COMPUTATIONAL NUMBER THEORY

Secret key exchange

Problem: Obtain a joint secret key via interaction over a public channel:

Alice		**Bob**
$x \xleftarrow{\$} \ldots;\ X \leftarrow \ldots$		
	$\xrightarrow{X}$	
		$y \xleftarrow{\$} \ldots;\ Y \leftarrow \ldots$
	$\xleftarrow{Y}$	
$K_A \leftarrow F_A(x, Y)$		$K_B \leftarrow F_B(y, X)$

Desired properties of the protocol:

- $K_A = K_B$, meaning Alice and Bob agree on a key
- Adversary given X, Y can't compute K_A

Secret Key Exchange

Can you build a secret key exchange protocol?

Secret Key Exchange

Can you build a secret key exchange protocol?

Symmetric cryptography has existed for thousands of years.

But no secret key exchange protocol was found in that time.

Many people thought it was impossible.

Secret Key Exchange

Can you build a secret key exchange protocol?

Symmetric cryptography has existed for thousands of years.

But no secret key exchange protocol was found in that time.

Many people thought it was impossible.

In 1976, Diffie and Hellman proposed one.

This was the birth of public-key (asymmetric) cryptography.

DH Key Exchange Video

`http://www.youtube.com/watch?v=3QnD2c4Xovk`

DH Secret Key Exchange

The following are assumed to be public: A large prime p and a number g called a generator mod p. Let $\mathbf{Z}_{p-1} = \{0, 1, \ldots, p-2\}$.

Alice		**Bob**
$x \xleftarrow{\$} \mathbf{Z}_{p-1}$; $X \leftarrow g^x \bmod p$		
	$\xrightarrow{X}$	
		$y \xleftarrow{\$} \mathbf{Z}_{p-1}$; $Y \leftarrow g^y \bmod p$
	$\xleftarrow{Y}$	
$K_A \leftarrow Y^x \bmod p$		$K_B \leftarrow X^y \bmod p$

- $Y^x = (g^y)^x = g^{xy} = (g^x)^y = X^y$ modulo p, so $K_A = K_B$
- Adversary is faced with computing $g^{xy} \bmod p$ given $g^x \bmod p$ and $g^y \bmod p$, which nobody knows how to do efficiently for large p.

DH Secret Key Exchange: Questions

- How do we pick a large prime p, and how large is large enough?
- What does it mean for g to be a generator modulo p?
- How do we find a generator modulo p?
- How can Alice quickly compute $x \mapsto g^x \bmod p$?
- How can Bob quickly compute $y \mapsto g^y \bmod p$?
- Why is it hard to compute $(g^x \bmod p, g^y \bmod p) \mapsto g^{xy} \bmod p$?
- ...

To answer all that and more, we will forget about DH secret key exchange for a while and take a trip into computational number theory ...

Notation

$\mathbf{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$

$\mathbf{N} = \{0, 1, 2, \ldots\}$

$\mathbf{Z}_+ = \{1, 2, 3, \ldots\}$

For $a, N \in \mathbf{Z}$ let $\gcd(a, N)$ be the largest $d \in \mathbf{Z}_+$ such that d divides both a and N.

Example: $\gcd(30, 70) = 10$.

Integers mod N

For $N \in \mathbf{Z}_+$, let

- $\mathbf{Z}_N = \{0, 1, \ldots, N-1\}$
- $\mathbf{Z}_N^* = \{a \in \mathbf{Z}_N \ : \ \gcd(a, N) = 1\}$
- $\varphi(N) = |\mathbf{Z}_N^*|$

Example: $N = 12$

- $\mathbf{Z}_{12} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$
- $\mathbf{Z}_{12}^* =$

Integers mod N

For $N \in \mathbf{Z}_+$, let

- $\mathbf{Z}_N = \{0, 1, \ldots, N-1\}$
- $\mathbf{Z}_N^* = \{a \in \mathbf{Z}_N \ : \ \gcd(a, N) = 1\}$
- $\varphi(N) = |\mathbf{Z}_N^*|$

Example: $N = 12$

- $\mathbf{Z}_{12} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$
- $\mathbf{Z}_{12}^* = \{1, 5, 7, 11\}$
- $\varphi(12) =$

Integers mod N

For $N \in \mathbf{Z}_+$, let

- $\mathbf{Z}_N = \{0, 1, \ldots, N-1\}$
- $\mathbf{Z}_N^* = \{a \in \mathbf{Z}_N \ : \ \gcd(a, N) = 1\}$
- $\varphi(N) = |\mathbf{Z}_N^*|$

Example: $N = 12$

- $\mathbf{Z}_{12} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$
- $\mathbf{Z}_{12}^* = \{1, 5, 7, 11\}$
- $\varphi(12) = 4$

Division and mod

$\text{INT-DIV}(a, N)$ returns (q, r) such that

- $a = qN + r$
- $0 \leq r < N$

Refer to q as the quotient and r as the remainder. Then

$$a \bmod N = r \in \mathbf{Z}_N$$

is the remainder when a is divided by N.

