Cryptography

Complexity Theory

Faculty of Information Technology
Brno University of Technology
Brno, Czech Republic

Ondřej Lengál

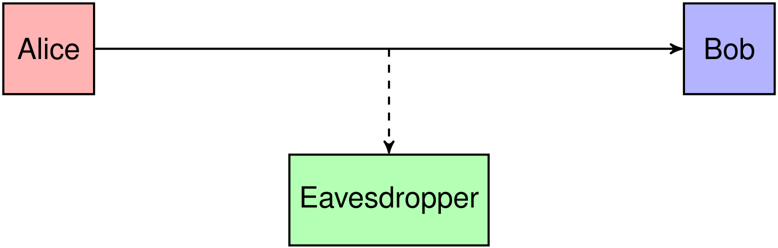
This material was created with the support of the Czech Ministry of Education, Youth and Sports (project FRVŠ 166/2013/G1).

Motivation

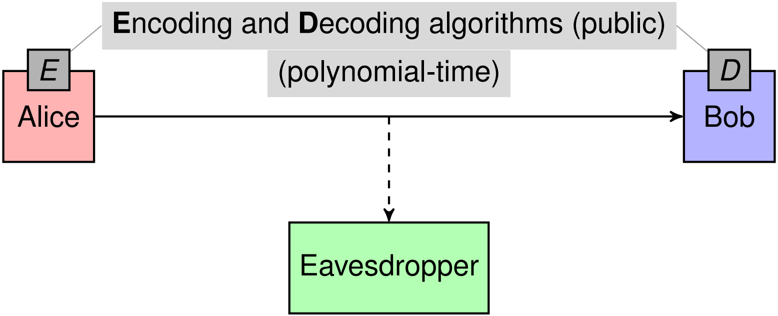
- Hardness of problems is not always bad ...
- ...sometimes, it is a resource to be exploited!
- We wish to find problems that are quickly solvable with a partial knowledge of the solution, but very hard without it (including approximation/probabilistic algorithms).
- We will look at cryptography from the complexity's point of view. For history, side channel attacks, etc., refer to the KRY class.

Note: in this lecture we fix $\Sigma = \{0, 1\}$.

