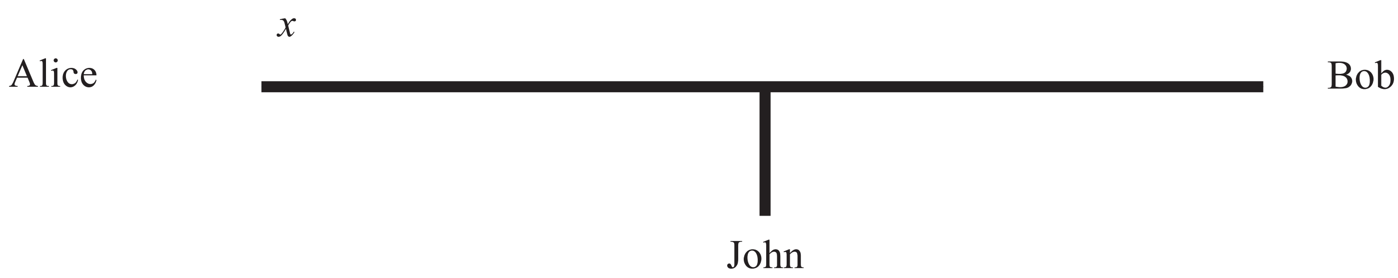
Theory of Computation
Chapter 12: Cryptography

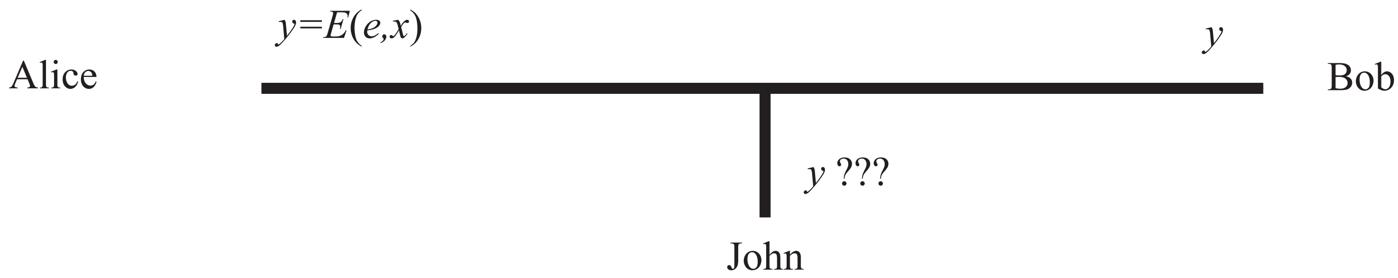
Guan-Shieng Huang

Dec. 20, 2006

Introduction

Alice wants to communicate with Bob secretely.





Assumption

- The encryption method is publicly known.
- The transmission is intercepted by John.
- John is malevolent; he may send fake messages to deceive Bob.

Requirements

1. $D(d, E(e, x)) = x$
2. D and E are polynomial-time algorithms
3. John cannot compute x from y without knowing d.

One-time pad (information secure)

Let $e = d$, a random string of length the same as x.
Let $E(e, x) = e \oplus x$ and $D(d, y) = d \oplus y$.
Then $D(d, E(e, x)) = d \oplus (e \oplus x) = x$.
And if John knows x and y, he knows d.

Problems with one-time pad

- How to agree upon the key (i.e. d and e)?
- The keys are too long, and this makes frequent routine communication impossible.

Remarks

- One-time pad is information secure.
- Computer scientists focus on computational secure protocols.

Public-Key Cryptography

Scheme

1. Bob: generates (e, d) and announces e.
   (d is kept secretly by Bob himself.)

2. Alice: sends a message x to Bob by computing and transmitting y where $y = E(e, x)$.

3. Bob: gets x by computing $D(d, y)$.

Requirements

- It is computationally infeasible to deduce d from e and x from y without knowing d.
- $E(e, x)$ and $D(d, y)$ can be computed in polynomial time.
- $x = D(d, E(e, x))$.

One-Way Function

f: a function from strings to string with

1. f is one-to-one;
2. for all x, $|x|^{\frac{1}{k}} \leq |f(x)| \leq |x|^k$ for some $k > 0$;
3. f can be computed in polynomial time;
4. there is no polynomial-time algorithm that computes x from $y = f(x)$ or returns "no" of no such an x exists. (or a stronger version requires no polynomial fraction of)

Remark

We still not yet know the existence of true one-way functions.