Example: $\text{INT-DIV}(17, 3) = (5, 2)$ and $17 \bmod 3 = 2$.

Def: $a \equiv b \pmod{N}$ if $a \bmod N = b \bmod N$.

Example: $17 \equiv 14 \pmod{3}$

Groups

Let G be a non-empty set, and let $\cdot$ be a binary operation on G. This means that for every two points $a, b \in G$, a value $a \cdot b$ is defined.

Example: $G = \mathbf{Z}_{12}^*$ and "$\cdot$" is multiplication modulo 12, meaning

$$a \cdot b = ab \bmod 12$$

Def: We say that G is a *group* if it has four properties called closure, associativity, identity and inverse that we present next.

Fact: If $N \in \mathbf{Z}_+$ then $G = \mathbf{Z}_N^*$ with $a \cdot b = ab \bmod N$ is a group.

Groups: Closure

Closure: For every $a, b \in G$ we have $a \cdot b$ is also in G.

Example: $G = \mathbf{Z}_{12}$ with $a \cdot b = ab$ does not have closure because $7 \cdot 5 = 35 \notin \mathbf{Z}_{12}$.

Fact: If $N \in \mathbf{Z}_+$ then $G = \mathbf{Z}_N^*$ with $a \cdot b = ab \bmod N$ satisfies closure, meaning

$$\gcd(a, N) = \gcd(b, N) = 1 \text{ implies } \gcd(ab \bmod N, N) = 1$$

Example: Let $G = \mathbf{Z}_{12}^* = \{1, 5, 7, 11\}$. Then

$$5 \cdot 7 \bmod 12 = 35 \bmod 12 = 11 \in \mathbf{Z}_{12}^*$$

Exercise: Prove the above Fact.

Groups: Associativity

Associativity: For every $a, b, c \in G$ we have $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.

Fact: If $N \in \mathbf{Z}_+$ then $G = \mathbf{Z}_N^*$ with $a \cdot b = ab \bmod N$ satisfies associativity, meaning

$$((ab \bmod N)c) \bmod N = (a(bc \bmod N)) \bmod N$$

Example:

$$\begin{aligned}(5 \cdot 7 \bmod 12) \cdot 11 \bmod 12 &= (35 \bmod 12) \cdot 11 \bmod 12 \\ &= 11 \cdot 11 \bmod 12 = 1 \\ 5 \cdot (7 \cdot 11 \bmod 12) \bmod 12 &= 5 \cdot (77 \bmod 12) \bmod 12 \\ &= 5 \cdot 5 \bmod 12 = 1\end{aligned}$$

Exercise: Given an example of a set G and a natural operation $a, b \mapsto a \cdot b$ on G that satisfies closure but *not* associativity.

Groups: Identity element

Identity element: There exists an element $\mathbf{1} \in G$ such that $a \cdot \mathbf{1} = \mathbf{1} \cdot a = a$ for all $a \in G$.

Fact: If $N \in \mathbf{Z}_+$ and $G = \mathbf{Z}_N^*$ with $a \cdot b = ab \bmod N$ then 1 is the identity element because $a \cdot 1 \bmod N = 1 \cdot a \bmod N = a$ for all a.

Groups: Inverses

Inverses: For every $a \in G$ there exists a unique $b \in G$ such that $a \cdot b = b \cdot a = \mathbf{1}$.

This b is called the inverse of a and is denoted a^{-1} if G is understood.

Fact: If $N \in \mathbf{Z}_+$ and $G = \mathbf{Z}_N^*$ with $a \cdot b = ab \bmod N$ then $\forall a \in \mathbf{Z}_N^* \quad \exists b \in \mathbf{Z}_N^*$ such that $a \cdot b \bmod N = 1$.

We denote this unique inverse b by $a^{-1} \bmod N$.

Example: $5^{-1} \bmod 12$ is the $b \in \mathbf{Z}_{12}^*$ satisfying $5b \bmod 12 = 1$, so $b =$

Groups: Inverses

Inverses: For every $a \in G$ there exists a unique $b \in G$ such that $a \cdot b = b \cdot a = \mathbf{1}$.

This b is called the inverse of a and is denoted a^{-1} if G is understood.

Fact: If $N \in \mathbf{Z}_+$ and $G = \mathbf{Z}_N^*$ with $a \cdot b = ab \bmod N$ then $\forall a \in \mathbf{Z}_N^* \quad \exists b \in \mathbf{Z}_N^*$ such that $a \cdot b \bmod N = 1$.

We denote this unique inverse b by $a^{-1} \bmod N$.

