T-79.5502 Advanced Course in Cryptology

RSA and Rabin Signatures Signcryption

Alessandro Tortelli

26-04-06

Overview

- Introduction
- Probabilistic Signature Scheme PSS
- PSS with message recovery
- Signcryption
  - CSC1
  - RSA-TBOS

RSA and Rabin textbook signatures

- Textbook RSA and Rabin signatures are deterministic algorithms:
  - Given
    - (sk, pk) a key pair
    - M message

  ⇒ Signature is uniquely determined by (sk, pk) and M
- Undesirable property
  - Adaptive chosen message attack permits Malice to obtain two different square roots of a chosen message and thereby factor the modulus (§10.4.5)
- Solution: probabilistic approach

Signatures with Randomized Padding

- Bellare and Rogaway initiate the work of signing with RSA and Rabin in a probabilistic method

→ **Probabilistic Signature Scheme PSS**

- PSS is a randomized padding-based fit-for-application digital signature scheme for the RSA and Rabin functions
- Similarities with the RSA-OAEP scheme even if:
  – OAEP encryption procedure makes use of the one-way part of the RSA fucntion
  – PSS signature scheme uses the trapdoor part of the RSA function

PSS key parameters

- Let $(N, e, d, G, H, k_0, k_1) \leftarrow_U Gen(1^k)$

  RSA key material: $(N, e)\, public$

  $$d = e^{-1} (\mathrm{mod}\, \phi(N))\, private$$

- $k = |N| = k_0 + k_1$ with 2^{-k_0} and 2^{-k_1} negligible quantities
- Signing and verifying algorithms make use of two hash functions:

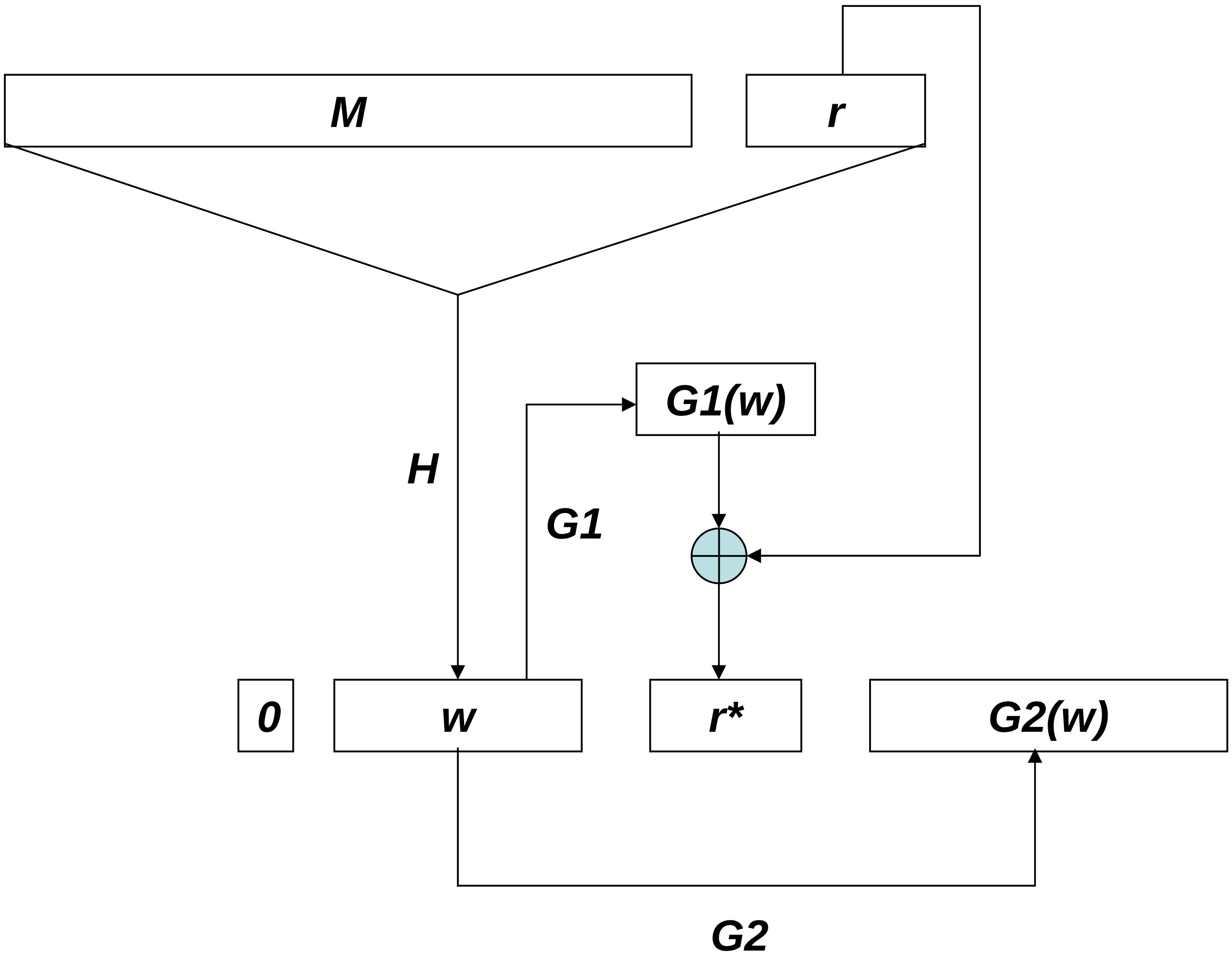
  $$H : \{0,1\}^* \mapsto \{0,1\}^{k1}$$ *compressor*

