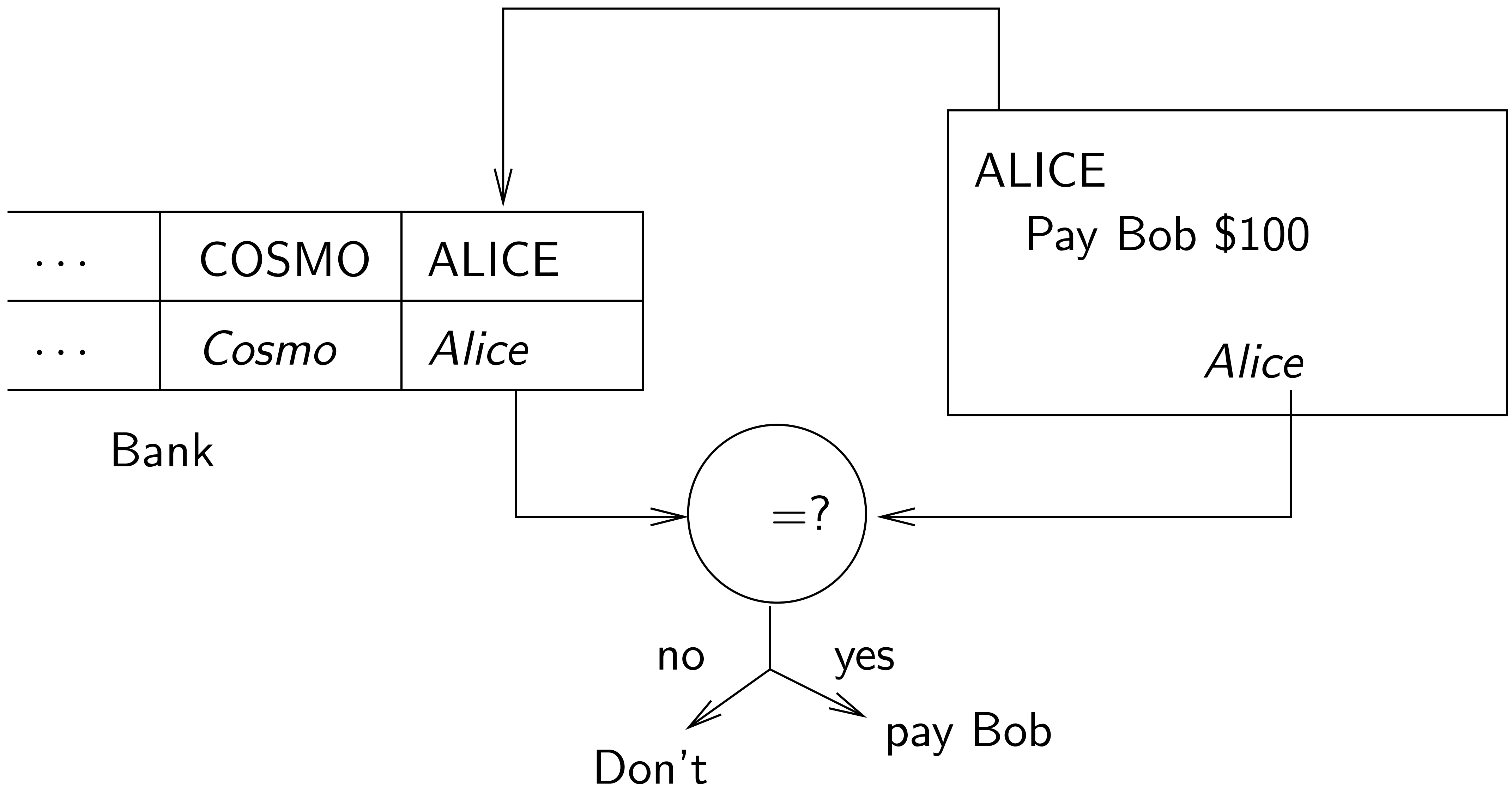
Digital Signatures

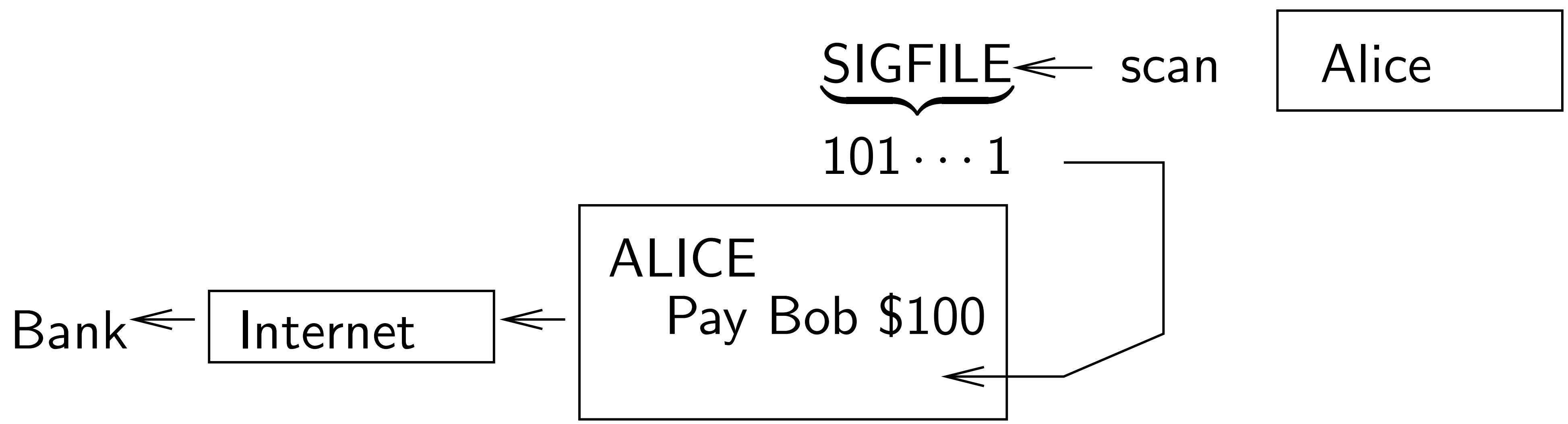
Adam O'Neill

based on http://cseweb.ucsd.edu/~mihir/cse207/

Signing by hand



Signing electronically



Problem: signature is easily copied

Inference: signature must be a function of the message that only Alice can compute

Digital signatures

