Randomization
Deterministic randomized algorithms

- Randomized algorithms for decision problems
  - Atlantic City (B) / Monte Carlo (R) / Las Vegas (Z)
- Complexity classes:
  - Atlantic City, polynomial time: BPP
  - Monte Carlo, polynomial time: RP
  - Las Vegas, polynomial time: ZP

Randomized algorithm and BPP

- Probabilistic algorithm:
  - Uses instruction Random() that returns 0 with probability ½ and 1 with probability ½.
- BPP = Bounded-error Probabilistic Polynomial time

  BPP = { *f* functions such that there exists a probabilistic polynomial time algorithm *A*: $\forall x \in \{0,1\}^*$ Prob[$A(x)=f(x)$] ≥ 2/3 }
- Equivalent def: random values are set in input:
  BPP = { *f*: it exists polynomial-time DTM *M* and a polynomial P $\forall x \in \{0,1\}^*$ $\text{Prob}_{r \text{ random} \in \{0,1\}^{P(|x|)}}$ [$M(x,r)=f(x)$] ≥ 2/3 }

One way function

- **Definition**: A polynomial time computable function
  f: {0,1}* -> {0,1}*
  is a one-way function iff

  $\forall$ probabilistic polynomial-time algorithm A, there exists a negigible function $\varepsilon = n^{-\omega(1)}$ such that $\forall n$

  $\text{Prob}_{y = f(x) \text{ with } x \text{ random} \in \{0,1\}^n}$ [A(y) = x' with f(x')=y] < ε(n)
- **Theorem**: if there exists a one-way function, P≠NP
  - Proof: contradiction
- *Conjecture: there exists a one-way function.*

Examples of presumed one-way (based on factorization)

- Ex1: multiplication $f(x_1 \| x_2) = x_1 . x_2$
- Ex2 : n bits of the input x used as random bits to generate two n/3 bits primes P_x and Q_x. $f(x) = P_x.Q_x$
- Ex3 : $RSA_{N,e}(x) = x^e \bmod N$ with N=PQ and e coprime to (P-1)(Q-1)
  - One-to-one mapping in Z_N^*
- Ex4: Rabin function: $f(X) = X^2 \bmod N$ for X in QR_N
  (X quadratic residue modulo N iff it exists W: $X=W^2 \bmod N$)
  - One-to-one mapping in QR_N

Levin's universal one-way function

- Let M_i= the i^{th} DTM (according to some arbitrary numbering M1, …, Mn, …)
  and let $M_i^t(x)$ be the output of $M_i(x)$ if $M_i(x)$ uses less than t steps, else $0^{|x|}$.
- Levin's universal one-way function f_U :
  - Input n bits treated as a list $x_1, \dots x_{\log n}$ of n/log n bit strings
  - Output: $M_1^T(x_1), \dots, M_{\log n}^T(x_{\log n})$ with $T = n^2$
- Theorem : if some one-way function g exists,
  then f_U is one way.

Encryption from one-way functions

- **Def**: (E,D) encryption with n-bits keys for m-bits messages.
  (E,D) is computationnaly secure iff, for every probabilistic polynomial-time algorithm A,
  there exists a negigible function $\varepsilon = n^{-\omega(1)}$ such that $\forall n$
  $\mathrm{Prob}_{k\in_R\{0,1\}^n,\, x\in_R\{0,1\}^m}$ [$A(E_k(x))=(i,b)$ such that $x_i=b$] $\leq \frac{1}{2} + \varepsilon(n)$
- Theorem: Suppose one-way functions exist.
  Then, for every integer c≥1, there exists a computationally secure encryption scheme (E,D) using n-length keys for n^c-length messages.

Semantic security

- The encryption scheme provides no additional information on the plaintext than its previously know distribution.
  - A sequence $X=(X_n)_{n\in N}$ of rand. var. with $X_n \in \{0,1\}^{m(n)}$ (m polynom) is ***sampleable*** if it exists a probabilistic polynomial time algorithm D such that, for any n, X_n = distribution $D(1^n)$.
  - Then the encryption should not provide more information than D
    - Ie ciphertext distribution is undistinguishable from distribution $E(D(1^n))$
- **Def**: (E,D) encryption with n-bits keys for m(n)-bits messages for some polynomial m. (E,D) is **semantically secure** iff
  - $\forall$ sampleable sequence $(X_n)_{n\in N}$ with $X_n \in \{0,1\}^{m(n)}$ (m polynom)
  - $\forall$ polynomial-time computable function f: {0,1}*-> {0,1}
  - $\forall$ probabilistic polynomial-time algorithm A,

  there exists a negigible function $\varepsilon = n^{-\omega(1)}$ and a probabilistic polynomial algorithm B such that $\forall n$
  $\mathrm{Prob}_{k\in_R\{0,1\}^n,\, x\in_R X_n}[A(E_k(x))= f(x)] \leq \mathrm{Prob}_{x\in_R X_n}[B(1^n) = f(x)] + \varepsilon(n)$

Outline Lecture 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)
- $\forall x \in \{0, \dots, n-1\}$:
  $D^{Bob}(x) = x^d \pmod n$
  $E^{Bob}(x) = x^e \pmod n$

$E^{Bob}(x)$

RSA

Alice

Wants to send secret M to Bob

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

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

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

Eva

$S_1 \dots S_i \dots S_m$

Bob

1/ **Building keys - Bob**

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

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

$M = M_1 M_2 \dots M_m$

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} :
   - $\forall x \in \{0, \dots, n-1\}$: $D^{Bob}(E^{Bob}(x)) = E^{Bob}(D^{Bob}(x)) = x$

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 Lecture 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.

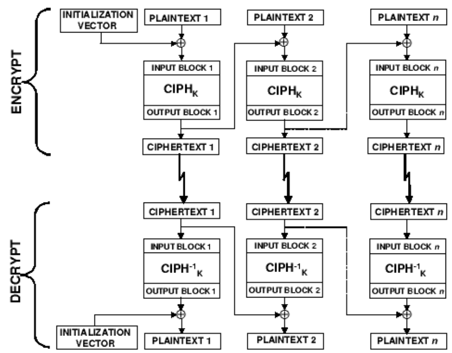
Complements on RSA

- 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

Assymmetric cryptography applications / RSA

- Authentication
- Signature

RSA Signature

Alice

Receives a message M signed by Bob and verifies its signature (authentication)

3. receive M, S

4. compute T = E(S)

5. Verifies Bob's signature
   By testing T = M ?

Eve

M S

$E^{Bob}(x)$

Bob

Signs message M and sends it to Alice

1. Compute S = $D^{Bob}(M)$
2. Sends (M, S) to Alice

RSA signature of the digest

Alice

Receives a message M signed by Bob and verifies its signature (authentication)

3. Receives M, S

4. Computes: T = E(S)

5. Verifies Bob's signature:
   is T = h(M) ?

Eve

M S

$E^{Bob}(x)$

Bob

Sends M signed to Alice but signs only the digest h(M) where h is a one-way function.

Ex: h = [SHA1, MD5], SHA2…

1. Computes h(M) and
   S = D^{Bob}(h(M))
2. Sends (M, S) to Alice

Outline Course 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 Course2

- 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: hash functions