  $$G : \{0,1\}^{k1} \mapsto \{0,1\}^{k-k1-1}$$ *generator*

  G output is split in two sub-strings:

  - G1 has the first k0 bits
  - G2 has the remaining k-k1-k0-1 bits

PSS Padding



PSS signature generation

$$SignPSS(M,d,N)=$$

$$r \leftarrow_U \{0,1\}^{k0}$$

$$w \leftarrow H(M \| r)$$

$$r^* \leftarrow G_1(w) \oplus r$$

K-bit string less then N, necessary in order for the modulo exponentiation to be contucted correctly

$$y \leftarrow 0 \| w \| r^* \| G_2(w)$$

$$return(y^d \pmod N))$$

PSS signature verification

$$VerifyPSS(M, U, e, N) =$$

$$y \leftarrow U^e \pmod N$$

Parse y as $b \| w \| r^* \| \gamma$

$$r \leftarrow r^* \oplus G1(w)$$

if $(H(M \| r)) = w \wedge G2(w) = \gamma \wedge b = 0)$

return(**True**)

else return(**False**)

PSS Security

- As with the security proof of RSA-OAEP the security proof of RSA-PSS takes place in the random oracle model. Thus the security proof only provides *heuristic* evidence for security in the real world.

- Formal evidence is again derived from reduction to contradiction: breaking RSA-PSS will require roughly the same amount of work as it takes to solve the RSA problem ⇒ HARD PROBLEM!
  - forgery -> full invertion in one go -> exact security

- RSA-PSS is existentially unforgeable against adaptive chosen-message attacks in the random oracle model under the assumption that the RSA problem is intractable

Signing with Message Recovery PSS-R

- Main idea: a padding-signature scheme that also permits everybody to recover a signed message

- Original Scheme: Bellare and Rogaway

- Variation: Coron et al.
  - Is secure for signature usage (trapdoor part of RSA function)

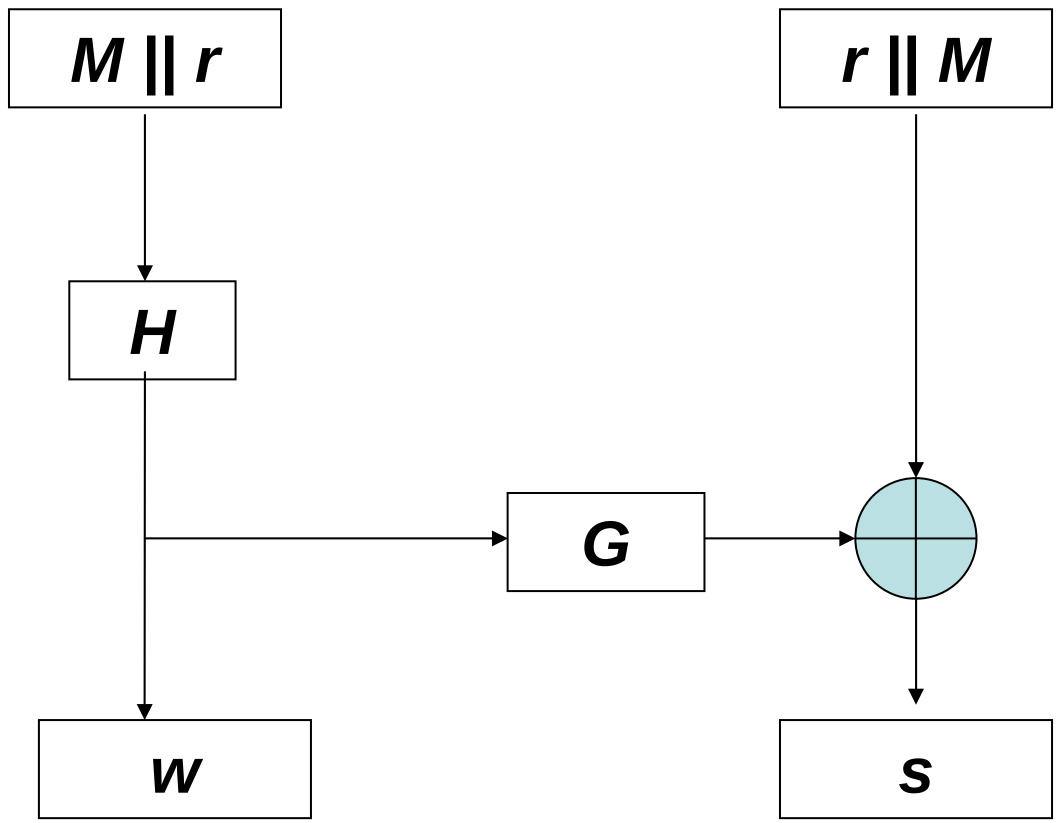
    unforgeability under adaptive chosen-message attack

