Introduction to Cryptology

11.1 - RSA and other factoring-based schemes

Federico Pintore

Mathematical Institute, University of Oxford (UK)

Michaelmas term 2020

RSA Encryption Scheme

Designed by R. Rivest, A. Shamir and L. Adleman in 1977.

Variants of the original scheme still used nowadays, for both public-key encryption and digital signatures.

Security is based on the RSA problem.

Plain RSA encryption algorithm

The RSA encryption scheme $E_{\mathrm{RSA}} = (\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Dec})$ is composed by three PPT algorithms defined as follows.

- $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathrm{KeyGen}(n)$: it runs a GenRSA algorithm on input a security parameter n. Then PK is set to (N, e), while SK is set to (N, d)[1].
- $c \leftarrow \mathrm{Enc}(\mathrm{PK}, m \in \mathbb{Z}_N^*)$: on input a public key (N, e) and a message m, it outputs $c = m^e$.
- $m \leftarrow \mathrm{Dec}(\mathrm{SK}, c)$: on input a secret key (N, d) and a ciphertext c, it computes $m = c^d$.

[1] We recall that $N = pq$, where p and q are two distinct n-bit odd primes, while $[e]_{\varphi(N)}[d]_{\varphi(N)} = [1]_{\varphi(N)}$.

Plain RSA encryption algorithm

The RSA encryption scheme $E_{\mathrm{RSA}} = (\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Dec})$ is composed by three PPT algorithms defined as follows.

- $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathrm{KeyGen}(n)$: it runs a GenRSA algorithm on input a security parameter n. Then PK is set to (N, e), while SK is set to (N, d)[1].
- $c \leftarrow \mathrm{Enc}(\mathrm{PK}, m \in \mathbb{Z}_N^*)$: on input a public key (N, e) and a message m, it outputs $c = m^e$.
- $m \leftarrow \mathrm{Dec}(\mathrm{SK}, c)$: on input a secret key (N, d) and a ciphertext c, it computes $m = c^d$.

Correctness: $c^d = (m^e)^d = m^{ed} = m^{\ell\varphi(N)+1} = m$.

[1] We recall that $N = pq$, where p and q are two distinct n-bit odd primes, while $[e]_{\varphi(N)}[d]_{\varphi(N)} = [1]_{\varphi(N)}$.

Plain RSA security

E_{RSA} is deterministic, so it does not have indistinguishable encryptions in the presence of an eavesdropper.

The factoring assumption implies that it is computationally infeasible to recover the private key from the public key.

The RSA assumption implies that an adversary $\mathcal{A}$ cannot recover m from (N, e) and c if m is uniform in $\mathbb{Z}_N^*$.

- What if m is not chosen uniformly from $\mathbb{Z}_N^*$?
- What if $\mathcal{A}$ learns partial information about m?

Padded RSA

Idea: to encrypt m, first map it to an element $\tilde{m} \in \mathbb{Z}_N^*$. The map should be randomised and reversible.

Padded RSA

Idea: to encrypt m, first map it to an element $\tilde{m} \in \mathbb{Z}_N^*$. The map should be randomised and reversible.

- Enc chooses a uniform $r \in \{0,1\}^{\ell(n)}$, and sets $\tilde{m} = r||m$.
- The security of the padded variant depends on $\ell(n)$.
  - $\ell(n) = \mathcal{O}(\log n)$ is a bad choice;
  - provable security based on the RSA assumption when $m \in \{0,1\}$ and ℓ is as large as possible;
  - for other cases, no security proof, but no attacks are known either!