- A digital signature will have the following attributes:

Digital signatures

- A digital signature will have the following attributes:
  - Even the bank cannot forge

Digital signatures

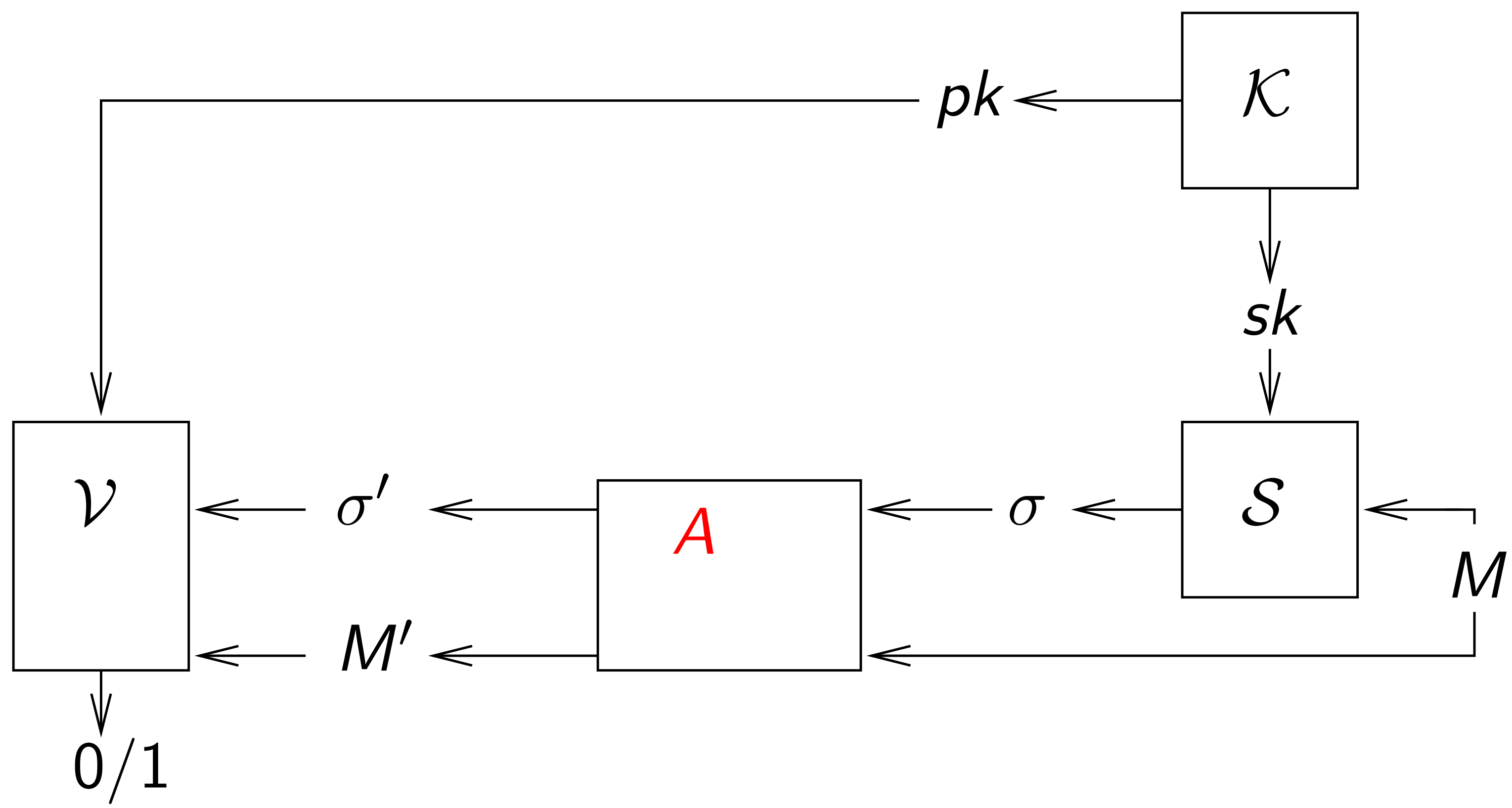
- A digital signature will have the following attributes:
  - Even the bank cannot forge
  - Verifier does not need to share a key with signer or have any secrets at all

