Hard problems in cryptography
A brief introduction to complexity

Sorina Ionica

Laboratoire MIS, Université de Picardie Jules Verne, France

May 11, 2018

Big-O notation

Let $f : \mathbb{N} \to \mathbb{N}$.

We say that an algorithm has $O(f(n))$ if it runs in less than $cf(n)$ operations, with $c > 0$ a constant.

Complexity classes

- Constant time $O(1)$.
- Polynomial time $O(n)$, $O(n^2)$ etc.
- Logarithmic time $O(\log n)$.
- Exponential time $O(e^n)$.

Exercise : Given a group $(\mathbb{G}, \cdot)$ and $g \in \mathbb{G}$, compute g^d with the naive method. What is the complexity?

Square-and-multiply

Let $(\mathbb{G}, \cdot)$ be a group and $g \in \mathbb{G}$, compute g^d.

Write $d = d_{k-1}2^{k-1} + d_{k-2}2^{k-2} + ... + d_1 2 + d_0$.

Require: g, d
Ensure: $y = g^d$
  $y := g$
  for $i = k - 2$ down to 0 **do**
    $g := g^2$
    if $d_i = 1$ **then** $y := y \times g$
  end for
  Return y

What is the complexity?

Square-and-multiply

Let $(\mathbb{G}, \cdot)$ be a group and $g \in \mathbb{G}$, compute g^d.

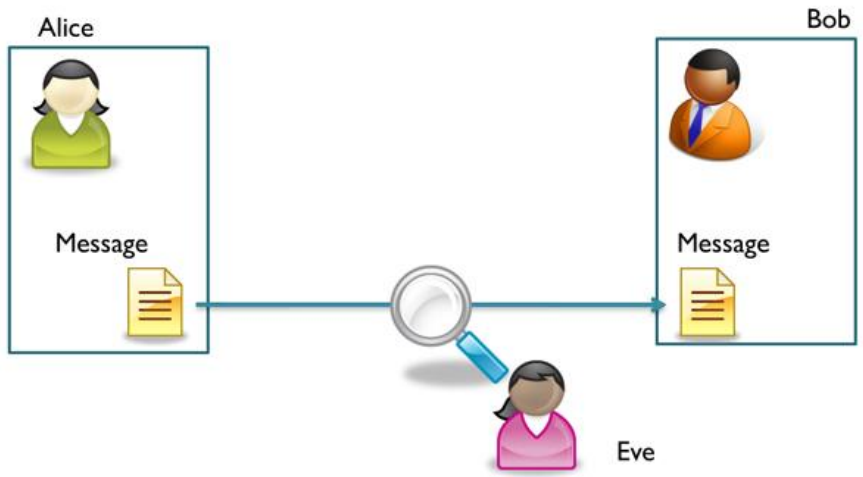
Write $d = d_{k-1}2^{k-1} + d_{k-2}2^{k-2} + \ldots + d_1 2 + d_0$.

Require: g, d
Ensure: $y = g^d$
  $y := g$
  for $i = k - 2$ down to 0 **do**
    $g := g^2$
    if $d_i = 1$ **then** $y := y \times g$
  end for
  Return y

$O(\log d)$

Public key cryptography

Symetric encryption schemes (DES, AES) work well but they require that both parties have à priori exchanged a shared secret key.



How to share a common secret over an insecure channel?

One way functions

One way function "one-way": $f : X \to Y$

- Computable in polynomial time
- Hard to invert :
  Let n be the size of the input x.
  - For any algorithm A running in polynomial time in n and every $\epsilon > 0$, there is a B such that *for almost all* x with $n > B$, we have that

$$Pr(A(f(x)) = x) < \epsilon$$

One-way: a function difficult to invert on average (not in the worst case).

One-way functions from groups

Let $(\mathbb{G}, \cdot)$ be a cyclic group and g a generator.

A function easy to compute : $d \in \mathbb{Z} \to g^d$ $\quad O(\log d)$

Hard to invert: given $h = g^d$ it is hard to compute d. $\quad O(d)$

The discrete logarithm problem (DLP)

Let h be a element of G. Find $d \in \mathbb{Z}$ such that $g^d = h$.

What groups?

Denote by $N = \#G$.

Exhaustive search : one can find the discrete log in time $O(N)$.

The challenge: Find $\mathbb{G}$ such that

- The discrete log is hard.
- Group operations are fast.
- There is a short, compact representation of group elements.

Finite fields

- Let p be a prime number. Then $\mathbb{F}_p := \mathbb{Z}/p\mathbb{Z}$.
- q is a power of some prime p
- $\mathbb{F}_q$ is finite field with q elements

$\mathbb{F}_q \simeq \mathbb{F}_p[x]/f(x)\mathbb{F}_p[x]$ for any irreducible f (with $q = p^{\deg(f)}$)

Subexponential attacks for discrete log in finite field

Subexponential complexity $L(\alpha, x) = e^{(\log x)^{\alpha}(\log\log x)^{1-\alpha}}$, $\alpha \in (0,1)$

- $\mathbb{G} \subset (\mathbb{F}_q^*, \times)$, $\mathrm{char}\ \mathbb{F}_q$ large : subexponential time $O(e^{c(\log q)^{1/3}(\log\log q)^{2/3}})$
- $\mathbb{G} \subset (\mathbb{F}_q^*, \times)$, $\mathrm{char}\ \mathbb{F}_q$ small : subexponential time $O(exp(c'(\log q)^{1/4}(\log\log q)^{3/4})$

Elliptic curves

- 1985 Neal Koblitz and Victor Miller : groups of elliptic curves
- For $\mathbb{G} \subset E(\mathbb{F}_p)$, p prime (and a few extra conditions) $O(\sqrt{p})$.

Another example: RSA and factoring

Let $N = p \cdot q$ with p, q large prime numbers and $(e, \varphi(N)) = 1$.

Assuming factoring is hard, this function is one-way:

$$\begin{aligned} f : \mathbb{Z}_N &\rightarrow \mathbb{Z}_N \\ m &\rightarrow m^e \pmod N. \end{aligned}$$

Subexponential attacks for factoring

- L'algorithme de Dixon, le crible quadratique

$$L_{1/2}(N) = e^{(\log N)^{1/2}(\log\log N)^{1/2}}$$

- Nowadays the Number Field Sieve

$$L_{1/3}(N) = e^{(\log N)^{1/3}(\log\log N)^{2/3}}$$

Complexity classes

- **P** Polynomial time.
- **NP** Non-deterministic polynomial time.
- **NP-complete** A problem in NP such that any other problem in NP has polynomial time reduction to it.

Complexity classes

- **P** Polynomial time.
- **NP** Non-deterministic polynomial time.
- **NP-complete** A problem in NP such that any other problem in NP has polynomial time reduction to it.