RSA-OAEP

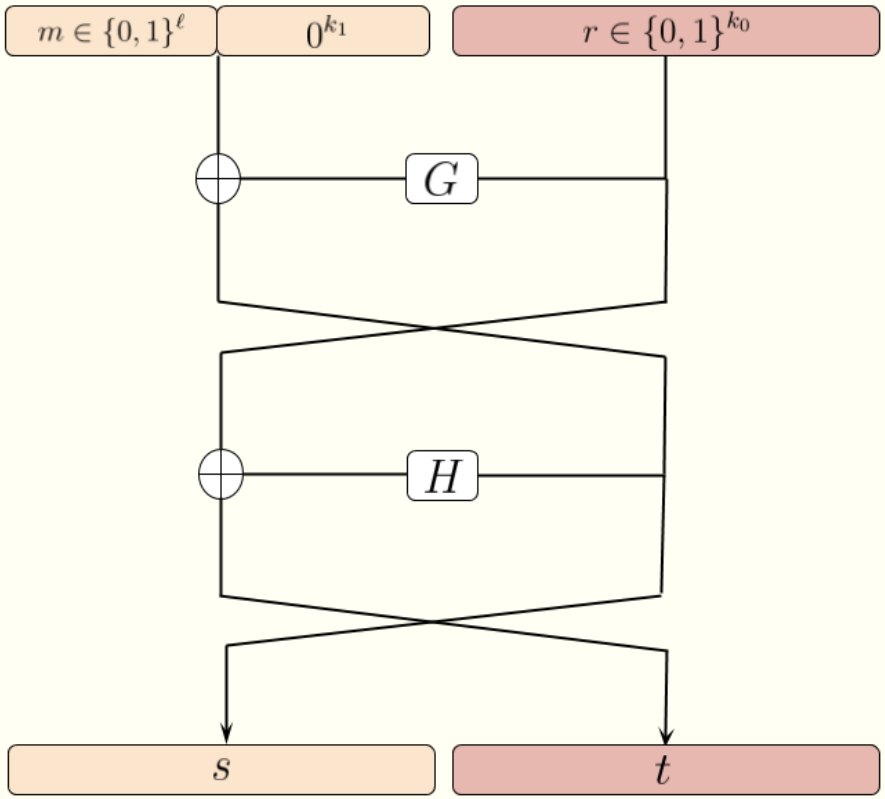
CCA-secure variant which uses optimal asymmetric encryption padding OAEP. Part of RSA PKCS#1 since version 2.0.

RSA-OAEP

CCA-secure variant which uses optimal asymmetric encryption padding OAEP. Part of RSA PKCS#1 since version 2.0.

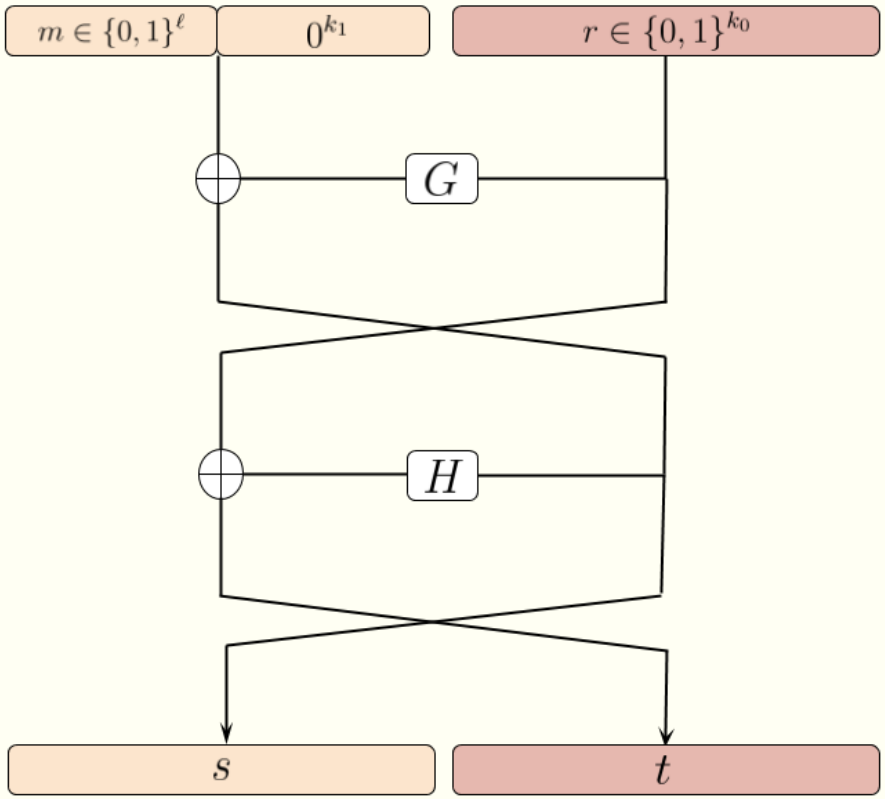
- It uses integer-valued functions $\ell(n), k_0(n), k_1(n)$ where $k_0(n)$, $k_1(n) = \Theta(n)$ and $\ell(n) + k_0(n) + k_1(n)$ smaller than the minimum bit-length of the moduli output by GenRSA(n).
- Two hash functions H and G are also employed. They are modelled as random oracles in the security proof.
- The transformation executed by OAEP is a two-round Feistel network (G and H are the round functions).