Example: $5^{-1} \bmod 12$ is the $b \in \mathbf{Z}_{12}^*$ satisfying $5b \bmod 12 = 1$, so $b = 5$

Exercises

Let $N \in \mathbf{Z}_+$ and let $G = \mathbf{Z}_N$. Prove that G is a group under the operation $a \cdot b = (a + b) \bmod N$.

Let $n \in \mathbf{Z}_+$ and let $G = \{0,1\}^n$. Prove that G is a group under the operation $a \cdot b = a \oplus b$.

Let $n \in \mathbf{Z}_+$ and let $G = \{0,1\}^n$. Prove that G is *not* a group under the operation $a \cdot b = a \wedge b$. (This is bit-wise AND, for example $0110 \wedge 1101 = 0100$.)

Computational Shortcuts

What is $5 \cdot 8 \cdot 10 \cdot 16 \bmod 21$?

Computational Shortcuts

What is $5 \cdot 8 \cdot 10 \cdot 16 \bmod 21$?

Slow way: First compute

$$5 \cdot 8 \cdot 10 \cdot 16 = 40 \cdot 10 \cdot 16 = 400 \cdot 16 = 6400$$

and then compute $6400 \bmod 21 =$

Computational Shortcuts

What is $5 \cdot 8 \cdot 10 \cdot 16 \bmod 21$?

Slow way: First compute

$$5 \cdot 8 \cdot 10 \cdot 16 = 40 \cdot 10 \cdot 16 = 400 \cdot 16 = 6400$$

and then compute $6400 \bmod 21 = 16$

Fast way:

- $5 \cdot 8 \bmod 21 = 40 \bmod 21 = 19$
- $19 \cdot 10 \bmod 21 = 190 \bmod 21 = 1$
- $1 \cdot 16 \bmod 21 = 16$

Exponentiation

Let G be a group and $a \in G$. We let $a^0 = \mathbf{1}$ be the identity element and for $n \geq 1$, we let

$$a^n = \underbrace{a \cdot a \cdots a}_{n}.$$

Also we let

$$a^{-n} = \underbrace{a^{-1} \cdot a^{-1} \cdots a^{-1}}_{n}.$$

This ensures that for all $i, j \in \mathbf{Z}$,

- $a^{i+j} = a^i \cdot a^j$
- $a^{ij} = (a^i)^j = (a^j)^i$
- $a^{-i} = (a^i)^{-1} = (a^{-1})^i$

Meaning we can manipulate exponents "as usual".

Examples

Let $N = 14$ and $G = \mathbf{Z}_N^*$. Then modulo N we have

$$5^3 =$$

Examples

Let $N = 14$ and $G = \mathbf{Z}_N^*$. Then modulo N we have

$$5^3 = 5 \cdot 5 \cdot 5$$

Examples

Let $N = 14$ and $G = \mathbf{Z}_N^*$. Then modulo N we have

$$5^3 = 5 \cdot 5 \cdot 5 \equiv 25 \cdot 5 \equiv 11 \cdot 5 \equiv 55 \equiv 13$$

and

$$5^{-3} =$$

Examples

Let $N = 14$ and $G = \mathbf{Z}_N^*$. Then modulo N we have

$$5^3 = 5 \cdot 5 \cdot 5 \equiv 25 \cdot 5 \equiv 11 \cdot 5 \equiv 55 \equiv 13$$

and

$$5^{-3} = 5^{-1} \cdot 5^{-1} \cdot 5^{-1}$$

Examples

Let $N = 14$ and $G = \mathbf{Z}_N^*$. Then modulo N we have

$$5^3 = 5 \cdot 5 \cdot 5 \equiv 25 \cdot 5 \equiv 11 \cdot 5 \equiv 55 \equiv 13$$

and

$$5^{-3} = 5^{-1} \cdot 5^{-1} \cdot 5^{-1} \equiv 3 \cdot 3 \cdot 3$$

Examples

Let $N = 14$ and $G = \mathbf{Z}_N^*$. Then modulo N we have

$$5^3 = 5 \cdot 5 \cdot 5 \equiv 25 \cdot 5 \equiv 11 \cdot 5 \equiv 55 \equiv 13$$

and

$$5^{-3} = 5^{-1} \cdot 5^{-1} \cdot 5^{-1} \equiv 3 \cdot 3 \cdot 3 \equiv 27 \equiv 13$$

Group Orders

The order of a group G is its size $|G|$, meaning the number of elements in it.

Example: The order of $\mathbf{Z}_{21}^*$ is

Group Orders

The order of a group G is its size $|G|$, meaning the number of elements in it.

Example: The order of $\mathbf{Z}_{21}^*$ is 12 because

$$\mathbf{Z}_{21}^* = \{1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20\}$$

Fact: Let G be a group of order m and $a \in G$. Then, $a^m = \mathbf{1}$.