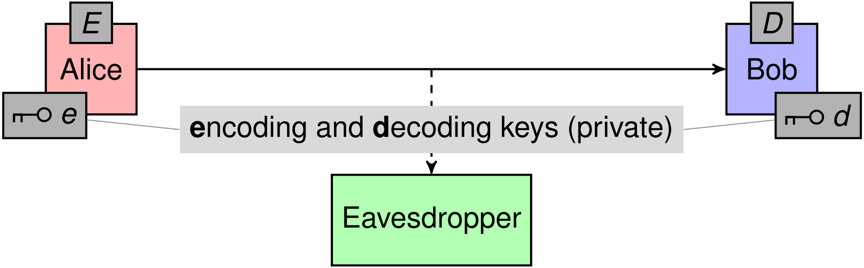
General Setting



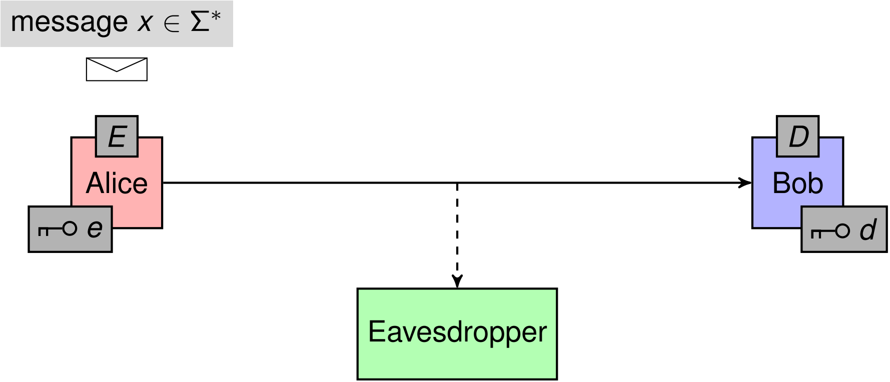
General Setting



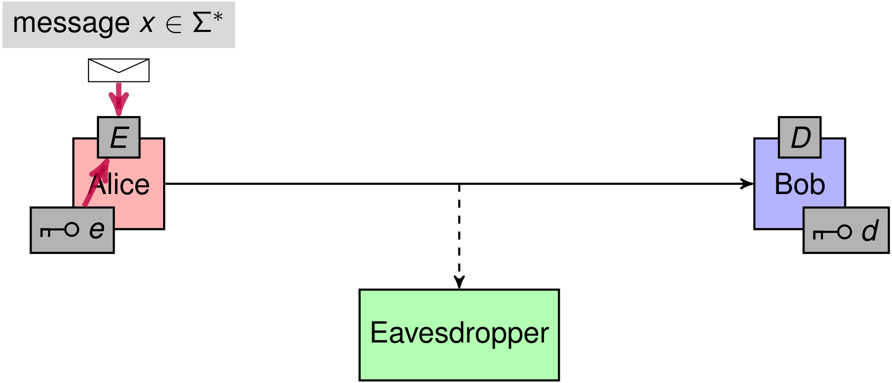
General Setting



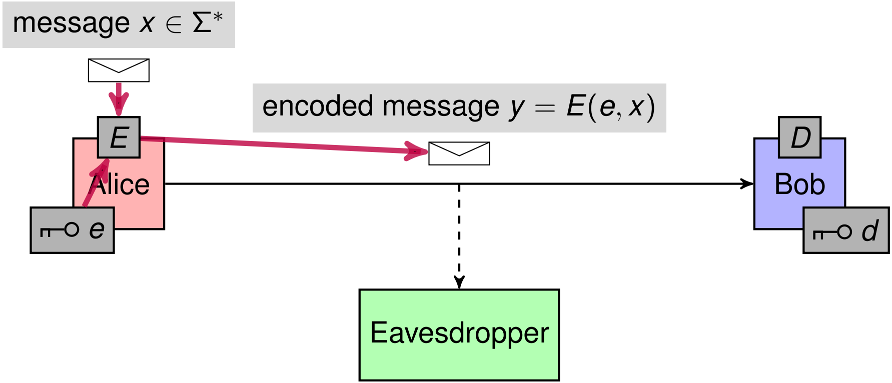
General Setting



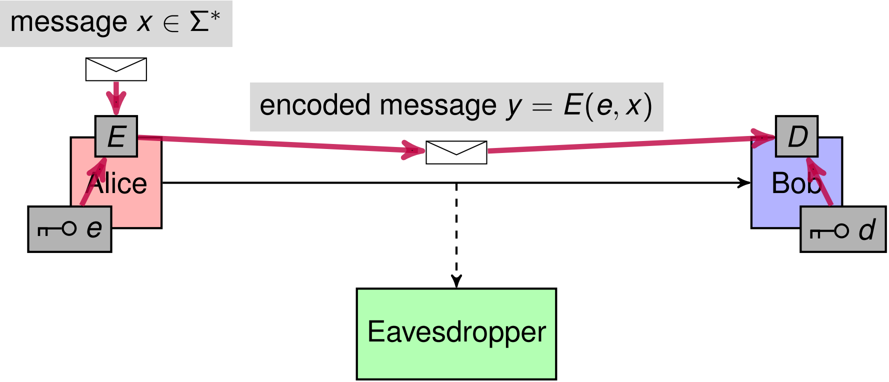
General Setting



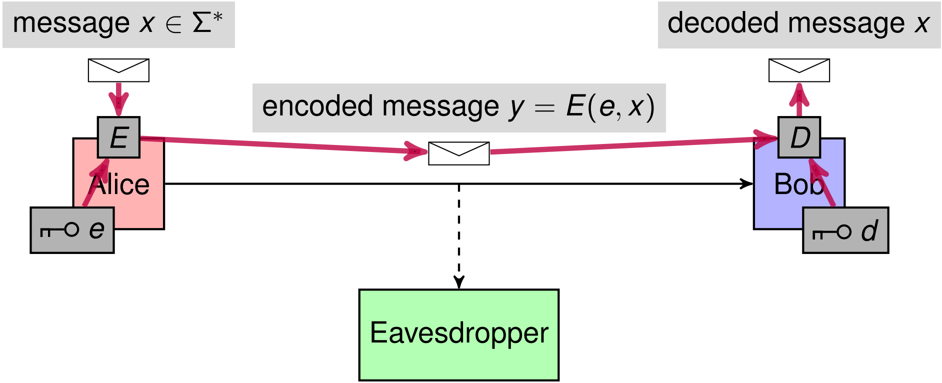
General Setting



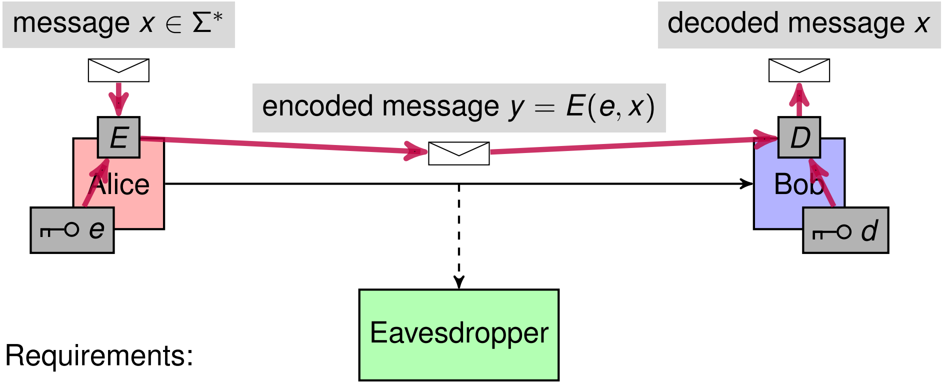
General Setting



General Setting



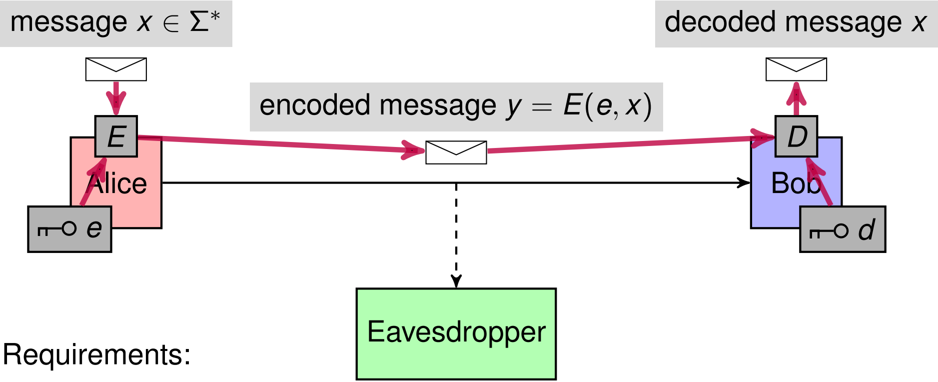
General Setting



Requirements:

1. $x = D(d, E(e, x))$

General Setting



Requirements:

1. $x = D(d, E(e, x))$
2. Eavesdropper not able to compute x from y without knowing d

One-Time Pad

Example (one-time pad):

- let $e = d$ be a string $w \in \Sigma^*$ of length $|x|$ and

$$D = E = \lambda\, a\, b\, .\, a \oplus b$$

One-Time Pad

Example (one-time pad):

- let $e = d$ be a string $w \in \Sigma^*$ of length $|x|$ and

$$D = E = \lambda\, a\, b\, .\, a \oplus b$$

The requirements hold:

1.
   - $y = E(w, x) = w \oplus x$
   - $D(w, y) = w \oplus (w \oplus x) = (w \oplus w) \oplus x = 0^{|x|} \oplus x = x$

One-Time Pad

Example (one-time pad):

- let $e = d$ be a string $w \in \Sigma^*$ of length $|x|$ and

$$D = E = \lambda\, a\, b\, .\, a \oplus b$$

The requirements hold:

1. - $y = E(w, x) = w \oplus x$
   - $D(w, y) = w \oplus (w \oplus x) = (w \oplus w) \oplus x = 0^{|x|} \oplus x = x$
2. If Eavesdropper could derive x from y, then she knows $d = x \oplus y$.

