CRYPTOGRAPHY

- Public-key cryptography
- Cryptography and complexity
- Randomized cryptography
- Signatures
- Mental poker
- Interactive proofs
- Zero knowledge

(C. Papadimitriou: *Computational Complexity*, Chapter 12)

© 2007 TKK, Laboratory for Theoretical Computer Science

Cryptography

- Two parties Alice and Bob wish to communicate in the presence of malevolent eavesdroppers.
- Alice and Bob agree on two algorithms E (encoding) and D (decoding) which are assumed to be known to general public.
- Privacy is ensured by two strings $e,d \in \Sigma^*$ (here $\Sigma = \{0,1\}$), the *encoding and decoding key*, respectively.
- If Alice wants to send a message $x \in \Sigma^*$ to Bob, she computes the encrypted message $y = E(e,x)$ and transmits this to Bob over the unreliable channel.
- Bob receives y and computes $D(d,y) = x$.
  (e and d have been carefully selected to make D an inverse of E).

© 2007 TKK, Laboratory for Theoretical Computer Science

Cryptography

- E and D should be polynomial-time algorithms
- There should be no way for an eavesdropper to compute x from y without knowing d.
- A simple solution: *one-time pad*
  - Choose d and e be the same arbitrary string e of length $|x|$.
  - Let both $E(e,x) = e \oplus x$ and $D(e,y) = e \oplus y$ (the exclusive or of the corresponding strings)
  - Now $e \oplus (e \oplus x) = x$ and hence $D(e,E(e,x)) = x$
  - Furthermore, if an eavesdropper could derive x from y, then she or he knows $e = x \oplus y$.
- One-time pads have limited usability:
  (i) How to protect communication agreeing on the keys?
  (ii) Long keys are needed (as long as the messages).

© 2007 TKK, Laboratory for Theoretical Computer Science

A public-key cryptosystem

- Bob generates a pair of keys (d,e) and announces e openly.
  (d is private but e is well-known to Alice and the general public.)
- Alice can send a message x to Bob by transmitting $E(e,x)$.
- Bob can decode the message by $D(d,E(e,x)) = x$.
- It should be computationally infeasible to deduce d from e and x from y without knowing d.
- The difficulty of compromising a public-key cryptosystem rests with the difficulty of guessing x from y.
- Once we have a correct guess x, it can be checked simply by testing $E(e,x) = y$.
- Since x cannot be more than polynomially longer than y, compromising a public-key cryptosystem (given y find x such that $E(e,x) = y$) is a problem in $\mathbf{FNP}$.

© 2007 TKK, Laboratory for Theoretical Computer Science

One-Way Functions

- Secure public-key cryptosystems can exist only if $\mathbf{P} \neq \mathbf{NP}$.
- Even if we assume $\mathbf{P} \neq \mathbf{NP}$, the existence of a secure public-key cryptosystem is not immediate but what is needed is a *one-way function* (for which the inverse is in $\mathbf{FNP} - \mathbf{FP}$).

Definition. Let f a function from strings to strings. We say that f is a one-way function if the following holds:
(i) f is one-to-one and for all $x \in \Sigma^*$, $|x|^{\frac{1}{k}} \leq |f(x)| \leq |x|^k$.
(ii) f is in $\mathbf{FP}$.
(iii) The inverse f^{-1} of f is not in $\mathbf{FP}$.

- Notice that f^{-1} is in $\mathbf{FNP}$.
  ($x = f^{-1}(y)$ if $f(x) = y$ which can be checked in polynomial time).

© 2007 TKK, Laboratory for Theoretical Computer Science

Candidates for One-Way Functions

- *Integer multiplication* $f_{\mathrm{MULT}}(p, C(p), q, C(q)) = pq$
  where $p < q$ are prime numbers and $C(p), C(q)$ are their primality certificates.
  (i) f_{MULT} is one-to-one, (ii) polynomial-time computable and (iii) we know of no polynomial-time algorithm which inverts f_{MULT}, i.e., factors products of large primes.
- *Exponentiation modulo a prime*
  $f_{\mathrm{EXP}}(p, C(p), r, x) = (p, C(p), r^x \bmod p)$
  where p is a prime number, $C(p)$ is its primality certificate, r is a primitive root modulo p ($r^{p-1} = 1 \bmod p$) and integer $x < p$.
  No polynomial-time algorithm inverting f_{EXP} is known (the *discrete logarithm problem*).

