020

Announcements

PR2 Due April 19th

HW6 Due April 23rd

Previously on COS 433...

PKE Syntax

Message space \mathbf{M}

Algorithms:

- $(\mathbf{sk}, \mathbf{pk}) \leftarrow \mathbf{Gen}(\lambda)$
- $\mathbf{Enc}(\mathbf{pk}, m)$
- $\mathbf{Dec}(\mathbf{sk}, m)$

Correctness:

$$\Pr[\mathbf{Dec}(\mathbf{sk}, \mathbf{Enc}(\mathbf{pk}, m)) = m : (\mathbf{sk}, \mathbf{pk}) \leftarrow \mathbf{Gen}(\lambda)] = 1$$

Security

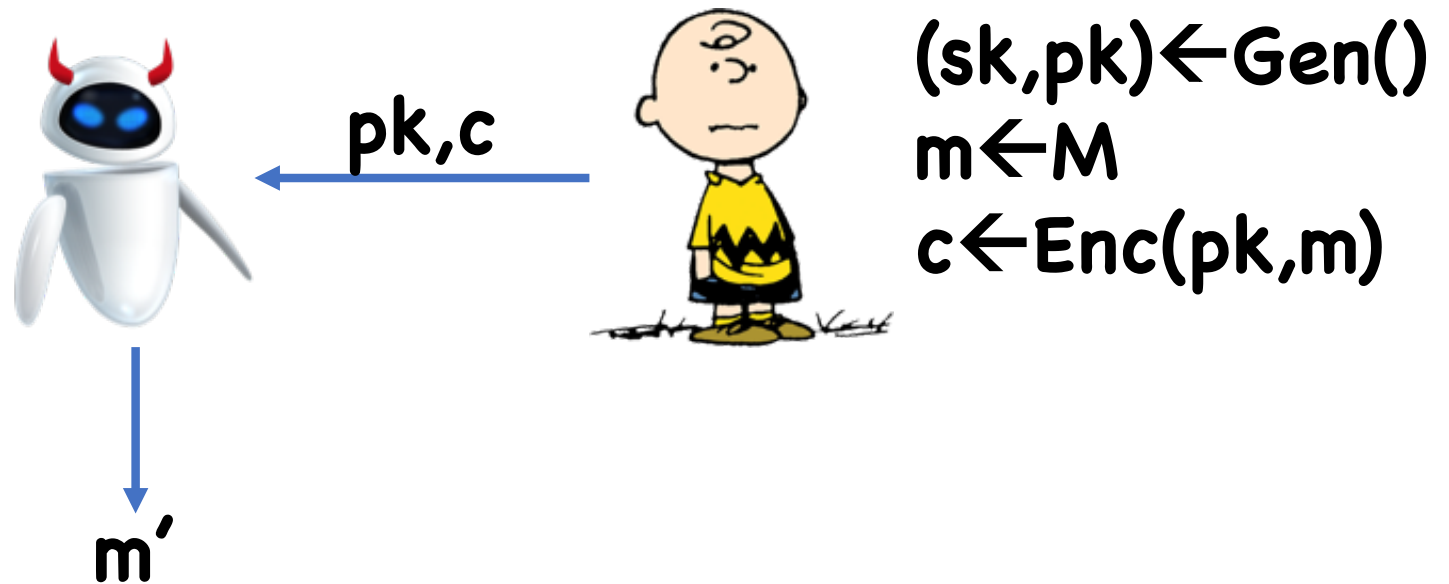
One-way security

Semantic Security

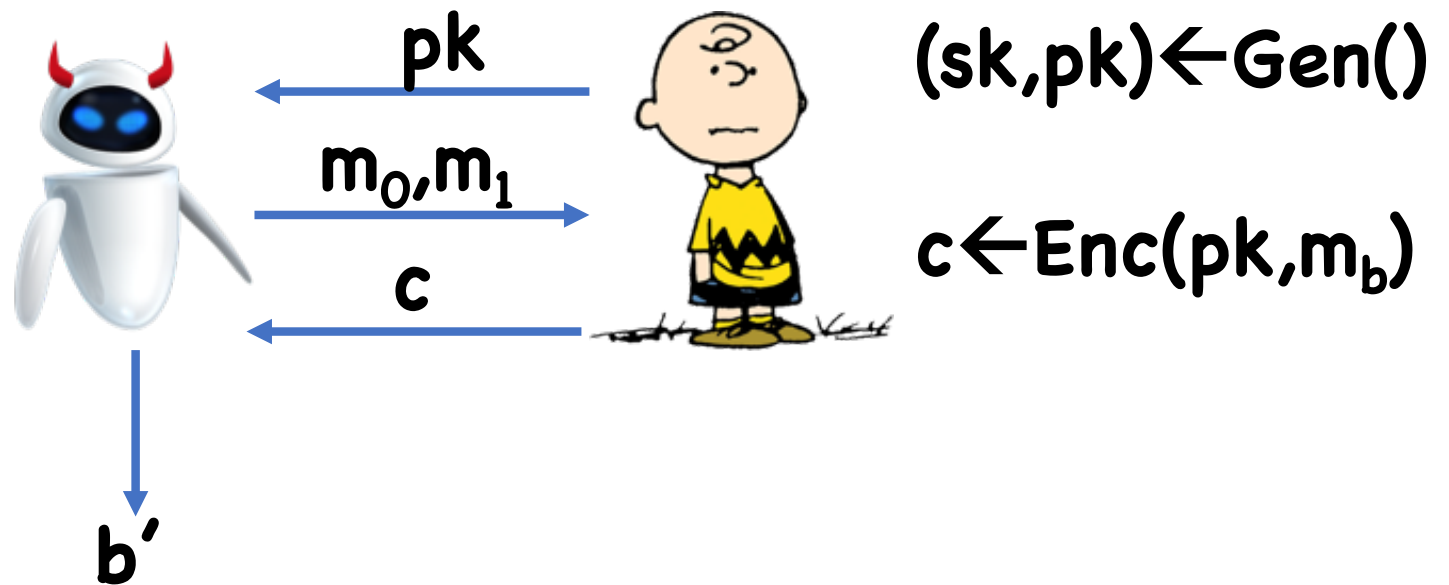
CPA security

CCA Security

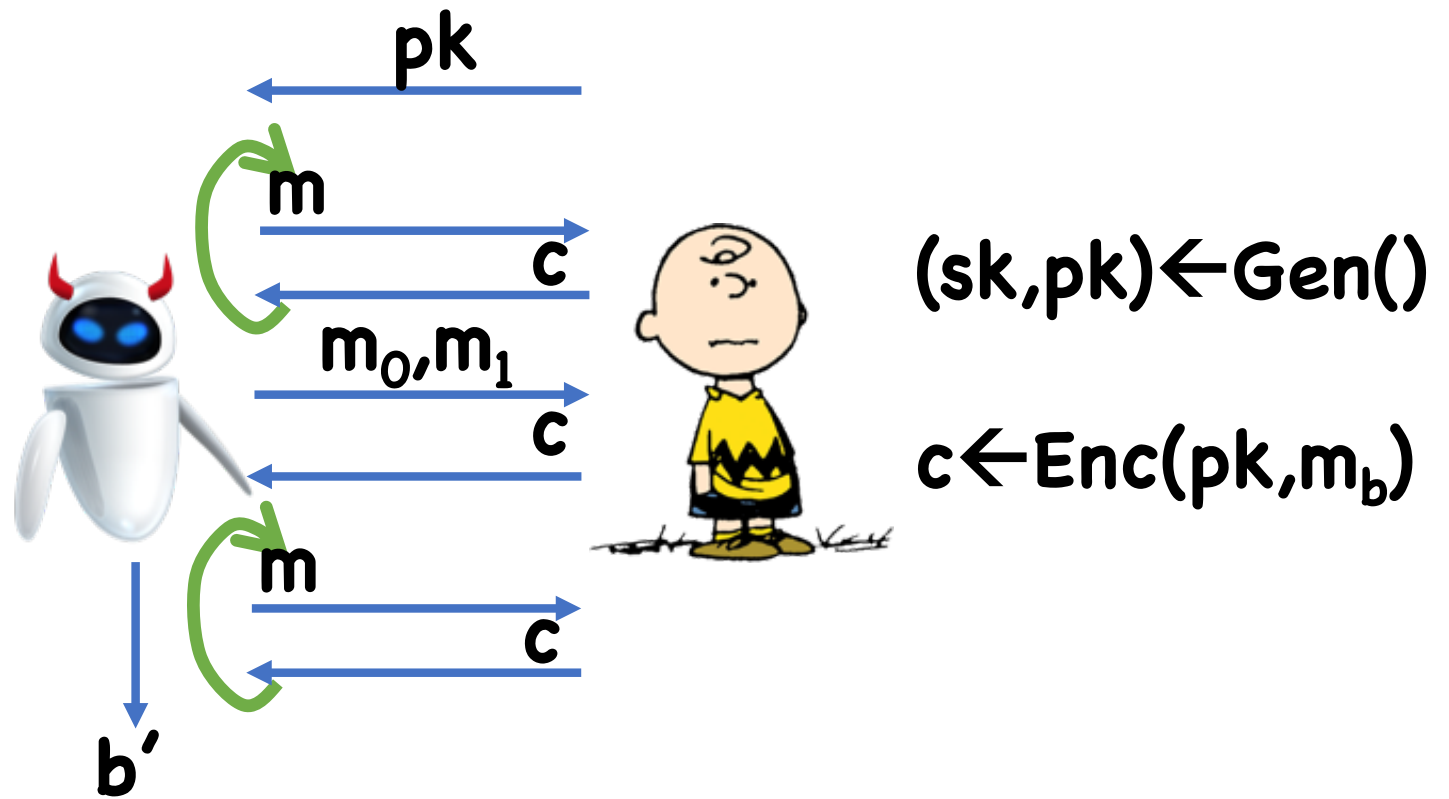
One-way Security



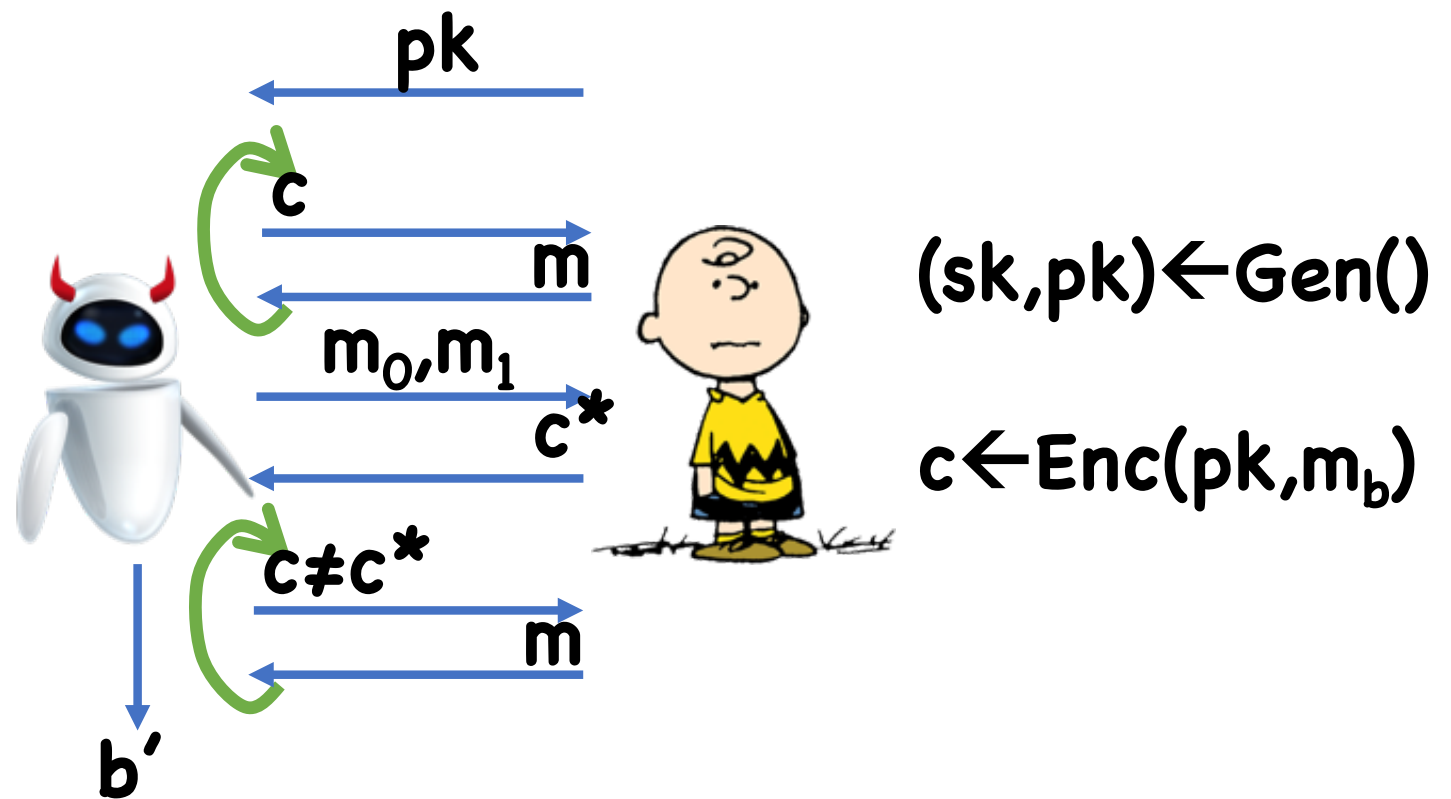
Semantic Security



CPA Security



CCA Security



Q: Why no public key
authenticated encryption?

One-Way Encryption from TDPs

$\text{Gen}_E() = \text{Gen}_{\text{TDP}}()$

$\text{Enc}(\text{pk}, m)$: Output $c = F(\text{pk}, m)$

$\text{Dec}(\text{sk}, c)$: Output $m' = F^{-1}(\text{sk}, c)$

ElGamal

Group \mathbf{G} of order \mathbf{p} , generator \mathbf{g}
Message space = \mathbf{G}

Gen():

- Choose random $\mathbf{a} \leftarrow \mathbb{Z}_p^*$, let $\mathbf{h} \leftarrow \mathbf{g}^{\mathbf{a}}$
- $\mathbf{pk}=\mathbf{h}$, $\mathbf{sk}=\mathbf{a}$

Enc(pk, $m \in \{0,1\}$):

- $\mathbf{r} \leftarrow \mathbb{Z}_p$
- $\mathbf{c} = (\mathbf{g}^{\mathbf{r}}, \mathbf{h}^{\mathbf{r}} \times \mathbf{m})$

Dec?

Today

CCA Secure Encryption

Digital Signatures

CCA-Secure Encryption

Non-trivial to construct with provable security

Most efficient constructions have heuristic security

CCA Secure PKE from TDPs

Let $(\mathbf{Enc}_{SKE}, \mathbf{Dec}_{SKE})$ be a CCA-secure secret key encryption scheme.

