COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Fall 2020

Announcements/Reminders

Last day to turn in HW3

HW4 due Oct 27

Previously on COS 433…

Number Theory and Crypto

(Handout on course website with basic number theory primer)

Number-theory Constructions

Goal: base security on hard problems of interest to mathematicians

- Wider set of people trying to solve problem
- Longer history
- Ultimately, new applications

Number Theory

$\mathbb{Z}_N$: integers mod **N**
$\mathbb{Z}_N^*$: integers mod **N** that are relatively prime to **N**

- $x \in \mathbb{Z}_N^*$ iff **x** has an "inverse" **y** s.t. $xy \bmod N = 1$
  $\Rightarrow \mathbb{Z}_N^*$ is a multiplicative group

- For prime **N**, $\mathbb{Z}_N^* = \mathbb{Z}_N \setminus \{0\} = \{1,\ldots,N-1\}$
  $\Rightarrow \mathbb{Z}_N$ for prime **N** is a field

Totient function: $\Phi(N) := |\mathbb{Z}_N^*|$

Euler's theorem: for any $x \in \mathbb{Z}_N^*$, $x^{\Phi(N)} \bmod N = 1$

Today

Number theory continued

Cyclic Groups

For prime $\mathbf{p}$, $\mathbb{Z}_p^*$ is cyclic, meaning

$$\exists\ \mathbf{g}\ \text{s.t.}\ \mathbb{Z}_p^* = \{1, g, g^2, \ldots, g^{p-2}\}$$

(we call such a **g** a generator)

However, not all **g** are generators

- If $\mathbf{g_0}$ is a generator, then $\mathbf{g=g_0^2}$ is not:

  $g_0^{(p-1)/2} = g^{p-1} = 1$, so $|\{1,g,\ldots\}| \leq (p-1)/2$

- How to test for generator?

Discrete Log

Discrete Log

Let **p** be a large number (usually prime)

Given $\mathbf{g} \in \mathbb{Z}_p^*$, $\mathbf{a} \in \mathbb{Z}$, "easy" to compute $\mathbf{g^a \bmod p}$

- Time **poly(log a, log p)**
- How?

However, no known efficient ways to recover **a** (mod $\mathbf{\Phi(p)=p-1}$) from **g** and $\mathbf{g^a \bmod p}$

Hardness of DLog

For prime **p**, best know algorithms:

- Brute force: $O(p)$
- Better algs based on birthday paradox: $O(p^{1/2})$
- Even better heuristic algorithms:

$$\exp(C\, (\log p)^{1/3}\, (\log \log p)^{2/3})$$

(super polynomial in **log p**)

For non-prime **p**, some cases are easy

Sampling Large Random Primes

Prime Number Theorem: A random **λ**-bit number is prime with probability **$\approx 1/\lambda$**

Primality Testing: It is possible in polynomial time to decide if an integer is prime

Fermat Primality Test (randomized, some false positives):

- Choose a random integer $a \in \{0,\dots,N-1\}$
- Test if $a^N = a \bmod N$
- Repeat many times

Discrete Log Assumption: For any discrete log algorithm  running in time polynomial time, there exists negligible **ε** such that:

$$\Pr\left[a \leftarrow \text{[algorithm]}(p, g, g^a \bmod p) : \begin{array}{l} p \leftarrow \text{random } \lambda\text{-bit prime} \\ g \leftarrow \text{random generator of } \mathbb{Z}_p^*, \\ a \leftarrow \mathbb{Z}_{p-1} \end{array}\right] \leq \varepsilon(\lambda)$$

Collision Resistance from DLog

Let **p** be a prime

- Key space = $\mathbb{Z}_p^{\,2}$
- Domain: $\mathbb{Z}_{p-1}^{\,2}$
- Range: $\mathbb{Z}_p$
- $\mathbf{H(\ (g,h),\ (x,y)\ ) = g^x h^y}$