Digital signatures

- A digital signature will have the following attributes:
  - Even the bank cannot forge
  - Verifier does not need to share a key with signer or have any secrets at all
  - Public keys of signers distributed as for encryption

Digital signatures

A digital signature scheme $\mathcal{DS} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ is a triple of algorithms where



Correctness: $\mathcal{V}(pk, M, \mathcal{S}(sk, M)) = 1$ with probability one for all M.

Usage

Step 1: key generation

Alice lets $(pk, sk) \xleftarrow{\$} \mathcal{K}$ and stores sk (securely).

Step 2: pk dissemination

Alice enables any potential verifier to get pk.

Step 3: sign

Alice can generate a signature σ of a document M using sk.

Step 4: verify

Anyone holding pk can verify that σ is Alice's signature on M.

Security of a DS Scheme

Possible adversary goals

- find *sk*
- Forge

Possible adversary abilities

- can get *pk*
- known message attack
- chosen message attack

Security of a DS Scheme

Interpretation: adversary cannot get a verifier to accept σ as Alice's signature of M unless Alice has really previously signed M, even if adversary can obtain Alice's signatures on messages of the adversary's choice.

As with MA schemes, the definition does not require security against replay. That is handled on top, via counters or time stamps.

UF-CMA

Let $\mathcal{DS} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ be a signature scheme and A an adversary.

Game UF-CMA$_{\mathcal{DS}}$

procedure Initialize
$(pk, sk) \xleftarrow{\$} \mathcal{K}$; $S \leftarrow \emptyset$
return pk

procedure Finalize(M, σ)
$d \leftarrow \mathcal{V}(pk, M, \sigma)$
return $(d = 1 \wedge M \notin S)$

procedure Sign(M):
$\sigma \xleftarrow{\$} \mathcal{S}(sk, M)$
$S \leftarrow S \cup \{M\}$
return σ

The uf-cma advantage of A is

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{DS}}(A) = \Pr\left[\text{UF-CMA}^{A}_{\mathcal{DS}} \Rightarrow \text{true}\right]$$

Strong unforgeability

Adversary can't get receiver to accept σ as Alice's signatre on M unless

- UF: Alice previously signed M
- SUF: Alice previously signed M and produced signature σ

Adversary wins if it gets receiver to accept σ as Alice's signature on M and

- UF: Alice did not previously sign M
- SUF: Alice may have previously signed M but the signature(s) produced were different from σ

SUF-CMA

Let $\mathcal{DS} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ be a signature scheme and A an adversary.

Game SUF-CMA$_{\mathcal{DS}}$

procedure Initialize: $(pk, sk) \xleftarrow{\$} \mathcal{K}$; $S \leftarrow \emptyset$ **return** pk **procedure Finalize**(M, σ): **return** $\mathcal{V}(pk, M, \sigma) = 1$ and $(M, \sigma) \notin S$	**procedure Sign**(M): $\sigma \xleftarrow{\$} \mathcal{S}(sk, M)$ $S \leftarrow S \cup \{(M, \sigma)\}$ **return** σ

The suf-cma advantage of A is

$$\mathbf{Adv}^{\text{suf-cma}}_{\mathcal{DS}}(A) = \Pr\left[\text{SUF-CMA}^{A}_{\mathcal{DS}} \Rightarrow \text{true}\right]$$

RSA signatures

Fix an RSA generator $\mathcal{K}_{rsa}$ and let the key generation algorithm be

Alg $\mathcal{K}$
$(N, p, q, e, d) \xleftarrow{\$} \mathcal{K}_{rsa}$
$pk \leftarrow (N, e)$; $sk \leftarrow (N, d)$
return pk, sk

We will use these keys in all our RSA-based schemes and only describe signing and verifying.