The OAEP mechanism



$$H : \{0,1\}^{\ell+k_1} \to \{0,1\}^{k_0}$$
$$G : \{0,1\}^{k_0} \to \{0,1\}^{\ell+k_1}$$

The OAEP mechanism



$H : \{0,1\}^{\ell+k_1} \to \{0,1\}^{k_0}$

$G : \{0,1\}^{k_0} \to \{0,1\}^{\ell+k_1}$

Input: $m \in \{0,1\}^{\ell}$

$m' := m||0^{k_1}$

$r \leftarrow \{0,1\}^{k_0}$

$s := m' \oplus G(r) \in \{0,1\}^{\ell+k_1}$

$t := r \oplus H(s) \in \{0,1\}^{k_0}$

$\tilde{m} := s||t$

return $\tilde{m}$

RSA-OAEP

The RSA-OAEP encryption scheme $(\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Dec})$ is composed by three PPT algorithms defined as follows.

- $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathrm{KeyGen}(n)$: it runs a GenRSA algorithm on input a security parameter n and sets PK to (N, e) and SK to (N, d).
- $c \leftarrow \mathrm{Enc}(\mathrm{PK}, m \in \{0,1\}^{\ell(n)})$: m is padded with the OAEP mechanism, obtaining $\tilde{m}$. The ciphertext c is set to $([\tilde{m}]_N)^e$, where $\mathrm{PK} = (N, e)$.
- $m \leftarrow \mathrm{Dec}(\mathrm{SK}, c)$: on input a secret key $\mathrm{SK} = (N, d)$ and a ciphertext c, $\tilde{m}$ is set to c^d. If $|\tilde{m}| \neq \ell + k_0 + k_1$, the algorithm outputs $\perp$. Otherwise:
  - it parses $\tilde{m}$ as $(s||t)$, where $s \in \{0,1\}^{\ell+k_1}, t \in \{0,1\}^{k_0}$;
  - it computes $r := H(s) \oplus t$;
  - it computes $m' := G(r) \oplus s$. If the most-significant k_1 bits of m' are not all 0, it outputs $\perp$. Otherwise, it outputs the ℓ least-significant bits of m'.

Security of RSA-OAEP

RSA-OAEP is CCA-secure in the ROM.

Security of RSA-OAEP

RSA-OAEP is CCA-secure in the ROM.

In 2001, James Manger showed an attack on a variant of RSA-OAEP specified in PKCS#1 v2.0.

- It exploited implementation weaknesses, prone to side-channel attack.
- Two different conditions make Dec output $\perp$. The times to return the message errors were not identical.
- The attack recovered the plaintext m with $||N||$ queries to an oracle leaking the error.

Security of RSA-OAEP

RSA-OAEP is CCA-secure in the ROM.

In 2001, James Manger showed an attack on a variant of RSA-OAEP specified in PKCS#1 v2.0.

- It exploited implementation weaknesses, prone to side-channel attack.
- Two different conditions make Dec output $\perp$. The times to return the message errors were not identical.
- The attack recovered the plaintext m with $||N||$ queries to an oracle leaking the error.

Cryptographic implementations should adhere to the theoretical formalisations.

RSA - Weak-key Generator Attack

Suppose Alice computes a composite number $N_A = pq_A$, while Bob computes $N_B = pq_B$.

RSA - Weak-key Generator Attack

Suppose Alice computes a composite number $N_A = pq_A$, while Bob computes $N_B = pq_B$.

- Alice can compute $q_B = N_B/p$ (Bob $q_A = N_A/p$).

RSA - Weak-key Generator Attack

Suppose Alice computes a composite number $N_A = pq_A$, while Bob computes $N_B = pq_B$.

- Alice can compute $q_B = N_B/p$ (Bob $q_A = N_A/p$).
- Anyone can compute $\gcd(N_A, N_B) = p$, and then q_A and q_B.

RSA - Weak-key Generator Attack

Suppose Alice computes a composite number $N_A = pq_A$, while Bob computes $N_B = pq_B$.

- Alice can compute $q_B = N_B/p$ (Bob $q_A = N_A/p$).
- Anyone can compute $\gcd(N_A, N_B) = p$, and then q_A and q_B.

A concrete attack in 2012:

Ron was wrong, *Whit is right*, Lenstra *et al.*

It showed that 2/1000 of the gathered RSA keys were weak.

A CCA-secure KEM in the ROM

Consider a KEM $(\mathrm{KeyGen}, \mathrm{Encaps}, \mathrm{Decaps})$ defined as follows:

- $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathrm{KeyGen}(n)$: given a security parameter n, it runs a GenRSA algorithm on n and specifies a hash function $H : \mathbb{Z}_N^* \to \{0,1\}^n$. Then it sets PK to (N, e, H) and SK to (N, d, H).

- $(c, k) \leftarrow \mathrm{Encaps}(\mathrm{PK}, n)$: on input a public key $\mathrm{PK} = (N, e, H)$ and n, it picks a random $r \in \mathbb{Z}_N^*$ and outputs the ciphertext $c := r^e$ and the key $k := H(r)$.