One-Time Pad

Example (one-time pad):

- let $e = d$ be a string $w \in \Sigma^*$ of length $|x|$ and

$$D = E = \lambda\, a\, b\, .\, a \oplus b$$

The requirements hold:

1.
   - $y = E(w, x) = w \oplus x$
   - $D(w, y) = w \oplus (w \oplus x) = (w \oplus w) \oplus x = 0^{|x|} \oplus x = x$
2. If Eavesdropper could derive x from y, then she knows $d = x \oplus y$.

Issues:

One-Time Pad

Example (one-time pad):

- let $e = d$ be a string $w \in \Sigma^*$ of length $|x|$ and

$$D = E = \lambda\, a\, b\, .\, a \oplus b$$

The requirements hold:

1.
   - $y = E(w, x) = w \oplus x$
   - $D(w, y) = w \oplus (w \oplus x) = (w \oplus w) \oplus x = 0^{|x|} \oplus x = x$
2. If Eavesdropper could derive x from y, then she knows $d = x \oplus y$.

Issues:

1. w is usable only once
   - suppose $y_1 = E(w, x_1)$, $y_2 = E(w, x_2)$
   - then Eavesdropper may obtain $y_1 \oplus y_2$ (and use it for an attack)

One-Time Pad

Example (one-time pad):

- let $e = d$ be a string $w \in \Sigma^*$ of length $|x|$ and

$$D = E = \lambda\, a\, b\, .\, a \oplus b$$

The requirements hold:

1.
   - $y = E(w, x) = w \oplus x$
   - $D(w, y) = w \oplus (w \oplus x) = (w \oplus w) \oplus x = 0^{|x|} \oplus x = x$
2. If Eavesdropper could derive x from y, then she knows $d = x \oplus y$.

Issues:

1. w is usable only once
   - suppose $y_1 = E(w, x_1)$, $y_2 = E(w, x_2)$
   - then Eavesdropper may obtain $y_1 \oplus y_2$ (and use it for an attack)
2. distribution of keys to the parties

Public-Key Cryptography

Public-key Cryptosystem

- d — secret and private for Bob,
- e — public,
- it is computationally infeasible to deduce d from e, and x from y without knowing d

Issues:

- when guessing x, it is easy to check whether $x \stackrel{?}{=} D(d, y)$ by checking whether $y = E(e, x)$
- and since $|x| \leq |y|^k$ for some $k > 0$, compromising it is in **FNP**,
- $\implies$ public-key cryptosystems exists only if $\mathbf{P} \neq \mathbf{NP}$.

... one-way functions (inhabitants of $\mathbf{FNP} \setminus \mathbf{FP}$)

One-way Functions

A function $f : \Sigma^* \to \Sigma^*$ is one-way if:

1. f is injective and $\forall x \in \Sigma^*, |x|^{\frac{1}{k}} \leq |f(x)| \leq |x|^k$ for some $k > 0$,
2. $f \in \mathbf{FP}$,
3. $f^{-1} \notin \mathbf{FP}$ (and therefore $f^{-1} \in \mathbf{FNP} \setminus \mathbf{FP}$).

If there exist one-way functions, then $\mathbf{P} \neq \mathbf{NP}$.

RSA

The RSA function:

- Proposed by Ron **R**ivest, Adi **S**hamir, and Leonard **A**dleman.
- Uses integer multiplication and exponentiation modulo a prime.
- $p, q \quad \ldots$ two large primes (private), their product pq (public)
- $1 < e < \phi(pq) \quad \ldots$ an integer coprime with $\phi(pq)$ (public)
  - $\phi(pq) = pq(1 - \frac{1}{p})(1 - \frac{1}{q}) = pq - p - q + 1$ Euler's totient function
- $d \quad \ldots$ an integer s.t. $\quad e \cdot d \equiv 1 \mod \phi(pq)$ (private)
- $E = \lambda\, x \,.\, x^e \mod pq$
- $D = \lambda\, y \,.\, y^d \quad (= (x^e)^d = x^{e \cdot d} = x^{1+k\phi(pq)} = x \mod pq)$
  - if $1 \leq x < pq$ and x and pq are coprime, then $x^{\phi(pq)} = 1 \mod pq$ Euler's totient theorem (generalization of Fermat's little theorem)
- fast factoring can break RSA (p, q, and e can be used to get d)

UP

Definition (UP)

UP is the class of languages accepted by unambiguous polynomial-time bounded nondeterministic Turing machines.