Examples: Modulo 21 we have

- $5^{12} \equiv (5^3)^4 \equiv 20^4 \equiv (-1)^4 \equiv 1$
- $8^{12} \equiv (8^2)^6 \equiv (1)^6 \equiv 1$

Simplifying exponentiation

Fact: Let G be a group of order m and $a \in G$. Then, $a^m = \mathbf{1}$.

Corollary: Let G be a group of order m and $a \in G$. Then for any $i \in \mathbf{Z}$,

$$a^i = a^{i \bmod m}.$$

Proof: Let $(q, r) \leftarrow \text{INT-DIV}(i, m)$, so that $i = mq + r$ and $r = i \bmod m$. Then

$$a^i = a^{mq+r} = (a^m)^q \cdot a^r$$

But $a^m = \mathbf{1}$ by Fact.

Simplifying exponentiation

Corollary: Let G be a group of order m and $a \in G$. Then for any $i \in \mathbf{Z}$,

$$a^i = a^{i \bmod m}.$$

Example: What is $5^{74} \bmod 21$?

Simplifying exponentiation

Corollary: Let G be a group of order m and $a \in G$. Then for any $i \in \mathbf{Z}$,

$$a^i = a^{i \bmod m}.$$

Example: What is $5^{74} \bmod 21$?

Solution: Let $G = \mathbf{Z}_{21}^*$ and $a = 5$. Then, $m = 12$, so

$$\begin{aligned} 5^{74} \bmod 21 &= 5^{74 \bmod 12} \bmod 21 \\ &= 5^2 \bmod 21 \\ &= 4. \end{aligned}$$

Exercise

Evaluate the expressions shown in the first column. Your answer, in the second column, should be a member of the set shown in the third column. In the first case, the inverse refers to the group $\mathbf{Z}_{101}^*$. Don't use any electronic tools; these are designed to be done by hand.

Expression	Value	In
$34^{-1} \bmod 101$		$\mathbf{Z}_{101}^*$
$5^{1602} \bmod 17$		$\mathbf{Z}_{17}^*$
$\lvert\mathbf{Z}_{24}^*\rvert$		$\mathbf{N}$

Measuring Running Time of Algorithms on Numbers

In an algorithms course, the cost of arithmetic is often assumed to be $\mathcal{O}(1)$, because numbers are small. In cryptography numbers are

very, very BIG!

Typical sizes are 2^{512}, 2^{1024}, 2^{2048}.

Numbers are provided to algorithms in binary. The length of a, denoted $|a|$, is the number of bits in the binary encoding of a.

Example: $|7| = 3$ because 7 is 111 in binary.

Running time is measured as a function of the lengths of the inputs.

Algorithms on numbers

Algorithm	Input	Output	Time
ADD	$a,\ b$	$a+b$	$\mathcal{O}(\|a\|+\|b\|)$
MULT	$a,\ b$	ab	$\mathcal{O}(\|a\|\cdot\|b\|)$
INT-DIV	$a,\ N$	q,r	$\mathcal{O}(\|a\|\cdot\|N\|)$
MOD	$a,\ N$	$a \bmod N$	$\mathcal{O}(\|a\|\cdot\|N\|)$
EXT-GCD	$a,\ N$	(d,a',N')	$\mathcal{O}(\|a\|\cdot\|N\|)$
MOD-INV	$a \in \mathbf{Z}_N^*,\ N$	$a^{-1} \bmod N$	$\mathcal{O}(\|N\|^2)$
MOD-EXP	$a \in \mathbf{Z}_N,\ n,\ N$	$a^n \bmod N$	$\mathcal{O}(\|n\|\cdot\|N\|^2)$
EXP_G	$a \in G,\ n$	$a^n \in G$	$\mathcal{O}(\|n\|)$ G-ops

Extended gcd

$\mathrm{EXT\text{-}GCD}(a,N)$ returns (d,a',N') such that

$$d = \gcd(a,N) = a\cdot a' + N\cdot N' .$$

Example: $\mathrm{EXT\text{-}GCD}(12,20) =$

Extended gcd

$\mathrm{EXT\text{-}GCD}(a,N)$ returns (d,a',N') such that

$$d = \gcd(a,N) = a\cdot a' + N\cdot N' .$$

Example: $\mathrm{EXT\text{-}GCD}(12,20) = (4,-3,2)$ because

$$4 = \gcd(12,20) = 12\cdot(-3) + 20\cdot 2 .$$

Extended gcd Algorithm

EXT-GCD$(a, N) \mapsto (d, a', N')$ such that

$$d = \gcd(a, N) = a \cdot a' + N \cdot N' .$$

Lemma: Let $(q, r) = \text{INT-DIV}(a, N)$. Then, $\gcd(a, N) = \gcd(N, r)$

Alg EXT-GCD(a, N) // $(a, N) \neq (0, 0)$
if $N = 0$ then return $(a, 1, 0)$
else
$(q, r) \leftarrow \text{INT-DIV}(a, N)$; $(d, x, y) \leftarrow \text{EXT-GCD}(N, r)$
$a' \leftarrow y$; $N' \leftarrow x - qy$; return (d, a', N')

Running time is $\mathcal{O}(|a| \cdot |N|)$, so the extended gcd can be computed in quadratic time. If $a \geq N > 0$ then $\text{abs}(a') \leq N$ and $\text{abs}(N') \leq a$ where $\text{abs}(\cdot)$ denotes the absolute value. Analysis showing all this is non-trivial (worst case is Fibonacci numbers).