Let $(\mathbf{Gen}, \mathbf{F}, \mathbf{F}^{-1})$ be a TDP

Let \mathbf{H} be a hash function

CCA Secure PKE from TDPs

$\text{Gen}_{\text{PKE}}() = \text{Gen}()$

$\text{Enc}_{\text{PKE}}(\text{pk}, m)$:

- Choose random r
- Let $c \leftarrow F(\text{pk}, r)$
- Let $d \leftarrow \text{Enc}_{\text{SKE}}(H(r), m)$
- Output (c_0, c_1)

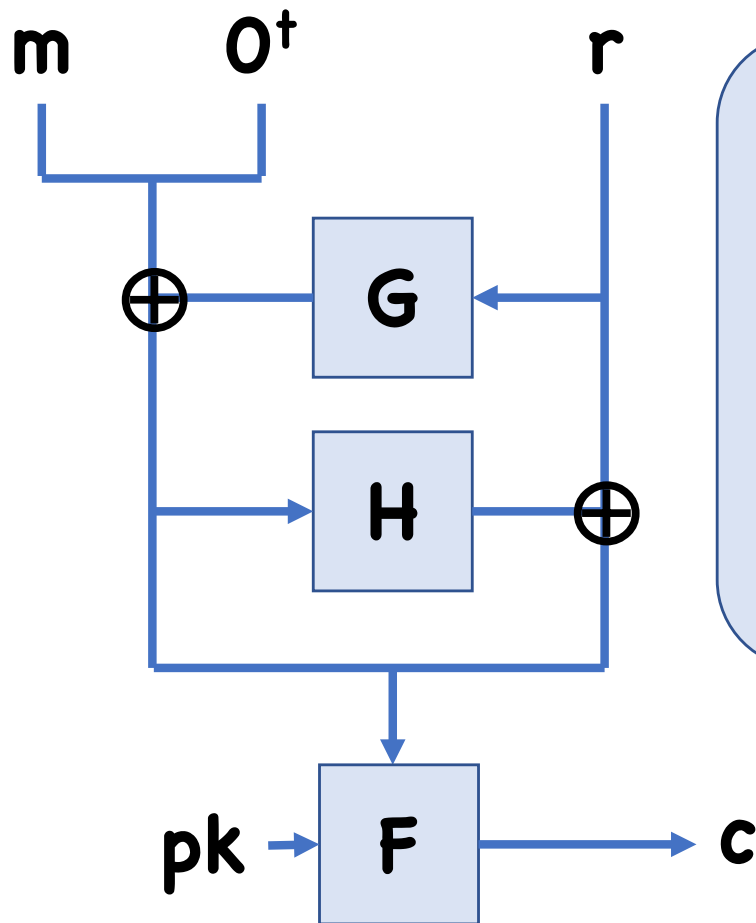
$\text{Dec}_{\text{PKE}}(\text{sk}, (c, d))$:

- Let $r \leftarrow F^{-1}(\text{sk}, c)$
- Let $m \leftarrow \text{Dec}_{\text{SKE}}(H(r), d)$

CCA Secure PKE from TDPs

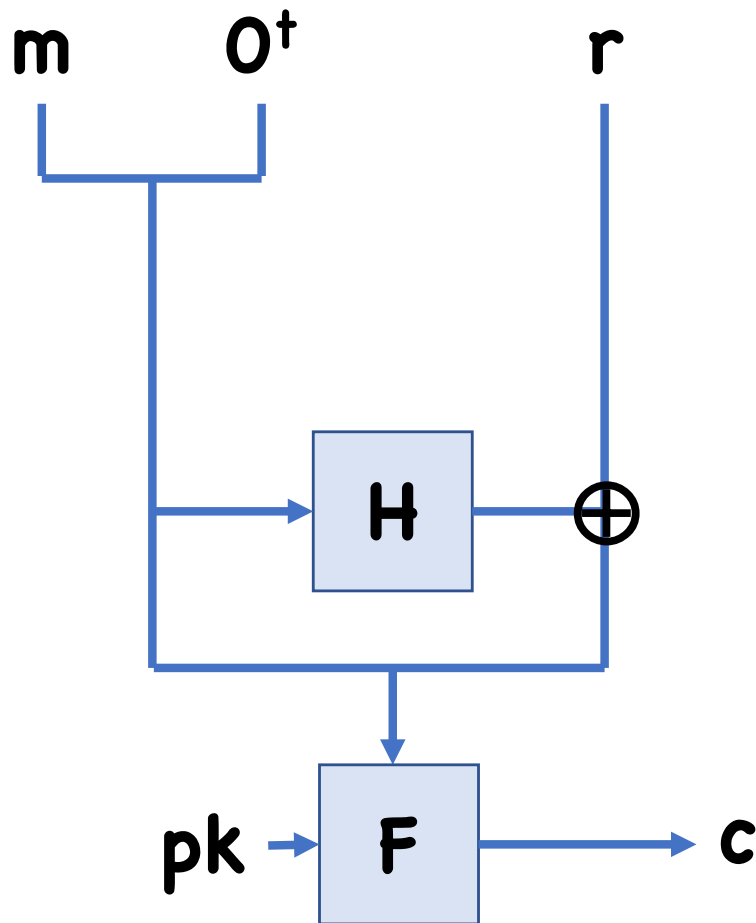
Theorem: If $(\text{Enc}_{\text{SKE}}, \text{Dec}_{\text{SKE}})$ is a CCA-secure secret key encryption scheme, $(\text{Gen}, \text{F}, \text{F}^{-1})$ is a TDP, and H is modeled as a random oracle, then $(\text{Gen}_{\text{PKE}}, \text{Enc}_{\text{PKE}}, \text{Dec}_{\text{PKE}})$ is a CCA secure public key encryption scheme

OAEP



Theorem: For RSA TDP, if G, H are modeled as a random oracles, then $(\text{Gen}_{\text{PKE}}, \text{Enc}_{\text{PKE}}, \text{Dec}_{\text{PKE}})$ is a CCA secure public key encryption scheme