© 2007 TKK, Laboratory for Theoretical Computer Science

The RSA function

- f_{MULT} and f_{EXP} are not directly usable as the basis of a public-key cryptosystem but their combination is.
- The RSA function $f_{\mathrm{RSA}}(x, e, p, C(p), q, C(q)) = (x^e \bmod pq, pq, e)$ where $p < q$ are prime numbers, $C(p), C(q)$ are their primality certificates, e is a relative prime to $\phi(pq) = pq - p - q + 1$ (Euler function) and integer $x < pq$.
- f_{RSA} is one-to-one.
- f_{RSA} is polynomial-time computable.
- No polynomial-time algorithm for inverting f_{RSA} has been announced.

© 2007 TKK, Laboratory for Theoretical Computer Science

The RSA public-key cryptosystem

- Bob knows p, q and announces the *public encryption key* (pq, e) where e is a relative prime to $\phi(pq) = pq - p - q + 1$.
- Alice encrypts a message x by $y = x^e \bmod pq$
- Bob knows an integer d which is another residue modulo pq such that $ed = 1 + k\phi(pq)$ for some integer k (can be computed by Euclid's algorithm) and the *private decryption key* is (pq, d).
- Bob decrypts a message y by
  $y^d = x^{ed} = x^{1+k\phi(pq)} = x(x^{\phi(pq)})^k = x \bmod pq$
  (where $x^{\phi(pq)} = 1 \bmod pq$)
- Notice that any algorithm that factors integers can be used to invert f_{RSA}: if we know p and q, then we can compute $\phi(pq) = pq - p - q + 1$ and from it and e we could recover d (by Euclid's algorithm).

© 2007 TKK, Laboratory for Theoretical Computer Science

Cryptography and complexity

- Linking the existence of one-way functions and **NP**-completeness?
- **UP** is a class closer to one-way functions:

  Definition. A nondeterministic Turing machine is called unambiguous if for any input x there is at most one accepting computation. **UP** is the class of languages accepted by unambiguous polynomial-time nondeterministic Turing machines.
- $\mathbf{P} \subseteq \mathbf{UP} \subseteq \mathbf{NP}$.

Theorem. $\mathbf{UP} = \mathbf{P}$ iff there are no one-way functions.

© 2007 TKK, Laboratory for Theoretical Computer Science

Cryptography and complexity

- We expect that $\mathbf{P} \neq \mathbf{UP}$ and $\mathbf{UP} \neq \mathbf{NP}$.
- **NP**-completeness is not useful in identifying one-way functions.
- **UP**-completeness does not seem to be useful either:

  UP is a semantical class with no known complete problems.
- Moreover, complexity theory does not seem to be the right tool for analyzing the security of cryptosystems because it is based on worst-case performance estimates:

  Even if we could show that compromising a cryptosystem is a hard computational problem in the worst case, this is not enough for the security of a cryptosystem.

© 2007 TKK, Laboratory for Theoretical Computer Science

Cryptography and complexity

- For the security it is unacceptable if an eavesdropper can easily decode half of the possible messages easily (even if decoding is very hard in the worst case).
- For the definition of a one-way function we need replace requirement (iii) f^{-1} not in **FP** by a stronger requirement:

  There is no polynomial-time algorithm for inverting f on a polynomial fraction of the inputs of length n.
- Often even this is not strong enough because it assumes that a deterministic algorithm is used but randomized algorithms should be allowed and even non-uniform families of circuits.
- In practice, an attack on a cryptosystem could focus only on the currently used key size and invest massive amounts of computation for constructing a circuit that works for the key size.

© 2007 TKK, Laboratory for Theoretical Computer Science

Trapdoor functions

- Moreover, not all one-functions are usable for cryptographic purposes.
- In addition to properties (i–iii) in the definition of a one-way function a couple of further requirements need to be satisfied:

  (iv) We can sample the domain of the function f efficiently (find efficiently arguments for which the function is "defined").

  (v) There is a polynomially computable function d of the input of f that makes the inversion problem computationally easy.
- One-function satisfying the two additional requirements are called trapdoor functions.
- f_{RSA} is a trapdoor function (with the necessary reservation about property (iii)).

© 2007 TKK, Laboratory for Theoretical Computer Science

Randomized cryptography

- Doubts on the security remain even if very strong one-way functions are used.
- For example, if f can be inverted only on a few strings, this could be a serious threat if the strings are important ones ("ATTACK NOW", "SELL NOKIA").
- An important case is to be able to send a single confidential bit $b \in \{0,1\}$.
- In RSA encoding a bit b is $b^e = b$ and the encrypted message would be the same as the original one!