Integer multiplication

$$f_{\mathtt{MULT}}(p, C(p), q, C(q)) = \begin{cases} pq & \text{if Condition (1) holds} \\ (q, C(q), q, C(q)) & \text{otherwise} \end{cases}$$

Condition (1): $C(p)$ and $C(q)$ are valid primality certifcates

Factoring the products of two primes is believed to be difficult.

Exponentiation modulo a prime

$$f_{\texttt{EXP}}(p, C(p), r, x) = (p, C(p), r^x \mod p)$$

where r is a primitive root modulo p, and it is included in the certificate $C(p)$.

The inverse of $f_{\texttt{EXP}}$ is the famous problem to evaluate the discrete logarithm, which is also believed to be very hard.

RSA

A (believed) realization of a public-key cryptosystem provided by Ron Rivest, Adi Shamir, and Len Adleman

Idea

1. Let p, q be two primes. Then
$$x^{\phi(pq)+1} \equiv x \pmod{pq}.$$
That is, $x^e \mod pq$ is invertible whenever $e \perp \phi(pq)$.
2. Let $ed \equiv 1 \pmod{\phi(pq)}$. That is, $ed = 1 + k\phi(pq)$. Then
$$(x^e)^d = x^{ed} = x^{1+k\phi(pq)} \equiv x \pmod{pq}.$$

Scheme

1. Find primes p and q.
2. Let $N = pq$. Then $\phi(N) = pq - p - q + 1$.
3. Find $e \perp \phi(N)$. Then there is d such that $ed \equiv 1 \pmod{\phi(N)}$.
4. Make (N, e) public.
5. Define

$$E(e, N, x) = x^e \mod N$$
$$D(d, N, y) = y^d \mod N$$

Each one keeps a private key d and announces the public key e and the modulus N.
Then

$$(x^e)^d \equiv x \pmod{N}.$$

The RSA function

$$f_{\texttt{RSA}}(x, e, p, C(p), q, C(q)) = (x^e \mod pq, pq, e)$$

whenever $e \perp pq$ and $C(p)$ and $C(q)$ are primality certificates for p and q.

Remarks

- Once we can factor pq, we can recover d from $\phi(pq)$.
  $\Longrightarrow$ Inverting $f_{\texttt{RSA}}$ can be reduced to inverting $f_{\texttt{MULT}}$.
- There are variants of the cryptosystem that are as hard as factoring the product of two primes.

Cryptography and Complexity

UP : Unambiguous non-deterministic Polynomial time
A language is in UP iff it can be decided by a non-deterministic Turing machine such that for any input x there is at most one accepting computation.
Clearly, $P \subseteq UP \subseteq NP$.

Theorem UP=P if and only if there are no one-way functions.

Remark The notion of worst-case performance of algorithms is inadequate for approaching the issue of secure cryptography.

Trapdoor Function

Randomized Cryptography

How to transmit a frequent message? Such as one bit $b \in \{0, 1\}$?

1. Generate an random number $x \leq \frac{pq}{2}$.
2. Transmit $y = (2x + b)^e \mod pq$.

Remark

The last bit of an integer is exactly as secure as the RSA public-key cryptosystem.

Protocols

- Signatures
- Mental Poker
- Zero Knowledge

Signature

It should

- contain the information of the original message;
- be modified in a way that unmistakably identifies the sender.

Protocol

$$S(x) = (x, x^d \mod pq) = (x, y)$$

And one who wants to verify the signature can test if

$$y^e \mod pq = x.$$

The point is that, one cannot generate y without knowing d.

Mental Poker

How to distribute a deck of cards fairly?

- One card can be distributed to only one player.
- The probability that all players get the same card are the same.
- There is no dealer.
- Some cards are more desired than others.
- Each player does not know other players' cards.

Let's consider three numbers $a < b < c$ as the cards, Alice and Bob as the players.
Each player gets one card, and the one who gets the larger number wins.

