Outline Cours 2

- Part 1 : Asymmetric cryptography, one way function, complexity
- Part 2 : arithmetic complexity and lower bounds : exponentiation

- **Part 3 : Provable security and polynomial time reduction :**
  - **P, NP classes. One-way function and NP class.**

- Part 4 : RSA : the algorithm
- Part 5 : Provable security of RSA
- Part 6 : Attacks and importance of padding.

One-way function and *NP class*

- E : $\{0,1\}^n \rightarrow \{0,1\}^n$ (or $Im(E) \subset \{0,1\}^{n+1}$)
  injective (one-to-one mapping),
  and ***easy to compute*** *i.e. ~linear time to compute E(X)*

- $D = E^{-1}$: must be computationaly impossible

- We do not know if such functions exist. But:
  - E « easy » to compute $\Rightarrow E \in P$
  - Then, since $D=E^{-1}$ $\Rightarrow D \in NP$
    - Proof: polynomial-time certificate

- Then, look for a convenient D among the most difficult problems inside NP… conjectured intractable
  - NP-complete ones: eg subset sum/knapsack [Merkle-Hellman, Chor-Rivest…]
  - Conjectured computationnaly imposible ones: factorization…

Some «hard » problems used to build one-way function

- Subset sum [NP-complete]
  - Input : S, $(a_1, \ldots, a_n)$; - Output : $(x_1, \ldots, x_n) \in \{0,1\}^n \sum_{i=1}^{n} x_i a_i = S$
- **Discrete logarithm** (not known in P nor NP-hard)
  - Input : g, M ; - Output : x tel que $g^x = M$
- **Factorization** (not known in P nor NP-hard…)
  1. Input : N - output : factorization of N
  2. Input: N, M, C ; - output : d s.t $M^d = C \bmod N$
  3. Input : N, e, C ; - output : M s.t. $M^e = C \bmod N$
  4. Input : N, x ; - output : YES iff ∃ y such that $x = y^2 \bmod N$

Example 1 : « Exponential and Discrete logarithm »

- (G, *) : cyclic group of order n; g a generator of G
  - $G = \{ g^i ; i = 0, \ldots, n-1 \}$
- **Exponential** : Exp: $\{ 0, \ldots, n-1\} \rightarrow G$ defined by $Exp(i) = g^i$

  Computation cost of Exp (i) = O(log (i)) = **O(log n)** [upper and lower bound, lect2]
  Example : 5^{11} [7] = $((5^2)^2\ 5)^2\ 5 = ((4)^2\ 5)^2\ 5 = (2.5)^2\ 5 = 2.5 = 3$
- **Discrete Logarithm**: Log : $G \rightarrow \{ 0, \ldots, n-1\}$ defined by Log(x) =i s.t. $x= g^i$

  Example : find x / $6^x = 8$ [11]

  (Réponse: x = 7)

  Best known algorithms for any G in **O($n^{0.5}$)** [Shanks]
  - Note : INTEGER-FACTORIZATION ≤P DISCRETE-LOGARITHM

  Conjectured hard to compute :
  - Very used in asymmetric cryptography: ex RSA, El Gamal, ECDLP
  - **But** : some specific instances are easy to compute

One-way trapdoor function

- Definition:
  - E is one-way
  - D(E(x)) = x [and E(D(x)) = x for signature]
  - But, given a trapdoor (the secret key), D is *easy* to compute (almost linear time)
- Provable security:
  - Given c = E(x), computing x is untractable
  - How to prove it? By reduction (contractiction) !
    - assume there exists an algorithm to compute x from c
    - then exhibit an algorithm that computes an untracatable problem !

