RSA Signatures

First, let's recall textbook RSA and its insecurity. Recall the e-RSA assumption:

1. $(N, d) \leftarrow \text{GenerateRSA}_e(1^\lambda)$
2. Sample $x \xleftarrow{R} \mathbb{Z}_N$
3. $\Pr[A(N, \underbrace{x^e}_{\text{mod } N}) \to x] = \text{negl}(\lambda)$

Textbook RSA signatures:

$\text{Setup}(1^\lambda)$: $(N, d) \leftarrow \text{GenerateRSA}_e(1^\lambda)$

vk: (N, e) sk: d

$\text{Sign}(sk, m)$: $\sigma \leftarrow m^d \pmod N$

$\text{Verify}(vk, m, \sigma)$: check that $\sigma^e = m \pmod N$

(there are also attacks!)

What goes wrong if we try to prove security? Adversary's forgery is on a particular message m and yet our challenge (from the RSA challenger) is a random one – unclear how to embed challenge. Also, can consider following attack:

$$\left.\begin{array}{l} \text{1. Choose random } \sigma \xleftarrow{R} \mathbb{Z}_N \text{ and compute } m = \sigma^e \\ \text{2. Output "forgery" } (m, \sigma). \end{array}\right\} \text{0-query adversary!}$$

How to fix? Hash the message before signing:

$\text{Sign}(sk, m) : \sigma \leftarrow H(m)^d \pmod N$ where $H : \{0,1\}^* \to \mathbb{Z}_N$

$\text{Verify}(vk, m, \sigma)$: Check that $\sigma^e = H(m)$

Can be shown to be secure if H is modeled as a random oracle.

1. Check: above attack does not work. (Need to find pre-image of H).
2. Can embed RSA challenge in security reduction.

Intuition: to forge on m, must query random oracle on $H(m)$

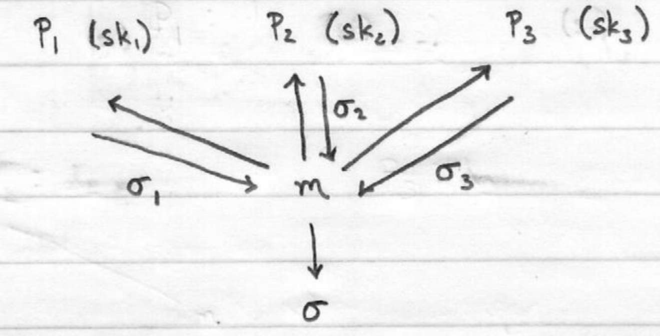
↳ otherwise $H(m)$ is information-theoretically hidden from adversary

↳ embed RSA challenge y as output of $H(m)$ for some m

↳ proper signature on m is then e^{th} root of y

Algebraic Properties of RSA Signatures

threshold signatures : to protect signing key, distribute the secret key across many parties (used by CAs, military - e.g. nuclear launch codes, etc.)



given a subset of the keys, cannot <u>forge</u> signatures

thresholdizing RSA: take signing key $d \in \mathbb{Z}_N$ and split it into random shares $d_1, \ldots, d_n$ such that $d = d_1 + d_2 + \cdots + d_n$

to sign, send $H(m)$ to $P_1, \ldots, P_n$ who each compute

$$\sigma_i \leftarrow H(m)^{d_i}$$

to reconstruct: $\sigma = \prod_{i \in [n]} \sigma_i = H(m)^{\sum_i d_i} = H(m)^d$

Security: each party is effectively computing an RSA signature on $H(m)$ with (secret) signing key d_i — security of RSA signatures translates to security against up to $(n-1)$ corrupt signers

more generally: using Shamir secret sharing, can extend to general "t-out-of-n" access structures (discussed later in the course)

More generally, can consider other forms of threshold cryptography (e.g. threshold encryption)

↳ recent work: universal thresholdizer general approach for thresholdizing cryptographic schemes + post-quantum security

↳ resolves several longstanding open problems!