Modular Inverse

For a, N such that $\gcd(a, N) = 1$, we want to compute $a^{-1} \bmod N$, meaning the unique $a' \in \mathbf{Z}_N^*$ satisfying $aa' \equiv 1 \pmod N$.

But if we let $(d, a', N') \leftarrow \text{EXT-GCD}(a, N)$ then

$$d = 1 = \gcd(a, N) = a \cdot a' + N \cdot N'$$

But $N \cdot N' \equiv 0 \pmod N$ so $aa' \equiv 1 \pmod N$

Alg MOD-INV(a, N)
$(d, a', N') \leftarrow \text{EXT-GCD}(a, N)$
return $a' \bmod N$

Modular inverse can be computed in quadratic time.

Modular Exponentiation

Let G be a group and $a \in G$. For $n \in \mathbf{N}$, we want to compute $a^n \in G$.

We know that

$$a^n = \underbrace{a \cdot a \cdots a}_{n}$$

Consider:

$y \leftarrow 1$
for $i = 1, \ldots, n$ do $y \leftarrow y \cdot a$
return y

Question: Is this a good algorithm?

Modular Exponentiation

Let G be a group and $a \in G$. For $n \in \mathbf{N}$, we want to compute $a^n \in G$.

We know that

$$a^n = \underbrace{a \cdot a \cdots a}_{n}$$

Consider:

$y \leftarrow 1$
for $i = 1, \ldots, n$ do $y \leftarrow y \cdot a$
return y

Question: Is this a good algorithm?

Answer: It is correct but VERY SLOW. The number of group operations is $\mathcal{O}(n) = \mathcal{O}(2^{|n|})$ so it is exponential time. For $n \approx 2^{512}$ it is prohibitively expensive.

Fast exponentiation idea

We can compute

$$a \longrightarrow a^2 \longrightarrow a^4 \longrightarrow a^8 \longrightarrow a^{16} \longrightarrow a^{32}$$

in just 5 steps by repeated squaring. So we can compute a^n in i steps when $n = 2^i$.

But what if n is not a power of 2?

Square-and-Multiply Exponentiation Example

Suppose the binary length of n is 5, meaning the binary representation of n has the form $b_4b_3b_2b_1b_0$. Then

$$\begin{aligned} n &= 2^4b_4 + 2^3b_3 + 2^2b_2 + 2^1b_1 + 2^0b_0 \\ &= 16b_4 + 8b_3 + 4b_2 + 2b_1 + b_0 \ . \end{aligned}$$

We want to compute a^n. Our exponentiation algorithm will proceed to compute the values $y_5, y_4, y_3, y_2, y_1, y_0$ in turn, as follows:

$$\begin{array}{lclcl} y_5 &=& \mathbf{1} && \\ y_4 &=& y_5^2 \cdot a^{b_4} &=& a^{b_4} \\ y_3 &=& y_4^2 \cdot a^{b_3} &=& a^{2b_4+b_3} \\ y_2 &=& y_3^2 \cdot a^{b_2} &=& a^{4b_4+2b_3+b_2} \\ y_1 &=& y_2^2 \cdot a^{b_1} &=& a^{8b_4+4b_3+2b_2+b_1} \\ y_0 &=& y_1^2 \cdot a^{b_0} &=& a^{16b_4+8b_3+4b_2+2b_1+b_0} \ . \end{array}$$

Square-and-Multiply Exponentiation Algorithm

Let $\text{bin}(n) = b_{k-1} \ldots b_0$ be the binary representation of n, meaning

$$n = \sum_{i=0}^{k-1} b_i 2^i$$

Alg $\text{EXP}_G(a, n)$ // $a \in G$, $n \geq 1$
$b_{k-1} \ldots b_0 \leftarrow \text{bin}(n)$
$y \leftarrow 1$
for $i = k-1$ downto 0 do $y \leftarrow y^2 \cdot a^{b_i}$
return y

The running time is $\mathcal{O}(|n|)$ group operations.

MOD-EXP(a, n, N) returns a^n mod N in time $\mathcal{O}(|n| \cdot |N|^2)$, meaning is cubic time.

Exercise

Consider the following computational problem:

INPUT: N, a, b, x, y where $N \geq 1$ is an integer, $a, b \in \mathbf{Z}_N^*$ and x, y are integers with $0 \leq x, y < N$
OUTPUT: $a^x b^y$ mod N

Let $k = |N|$.