Idea

Signer $pk = (N, e)$ and $sk = (N, d)$

Let f, f^{-1}: $\mathbb{Z}_N^* \to \mathbb{Z}_N^*$ be the RSA function (encryption) and inverse (decryption) defined by

$$f(x) = x^e \bmod N \quad \text{and} \quad f^{-1}(y) = y^d \bmod N\,.$$

Sign by "decrypting" the message y:

$$x = \mathcal{S}_{N,d}(y) = f^{-1}(y) = y^d \bmod N$$

Verify by "encrypting" signature x:

$$\mathcal{V}_{N,e}(x) = 1 \text{ iff } f(x) = y \text{ iff } x^e \equiv y \bmod N\,.$$

Plain RSA signatures

Signer $pk = (N, e)$ and $sk = (N, d)$

Alg $\mathcal{S}_{N,d}(y)$:	**Alg** $\mathcal{V}_{N,e}(y, x)$:
$x \leftarrow y^d \bmod N$	**if** $x^e \equiv y \pmod{N}$ **then return** 1
return x	**return** 0

Here $y \in \mathbb{Z}_N^*$ is the message and $x \in \mathbb{Z}_N^*$ is the signature.

Attacks

Historical perspective

When Diffie and Hellman introduced public-key cryptography they suggested the DS scheme

$$\begin{aligned}\mathcal{S}(sk, M) &= D(sk, M)\\ \mathcal{V}(pk, M, \sigma) &= 1 \text{ iff } E(pk, \sigma) = M\end{aligned}$$

where (E, D) is a public-key encryption scheme.

But

- This views public-key encryption as deterministic; they really mean trapdoor permutations in our language
- Plain RSA is an example
- It doesn't work!

Nonetheless, many textbooks still view digital signatures this way.

Another issue

In plain RSA, the message is an element of $\mathbb{Z}_N^*$. We really want to be able to sign strings of arbitrary length.

Hash-based RSA sigs

Let H: $\{0,1\}^* \to \mathbb{Z}_N^*$ be a public hash function and let $pk = (N, e)$ and $sk = (N, d)$ be the signer's keys. The hash-then-decrypt scheme is

Alg $\mathcal{S}_{N,d}(M)$:	**Alg** $\mathcal{V}_{N,e}(M, x)$:
$y \leftarrow H(M)$	$y \leftarrow H(M)$
$x \leftarrow y^d \bmod N$	**if** $x^e \equiv y \pmod N$ **then return** 1
return x	**return** 0

Succinctly,

$$\mathcal{S}_{N,d}(M) = H(M)^d \bmod N$$

Different choices of H give rise to different schemes.

Hash function requirements

Suppose an adversary can find a collision for H, meaning distinct M_1, M_2 with $H(M_1) = H(M_2)$.

Then

$$H(M_1)^d \equiv H(M_2)^d \pmod{N}$$

meaning M_1, M_2 have the same signature.

So forgery is easy:

- Obtain from signing oracle the signature $x_1 = H(M_1)^d \bmod N$ of M_1
- Output M_2 and its signature x_1

Conclusion: H needs to be collision-resistant

Preventing previous attacks

For plain RSA

- 1 is a signature of 1
- $\mathcal{S}_{N,d}(y_1 y_2) = \mathcal{S}_{N,d}(y_1) \cdot \mathcal{S}_{N,d}(y_2)$

But with hash-then-decrypt RSA

- $H(1)^d \not\equiv 1$ so 1 is not a signature of 1
- $\mathcal{S}_{N,d}(M_1 M_2) = H(M_1 M_2)^d \not\equiv H(M_1)^d \cdot H(M_2)^d \pmod{N}$

A "good" choice of H prevents known attacks.

RSA PKCS #1 sigs

Signer has $pk = (N, e)$ and $sk = (N, d)$ where $|N| = 1024$. Let $h\colon \{0,1\}^* \to \{0,1\}^{160}$ be a hash function (like SHA-1) and let $n = |N|_8 = 1024/8 = 128$.

Then

$$H_{PKCS}(M) = 00||01||\underbrace{FF||\ldots||FF}_{n-22}||\underbrace{h(M)}_{20}$$