Insecure OAEP Variants

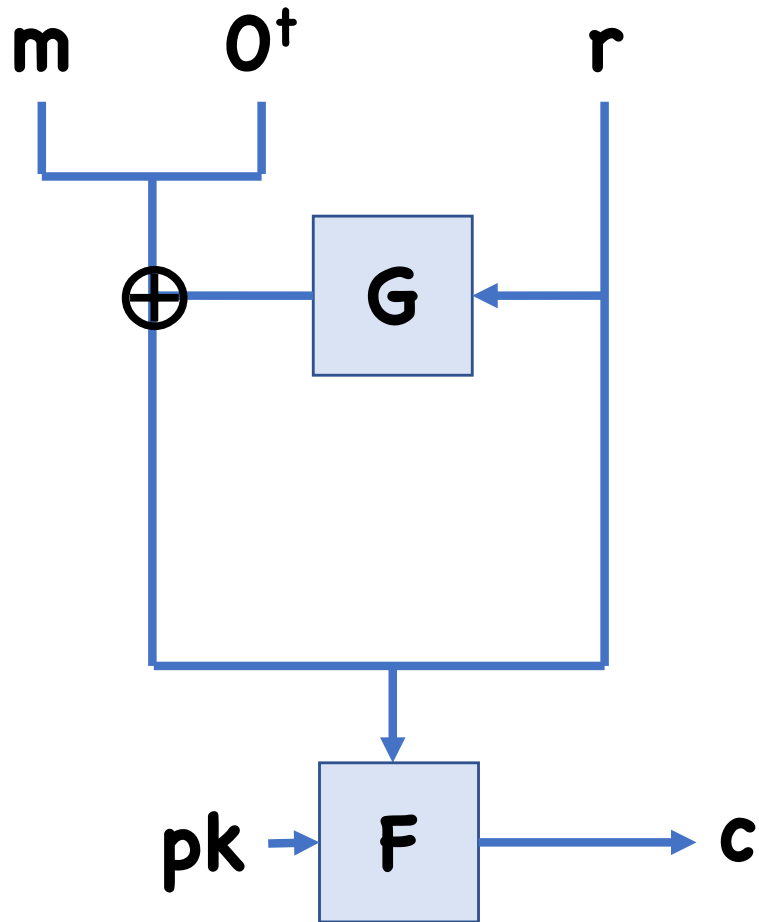


$$c = F(pk, (m, O^+, y))$$

May contain m in the clear

- $F(pk, (m, x, y))$
= $(m, F'(pk, (x, y)))$

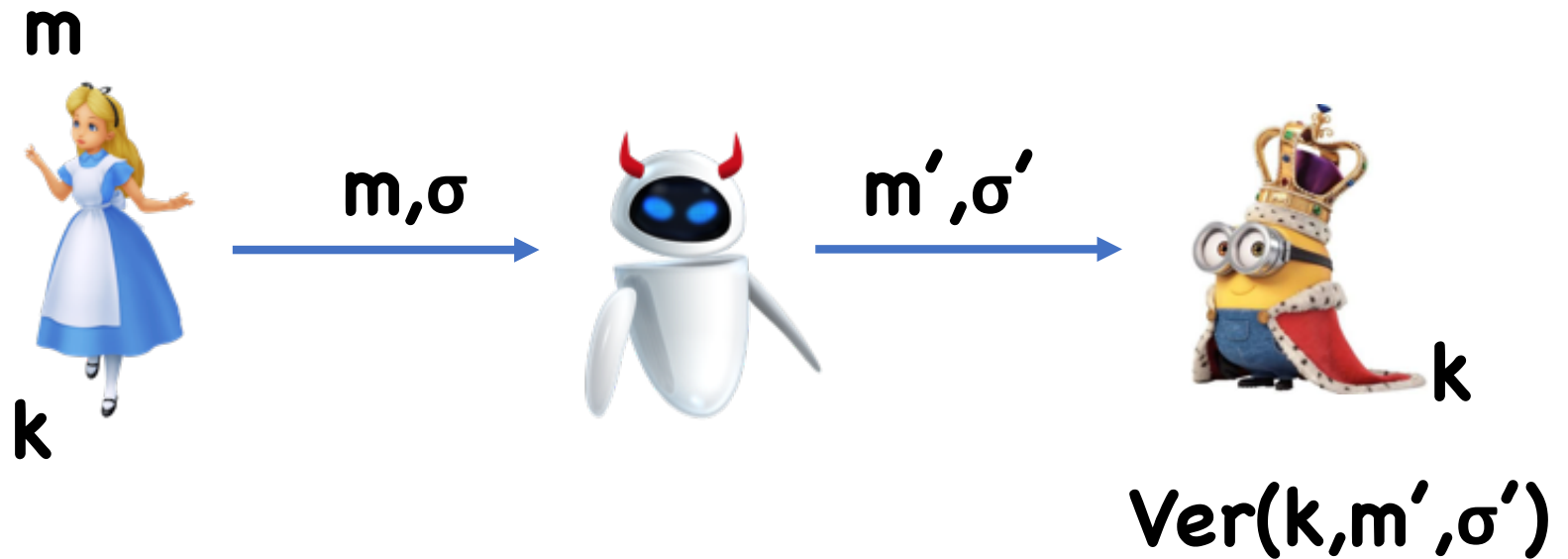
Insecure OAEP Variants



Digital Signatures

(aka public key MACs)

Message Authentication Codes



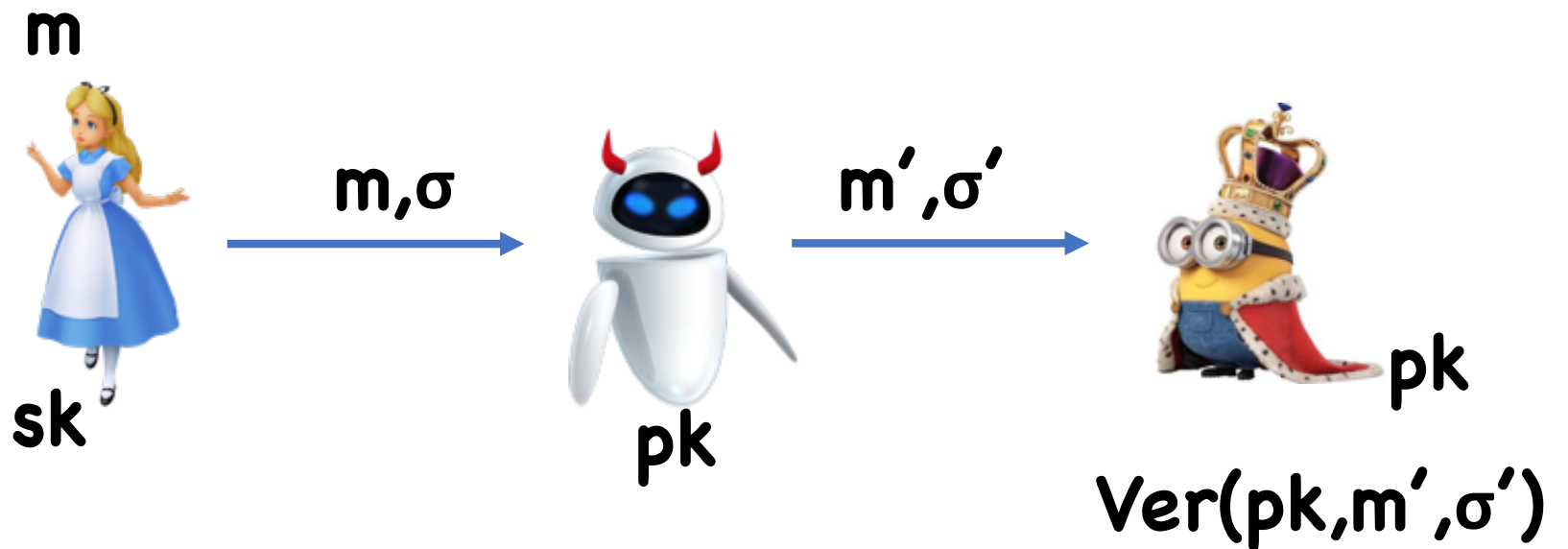
Goal: If Eve changed m , Bob should reject

Problem

What if Alice and Bob have never met before to exchange key \mathbf{k} ?

Want: a public key version of MACs where Bob can verify without having Alice's secret key

Message Integrity in Public Key Setting



Goal: If Eve changed m , Bob should reject

Digital Signatures

Algorithms:

- **Gen()** \rightarrow (sk, pk)
- **Sign(sk, m)** \rightarrow σ
- **Ver(pk, m, σ)** \rightarrow 0/1

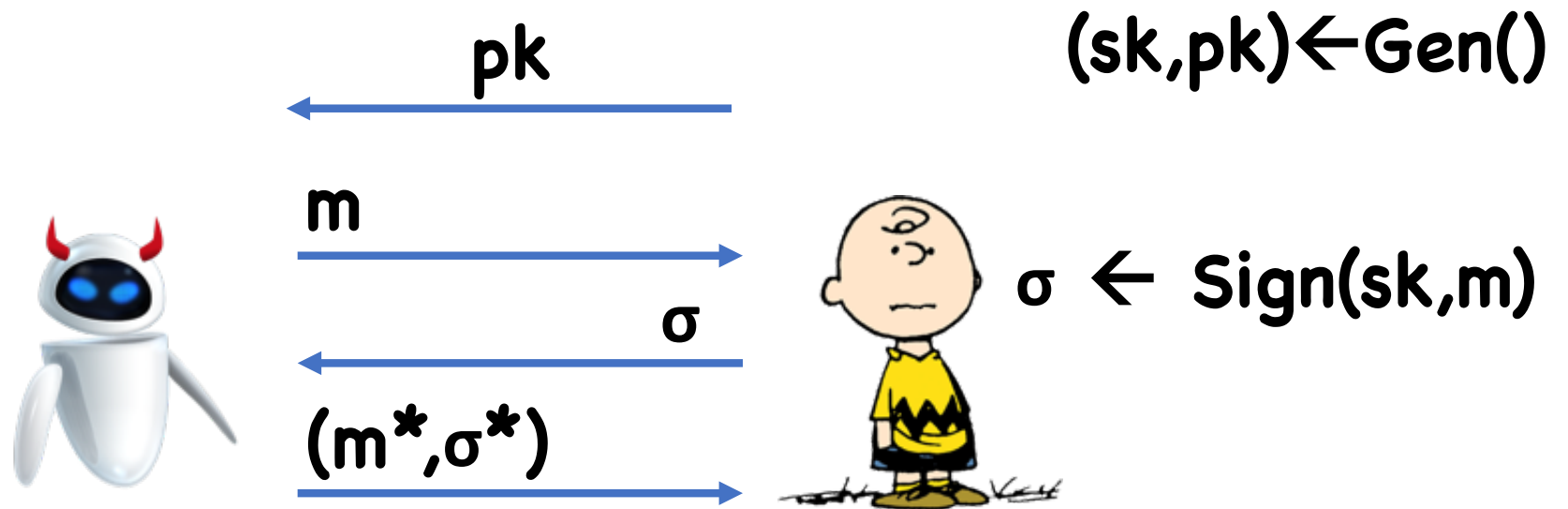
Correctness:

$$\Pr[\text{Ver}(\text{pk}, m, \text{Sign}(\text{sk}, m)) = 1 : (\text{sk}, \text{pk}) \leftarrow \text{Gen}()] = 1$$

Security Notions?