1. Consider the algorithm that first computes $X = a^x$ mod N, then computes $Y = b^y$ mod N, and returns XY mod N. Explain why this has worst case cost of $4k + 1$ multiplications modulo N.
2. Design an alternative, faster algorithm for this problem that uses at most $2k + 1$ multiplications modulo N.

Algorithms on numbers

Algorithm	Input	Output	Time
ADD	$a,\ b$	$a+b$	$\mathcal{O}(\lvert a\rvert + \lvert b\rvert)$
MULT	$a,\ b$	ab	$\mathcal{O}(\lvert a\rvert \cdot \lvert b\rvert)$
INT-DIV	$a,\ N$	q,r	$\mathcal{O}(\lvert a\rvert \cdot \lvert N\rvert)$
MOD	$a,\ N$	$a \bmod N$	$\mathcal{O}(\lvert a\rvert \cdot \lvert N\rvert)$
EXT-GCD	$a,\ N$	(d, a', N')	$\mathcal{O}(\lvert a\rvert \cdot \lvert N\rvert)$
MOD-INV	$a \in \mathbf{Z}_N^*,\ N$	$a^{-1} \bmod N$	$\mathcal{O}(\lvert N\rvert^2)$
MOD-EXP	$a \in \mathbf{Z}_N,\ n,\ N$	$a^n \bmod N$	$\mathcal{O}(\lvert n\rvert \cdot \lvert N\rvert^2)$
EXP_G	$a \in G,\ n$	$a^n \in G$	$\mathcal{O}(\lvert n\rvert)$ G-ops

Generators and cyclic groups

Let G be a group of order m and let $g \in G$. We let

$$\langle g \rangle = \{\, g^i \ : \ i \in \mathbf{Z} \,\} .$$

Fact: $\langle g \rangle = \{\, g^i \ : \ i \in \mathbf{Z}_m \,\}$

Exercise: Prove the above Fact.

Fact: The size $|\langle g \rangle|$ of the set $\langle g \rangle$ is a divisor of m

Note: $|\langle g \rangle|$ need not equal m!

Definition: $g \in G$ is a generator (or primitive element) of G if $\langle g \rangle = G$, meaning $|\langle g \rangle| = m$.

Definition: G is cyclic if it has a generator, meaning there exists $g \in G$ such that g is a generator of G.

Generators and cyclic groups: Example

Let $G = \mathbf{Z}_{11}^* = \{1,2,3,4,5,6,7,8,9,10\}$, which has order $m = 10$.

i	0	1	2	3	4	5	6	7	8	9	10
$2^i \bmod 11$	1	2	4	8	5	10	9	7	3	6	1
$5^i \bmod 11$	1	5	3	4	9	1	5	3	4	9	1

so

$$\begin{aligned} \langle 2 \rangle &= \{1,2,3,4,5,6,7,8,9,10\} \\ \langle 5 \rangle &= \{1,3,4,5,9\} \end{aligned}$$

- 2 a generator because $\langle 2 \rangle = \mathbf{Z}_{11}^*$.
- 5 is not a generator because $\langle 5 \rangle \neq \mathbf{Z}_{11}^*$.
- $\mathbf{Z}_{11}^*$ is cyclic because it has a generator.

Exercise

Let G be the group $\mathbf{Z}_{10}^*$ under the operation of multiplication modulo 10.

1. List the elements of G
2. What is the order of G?
3. Determine the set $\langle 3 \rangle$
4. Determine the set $\langle 9 \rangle$
5. Is G cyclic? Why or why not?

Discrete Logarithms

If $G = \langle g \rangle$ is a cyclic group of order m then for every $a \in G$ there is a unique exponent $i \in \mathbf{Z}_m$ such that $g^i = a$. We call i the discrete logarithm of a to base g and denote it by

$$\mathrm{DLog}_{G,g}(a)$$

The discrete log function is the inverse of the exponentiation function:

$$\begin{aligned} \mathrm{DLog}_{G,g}(g^i) &= i \quad \text{for all } i \in \mathbf{Z}_m \\ g^{\mathrm{DLog}_{G,g}(a)} &= a \quad \text{for all } a \in G. \end{aligned}$$

Discrete Logarithms: Example

Let $G = \mathbf{Z}_{11}^* = \{1,2,3,4,5,6,7,8,9,10\}$, which is a cyclic group of order $m = 10$. We know that 2 is a generator, so $\mathrm{DLog}_{G,2}(a)$ is the exponent $i \in \mathbf{Z}_{10}$ such that $2^i \bmod 11 = a$.

i	0	1	2	3	4	5	6	7	8	9
$2^i \bmod 11$	1	2	4	8	5	10	9	7	3	6

a	1	2	3	4	5	6	7	8	9	10
$\mathrm{DLog}_{G,2}(a)$										

Discrete Logarithms: Example

Let $G = \mathbf{Z}_{11}^* = \{1,2,3,4,5,6,7,8,9,10\}$, which is a cyclic group of order $m = 10$. We know that 2 is a generator, so $\mathrm{DLog}_{G,2}(a)$ is the exponent $i \in \mathbf{Z}_{10}$ such that $2^i \bmod 11 = a$.

i	0	1	2	3	4	5	6	7	8	9
$2^i \bmod 11$	1	2	4	8	5	10	9	7	3	6

a	1	2	3	4	5	6	7	8	9	10
$\mathrm{DLog}_{G,2}(a)$	0	1	8	2	4	9	7	3	6	5

Exercise

Let G be the group $\mathbf{Z}_{10}^*$ under the operation of multiplication modulo 10.