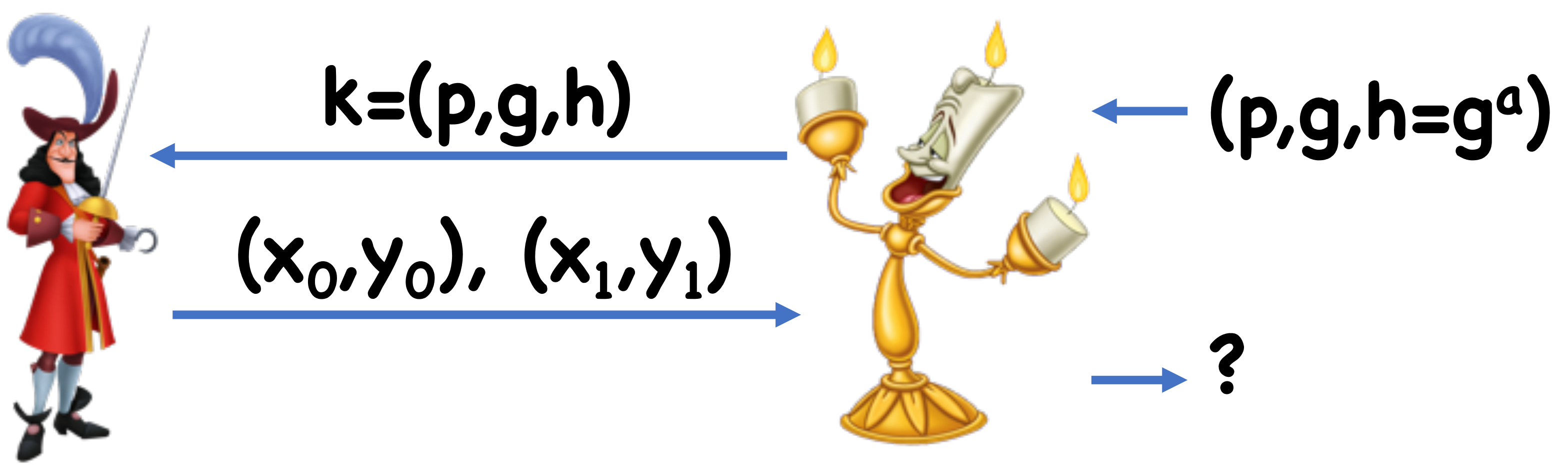
To generate key, choose random **p**, **g,h** $\in \mathbb{Z}_p^*$

- Require **g** a generator

Collision Resistance from Discrete Log

$H(\ (g,h),\ (x,y)\ ) = g^x h^y$

Theorem: If discrete log assumption holds, then **H** is collision resistant



Collision Resistance from Discrete Log

Proof idea:

- Input to **H** is equation for a line **line(a)=ay+x**
- **H(line) = $g^{line(a)}$** (evaluation "in the exponent")
- A collision is two different lines that intersect at **a**
- Use equations for two lines to solve for **a**:

$$a = -(x_1-x_0)/(y_1-y_0) \pmod{p-1}$$

Problem

For $p>2$, $p-1$ is not a prime, so has some factors

Therefore, (y_1-y_0) not necessarily invertible mod $p-1$

However, possible to show that if this is the case, either:

- (y_1-y_0) and (x_1-x_0) have common factor, so can remove factor and try again, or
- g is not a generator (which isn't allowed)

Blum-Micali PRG

Let $\mathbf{p}$ be a prime

Let $\mathbf{g} \in \mathbb{Z}_p^*$

Let $\mathbf{h}: \mathbb{Z}_p^* \rightarrow \{0,1\}$ be $\mathbf{h(x) = 1}$ **if** $\mathbf{0 < x < (p-1)/2}$

Seed space: $\mathbb{Z}_p^*$

Algorithm:

- Let $\mathbf{x_0}$ be seed
- For $\mathbf{i=0,\ldots}$
  - Let $\mathbf{x_{i+1} = g^{x_i} \bmod p}$
  - Output $\mathbf{h(x_i)}$

Theorem: If the discrete log assumption holds on $\mathbb{Z}_p^*$, then the Blum-Micali generator is a secure PRG

We will prove this eventually (if time)

Another PRG

p a prime
Let **g** be a generator

Seed space: $\mathbb{Z}_{p-1}^2$
Range: $\mathbb{Z}_p^3$

$$\mathbf{PRG(a,b) = (g^a, g^b, g^{ab})}$$

Don't know how to prove security from DLog

Stronger Assumptions on Groups

Sometimes, the discrete log assumption is not enough

Instead, define stronger assumptions on groups

Computational Diffie-Hellman:

- Given (g, g^a, g^b), compute g^{ab}

Decisional Diffie-Hellman:

- Distinguish (g, g^a, g^b, g^c) from (g, g^a, g^b, g^{ab})

Increasing Difficulty

DLog:

- Given (g, g^a), compute a

CDH:

- Given (g, g^a, g^b), compute g^{ab}

DDH:

- Distinguish (g, g^a, g^b, g^c) from (g, g^a, g^b, g^{ab})