And

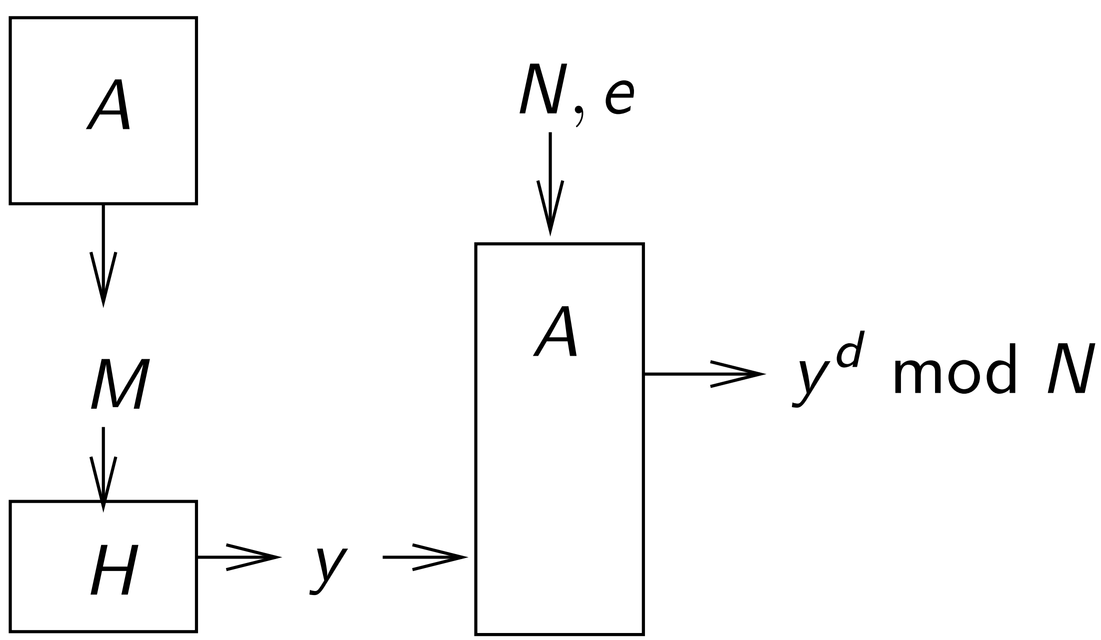
$$\mathcal{S}_{N,d}(M) = H_{PKCS}(M)^d \bmod N$$

Then

- H_{PKCS} is CR as long as h is CR
- $H_{PKCS}(1) \not\equiv 1 \pmod{N}$
- $H_{PKCS}(y_1 y_2) \not\equiv H_{PKCS}(y_1) \cdot H_{PKCS}(y_2) \pmod{N}$

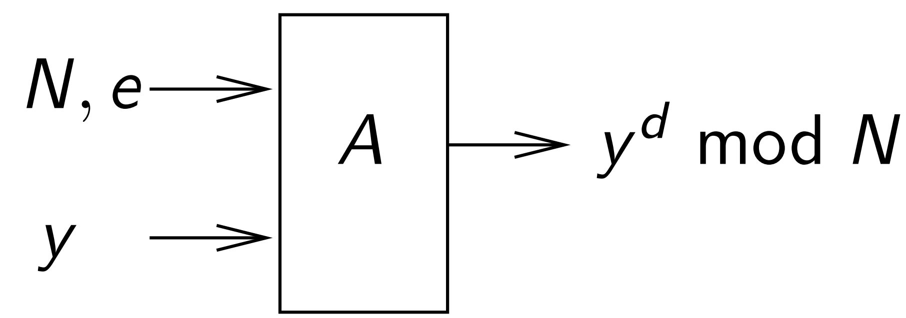
Security

Forger's goal



y here need not be random

Inverter's goal



y here is random

The Problem

Recall

$$H_{PKCS}(M) = 00||01||FF|| \ldots ||FF||h(M)$$

But first $n - 20 = 108$ bytes out of n are fixed so $H_{PKCS}(M)$ does not look "random" even if h is a RO or perfect.

We cannot hope to show RSA PKCS#1 signatures are secure assuming (only) that RSA is 1-wayno matter what we assume about h and even if h is a random oracle.

Goal

We will validate the hash-then-decrypt paradigm

$$\mathcal{S}_{N,d}(M) = H(M)^d \bmod N$$

by showing the signature scheme is provably UF-CMA assuming RSA is 1-way as long as H is a RO.

This says the paradigm has no "structural weaknesses" and we should be able to get security with "good" choices of H.

Choice of hash

A "good" choice of H might be something like

$$\begin{aligned} H(M) \quad &= \quad \text{first } n \text{ bytes of} \\ & \quad\;\; \mathrm{SHA1}(1 \,||\, M) \,||\, \mathrm{SHA1}(2 \,||\, M) \,||\, \cdots \,||\, \mathrm{SHA1}(11 \,||\, M) \end{aligned}$$

Full-Domain Hash

Signer $pk = (N, e)$ and $sk = (N, d)$

algorithm $\mathcal{S}^{H}_{N,d}(M)$	algorithm $\mathcal{V}^{H}_{N,e}(M, x)$
return $H(M)^d \bmod N$	**if** $x^e \equiv H(M) \pmod N$ **then return** 1
	else return 0

Here H: $\{0,1\}^* \to \mathbb{Z}^*_N$ is a random oracle.

UF-CMA in the RO Model

Let $\mathcal{DS} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ be a signature scheme and A an adversary.