Much the same as MACs, except adversary gets verification key

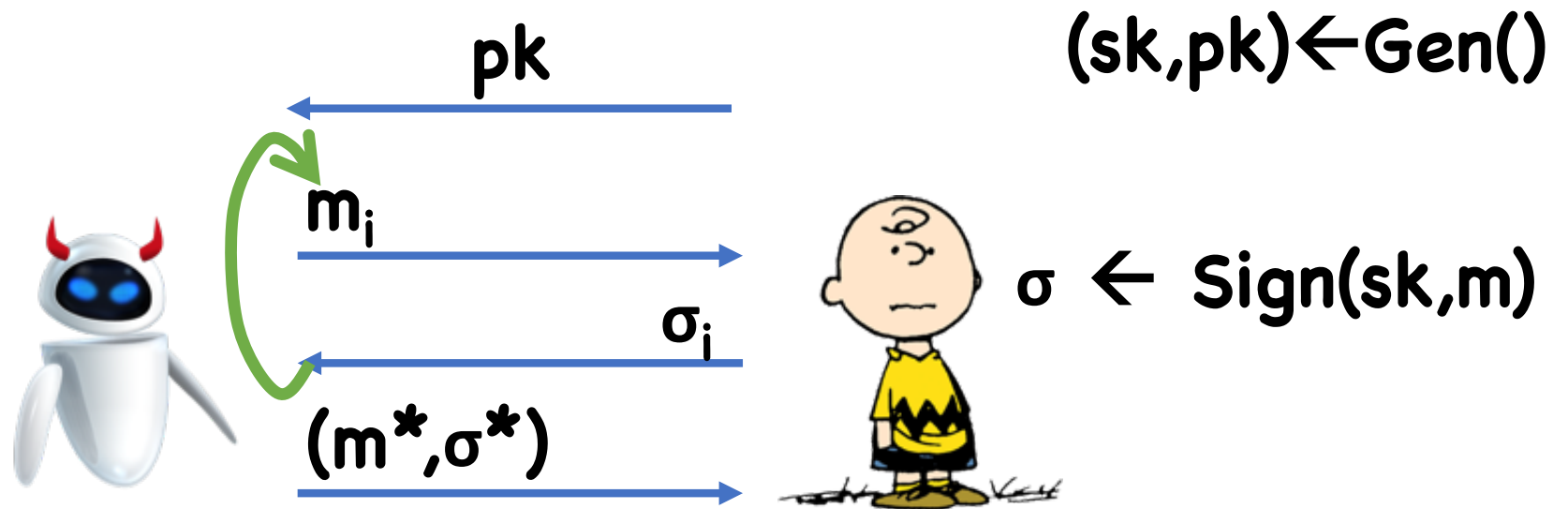
1-time Security For Signatures



- Output 1 iff:
- $m^* \neq m$
 - $\text{Ver}(pk, m^*, \sigma^*) = 1$

$$\text{1CMA-Adv}(\text{robot}) = \Pr[\text{Charlie Brown outputs 1}]$$

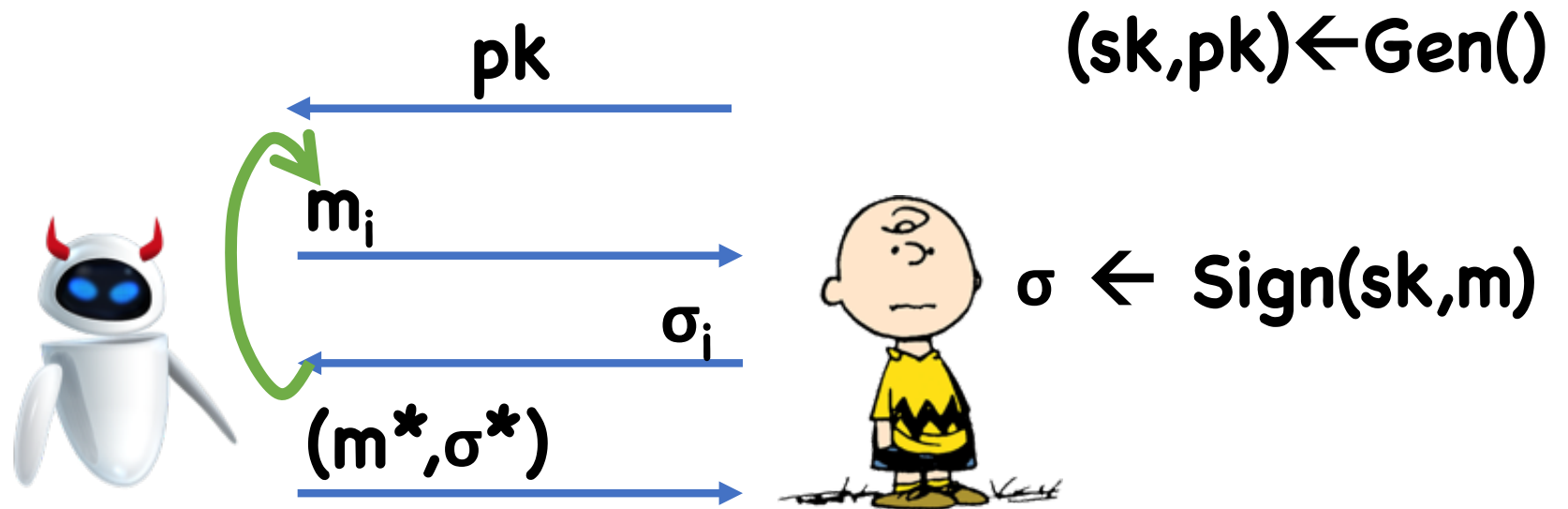
Many-time Signatures



- Output 1 iff:
- $m^* \notin \{m_1, \dots\}$
 - $\text{Ver}(pk, m^*, \sigma^*) = 1$

$$\text{CMA-Adv}(\text{devil robot}) = \Pr[\text{Charlie outputs 1}]$$

Strong Security



- Output 1 iff:
- $(m^*, \sigma^*) \notin \{(m_1, \sigma_1) \dots\}$
 - $\text{Ver}(pk, m^*, \sigma^*) = 1$

$$\text{CMA-Adv}(\text{robot}) = \Pr[\text{Carnegie outputs 1}]$$

Building Digital Signatures

Non-trivial to construct with provable security

Most efficient constructions have heuristic security

Signatures from TDPs?