  - Is secure for encryption usage (one-way part of RSA function)

    unforgeability under IND-CCA2 mode

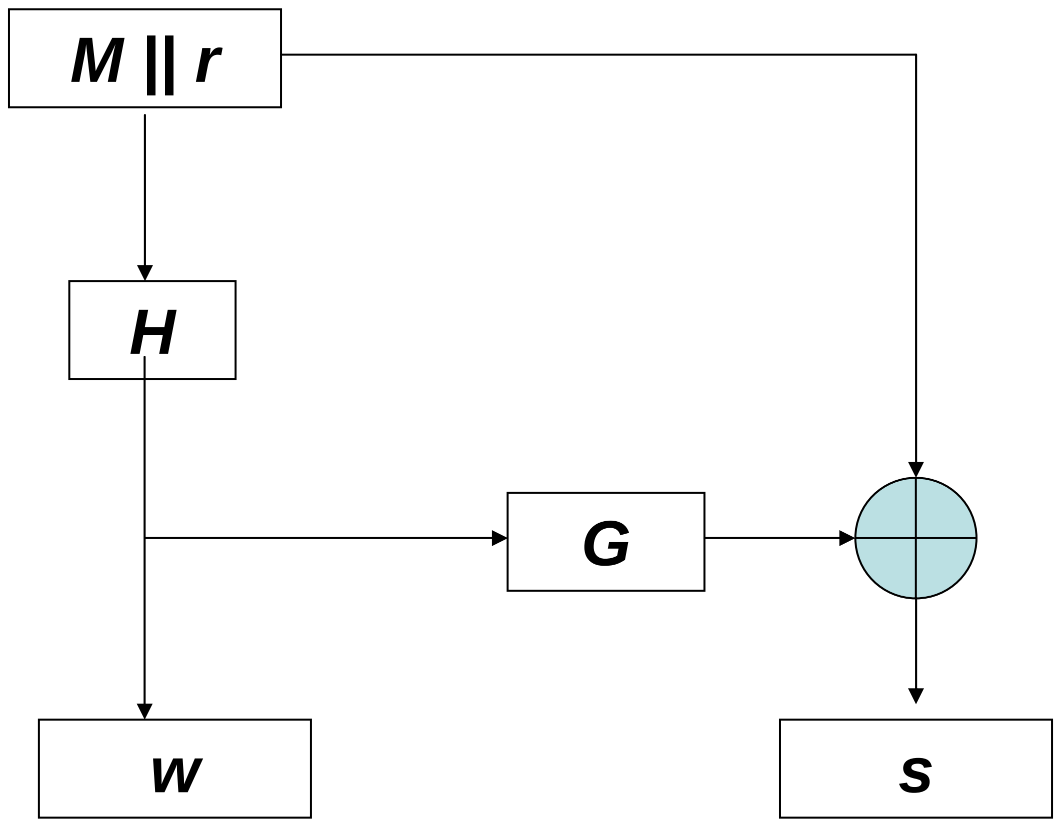
PSS-R Padding

Original of Bellare and Rogaway



PSS-R Padding

Variation of Coron et al.