Game UF-CMA$_{\mathcal{DS}}$

procedure Initialize: $(pk, sk) \xleftarrow{\$} \mathcal{K}$; $S \leftarrow \emptyset$ **return** pk **procedure Finalize**(M, σ): **return** $\mathcal{V}^H(pk, M, \sigma) = 1$ and $M \notin S$	**procedure Sign**(M): $\sigma \xleftarrow{\$} \mathcal{S}^H(sk, M)$ $S \leftarrow S \cup \{M\}$ **return** σ **procedure** $H(M)$: **if** $H[M] = \perp$ **then** $H[M] \xleftarrow{\$} \mathcal{R}$ **return** $H[M]$

Here $\mathcal{R}$ is the range of H.

Security of FDH

Theorem: [BR96] Let $\mathcal{K}_{rsa}$ be a RSA generator and $\mathcal{DS} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ the associated FDH RO-model signature scheme. Let A be a uf-cma adversary making q_s signing queries and q_H queries to the RO H and having running time at most t. Then there is an inverter I such that

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{DS}}(A) \leq (q_s + q_H + 1) \cdot \mathbf{Adv}^{\text{owf}}_{\mathcal{K}_{rsa}}(I)\,.$$

Furthermore the running time of I is that of A plus the time for $\mathcal{O}(q_s + q_H + 1)$ computations of the RSA function.

RO proofs for encryption

There is a "crucial" hash query Q such that

- If A does not query Q it has 0 advantage
- If A queries Q an overlying algorithm can "see" it and solve some presumed hard computational problem

Example: In the RO EG KEM, $Q = g^{xy}$ where $pk = g^{x}$ and g^{y} is in challenge ciphertext.

Programming the RO

For signatures we exploit the RO model in a new way by replying to RO queries with carefully constructed objects. In particular the inverter I that on input y aims to compute y^d mod N might reply to a RO query M made by A via

- y or some function thereof
- x^e mod N for $x \xleftarrow{\$} \mathbb{Z}_N^*$ chosen by I

Thus I is "programming" $H(M)$ to equal values of I's choice.

Inverter

adversary $I(N, e, y)$

$g \xleftarrow{\$} \{1, \ldots, q_H\}$; $j \leftarrow 0$
$(M, \sigma) \xleftarrow{\$} A^{\mathsf{HSim},\mathsf{SignSim}}(N, e)$
Return σ

subroutine SignSim(M)
$j \leftarrow \mathtt{Ind}(M)$
Return x_j

subroutine HSim(M)
$j \leftarrow j + 1$; $M_j \leftarrow M$; $\mathtt{Ind}(M) \leftarrow j$
If $j = g$ then $H[M] \leftarrow y$; $x_j \leftarrow \perp$
else $x_j \xleftarrow{\$} \mathbb{Z}_N^*$; $H[M] \leftarrow x_j^e \bmod N$
Return $H[M]$

Games G_0, $\boxed{G_1}$

procedure Initialize
$(N, p, q, e, d) \xleftarrow{\$} \mathcal{K}_{rsa}$
$g \xleftarrow{\$} \{1, \ldots, q_H\}$; $j \leftarrow 0$; $y \xleftarrow{\$} \mathbb{Z}_N^*$
return (N, e)

procedure Sign(M)
$j \leftarrow \mathtt{Ind}(M)$
if $j = g$ then $\boxed{x_j \leftarrow y^d \bmod N}$
return x_j

procedure $H(M)$
$j \leftarrow j + 1$; $M_j \leftarrow M$; $\mathtt{Ind}(M) \leftarrow j$
if $j = g$ **then** $H[M] \leftarrow y$; $x_j \leftarrow \perp$
else $x_j \xleftarrow{\$} \mathbb{Z}_N^*$; $H[M] \leftarrow x_j^e \bmod N$
return $H[M]$

procedure Finalize(M, σ)
$j \leftarrow \mathtt{Ind}(M)$
if $j \neq g$ **then** bad $\leftarrow$ true
return $(\sigma^e \equiv H[M] \pmod{N})$

Let Bad_i be the event that G_i sets bad. Then

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{owf}}_{\mathcal{K}_{rsa}}(I) &\geq \Pr\left[G_0^A \Rightarrow \mathsf{true} \wedge \overline{\mathsf{Bad}_0}\right] \\
&= \Pr\left[G_1^A \Rightarrow \mathsf{true} \wedge \overline{\mathsf{Bad}_1}\right]
\end{aligned}$$

where last line is due to Fundamental Lemma variant. But the events "$G_1^A \Rightarrow$ true and "$\overline{\mathsf{Bad}_1}$" are independent so

$$\begin{aligned}
&= \Pr\left[G_1^A \Rightarrow \mathsf{true}\right] \cdot \Pr\left[\overline{\mathsf{Bad}_1}\right] \\
&= \mathbf{Adv}^{\mathrm{uf\text{-}cma}}_{\mathcal{DS}}(A) \cdot \frac{1}{q}
\end{aligned}$$

Choosing a modulus size

Say we want 80-bits of security, meaning a time t attacker should have advantage at most $t \cdot 2^{-80}$. The following shows modulus size k and cost c of one exponentiation, with $q_H = 2^{60}$ and $q_s = 2^{45}$ in the FDH case:

Task	k	c
Inverting RSA	1024	1
Breaking FDH as per [BR96] reduction	3700	47

This (for simplicity) neglects the running time difference between A, I.