- $k \leftarrow \mathrm{Decaps}(\mathrm{SK}, c)$: on input a secret key $\mathrm{SK} = (N, d, H)$ and a ciphertext c, it outputs $k := H(c^d)$.

A CCA-secure KEM in the ROM

Consider a KEM $(\mathrm{KeyGen}, \mathrm{Encaps}, \mathrm{Decaps})$ defined as follows:

- $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathrm{KeyGen}(n)$: given a security parameter n, it runs a GenRSA algorithm on n and specifies a hash function $H : \mathbb{Z}_N^* \to \{0,1\}^n$. Then it sets PK to (N, e, H) and SK to (N, d, H).
- $(c, k) \leftarrow \mathrm{Encaps}(\mathrm{PK}, n)$: on input a public key $\mathrm{PK} = (N, e, H)$ and n, it picks a random $r \in \mathbb{Z}_N^*$ and outputs the ciphertext $c := r^e$ and the key $k := H(r)$.
- $k \leftarrow \mathrm{Decaps}(\mathrm{SK}, c)$: on input a secret key $\mathrm{SK} = (N, d, H)$ and a ciphertext c, it outputs $k := H(c^d)$.

Its security relies on the RSA assumption. Part of the ISO/IEC18033-2 standard for public-key encryption.

Rabin Encryption Scheme

The Rabin encryption scheme $(\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Dec})$ consists of three PPT algorithms:

- $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathrm{KeyGen}(n)$: it runs GenModulus on input a security parameter n, obtaining (N, p, q), where $N = pq$ and p, q are n-bit primes with $[p]_4 = [q]_4 = [3]_4$.
  PK is set to N, SK is set to (p, q).
- $c \leftarrow \mathrm{Enc}(\mathrm{PK}, m \in \{0,1\})$: on input a public key N and a message m, it chooses a uniform $r \in QR(N)$, where $lsb(r) = m$, and outputs $c := r^2$.
- $m \leftarrow \mathrm{Dec}(\mathrm{SK}, c)$: given a secret key (p, q) and a ciphertext c, it computes the unique $r \in QR(N)$ s.t. $r^2 = c$, and outputs $lsb(r)$.

Rabin Encryption Scheme

The Rabin encryption scheme $(\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Dec})$ consists of three PPT algorithms:

- $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathrm{KeyGen}(n)$: it runs GenModulus on input a security parameter n, obtaining (N, p, q), where $N = pq$ and p, q are n-bit primes with $[p]_4 = [q]_4 = [3]_4$.
  PK is set to N, SK is set to (p, q).
- $c \leftarrow \mathrm{Enc}(\mathrm{PK}, m \in \{0,1\})$: on input a public key N and a message m, it chooses a uniform $r \in QR(N)$, where $lsb(r) = m$, and outputs $c := r^2$.
- $m \leftarrow \mathrm{Dec}(\mathrm{SK}, c)$: given a secret key (p, q) and a ciphertext c, it computes the unique $r \in QR(N)$ s.t. $r^2 = c$, and outputs $lsb(r)$.

Theorem

If the factoring problem is hard relative to GenModulus*, then the encryption scheme is CPA-secure.*

Further Reading I

Mihir Bellare, Alexandra Boldyreva, and Silvio Micali.
Public-Key Encryption in a Multi-user Setting: Security Proofs and Improvements.
In Bart Preneel, editor, Advances in Cryptology — EUROCRYPT 2000, volume 1807 of Lecture Notes in Computer Science, pages 259–274. Springer Berlin Heidelberg, 2000.

Dan Boneh.
Simplified OAEP for the RSA and Rabin Functions.
In Joe Kilian, editor, Advances in Cryptology — CRYPTO 2001, volume 2139 of Lecture Notes in Computer Science, pages 275–291. Springer Berlin Heidelberg, 2001.

Further Reading II


Ronald Cramer and Victor Shoup.
Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack.
SIAM Journal on Computing, 33(1):167–226, 2003.


Whitfield Diffie and Martin E Hellman.
New directions in cryptography.
Information Theory, IEEE Transactions on, 22(6):644–654, 1976.


Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir.
New attacks on Feistel Structures with Improved Memory Complexities.
In Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I, pages 433–454, 2015.

Further Reading III

Naofumi Homma, Atsushi Miyamoto, Takafumi Aoki, Akashi Satoh, and Adi Shamir.
Collision-Based Power Analysis of Modular Exponentiation Using Chosen-Message Pairs.
In Cryptographic Hardware and Embedded Systems - CHES 2008, 10th International Workshop, Washington, D.C., USA, August 10-13, 2008. Proceedings, pages 15–29, 2008.