$\text{Gen}_{\text{sig}}() = \text{Gen}()$

$\text{Sign}(\text{sk}, m) = F^{-1}(\text{sk}, m)$

$\text{Ver}(\text{pk}, m, \sigma): F(\text{pk}, \sigma) == m$

Signatures from TDPs

$$\mathbf{Gen}_{\text{sig}}() = \mathbf{Gen}()$$

$$\mathbf{Sign}(\text{sk}, m) = \mathbf{F}^{-1}(\text{sk}, \mathbf{H}(m))$$

$$\mathbf{Ver}(\text{pk}, m, \sigma): \mathbf{F}(\text{pk}, \sigma) == \mathbf{H}(m)$$

Theorem: If $(\mathbf{Gen}, \mathbf{F}, \mathbf{F}^{-1})$ is a secure TDP, and \mathbf{H} is “modeled as a random oracle”, then $(\mathbf{Gen}_{\text{sig}}, \mathbf{Sign}, \mathbf{Ver})$ is (strongly) CMA-secure

Basic Rabin Signatures

Gen_{sig}(\cdot): let p, q be random large primes
 $sk = (p, q), pk = N = pq$

Sign(sk, m): Solve equation $\sigma^2 = H(m) \pmod N$
using factors p, q

- Output σ

Ver(pk, m, σ): $\sigma^2 \pmod N == H(m)$

Problems

H(m) might not be a quadratic residue

Can only sign roughly $\frac{1}{4}$ of messages

Suppose adversary makes multiple signing queries on the same message

- Receives $\sigma_1, \sigma_2, \dots$ such that $\sigma_i^2 \bmod N = H(m)$
- After enough tries, may get all 4 roots of **H(m)**
- Suppose $\sigma_1 \neq \pm \sigma_2 \bmod N$
- Then **GCD**($\sigma_1 - \sigma_2, N$) will give a factor

One Solution

Gen_{sig}(\cdot): let p, q be primes, a, b, c s.t.

- a is a non-residue **mod** p and q ,
- b is a residue **mod** p but not q ,
- c is a residue **mod** q but not p

$$\mathbf{sk} = (p, q, a, b, c), \mathbf{pk} = (N = pq, a, b, c)$$

Sign(sk, m):

- Solve equation $\sigma^2 \in \{1, a, b, c\} \times H(m) \bmod N$
- Output σ

Ver(pk, m, σ): $\sigma^2 \bmod N \in \{1, a, b, c\} \times H(m)$

One Solution

Exactly one of $\{1, a, b, c\} \times H(m)$ is a residue **mod N**

\Rightarrow Solution guaranteed to be found

Still have problem that multiple queries on same message will give different roots

One Solution

Possibilities:

- Have signer remember all messages signed
- Choose root that is itself a quadratic residue
(if **-1** is not a residue mod **p,q**,
there will be exactly one)

Another Solution

Gen_{sig}(\cdot): let p, q be random large primes
sk = (p, q), pk = N = pq

Sign(sk, m): Repeat until successful:

- Choose random $u \leftarrow \{0, 1\}^\lambda$
- Solve equation $\sigma^2 = H(m, u) \bmod N$
- Output (u, σ)

Ver(pk, m, (u, σ)): $\sigma^2 \bmod N == H(m, u)$

Another Solution

In expectation, after 4 tries will have success

(Whp) Only ever get a single root of a given $\mathbf{H(m,u)}$

Theorem: If factoring is hard and \mathbf{H} is modeled as a random oracle, then Rabin signatures are (weakly) CMA secure

Another Solution

Sign(sk,m): Repeat until successful:

- Choose random $\mathbf{u} \leftarrow \{0,1\}^\lambda$
- Solve equation $\sigma^2 = \mathbf{H}(m,\mathbf{u}) \bmod \mathbf{N}$ using factors \mathbf{p}, \mathbf{q} , where $\sigma < (\mathbf{N}-1)/2$
- Output (\mathbf{u}, σ)

Ver(pk,m,(u,σ)): $\sigma^2 \bmod \mathbf{N} == \mathbf{H}(m,\mathbf{u}) \wedge \sigma < (\mathbf{N}-1)/2$

Theorem: If factoring is hard and \mathbf{H} is modeled as a random oracle, then Rabin signatures are strongly CMA secure

Schnorr Signatures

$$\text{sk} = w$$

$$\text{pk} = h := g^w$$

Sign(sk,m):

- $r \leftarrow \mathbb{Z}_p$
- $a \leftarrow g^r$
- $b \leftarrow H(m,a)$
- $c \leftarrow r + wb$
- Output (a,c)

Ver(h,m,(a,c)):

$$b \leftarrow H(m,a)$$

$$a \times h^b == g^c?$$

Theorem: If Dlog is hard and H is modeled as a random oracle, then Schnorr signatures are strongly CMA secure

What's the Smallest Signature?

RSA Hash-and-Sign: 2 kilobits

ECDSA (variant of Schnorr using “elliptic curves”):
around 512 bits

BLS: 256 bits

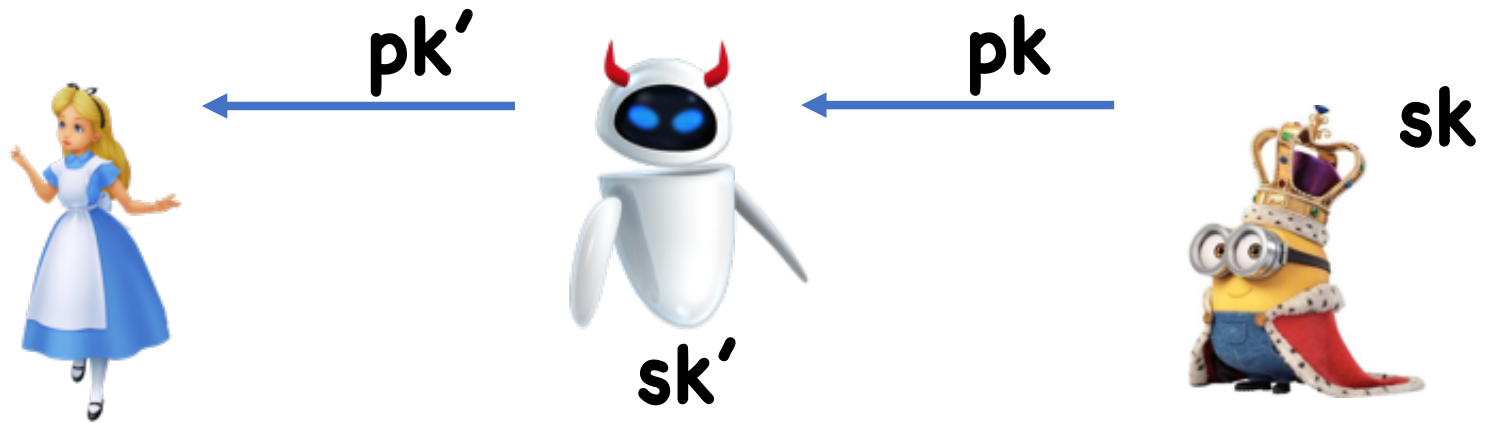
Are 128-bit signatures possible?

- No fundamental reason for impossibility, but all (practical) schemes require 256 bits or more

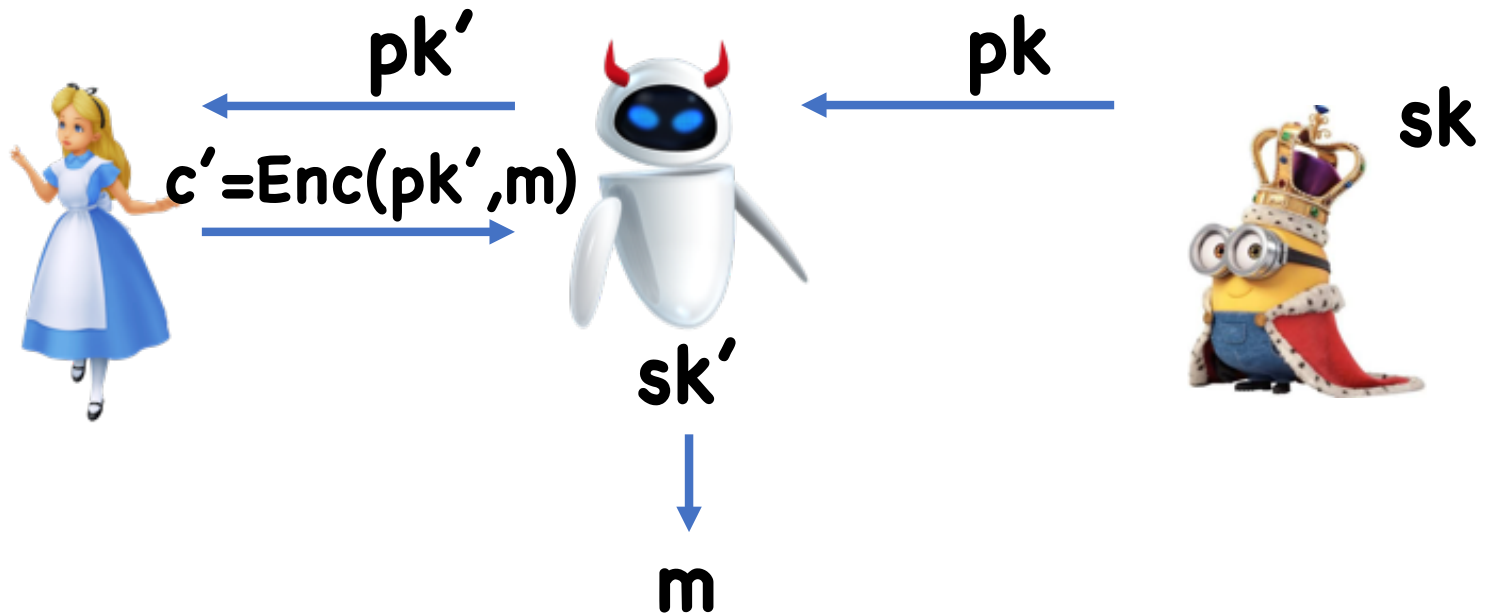
Digital Signatures and the Public Key Infrastructure