- Unambiguous NTM: for any input there is at most 1 accepting run.
- Obviously, $\mathbf{P} \subseteq \mathbf{UP} \subseteq \mathbf{NP}$.
- It is believed that $\mathbf{UP} \neq \mathbf{NP}$.

UP

Theorem

UP $\neq$ **P** *if and only if there exist one-way functions.*

UP

Theorem

UP $\neq$ **P** *if and only if there exist one-way functions.*

Proof (idea).

"$\Leftarrow$":

- Suppose there is a one-way function f.
- Consider the language $L_f = \{(x, y) \mid \exists z \in \Sigma^* \, . \, f(z) = y \wedge z \leq x\}$.
  (words over Σ ordered first by length and then lexicographically)

UP

Theorem

UP $\neq$ **P** *if and only if there exist one-way functions.*

Proof (idea).

"$\Leftarrow$":

- Suppose there is a one-way function f.
- Consider the language $L_f = \{(x, y) \mid \exists z \in \Sigma^* \,.\, f(z) = y \wedge z \leq x\}$.
  (words over Σ ordered first by length and then lexicographically)
- $L_f \in$ **UP**: a TM M for the input (x, y) guesses z and computes whether $y = f(z)$; if yes and $z \leq x$, M accepts, otherwise rejects

f being injective implies this happens at most once

UP

Theorem

UP $\neq$ **P** *if and only if there exist one-way functions.*

Proof (idea).

"$\Leftarrow$":

- Suppose there is a one-way function f.
- Consider the language $L_f = \{(x, y) \mid \exists z \in \Sigma^* \,.\, f(z) = y \wedge z \leq x\}$. (words over Σ ordered first by length and then lexicographically)
- $L_f \in$ **UP**: a TM M for the input (x, y) guesses z and computes whether $y = f(z)$; if yes and $z \leq x$, M accepts, otherwise rejects

  f being injective implies this happens at most once
- $L_f \notin$ **P**: if there were a **PTIME** algorithm for L_f, we could invert f in **PTIME** using binary search $\implies$ f would not be one-way
- therefore, **P** $\subset$ **UP** (because $L_f \in$ **UP** $\setminus$ **P**)

UP

"$\Rightarrow$":

- Suppose there is a language $L \in \textbf{UP} \setminus \textbf{P}$.
- Let U be an unambiguous TM accepting L.
- Let x be an encoding of an accepting computation of U on input y.
- Define $f_U(x) = 1y$ and $f_U(z) = 0z$ if z is not such an encoding.
- f_U is one-way, because
  - f_U is well-defined (y can be "read off" x in **PTIME**),
  - lengths of x and $f_U(x)$ are polynomially related,
  - f_U is injective ($f(x) = f(x') \implies x = x'$),
  - inverting f_U in **PTIME** would imply $L \in \textbf{P}$. $\square$

One-way Functions Revisited

Worst-case performance of algorithms

- not a good concept for cryptography!
- hard problems need to be densely populating the problem space,
- we need to refine the requirement for one-way functions:

> 3 $f^{-1} \notin$ **FP** (and therefore $f^{-1} \in$ **FNP** $\setminus$ **FP**).

to a stronger requirement:

> 3 there is no $k \in \mathbb{N}$, and no algorithm which, for large enough n, in time $\mathcal{O}(n^k)$ successfully computes $f^{-1}(y)$ for at least $\frac{2^n}{n^k}$ strings of length n.

- i.e. there is no **PTIME** algorithm that successfully inverts f on a polynomial fraction of the inputs of length n.

Randomized Cryptography

- Suppose Alice needs repeatedly send Bob a single bit $b \in \{0, 1\}$.

Randomized Cryptography

- Suppose Alice needs repeatedly send Bob a single bit $b \in \{0, 1\}$.
- Issue: $b^e = b$ for $b \in \{0, 1\}$!

Randomized Cryptography

- Suppose Alice needs repeatedly send Bob a single bit $b \in \{0, 1\}$.
- Issue: $b^e = b$ for $b \in \{0, 1\}$!
- Remedy: Alice generates a random integer $0 \leq x \leq \frac{pq}{2}$ and transmits to Bob $y = (2x + b)^e \mod pq$.