Example 2 : « knapsack » [Merkle-Hellman,78

- SUBSETSUM $\in$ *NP* -complete
  - Input : ($a_1, \ldots, a_n$) and *S* integers
  - Output : YES iff it exists ($x_1, \ldots, x_n$) $\in \{0,1\}^n$: $\sum_{i=1}^{n} x_i a_i = S$
- Idea for an encoding: $E(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i a_i$
- Building a trapdoor function
  - Easy to solve instance; choose ($a_1, \ldots, a_n$) ***super-increasing.***
    - What is the decoding algorithm?
  - Hiding simplicity $b_i = t.a_i \bmod m$ with t secret and prime to m
  - Public : ($b_1, \ldots, b_n$) and m : $E(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i b_i \bmod m$
  - Secret : ($a_1, \ldots, a_n$), t and $u = t^{-1} \bmod m$:
    - Decoding: just compute (S.u mod n) and decode from ($a_1, \ldots, a_n$)

Outline Cours 2

- Part 1 : Asymmetric cryptography, one way function, complexity
- Part 2 : arithmetic complexity and lower bounds : exponentiation
- Part 3 : Provable security. One-way function and NP class.

- **Part 4 : RSA : the algorithm**
- **Part 5 : Provable security of RSA**

- Part 6 : Importance of padding. Application to RSA signature.

Provable security of RSA

Rivest / Shamir / Adleman (1977)

Outlines:

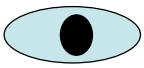
- RSA cipher: E and D
- Provable security of RSA
  1. E (D (x)) = D(E(x)) = x
  2. E is easy to compute
  3. E is hard to invert without knowing D

RSA

Alice

Wants to send secret M to Bob

Eva



Bob

1/ **Building keys - Bob**

- p, q large prime numbers
- n= p x q
- $\varphi(n) = (p-1)*(q-1)$
- e small, prime to $\varphi(n)$
- $d = e^{-1} \pmod{\varphi(n)}$

- Private key : (d, n)
  Public key : (e, n)
- $\forall x \in \{0, \ldots, n-1\}$:
  $D^{Bob}(x) = x^d \pmod n$
  $E^{Bob}(x) = x^e \pmod n$

$E^{Bob}(x)$

RSA

Alice

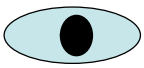
Wants to send secret M to Bob

2. $M = M_1 M_2 \ldots M_m$ such that
   $M_i \in \{0, \ldots, n-1\}$
   i.e. each block has $\log_2 n$ bits

3.Compute $S_i = E^{Bob}(M_i)$

4.Sends $S_1 \ldots S_i \ldots S_m$

Eva



$S_1 \ldots S_i \ldots S_m$

Bob

1/ **Building keys - Bob**

- $\forall x \in \{0, \ldots, n-1\}$:
  private: $D^{Bob}(x) = x^d \pmod n$
  public: $E^{Bob}(x) = x^e \pmod n$

5. Compute $M_i = D^{Bob}(S_i)$

$M = M_1 M_2 \ldots M_m$

Public: $E^{Bob}(x)$

Provable security of RSA

1. To generate a RSA key [(n,**d**), (n,e)] is easy (almost linear time)

2. D^{Bob} is the inverse of E^{Bob} :
   – $\forall x \in \{0, \ldots, n-1\}$: $D^{Bob}(E^{Bob}(x)) = E^{Bob}(D^{Bob}(x)) = x$

3. E^{Bob} is a one-way trap-door function :
   a) $E^{Bob}(x)$ is easy to compute (in almost linear time)
   b) $D^{Bob}(x)$ is easy to compute (in almost linear time) for the one who knows the trapdoor ***d***
   c) Recover x from $E^{Bob}(x)$ ***is computationally impossible***
      - Conjectured
      - **Theorem**: Breaking the RSA private key, ie computing *d* from *n* and *e* is computationally more difficult than factorising n

        => Believed secure if its hard to factor big numbers

Challenges RSA

-

Challenge	Price	Date
RSA-576	$10 000	3/12/2003 [Franke&al]
RSA-640	$ 20 000	2/12/2005 [Bahr&al]
RSA-704	$30 000	open
RSA-768	$50 000	open
RSA-896	$75 000	open
RSA-1024	$100 000	open
RSA-1536	$150 000	open
RSA-2048	$200 000	open

Outline Cours 2

- Part 1 : Asymmetric cryptography, one way function, complexity
- Part 2 : arithmetic complexity and lower bounds : exponentiation
- Part 3 : Provable security. One-way function and NP class.
- Part 4 : RSA : the algorithm
- Part 5 : Provable security of RSA

- **Part 6 : Attacks and importance of padding.**
  Application to RSA signature.