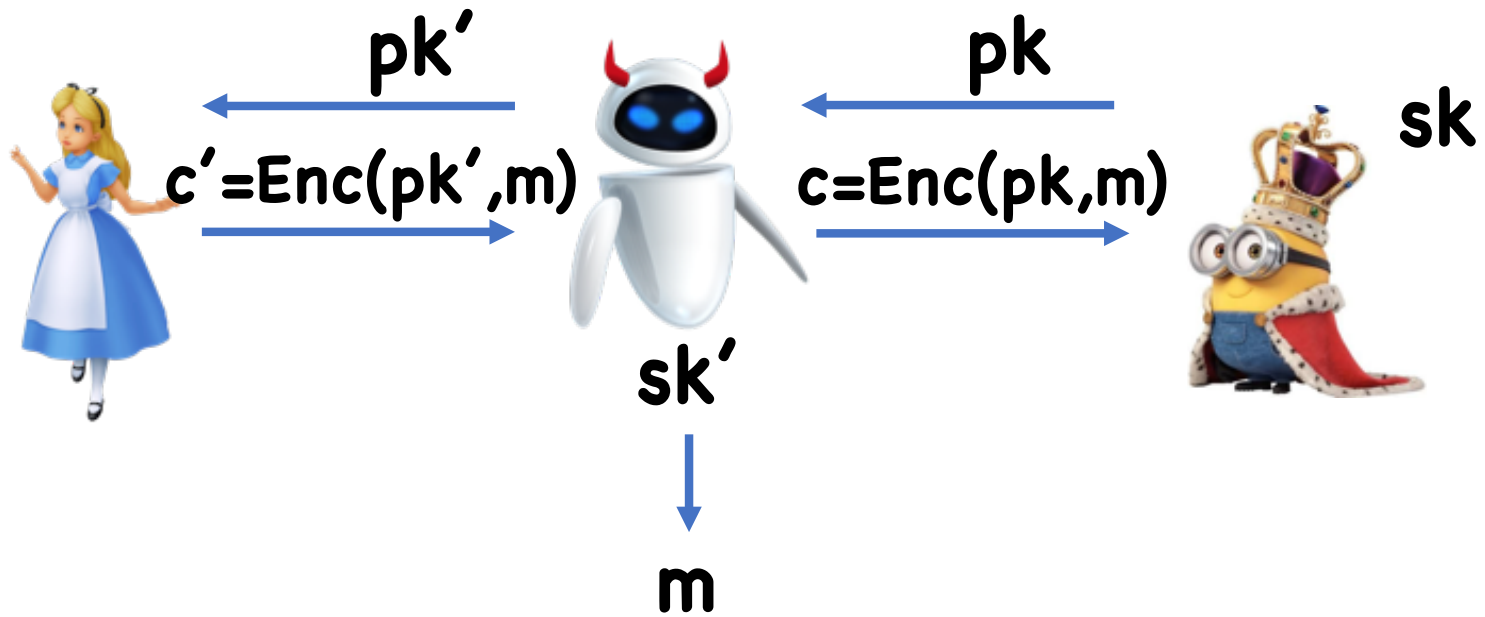
Digital Signatures and the Public Key Infrastructure