The protocol:

1. Alice and Bob agree on a large prime p.
2. Each has two secret keys: (e_A, d_A) and (e_B, d_B) such that
$$e_A d_A \equiv e_B d_B \equiv 1 \pmod{p-1}.$$
(This implies $x^{e_A d_A} \equiv x^{e_B d_B} \equiv x \pmod{p}$.)
Alice: $E(e_A, x) = x^{e_A} \mod p$; $D(d_A, y) = y^{e_A} \mod p$
Bob: $E(e_B, x) = x^{e_B} \mod p$; $D(d_B, y) = y^{e_B} \mod p$
3. Alice encodes a, b, c and sends them to Bob in a random order.
4. Bob chooses one number, say x, for Alice.
Alice decodes x and she knows her card.
5. Bob encodes the remaining two numbers, sends then to Alice in random order.
6. Alice chooses one from the two, decodes it by her d_A, and

sends it to Bob (say y).

7. Bob decodes y, and he knows his card.

Interactive Proofs

An interactive proof system (A, B) between Alice and Bob is

1. Alice runs an exponential-time algorithm;
2. Bob runs a poly.-time randomized algorithms;
3. Alice sends $m_{2i-1} = A(x; m_1; \ldots; m_{2i-2})$;
   Bob sends $m_{2i} = B(x; m_1; \ldots; m_{2i-1;r_i})$ where r_i is a random string;
   $i, |r_i|, |m_i| \leq |x|^k$ for some $k > 0$.
4. The last message, which is sent by Bob, $\in$ {"yes", "no"}.

(A, B) decides a language L iff

- $x \in L \Rightarrow x$ accepted by (A, B) with Prob. $\geq 1 - \frac{1}{2^{|x|}}$;
- $x \notin L \Rightarrow x$ accepted by (A', B) with Prob. $\leq \frac{1}{2^{|x|}}$ for any exponential-time algorithm A'.

Theorem NP $\subseteq$ IP, BPP $\subseteq$ IP.

Theorem Graph Non-isomorphism $\in$ IP

Given $x = (G, G')$, determine whether they are non-isomorphic.

Definition $G = (V, E)$ and $G' = (V', E')$ are isomorphic iff there is a bijection π from V to V' such that $(u, v) \in E$ iff $(\pi(u), \pi(v)) \in E'$. (WLOG, we may assume $V = V'$.)

Protocol: ith round

1. Bob:
   (a) generates a random bit b_i;
   (b) generates a graph G_i such that $G_i = G'$ if $b_i = 1$, and $G_i = G$ if $b_i = 0$;
   (c) sends $m_{2i-1} = (G, \pi_i(G_i))$ where π_i is a random permutation on the labels of the vertices.
2. Alice checks whether $(G, \pi_i(G_i))$ are non-isomorphic. If they are, $m_{2i} = 1$, otherwise $m_{2i} = 0$.

Finally, Bob checks if $(b_1, \ldots, b_{|x|})$ is identical to $(m_2, \ldots, m_{2|x|})$. Answer "yes" if it is the case; otherwise answer "no".

Zero Knowledge

Alice wants to convince Bob that she knows something, but she does not like to leak any other information about this except just convincing Bob.

Definition (3-Coloring) : Given a graph. decide whether the nodes can be colored by just three colors such that two adjacent nodes have different colors.

Suppose that Alice's coloring is $\chi : V \mapsto \{00, 01, 11\}$.

Protocol:

1. Alice:
   (a) Generate a random permutation π of the three colors.
   (b) Generate $|V|$ RSA public-private key pairs (p_i, q_i, d_i, e_i) for each node $i \in V$.
   (c) Compute the probabilistic encoding (y_i, y_i') according to $b_i b_i' = \pi(\chi(i))$ for $i \in V$. That is, $y_i = (2x_i + b_i)^{e_i} \mod p_i q_i$ and $y_i' = (2x_i' + b_i')^{e_i} \mod p_i q_i$ where $0 \le x_i, x_i' \le \frac{p_i q_i}{2}$.
   (d) Reveal $(e_i, p_i q_i, y_i, y_i')$ for each node $i \in V$ to Bob.
2. Bob picks at random an edge $(i, j) \in E$.
3. Alice reveals to Bob the private keys d_i and d_j.
4. Bob:
   (a) Compute $b_i = (y_i^{d_i} \mod p_i q_i) \mod 2$, and similarly for

b'_i, b_j, and b'_j.

(b) Check if $b_i b'_i \neq b_j b'_j$.

If Alice intends to cheat Bob, Bob has at least $|E|^{-1}$ prob. to identify this.
Repeat this protocol $k|E|$ times can reduce the prob. of false positive $\leq e^{-k}$.

Remark All problems in NP have zero-knowledge proofs.
(by reduction)