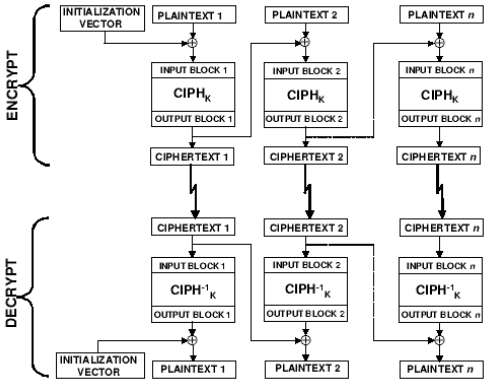
Complements on RSA

- Choice of the keys:
  - p, q: primes large enough [512 bits, 1024 bits=> RSA 2048]
  - d large (> N1/4 [attaque de Wiener]
  - e small (efficiency and ensures d to be large):
    - e=3, 17, 65537 [X.509 norm: **e=65537,** only 17 multiplication]
  - p such that p-1 has a large prime factor: p=2.p'+1 (idem for q)
    [Gordon algorithm based on Miller-Rabin primality test]

- Other attacks
  - Timing-attack: based on the analysis of the time to compute x^d mod n:
    - *Blinding* trick: to decode, choose a random r and compute $(r^e x)^d.r^{-1}$ mod n
  - Chosen-ciphertext attack, adpative chosen ciphertext attack
  - Frequency analysis

Protection: Padding and chaining

- Protection: always add some random initalization bits to the first block and use a chaining mode.
- Eg: mode **CBC** [Cipher Block Chaining]



- Other modes: OFB, Counter, GCM

Applications
Chiffrement à clef publique / RSA

- Authentification
- Signature

Signature RSA

Alice

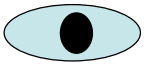
Reçoit un message M signé par Bob et vérifie l'authenticité de la signature

3. Reçoit M, S

4. Calcule : T = E(S)

5. Vérifie la signature de Bob en testant T = M ?

Eve



Bob

- veut envoyer M signé à Alice

1. Calcule S = $D^{Bob}(M)$
2. Envoie (M, S) à Alice

M S

$E^{Bob}(x)$

Signature RSA du résumé

Alice

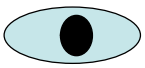
Reçoit un message M signé par Bob et vérifie l'authenticité de la signature

3. Reçoit M, S

4. Calcule : T = E(S)

5. Vérifie la signature de Bob : T = h(M) ?

Eve



Bob

veut envoyer M signé à Alice ne signe que le résumé h(M) où h est une fonction à sens unique.

Ex: h = SHA1, MD5, …

1. Calcule S = $D^{Bob}(h(M))$
2. Envoie (M, S) à Alice

M S

$E^{Bob}(x)$

Outline Cours 2

- Part 1 : Asymmetric cryptography, one way function, complexity
- Part 2 : arithmetic complexity and lower bounds : exponentiation
- Part 3 : Provable security. One-way function and NP class.
- Part 4 : RSA : the algorithm
- Part 5 : Provable security of RSA
- Part 6 : Importance of padding. Application to RSA signature.

Summary Lecture 2

- Provable security relies on complexity
- Breaking and RSA key is proved more difficult than factorization
  - But decrypting a message without computing d remains an open question
  - There exists variants that are proved more difficult than factorization [Rabin]:
    - But they are more expensive than RSA
  - Choices of the key (size and form of the primes) matters
- There exist other protocols with comparable security and smaller keys [ECDLP,..]
- Importance of padding and hash function
- -> Next lecture: provable random number generators and hash functions