PSS-R key parameters

- Let $(N, e, d, G, H, k_0, k_1) \leftarrow_U Gen(1^k)$

  RSA key material: $(N, e)\, public$

  $$d = e^{-1} (\bmod\, \phi(N))\, private$$

- $k = |N| = k_0 + k_1$ with 2^{-k_0} and 2^{-k_1} negligible quantities
- Signing and veifying algorithms make use of two hash functions:

$$G : \{0,1\}^{k1} \mapsto \{0,1\}^{k-k1-1}$$

$$H : \{0,1\}^{k-k1-1} \mapsto \{0,1\}^{k1}$$

PSS-R Signature Generation or Message Encryption

$$PSS-R-Padding(M, x, N) =$$

1. $r \leftarrow_U \{0,1\}^{k0}$

$w \leftarrow H(M \| r)$

$s \leftarrow G(w) \oplus (M \| r)$

$y \leftarrow (w \| s)$

2. $if\,(y \geq N)\,goto$ **1.**

3. $return(y^x (\mathrm{mod}\, N))$

x = d for signature generation

x = e for message encryption

PSS-R signature verification or decpryption with Ciphertext validation

$$PSS-R-UnPadding(U, x, N) =$$

$y \leftarrow U^x \pmod N$

Parse *y* as $w \| s$

Parse $G(w) \oplus s$ as $M \| r$

$if\,(H(M \| r) = w) \Longrightarrow return(True \| M)$

$else \Longrightarrow return(False \| Null)$

PSS-R Proof of security Encryption

- Proof of security is conceptually the same to that of RSA-OAEP
- A run of the attacker only causes a partial inversion
- Even running Malice twice, the reduction is far from tight (Number Field Sieve method works better if RSA modulus is less then 2048-bit)

$$\Rightarrow |w| > \frac{|N|}{2} \quad \Rightarrow |M||r| \le \frac{|N|}{2} \quad \Rightarrow |M| \le \frac{|N|}{2} - k_0$$

Rahter low bandwith for message recovery

PSS-R Proof of security Signature

- Proof of security is conceptually the same to that of RSA-PPS
- Successful forgery of a signature can lead to full inversion of RSA function in one go
- It suffices for k0 and k1 to have size with $2^{k\,0}$ $2^{k\,1}$ being negligible

$$\Rightarrow \quad |M| = k - k_0 - k_1$$

Signcryption

- Common practice:

  digital signature and then data encryption

  ➡ Message expantion rate

  ➡ Computational time spent

- Signcryption: is a public key primitive to achieve the combined functionality of digital signature and encryption
  - Zheng's Signcryption Scheme: SCS1(ElGamal)
  - Malone-Lee and Mao: Two Birds One Stone TBOS (RSA)

SCS1 parameters setup

- Public Parameters
  - p a large prime
  - q a large prime factor of p-1 (q|(p-1))
  - g an element of Zp* of order q
  - H: a oneway hash function
  - Setup a symmetric encryption algorithm $\mathcal{E}$

SCS1 keys setup

- Alice's keys
  - x_a: Alice's private key, $x_a \in Z_q^*$
  - y_a: Alice's public key, $y_a = g^{x_a} \bmod p$
- Bob's keys:
  - x_b: Bob's private key, $x_b \in Z_q^*$
  - y_b: Bob's public key, $y_b = g^{x_b} \bmod p$

SCS1 Signcryption

- To send to Bob *M*, Alice performs:

1. Pick *u* randomly from *[1 q]*, computes
   $$K = y_b^u \bmod p$$
   split *K* into *K1* and *K2* of appropriate lengths
2. $e \leftarrow H(K_2, M)$
3. $s \leftarrow u(e + x_a)^{-1} (\bmod q)$
4. $c \leftarrow \varepsilon_{k1}(M)$
5. Send to Bob the signcypted text *(c,e,s)*

SCS1 Unsigncryption

- Recived *(c,e,s)* from Alice, Bob performs:

1. Recover *K* from *e,s,g,p,ya* and *xb*:
   $$K \leftarrow (g^{e} y_{a})^{s x_{b}} \bmod p$$
2. Split *K* into *K1* and *K2*
3. $M \leftarrow D_{K1}(c)$
4. Accept *M* as a valid message originated from Alice only if:
   $$e = H(K_{2}, M)$$

SCS1 Efficency/1

- **Computation:**
  - Sygncryption:
    - One modulo exponentation
    - One hashing
    - One symmetric encryption
  - Unsigncryption
    - Similar amount of computation if $(g^e y_a)^{sx_b}$ is rewritten to $g^{esx_b} y_a^{sx_b}$ and computed using the "Product of Exponentiations Algorithm". Otherwise it needs two modulo exponentiations.