Stronger Assumptions

Computational Diffie Hellman: For any algorithm  running in polynomial time, there exists negligible **ε** such that:

Pr[g^{ab} **←**  **(p,g,** g^a**,** g^b**):**

p←random **λ**-bit prime

g←random generator of $\mathbb{Z}_p^*$,

a,b← $\mathbb{Z}_{p-1}$ **] ≤ ε(λ)**

Decisional Diffi … algorithm
running in po… time, the… ists negligible **ε**
such that:

$|Pr[1 \leftarrow$ (p… $^a, g^b, g^{a}$… $\leftarrow$ …]
$-Pr[1 \leftarrow$ (… $, g^b, g^c)$: … $\mathbb{Z}_{p-1}]| \leq \varepsilon(\lambda)$

Hardness of DDH

Need to be careful about DDH

Turns out that DDH as described is usually easy:

- For prime **$p>2$, $\Phi(p)=p-1$** will have small factors
- Can essentially reduce solving DDH to solving DDH over a small factor

Fixing DDH

Let $\mathbf{g_0}$ be a generator

Suppose $\mathbf{p-1 = qr}$ for prime $\mathbf{q}$, integer $\mathbf{r}$

Let $\mathbf{g=g_0^r}$

$\mathbf{g^q \bmod p = 1}$, but $\mathbf{g^{q'} \bmod p \neq 1}$ for any $\mathbf{q'<q}$

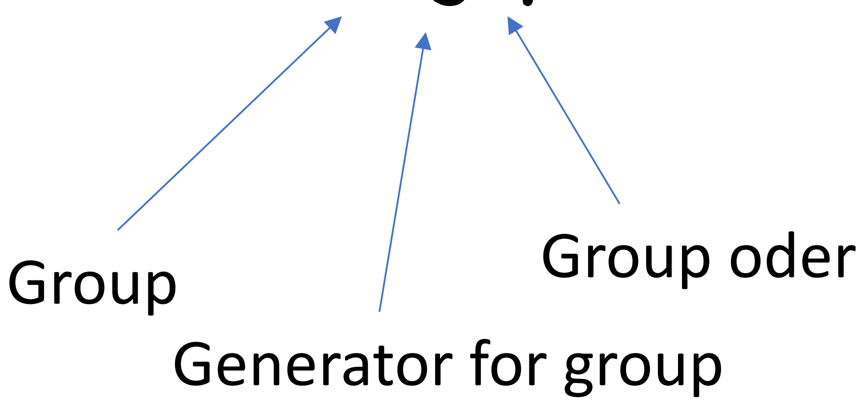
- So $\mathbf{g}$ has "order" $\mathbf{q}$

Let $\mathbf{G = \{1,g,g^2,\ldots\}}$ be group "generated by" $\mathbf{g}$

Generalizing Cryptographic Groups

Replace fixed family of groups with "group generator" algorithm

$$(G,g,q) \leftarrow \mathbf{GrGen}(\lambda)$$



Decisional Diffie Hellman for GrGen:

For any algorithm  running in polynomial time, there exists negligible **ε** such that:

| Pr[1←(g, g^a, g^b, g^{ab}):

$(G,g,q) \leftarrow GrGen(\lambda)$, $a,b \leftarrow \mathbb{Z}_q$]

–Pr[1←(g, g^a, g^b, g^c):

$(G,g,q) \leftarrow GrGen(\lambda)$, $a,b,c \leftarrow \mathbb{Z}_q$] | $\leq$ ε(λ)

Back to our PRG

Seed space: $\mathbf{Z}_q^{\,2}$
Range: $\mathbf{G}^3$

$$\mathbf{PRG(a,b) = (g^a, g^b, g^{ab})}$$

Security almost immediately follows from DDH

Generalizing Cryptographic Groups

Can also define Dlog, CDH relative to general **GrGen**

In many cases, problems turns out easy

Ex: $\mathbf{G} = \mathbf{Z_q}$, where $\mathbf{g \otimes h = g+h \bmod q}$

- What is exponentiation in **G**?
- What is discrete log in **G**?

Essentially only two groups where Dlog/CDH/DDH is conjectured to be hard:

- $\mathbb{Z}_p^*$ and its subgroups
- “Elliptic curve” groups

Parameter Size in Practice?