1. Show that 3 and 7 are generators of G

2. What is $\mathrm{DLog}_{G,3}(7)$?

3. What is $\mathrm{DLog}_{G,7}(9)$?

Finding Cyclic Groups

Fact 1: Let p be a prime. Then $\mathbf{Z}_p^*$ is cyclic.

Fact 2: Let G be any group whose order $m = |G|$ is a prime number. Then G is cyclic.

Note: $|\mathbf{Z}_p^*| = p - 1$ is not prime, so Fact 2 doesn't imply Fact 1!

Computing Discrete Logs

Let $G = \langle g \rangle$ be a cyclic group of order m with generator $g \in G$.

Input: $X \in G$
Desired Output: $\mathrm{DLog}_{G,g}(X)$

That is, we want x such that $g^x = X$.

for $x = 0, \ldots, m-1$ do
  if $g^x = X$ then return x

Is this a good algorithm?

Computing Discrete Logs

Let $G = \langle g \rangle$ be a cyclic group of order m with generator $g \in G$.

Input: $X \in G$
Desired Output: $\mathrm{DLog}_{G,g}(X)$

That is, we want x such that $g^x = X$.

for $x = 0, \ldots, m-1$ do
  if $g^x = X$ then return x

Is this a good algorithm? It is

- Correct (always returns the right answer)

Computing Discrete Logs

Let $G = \langle g \rangle$ be a cyclic group of order m with generator $g \in G$.

Input: $X \in G$
Desired Output: $\mathrm{DLog}_{G,g}(X)$

That is, we want x such that $g^x = X$.

for $x = 0, \ldots, m-1$ do
  if $g^x = X$ then return x

Is this a good algorithm? It is

- Correct (always returns the right answer), but
- SLOW!

Run time is $O(m)$ exponentiations, which for $G = \mathbf{Z}_p^*$ is $O(p)$, which is exponential time and prohibitive for large p.

Computing Discrete Logs: Best known algorithms

Group	Time to find discrete logarithms
$\mathbf{Z}_p^*$	$e^{1.92(\ln p)^{1/3}(\ln\ln p)^{2/3}}$
EC_p	$\sqrt{p} = e^{\ln(p)/2}$

Here p is a prime and EC_p represents an elliptic curve group of order p.

Note: In the first case the actual running time is $e^{1.92(\ln q)^{1/3}(\ln\ln q)^{2/3}}$ where q is the largest prime factor of $p-1$.

In neither case is a polynomial-time algorithm known.

This (apparent, conjectured) computational intractability of the discrete log problem makes it the basis for cryptographic schemes in which breaking the scheme requires discrete log computation.

Discrete logarithm computation records

In $\mathbf{Z}_p^*$:

$\|p\|$ in bits	When
431	2005
530	2007
596	2014

For elliptic curves, current record seems to be for $|p|$ around 113.

EC: More bang for the buck

Say we want 80-bits of security, meaning discrete log computation by the best known algorithm should take time 2^{80}. Then

- If we work in $\mathbf{Z}_p^*$ (p a prime) we need to set $|\mathbf{Z}_p^*| = p-1 \approx 2^{1024}$
- But if we work on an elliptic curve group of prime order p then it suffices to set $p \approx 2^{160}$.

Why? Because

$$e^{1.92(\ln 2^{1024})^{1/3}(\ln\ln 2^{1024})^{2/3}} \approx \sqrt{2^{160}} = 2^{80}$$

But now:

Group Size	Cost of Exponentiation
2^{160}	1
2^{1024}	260

Exponentiation will be 260 times faster in the smaller group!

DL Formally

Let $G = \langle g \rangle$ be a cyclic group of order m, and A an adversary.

Game $\mathrm{DL}_{G,g}$

procedure Initialize
$x \xleftarrow{\$} \mathbf{Z}_m$; $X \leftarrow g^x$
return X

procedure Finalize(x')
return $(x = x')$

The dl-advantage of A is

$$\mathbf{Adv}^{\mathrm{dl}}_{G,g}(A) = \Pr\left[\mathrm{DL}^A_{G,g} \Rightarrow \text{true}\right]$$

CDH: The Computational Diffie-Hellman Problem

Let $G = \langle g \rangle$ be a cyclic group of order m with generator $g \in G$. The CDH problem is:

Input: $X = g^x \in G$ and $Y = g^y \in G$
Desired Output: $g^{xy} \in G$

This underlies security of the DH Secret Key Exchange Protocol.