Digital Signatures and the Public Key Infrastructure



Digital Signatures and the Public Key Infrastructure

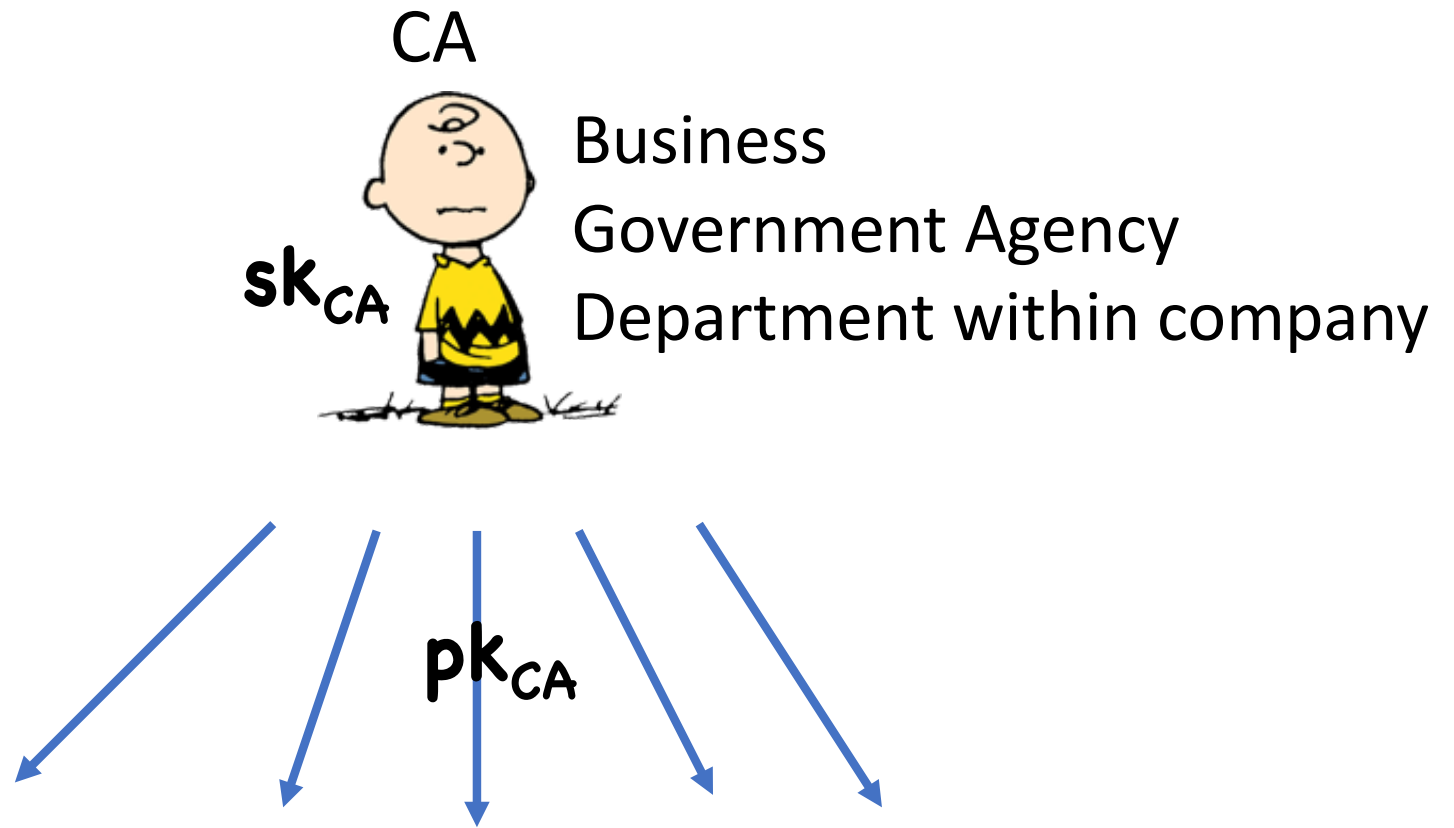


Takeaway

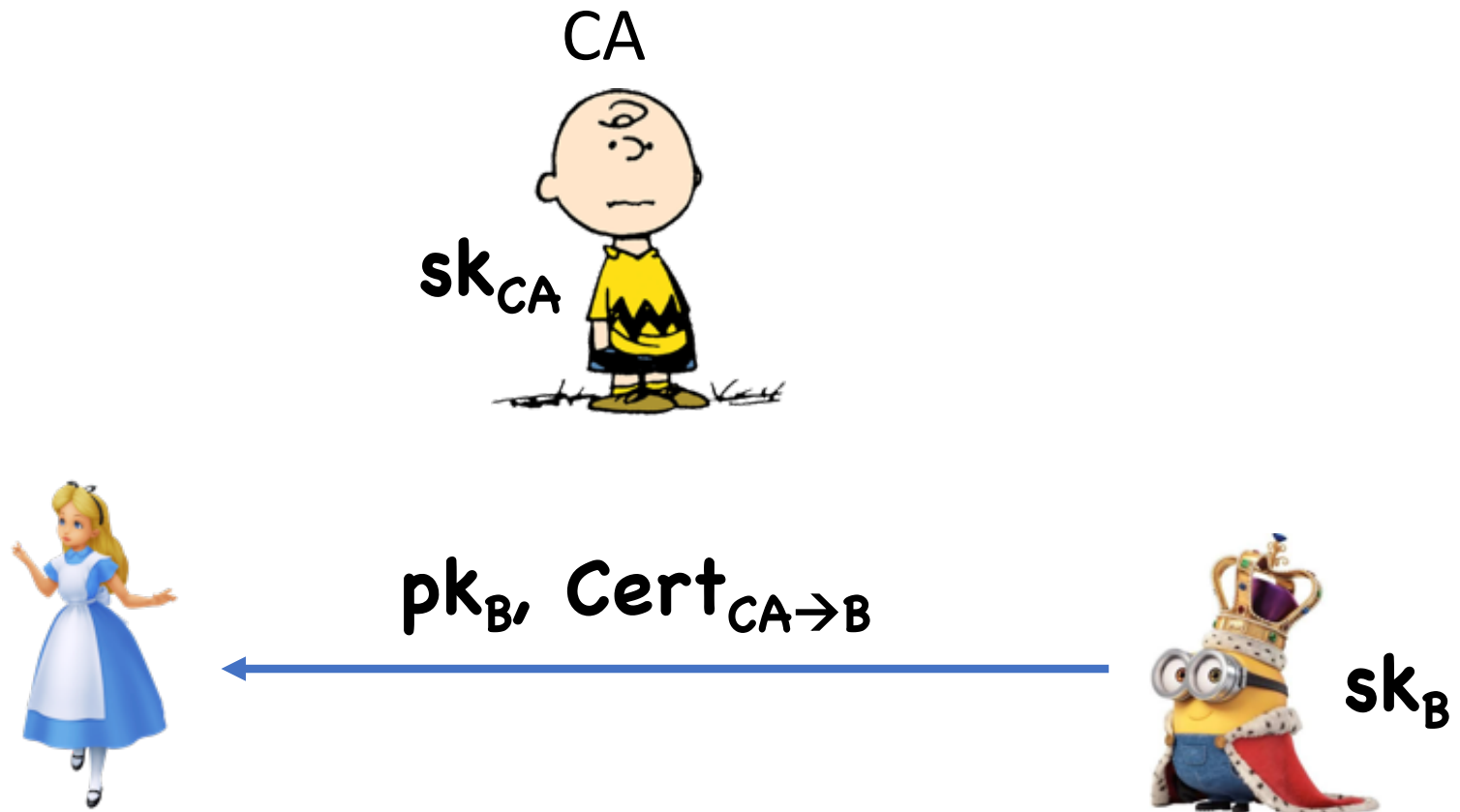
Need some authenticated channel to ensure distribution of public keys

But how to authenticate channel in the first place without being able to distribute public keys?

Solution: Certificate Authorities



Solution: Certificate Authorities



$$\text{Cert}_{CA \rightarrow B} = \text{Sign}(\text{sk}_{CA}, \text{"Bob's public key is } \text{pk}_B \text{"})$$

Solution: Certificate Authorities

Bob is typically some website

- Obtains **Cert** by, say, sending someone in person to CA with **pk_B**
- Only needs to be done once

If Alice trusts CA, then Alice will be convinced that **pk_B** belongs to Bob

Alice typically gets **pk_{CA}** bundled in browser

Limitations

Everyone must trust same CA

- May have different standards for issuing certs

Single point of failure: if sk_{CA} is compromised, whole system is compromised

Single CA must handle all verification

Multiple CAs

There are actually many CA's, CA_1, CA_2, \dots

Bob obtains cert from all of them, sends all the certs with his public key

As long as Alice trusts one of the CA's, she will be convinced about Bob's public key

Certificate Chaining

CA issues **Cert**_{CA→B} for Bob

Bob can now use his signing key to issue **Cert**_{B→D} to Donald

Donald can now prove his public key by sending
(Cert_{CA→B}, **Cert**_{B→D})

- Proves that CA authenticated Bob, and Bob authenticated Donald

Certificate Chaining

For Bob to issue his own certificates, a standard cert should be insufficient

- CA knows who Bob is, but does not trust him to issue certs on its behalf

Therefore, Bob should have a stronger cert:

$\text{Cert}_{CA \rightarrow B} = \text{Sign}(\text{sk}_{CA}, \text{"Bob's public key is } \mathbf{pk}_B \text{ and he can issue certificates on behalf of CA"})$

Certificate Chaining

One root CA

Many second level CAs CA_1, CA_2, \dots

- Each has **Cert**_{CA→CA_i}

Advantage: eases burden on root

Disadvantage: now multiple points of failure

Invalidating Certificates

Sometimes, need to invalidate certificates

- Private key stolen
- User leaves company
- Etc

Options:

- Expiration
- Explicit revocation

Announcements

PR2 Due April 19th

HW6 Due April 23rd

Crypto from Minimal Assumptions

Many ways to build crypto

We've seen many ways to build crypto

- SPN networks
- LFSR's
- Discrete Log
- Factoring

Questions:

- Can common techniques be abstracted out as theorem statements?
- Can every technique be used to build every application?

One-way Functions


The minimal assumption for crypto


Syntax:

- Domain \mathbf{D}
- Range \mathbf{R}
- Function $\mathbf{F: D \rightarrow R}$

No correctness properties other than deterministic

Security?

Definition: F is (t, ϵ) -One-Way if, for all  running in time at most t ,


$$\Pr[x \leftarrow \text{}(F(x)), x \leftarrow D] < \epsilon$$

Trivial example:

$F(x)$ = parity of x

Given $F(x)$, impossible to predict x

Security

Definition: F is (t, ϵ) -One-Way if, for all  running in time at most t ,

$$\Pr[F(x) = F(y) : y \leftarrow \text{img alt="cartoon character" data-bbox="468 438 492 522"} (F(x)), x \leftarrow D] < \epsilon$$

Examples

Any PRG

Any Collision Resistant Hash Function (with sufficient compression)

$$F(p,q) = pq$$

$$F(g,a) = (g, g^a)$$

$$F(N,x) = (N, x^3 \bmod N) \text{ or } F(N,x) = (N, x^2 \bmod N)$$

What's Known