Randomized Cryptography

- Suppose Alice needs repeatedly send Bob a single bit $b \in \{0, 1\}$.
- Issue: $b^e = b$ for $b \in \{0, 1\}$!
- Remedy: Alice generates a random integer $0 \leq x \leq \frac{pq}{2}$ and transmits to Bob $y = (2x + b)^e \mod pq$.
- Note: any message can be split into bits and send using this scheme. This avoids the problems of repetition, guessing messages, etc.

Protocols

Signature

- modification of a document that unmistakably identifies the sender,
- commutative public-key cryptosystems can be exploited:
- Alice sends a signed message

$E(e) \circ D(d) = D(d) \circ E(e) = \mathrm{id}$

$$S_{Alice}(x) = (x, D(d_{Alice}, x))$$

private

  - i.e. Alice sends the original message with its decoded counterpart
- given a signed message (x, s) anyone can check whether

$$E(e_{Alice}, s) = x$$

public

  - i.e. check that the signature is valid
- the RSA cryptosystem can be used.

Protocols

Mental Poker

- 3 n-bit numbers $a < b < c$ (cards)
- Alice and Bob to randomly choose one card each, such that:
  1. their cards are different,
  2. all 6 allowed outcomes have the same probability,
  3. Alice's (B's) card is known only to Alice (B) until she announces it,
  4. the outcome is indisputable.
- The person with the highest number wins.

Protocols

Mental Poker — a solution:

1. The players agree on a single large prime number p.
2. Each player generates two secret keys:
   - an encryption key e_{Alice}, e_{Bob},
   - a decryption key d_{Alice}, d_{Bob},
   - such that $e_{Alice}d_{Alice} = e_{Bob}d_{Bob} = 1 \mod p - 1$.
3. Alice encrypts and sends to Bob $a^{e_{Alice}}, b^{e_{Alice}}, c^{e_{Alice}} \pmod p$.
4. Bob randomly chooses one message, say $b^{e_{Alice}}$, and returns it to Alice to be her card (Alice decodes it with d_{Alice} to obtain b).
5. Bob encrypts and sends to Alice $a^{e_{Alice}e_{Bob}}, c^{e_{Alice}e_{Bob}} \pmod p$.
6. Alice randomly chooses one message, say $a^{e_{Alice}e_{Bob}}$, decodes it with d_{Alice} and sends $a^{e_{Bob}} \mod p$ to Bob as his card.

Protocols

Zero Knowledge Example: consider the problem of 3-COLOURING of a graph $G = (V, E)$. Suppose Alice knows the colouring $\chi : V \to \{00, 11, 01\}$ and wants to persuade Bob of the fact, without revealing χ to him.

Protocols

Zero Knowledge Example: consider the problem of 3-COLOURING of a graph $G = (V, E)$. Suppose Alice knows the colouring $\chi : V \to \{00, 11, 01\}$ and wants to persuade Bob of the fact, without revealing χ to him.

A multiple round protocol, where in each step

1. Alice generates a random permutation π of the 3 colours.
2. Then she generates an RSA key pair (p_i, q_i, d_i, e_i) for each $i \in V$.
3. For every $i \in V$ she computes the probabilistic encoding (y_i, y_i'), according to the i-th RSA system, of i's new colour $b_i b_i' = \pi(\chi(i))$
4. For every $i \in V$ she sends $(e_i, p_i q_i, y_i, y_i')$ to Bob.
5. Now, Bob picks a random edge $(k, l) \in E$ and Alice reveals the secret keys d_k and d_l of the endpoints.
6. Bob computes $b_k b_k'$ and $b_l b_l'$ and checks that indeed $b_k b_k' \neq b_l b_l'$.

Protocols

- If Alice does not have a legal colouring, then the probability of finding an edge $(k, l) \in E$, s.t. $b_k b'_k = b_l b'_l$, is at least $\frac{1}{|E|}$.
- After $n|E|$ rounds, the probability of Bob finding out Alice has no legal colouring is at least $1 - e^{-n}$.
- But if Alice has a legal colouring, Bob has not learned anything about it.