© 2007 TKK, Laboratory for Theoretical Computer Science

Randomized cryptography

- A remedy for this problem is a *randomized public-key cryptosystem*:
- For sending a bit b, generate a random integer $x \leq \frac{pq}{2}$ and then transmit $y = (2x+b)^e \bmod pq$.
- After receiving, decode the original message $(2x+b)$ and b can be read as the least significant bit of the decrypted integer.
- A longer message can be broken to a sequence of bits which could be sent individually as above.

© 2007 TKK, Laboratory for Theoretical Computer Science

Signatures

- Problem: Alice wants to send Bob a signed document x, i.e., a signed message $\mathrm{S}_{\mathrm{Alice}}(x)$ that contains x and identifies unmistakably the sender.
- A solution: use a public-key cryptosystem

  Alice and Bob have public and private keys: $e_{\mathrm{Alice}}, d_{\mathrm{Alice}}, e_{\mathrm{Bob}}, d_{\mathrm{Bob}}$
  (i) Alice sends: $\mathrm{S}_{\mathrm{Alice}}(x) = (x, D(d_{\mathrm{Alice}}, x))$
  (ii) Bob takes the second part and "decodes" it:
  $E(e_{\mathrm{Alice}}, D(d_{\mathrm{Alice}}, x)) = D(d_{\mathrm{Alice}}, E(e_{\mathrm{Alice}}, x)) = x$
  (This works for commutative systems like RSA)
  (iii) If decoding equals to the first part (x), Bob accepts x.

© 2007 TKK, Laboratory for Theoretical Computer Science

Mental Poker

- Problem: Alice and Bob have agreed upon three n-bit numbers $a < b < c$ (cards). They need to randomly choose one card each such that:
  (i) Their cards are different.
  (ii) All six pairs of distinct cards are equiprobable as outcomes.
  (iii) Alice's card is known to Alice but not to Bob and similarly for Bob.
  (iv) The outcome should be indisputable.
- A solution: The players agree on a single large prime p.
  Each player has two *secret* keys: $e_{\mathrm{Alice}}, d_{\mathrm{Alice}}, e_{\mathrm{Bob}}, d_{\mathrm{Bob}}$ such that $e_{\mathrm{Alice}} d_{\mathrm{Alice}} = e_{\mathrm{Bob}} d_{\mathrm{Bob}} = 1 \bmod p-1$.
  Now, e.g., $e_{\mathrm{Alice}} d_{\mathrm{Alice}} = k(p-1)+1$ and hence,
  $x^{e_{\mathrm{Alice}} d_{\mathrm{Alice}}} = x(x^{p-1})^k = x \bmod p$ (by Fermat's theorem)

© 2007 TKK, Laboratory for Theoretical Computer Science

Mental Poker: the protocol

- Alice encrypts the three cards and sends to Bob the encrypted messages $a^{e_{\text{Alice}}} \bmod p, b^{e_{\text{Alice}}} \bmod p, c^{e_{\text{Alice}}} \bmod p$ in some random order.
- Bob picks one, say $a^{e_{\text{Alice}}} \bmod p$, and sends it to Alice who decrypts it with d_{Alice} and keeps as her card).
- Bob encrypts the two remaining cards $b^{e_{\text{Alice}} e_{\text{Bob}}} \bmod p, c^{e_{\text{Alice}} e_{\text{Bob}}} \bmod p$ and sends them to Alice in some random order.
- Alice picks one, say $b^{e_{\text{Alice}} e_{\text{Bob}}} \bmod p$, decodes it with d_{Alice} and sends $b^{e_{\text{Alice}} e_{\text{Bob}} d_{\text{Alice}}} \bmod p$ to Bob.
- Bob decrypts this with d_{Bob} and takes as his card.

☞ Conditions (i–iv) satisfied.

© 2007 TKK, Laboratory for Theoretical Computer Science

Interactive Proofs

- A nondeterministic algorithm can be seen as a simple protocol: Alice has exponential computing power (to find a good certificate) and Bob has polynomial (to check the certificate).
- What languages can be accepted if Bob can use randomization?

Definition. An interactive proof system (A,B) is a protocol between Alice and Bob. Alice runs an exponential time algorithm A while Bob has a polynomial-time randomized algorithm B.
The input x is known to both algorithms.
The two exchange a sequence of messages $m_1, m_2, \ldots, m_{2|x|^k}$ where Alice sends the odd-numbered ones and Bob even-numbered and $|m_i| \le |x|^k$ for all i.