[BGGK17]

Algebraic Properties of RSA Signatures

blind signatures: digital signature where signer does not what message is being signed

- useful in cryptographic voting (voting authority certifies vote but does not know your vote)

  digital cash schemes (transactions validated but details hidden)

- build from RSA: $vk = (N, e)$

client (m) signer $(sk = d)$

$r \xleftarrow{R} \mathbb{Z}_N$ $\xrightarrow{H(m) \cdot r^e}$

$\xleftarrow{\sigma' = (H(m) \cdot r^e)^d}$

$\sigma = \sigma' \cdot r^{-1} \pmod N$

- correctness: $\sigma' = (H(m) \cdot r^e)^d = H(m)^d \cdot r^{de} = H(m)^d \cdot r \pmod N$ ✓

  $\sigma = \sigma' \cdot r^{-1} = H(m)^d \pmod N$

- message privacy: r is uniform over $\mathbb{Z}_N \Rightarrow$ uniform over $\mathbb{Z}_N^*$

  (statistical) $\Rightarrow r^e$ is uniform over $\mathbb{Z}_N^*$

  $\therefore H(m) \cdot r^e$ <u>statistically</u> hides $H(m)$

- unforgeability: given n signatures, cannot output $n+1$ signatures

  ↳ follows from stronger variant of RSA assumption

Another application of RSA: cryptographic accumulators

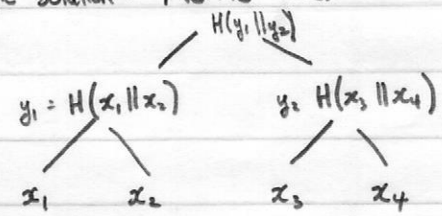
Idea: find a compact representation of a set of values $S = \{x_1, \ldots, x_n\}$

↳ for every value in the accumulator, there is a short witness w that can convince a verifier that the value is indeed in the accumulator

↳ example application (Benaloh and De Mare): club administrator give credential to each member and adds them to an accumulator – club members can prove their membership by providing name and witness

↳ more recent application (Zerocoin): bulletin board (e.g. block chain) containing cryptographic commitments (coins) – to spend a coin, users would provide a proof that their coin is in the accumulator (in particular, need ZK proof of serial number of the coin)

Simple solution : Merkle trees



root node is accumulator value: $O(\lambda)$ bits

witness that x_i is in accumulator : sibling nodes along path

$\hookrightarrow$ witness grows $O(\log |S|)$

Introduce strong RSA first

Accumulators from (strong) RSA [Baric-Pfitzmann] witness size almost independent of set size!

Note: $x_1, \dots, x_n$ are primes

Setup(1^λ) : $(N, x) \leftarrow$ GenerateStrongRSA(1^λ) $\quad$ pp : (N, x)

Accumulate $(pp,\ S = \{y_1, \dots, y_n\}) \rightarrow$ output $g^{\prod_{i \in [n]} y_i} \pmod N$

Witness (pp, S, y_j) : output $g^{\prod_{i \neq j} y_i} \pmod N$

Verify (pp, A, y, w) : check $w^y \stackrel{?}{=} A \pmod N$

Secure if no efficient adversary can convince verifier that non-member is in the set

Given pp, no efficient adversary can produce set S, $y \notin S$, w such that Verify$(pp, A, y, w) = 1$

Proof. Given a strong-RSA challenge (N, x), run adversary on (N, x).

Adversary outputs set $S = \{y_1, \dots, y_n\}$, $y^* \notin S$, and w^* such that

Verify$(pp, A, y^*, w^*) = 1$. This means that

common trick to extract an e^{th} root

$$A = x^{\prod_{i \in [n]} y_i} = (w^*)^{y^*} \pmod N$$

Let $e = y^*$ and $r = \prod_{i \in [n]} y_i$. Since $y^* \notin S$ and $y^*, y_1, \dots, y_n$ are relatively prime,

$\gcd(y^*, r) = 1$ so by Bezout's Lemma, there are coefficients a, b such that

$$ar + by^* = 1 \qquad \text{(found via extended Euclidean algorithm)}$$

might be easier:

$= x^{ar + by^*}$

$= (w^*)^{y^*} x^{by^*}$

$= (w^* x^b)^{y^*}$

Take $y = (w^*)^a x^b$. Then, observe that

$$y^e = (w^*)^{ae} x^{be} = (w^*)^{ay^*} x^{by^*} \qquad (\text{since } e = y^* \text{ and } x^r = (w^*)^{y^*})$$
$$= x^{ar + by^*} = x \pmod N$$

$\therefore$ y is an e^{th} root of x $\Rightarrow$ broke strong RSA!

Strong RSA assumption :

Given $(N, x) \leftarrow$ GenerateStrongRSA(N, x), difficult to find (y, e) such that $e \neq \pm 1$

$$y^e = x \pmod N$$

That is, adversary can choose the exponent e. Normal RSA : fixed e