SCS1 Efficency/2

- **Communication bandwidth:**
  - Symmetric encryption doesn’t cause data expantion
  - Message signcrypted + 2|q| bits

    (same bandwidth for trasmitting a signature and signed message in the ElGamal-family signature)
  - Suitable for sending bulk volume of data efficiently (for example using a block cipher with the CBC mode of operation)

SCS1 Security

- SCS1 is essentialy a triplet ElGamal signature with a recoverable commitment

  unforgeability of signature under adaptive chosen-message attack
- Zheng has not given a a reductionist proof on the IND-CCA2 security
- Perhaps only the intened receiver is able to recover the commitment value K, under adaptive chosen-ciphertext attack

SCS1 Non-repudiation

- i.e. a principal cannot deny the authorship of a message.
- In Zheng's scheme, verification of a (triplet) signature requires recovery of the commitment K and the recovery needs to use the receiver's private key DRAWBACK!
- Third party's arbitration cannot be done!

⇨ Bob can conduct a Zero Knowledge Proof to convince the arbitrator (tricky)

Two Birds One Stone

- Main idea: “double-wraps”
  - Alice first signs a message by “wrapping” it inside the trapdoor part of her own RSA function
  - Then encrypts the signature by further “wrapping” it inside the one-way part of the RSA function of an intended receiver (Bob)

(N_A, e_A) (N_A, d_A) Alice’s RSA public, private key material

(N_B, e_B) (N_B, d_B) Bob’s public, private key material

$$\left[M^{d_A} (\bmod N_A)\right]^{e_B} \left(\bmod N_B\right)$$

RSA-TBOS observations

- Alice’s RSA modulus may be larger than Bob’s one → same moduli size
- In general a message is wrapped after the message has been processed with a randomized padding scheme
- If an “inner wrapping” result exceeds the modulus for an “outer wrapping”?

→ sender chops of the most significant bit

→ receiver uses trial-and-error test to put it back

RSA-TBOS Key Parameters

- Let *k* an even positive integer
- Let:

(N_A, e_A) (N_A, d_A) Alice's RSA public, private key material

(N_B, e_B) (N_B, d_B) Bob's public, private key material

satisfying $|N_A| = |N_B| = k$

- Signing and veifying algorithms make use of two hash functions:

⇨ $H : \{0,1\}^{n+k0} \mapsto \{0,1\}^{k1}$

⇨ $G : \{0,1\}^{k1} \mapsto \{0,1\}^{n+k0}$

Where $k = n + k_0 + k_1$ with 2^{-k_0} and 2^{-k_1} neglibibible quantities

RSA-TBOS Signcryption

- When Alice signcrypts a message $M \in \{0,1\}^n$ for Bob, she performs:

1. $r \leftarrow_U \{0,1\}^{k0}$
2. $w \leftarrow H(M \| r)$
3. $s \leftarrow g(w) \oplus (M \| r)$
4. $if\ \left(s \| w > N_A\right) goto\ (1.)$
5. $c' \leftarrow (s \| w)^{d_A} \pmod{N_A}$
6. $if\ (c' > N_B),\ c' \leftarrow c' - 2^{k-1}$
7. $c \leftarrow c'^{e_B} \pmod{N_B}$
8. Send *c* to Bob

RSA-TBOS Unsigncryption/1

- When Bob unsigncrypts a cryptogram *c* from Alice, he performs:

1. $c' \leftarrow c^{d_B} \pmod{N_B}$
2. $if\ (c' > N_A),\ reject$
3. $\mu \leftarrow c'^{e_A} \pmod{N_A}$
4. $Parse\ (\mu)\ as\ (s \,\|\, w)$
5. $M \,\|\, r \leftarrow G(w) \oplus s$
6. $if\ (H(M \,\|\, r) = w),\ return\ (M)$

RSA-TBOS Unsigncryption/2

7. $c' \leftarrow c' + 2^{k-1}$
8. $if\,(c' > N_A), reject$
9. $\mu \leftarrow c'^{e_A} \pmod{N_A}$
10. $Parse(\mu) as (s \,\|\, w)$
11. $M \,\|\, r \leftarrow G(w) \oplus s$
12. $if\,(w) \neq H(M \,\|\, r), reject$
13. Return *M*

RSA-TBOS features

- ✓ Non-repudiation is very simple
  - o The receiver of a signcryption follows the unsigncryption procedure up until stage 2, c’ may then be given to a third party who can verify its validity
- ✓ Message confidentiality under the IND-CCA2
- ✓ Signature unforgeability under the chosen message attack
- ❖ Rather low message bandwidth for message recovery due to the application of the RSA-PSS-R padding scheme