This motivates getting tighter reductions for FDH, or alternative schemes with tighter reductions.

Better analysis

Theorem: [Co00] Let $\mathcal{K}_{rsa}$ be a RSA generator and $\mathcal{DS} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ the associated FDH RO-model signature scheme. Let A be a uf-cma adversary making q_s signing queries and q_H queries to the RO H and having running time at most t. Then there is an inverter I such that

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{DS}}(A) \leq \mathcal{O}(q_s) \cdot \mathbf{Adv}^{\text{owf}}_{\mathcal{K}_{rsa}}(I)\,.$$

Furthermore the running time of I is that of A plus the time for $\mathcal{O}(q_s + q_H + 1)$ computations of the RSA function.

Proof Idea

Choosing a modulus size

Say we want 80-bits of security, meaning a time t attacker should have advantage at most $t \cdot 2^{-80}$. The following shows modulus size k and cost c of one exponentiation, with $q_H = 2^{60}$ and $q_s = 2^{45}$ in the FDH case:

Task	k	c
Inverting RSA	1024	1
Breaking FDH as per [BR96] reduction	3700	47
Breaking FDH as per [Co00] reduction	2800	21

RSA-PSS

Signer $pk = (N, e)$ and $sk = (N, d)$

algorithm $\mathcal{S}_{N,d}^{h,g_1,g_2}(M)$	algorithm $\mathcal{V}_{N,e}^{h,g_1,g_2}(M, x)$
$r \xleftarrow{\$} \{0,1\}^{160}$	$y \leftarrow x^e \bmod N$
$w \leftarrow h(M \,\|\, r)$	$b \,\|\, w \,\|\, r^* \,\|\, P \leftarrow y$
$r^* \leftarrow g_1(w) \oplus r$	$r \leftarrow r^* \oplus g_1(w)$
$y \leftarrow 0 \,\|\, w \,\|\, r^* \,\|\, g_2(w)$	if $(g_2(w) \neq P)$ then return 0
return $y^d \bmod N$	if $(b = 1)$ then return 0
	if $(h(M \,\|\, r) \neq w)$ then return 0
	return 1

Here $h, g_1\colon \{0,1\}^* \to \{0,1\}^{160}$ and $g_2\colon \{0,1\}^* \to \{0,1\}^{k-321}$ are random oracles where $k = |N|$.

Choosing a modulus size

Say we want 80-bits of security, meaning a time t attacker should have advantage at most $t \cdot 2^{-80}$. The following shows modulus size k and cost c of one exponentiation, with $q_H = 2^{60}$ and $q_s = 2^{45}$ in the FDH and PSS cases:

Task	k	c
Inverting RSA	1024	1
Breaking FDH as per [BR96] reduction	3700	47
Breaking FDH as per [Co00] reduction	2800	21
Breaking PSS as per [BR96] reduction	1024	1

Choosing a modulus size

There are no attacks showing that FDH is less secure than RSA, meaning there are no attacks indicating FDH with a 1024 bit modulus has less than 80 bits of security. But to get the provable guarantees we must use larger modulii as shown, or use PSS.

ElGamal sigs

Let $G = \mathbf{Z}_p^* = \langle g \rangle$ where p is prime.

Signer keys: $pk = X = g^x \in \mathbf{Z}_p^*$ and $sk = x \xleftarrow{\$} \mathbf{Z}_{p-1}$

Algorithm $\mathcal{S}_x(m)$
$k \xleftarrow{\$} \mathbf{Z}_{p-1}^*$
$r \leftarrow g^k \mod p$
$s \leftarrow (m - xr) \cdot k^{-1} \mod (p-1)$
return (r, s)

Algorithm $\mathcal{V}_X(m, (r, s))$
if $(r \notin G$ or $s \notin \mathbf{Z}_{p-1})$
    then return 0
if $(X^r \cdot r^s \equiv g^m \mod p)$
    then return 1
else return 0

Correctness check: If $(r, s) \xleftarrow{\$} \mathcal{S}_x(m)$ then

$$X^r \cdot r^s = g^{xr} g^{ks} = g^{xr+ks} = g^{xr+k(m-xr)k^{-1} \mod (p-1)} = g^{xr+m-xr} = g^m$$

so $\mathcal{V}_X(m, (r, s)) = 1$.