$\mathbf{G}$ = subgroup of $\mathbb{Z}_p^*$ of order $\mathbf{q}$, where $\mathbf{q} \mid \mathbf{p-1}$

- In practice, best algorithms require $\mathbf{p} \geq \mathbf{2^{1024}}$ or so

- $\mathbf{G}$ = "elliptic curve" group
- Can set $\mathbf{p} \approx \mathbf{2^{256}}$ to have security
  $\Rightarrow$ best attacks run in time $\mathbf{2^{128}}$

Therefore, elliptic curve groups tend to be much more efficient $\Rightarrow$ preferred in practice

Naor-Reingold PRF

Domain: $\{0,1\}^n$

Key space: $\mathbb{Z}_q^{n+1}$

Range: $\mathbb{G}$

$$F(\ (a, b_1, b_2, \ldots, b_n),\ x\ ) = g^{a\ b_1^{x1}\ b_2^{x2}\ \ldots\ b_n^{xn}}$$

Theorem: If DDH assumption holds on $\mathbb{G}$, then the Naor-Reingold PRF is secure

Proof by Hybrids

Hybrids **0**: $H(x) = g^{a\, b_1^{x1}\, b_2^{x2}\, \dots\, b_n^{xn}}$

Hybrid **i**: $H(x) = H_i(x_{[1,i]})^{b_{i+1}^{x_{i+1}}\, \dots\, b_n^{xn}}$

- H_i is a random function from $\{0,1\}^i \rightarrow G$

Hybrid **n**: $H(x)$ is truly random

Proof

Suppose adversary can distinguish Hybrid **i-1** from Hybrid **i** for some **i**

Easy to construct adversary that distinguishes:

$$\mathbf{x \rightarrow H_i(x)} \text{ from } \mathbf{x \rightarrow H_{i-1}(x_{[1,i-1]})^{b^{x_i}}}$$

Proof

Suppose adversary makes $\mathbf{2r}$ queries

- Assume wlog that queries are in pairs $\mathbf{x||0}$, $\mathbf{x||1}$

What does the adversary see?

- $\mathbf{H_i(x)}$: $\mathbf{2r}$ random elements in $\mathbf{G}$

- $\mathbf{H_{i-1}(x_{[1,i-1]})^{b_i^{xi}}}$: $\mathbf{r}$ random elements in $\mathbf{G}$, $\mathbf{h_1,\ldots,h_q}$
as well as $\mathbf{h_1^b, \ldots, h_q^b}$

Lemma: Assuming the DDH assumption on **G**, for any polynomial **r**, the following distributions are indistinguishable:

$$(g, g^{x_1}, g^{y_1}, \ldots, g^{x_q}, g^{y_q}) \text{ and}$$
$$(g, g^{x_1}, g^{b\,x_1}, \ldots, g^{x_q}, g^{b\,x_q})$$

Suffices to finish proof of NR-PRF

Proof of Lemma

Hybrids **0**: $(g, g^{x_1}, g^{b\, x_1}, \ldots, g^{x_r}, g^{b\, x_r})$

Hybrid **i**:

$$(g, g^{x_1}, g^{y_1}, \ldots, g^{x_i}, g^{y_i}, g^{x_{i+1}}, g^{b\, x_{i+1}}, \ldots g^{x_r}, g^{b\, x_r})$$

Hybrid **q**: $(g, g^{x_1}, g^{y_1}, \ldots, g^{x_r}, g^{y_r})$

Proof of Lemma

Suppose adversary distinguishes Hybrid **i-1** from Hybrid **i**

Use adversary to break DDH:

$$(g, g^{x_1}, g^{y_1}, \ldots, g^{x_{i-1}}, g^{y_{i-1}}, u, v, g^{x_{i+1}}, h^{x_{i+1}}, \ \ldots g^{x_r}, h^{x_r})$$

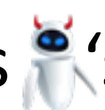
$$(g, h, u, v)$$

Proof of Lemma

$(g,g^{x_1},g^{y_1},\ldots,g^{x_{i-1}},g^{y_{i-1}},u,v,\ g^{x_{i+1}},h^{x_{i+1}},\ \ldots g^{x_r},h^{x_r})$

If $(g,h,u,v) = (g,g^{b},g^{x_i},g^{b\,x_i})$, then Hybrid **i–1**

If $(g,h,u,v) = (g,g^{b},g^{x_i},g^{y_i})$, then Hybrid **i**

Therefore, ‘s advantage is the same as ‘s

Further Applications

From NR-PRF can construct:

- CPA-secure encryption
- Block Ciphers
- MACs
- Authenticated Encryption

Announcements/Reminders

Last day to turn in HW3

HW4 due Oct 27