© 2007 TKK, Laboratory for Theoretical Computer Science

The protocol

- $m_1 = A(x)$.
- For all $i \le 2|x|^k$, $m_{2i} = B(x; m_1; \ldots; m_{2i-1}, r_i)$ and $m_{2i+1} = A(x; m_1; \ldots; m_{2i})$ where r_i is the polynomially long random string used by Bob at the ith stage (r_i is not known to A).
- For the last message $m_{2|x|^k} \in \{\text{“yes”}, \text{“no”}\}$ signaling accept/reject.
- (A,B) decides a language L iff for all strings x,
  (i) if $x \in L$, then the probability that x is accepted by (A,B) is at least $1 - \frac{1}{2^{|x|}}$.
  (ii) if $x \notin L$, then the probability that x is accepted by (A',B) is at most $\frac{1}{2^{|x|}}$ for any exponential algorithm A' replacing A.

IP is the class of languages decided by interactive proof systems.

© 2007 TKK, Laboratory for Theoretical Computer Science

Interactive Proofs

- $\mathbf{NP} \subseteq \mathbf{IP}$
- $\mathbf{BPP} \subseteq \mathbf{IP}$
- GRAPH NONISOMORPHISM $\in \mathbf{IP}$ (not known to be in $\mathbf{NP}/\mathbf{BPP}$)
- GRAPH ISOMORPHISM $\in \mathbf{NP}$ (not known to be **NP**-complete or in **P**)
- An interactive proof system deciding GRAPH NONISOMORPHISM:
  Bob: on input $x = (G, G')$ repeats for $i = 1, \ldots, |x|$ rounds:
  Chooses random bit b_i and if $b_i = 1$ then $G_i = G$ else $G_i = G'$;
  generates a random permutation π_i and sends $m_{2i-1} = (G, \pi(G_i))$
  Alice: checks whether the two graphs received are isomorphic. If they are $m_{2i} = 1$ else $m_{2i} = 0$
  After $|x|$ rounds Bob accepts if random bits $b_1, \ldots, b_{|x|}$ and Alice's replies $m_2, \ldots, m_{2|x|}$ are identical.

© 2007 TKK, Laboratory for Theoretical Computer Science

Interactive Proofs

- If G and G' are not isomorphic, then x is accepted since $b_i = 1$ iff m_{2i-1} contains two isomorphic graphs iff $m_{2i} = 1$.
- If G and G' are isomorphic, then Alice always sees a graph and its permuted copy and from that she needs to guess a random bit $|x|$ times:

  perfect success with probability $\frac{1}{2^{|x|}}$

© 2007 TKK, Laboratory for Theoretical Computer Science

Zero knowledge

- Problem: Find an interactive protocol (zero-knowledge proof) such that given a graph (V, E), in the end Bob is convinced that with very high probability Alice has a legal 3-coloring of the graph but Bob has no clue about the actual 3-coloring.
- Protocol: Suppose Alice's coloring is $\chi : V \mapsto \{00, 11, 01\}$
  For each round:
  Alice: generates a random permutation π of the colors;
  generates a RSA public-private keys (p_i, q_i, e_i, d_i) for each node i;
  for each node i, computes (y_i, y_i'), a RSA coding of $\pi(\chi(i))$;
  reveals $(e_i, p_i q_i, y_i, y_i')$ for each node $i \in V$ to Bob.
  Bob picks at random an edge $(i, j) \in E$ and Alice reveals d_i, d_j
  Bob: decodes y_i, y_i' and y_j, y_j' to obtain colors $\pi(\chi(i))$ and $\pi(\chi(j))$ and checks that they are different.

© 2007 TKK, Laboratory for Theoretical Computer Science

Zero knowledge

- If Alice has no legal coloring, Bob has at least $\frac{1}{|E|}$ probability of finding an edge $(i, j) \in E$ such that $\chi(i) = \chi(j)$.
- If this is repeated $k|E|$ times, the probability for Bob to find out that Alice has no legal coloring is at least $1 - e^{-k}$.
- Note: Bob has learned nothing about the coloring.

  Bob sees random public keys, encryptions of colors, and colors $\pi(\chi(i))$ and $\pi(\chi(j))$ but these are randomly chosen pairs of different colors (π changes after each round).
- Zero knowledge: the interactions in the protocol form a random string drawn from a distribution that was available in the beginning.
- All problems in **NP** have zero-knowledge proofs.

© 2007 TKK, Laboratory for Theoretical Computer Science

Learning Objectives

- The concepts of a public-key cryptosystem and a one-way function and their relationship
- The role of complexity theory in analyzing the security of cryptosystems
- Examples of cryptographic protocols: signatures, mental poker, interactive proofs, zero knowledge

© 2007 TKK, Laboratory for Theoretical Computer Science