Security

Signer keys: $pk = X = g^x \in \mathbf{Z}_p^*$ and $sk = x \xleftarrow{\$} \mathbf{Z}_{p-1}$

Algorithm $\mathcal{S}_x(m)$
$k \xleftarrow{\$} \mathbf{Z}_{p-1}^*$
$r \leftarrow g^k \mod p$
$s \leftarrow (m - xr) \cdot k^{-1} \mod (p-1)$
return (r, s)
	Algorithm $\mathcal{V}_X(m, (r, s))$
	if $(r \notin G$ or $s \notin \mathbf{Z}_{p-1})$
	    then return 0
	if $(X^r \cdot r^s \equiv g^m \mod p)$
	    then return 1
	else return 0

Suppose given $X = g^x$ and m the adversary wants to compute r, s so that $X^r \cdot r^s \equiv g^m \mod p$. It could:

- Pick r and try to solve for $s = \mathrm{DLog}_{\mathbf{Z}_p^*, r}(g^m X^{-r})$
- Pick s and try to solve for r ...?

Forgery!

Adversary has better luck if it picks m itself:

Adversary $A(X)$
$r \leftarrow gX \mod p$; $s \leftarrow (-r) \mod (p-1)$; $m \leftarrow s$
return $(m, (r, s))$

Then:

$$\begin{aligned} X^r \cdot r^s = X^{gX}(gX)^{-gX} = X^{gX} g^{-gX} X^{-gX} = g^{-gX} \\ = g^{-r} = g^m \end{aligned}$$

so (r, s) is a valid forgery on m.

ElGamal with hashing

Let $G = \mathbf{Z}_p^* = \langle g \rangle$ where p is a prime.
Signer keys: $pk = X = g^x \in \mathbf{Z}_p^*$ and $sk = x \xleftarrow{\$} \mathbf{Z}_{p-1}$
$H : \{0,1\}^* \to \mathbf{Z}_{p-1}$ a hash function.

Algorithm $\mathcal{S}_x(M)$
$m \leftarrow H(M)$
$k \xleftarrow{\$} \mathbf{Z}_{p-1}^*$
$r \leftarrow g^k \mod p$
$s \leftarrow (m - xr) \cdot k^{-1} \mod (p-1)$
return (r, s)

Algorithm $\mathcal{V}_X(M, (r, s))$
$m \leftarrow H(M)$
if $(r \notin G$ or $s \notin \mathbf{Z}_{p-1})$
    then return 0
if $(X^r \cdot r^s \equiv g^m \mod p)$
    then return 1
else return 0

DSA

- Fix primes p, q such that q divides $p-1$
- Let $G = \mathbf{Z}_p^* = \langle h \rangle$ and $g = h^{(p-1)/q}$ so that $g \in G$ has order q
- H: $\{0,1\}^* \to \mathbf{Z}_q$ a hash function
- Signer keys: $pk = X = g^x \in \mathbf{Z}_p^*$ and $sk = x \xleftarrow{\$} \mathbf{Z}_q$

Algorithm $\mathcal{S}_x(M)$
$m \leftarrow H(M)$
$k \xleftarrow{\$} \mathbf{Z}_q^*$
$r \leftarrow (g^k \mod p) \mod q$
$s \leftarrow (m + xr) \cdot k^{-1} \mod q$
return (r, s)

Algorithm $\mathcal{V}_X(M, (r, s))$
$m \leftarrow H(M)$
$w \leftarrow s^{-1} \mod q$
$u_1 \leftarrow mw \mod q$
$u_2 \leftarrow rw \mod q$
$v \leftarrow (g^{u_1} X^{u_2} \mod p) \mod q$
if $(v = r)$ **then return** 1
else return 0

Discussion

DSA as shown works only over the group of integers modulo a prime, but there is also a version ECDSA of it for elliptic curve groups.

In ElGamal and DSA/ECDSA, the expensive part of signing, namely the exponentiation, can be done off-line.

No proof that ElGamal or DSA is UF-CMA under a standard assumption (DL, CDH, ...) is known, even if H is a RO. Proofs are known for variants.

Schnorr Signatures

- Let $G = \langle g \rangle$ be a cyclic group of prime order p
- H: $\{0,1\}^* \to \mathbf{Z}_p$ a hash function
- Signer keys: $pk = X = g^x \in G$ and $sk = x \xleftarrow{\$} \mathbf{Z}_p$

Algorithm $\mathcal{S}_x(M)$
$r \xleftarrow{\$} \mathbf{Z}_p$
$R \leftarrow g^r$
$c \leftarrow H(R \| M)$
$a \leftarrow xc + r \bmod p$
return (R, a)

Algorithm $\mathcal{V}_X(M, (R, a))$
if $R \notin G$ then return 0
$c \leftarrow H(R \| M)$
if $g^a = RX^c$ then return 1
else return 0

Discussion

The Schnorr scheme works in an arbitrary (prime-order) group. When implemented in a 160-bit elliptic curve group, it is as efficient as ECDSA. It can be proven UF-CMA in the random oracle model under the discrete log assumption [PS,AABN]. The security reduction, however, is quite loose.