Obvious algorithm: $x \leftarrow \mathrm{DLog}_{G,g}(X)$; Return Y^x.

So if one can compute discrete logarithms then one can solve the CDH problem.

The converse is an open question. Potentially, there is a way to quickly solve CDH that avoids computing discrete logarithms. But no such way is known.

CDH Formally

Let $G = \langle g \rangle$ be a cyclic group of order m, and A an adversary.

Game $\mathrm{CDH}_{G,g}$

procedure Initialize
$x, y \xleftarrow{\$} \mathbf{Z}_m$
$X \leftarrow g^x$; $Y \leftarrow g^y$
return X, Y

procedure Finalize(Z)
return $(Z = g^{xy})$

The cdh-advantage of A is

$$\mathbf{Adv}^{\mathrm{cdh}}_{G,g}(A) = \Pr\left[\mathrm{CDH}^A_{G,g} \Rightarrow \mathsf{true}\right]$$

Building cyclic groups

We will need to build (large) groups over which our cryptographic schemes can work, and find generators in these groups.

How do we do this efficiently?

Building cyclic groups

To find a suitable prime p and generator g of $\mathbf{Z}_p^*$:

- Pick numbers p at random until p is a prime of the desired form
- Pick elements g from $\mathbf{Z}_p^*$ at random until g is a generator

For this to work we need to know

- How to test if p is prime
- How many numbers in a given range are primes of the desired form
- How to test if g is a generator of $\mathbf{Z}_p^*$ when p is prime
- How many elements of $\mathbf{Z}_p^*$ are generators

Finding primes

Desired: An efficient algorithm that given an integer k returns a prime $p \in \{2^{k-1}, \ldots, 2^k - 1\}$ such that $q = (p-1)/2$ is also prime.

Alg Findprime(k)
do
  $p \stackrel{\$}{\leftarrow} \{2^{k-1}, \ldots, 2^k - 1\}$
until (p is prime and $(p-1)/2$ is prime)
return p

- How do we test primality?
- How many iterations do we need to succeed?

Primality Testing

Given: integer N
Output: TRUE if N is prime, FALSE otherwise.

for $i = 2, \ldots, \lceil \sqrt{N} \rceil$ do
  if $N \bmod i = 0$ then return false
return true

Primality Testing

Given: integer N
Output: TRUE if N is prime, FALSE otherwise.

for $i = 2, \ldots, \lceil \sqrt{N} \rceil$ do
  if $N \bmod i = 0$ then return false
return true

Correct but SLOW! $O(N)$ running time, exponential. However, we have:

- $O(|N|^3)$ time randomized algorithms
- Even a $O(|N|^8)$ time deterministic algorithm

Density of primes

Let $\pi(N)$ be the number of primes in the range $1, \ldots, N$. So if $p \stackrel{\$}{\leftarrow} \{1, \ldots, N\}$ then

$$\Pr[p \text{ is a prime}] = \frac{\pi(N)}{N}$$

Fact: $\pi(N) \sim \dfrac{N}{\ln(N)}$

So

$$\Pr[p \text{ is a prime}] \sim \frac{1}{\ln(N)}$$

If $N = 2^{1024}$ this is about $0.001488 \approx 1/1000$.

So the number of iterations taken by our algorithm to find a prime is not too big.

Recall DH Secret Key Exchange

The following are assumed to be public: A large prime p and a generator g of $\mathbf{Z}_p^*$.

Alice		**Bob**
$x \stackrel{\$}{\leftarrow} \mathbf{Z}_{p-1};\ X \leftarrow g^x \bmod p$		
	$\xrightarrow{X}$	
		$y \stackrel{\$}{\leftarrow} \mathbf{Z}_{p-1};\ Y \leftarrow g^y \bmod p$
	$\xleftarrow{Y}$	
$K_A \leftarrow Y^x \bmod p$		$K_B \leftarrow X^y \bmod p$

- $Y^x = (g^y)^x = g^{xy} = (g^x)^y = X^y$ modulo p, so $K_A = K_B$
- Adversary is faced with the CDH problem.

DH Secret Key Exchange: Questions

- How do we pick a large prime p, and how large is large enough?
- What does it mean for g to be a generator modulo p?
- How do we find a generator modulo p?
- How can Alice quickly compute $x \mapsto g^x \bmod p$?
- How can Bob quickly compute $y \mapsto g^y \bmod p$?
- Why is it hard to compute $(g^x \bmod p, g^y \bmod p) \mapsto g^{xy} \bmod p$?
- ...

Exercise: Answer as many of these questions as you can based on the content of this chapter.