COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Spring 2020

Announcements

HW5 Due April 9^{th}
PR2 Due April 19^{th}

Previously on COS 433…

Discrete Log

Discrete Log

Let $\mathbf{p}$ be a large number (usually prime)

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

- Time $\mathbf{poly(\log a, \log p)}$
- How?

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

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

Pr[$a \leftarrow$  $(p, g, g^a \bmod p)$:

$p \leftarrow$ random λ-bit prime

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

$a \leftarrow \mathbb{Z}_{p-1}$] $\leq \varepsilon(\lambda)$

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

Blum-Micali PRG

Let $\mathbf{p}$ be a prime

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

Let $\mathbf{h}: \mathbf{G} \to \{0,1\}$ be $\mathbf{h(x) = 1 \text{ if } 0<x<(p-1)/2}$

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 next time (if time)

Today

Discrete log continued

Factoring

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:

$\Pr[g^{ab} \leftarrow$  $(p,g,g^a,g^b):$

$p \leftarrow$ random $\boldsymbol{\lambda}$-bit prime

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

$a,b \leftarrow \mathbb{Z}_{p-1}$ $] \leq \varepsilon(\lambda)$

Decisional Diffi … algorithm

running in po… time, the… ists negligible **ε**

such that:

|Pr[1← (… ^{a}, g^b, g$^{a…}$) … ← …]

−Pr[1← (… , g^b, g^c): … $\mathbb{Z}_{p-1}$]| ≤ ε(λ)

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

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

Group oder

Group

Generator for group

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):**

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

Another PRG

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

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

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?

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

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

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

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

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

Proof by Hybrids

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

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

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

Hybrid **n**: $\mathbf{H(x)}$ 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 $\mathbf{G}$

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

Lemma: Assuming the DDH assumption on $\mathbf{G}$, for any polynomial $\mathbf{r}$, the following distributions are indistinguishable:

$$(g, g^{x_1}, g^{y_1}, \ldots, g^{x_r}, g^{y_r}) \text{ and}$$
$$(g, g^{x_1}, g^{b\,x_1}, \ldots, g^{x_r}, g^{b\,x_r})$$

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,h,u,v)

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

Proof of Lemma

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

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

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

Therefore, ‘s advantage is the same as ‘s

Further Applications

From NR-PRF can construct:

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

Integer Factorization

Integer Factorization

Given an integer **N**, find it's prime factors

Studied for centuries, presumed difficult

- Grade school algorithm: $O(N^{1/2})$
- Better algorithms using birthday paradox: $O(N^{1/4})$
- Even better assuming G. Riemann Hyp.: $O(N^{1/5})$
- Still better heuristic algorithms:

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

- However, all require super-polynomial time in bit-length of **N**

Factoring Assumption: For any factoring algorithm  running in polynomial time, $\exists$ negligible $\boldsymbol{\varepsilon}$ such that:

Pr[(p,q)←**(N):**

N=pq

p,q ←random **λ**-bit primes**]** $\leq$ $\boldsymbol{\varepsilon(\lambda)}$

Chinese Remainder Theorem

Let $\mathbf{N} = \mathbf{pq}$ for distinct prime $\mathbf{p,q}$

Let $\mathbf{x} \in \mathbb{Z}_\mathbf{p}$, $\mathbf{y} \in \mathbb{Z}_\mathbf{q}$

Then there exists a unique integer $\mathbf{z} \in \mathbb{Z}_\mathbf{N}$ such that

- $\mathbf{x = z \bmod p}$, and
- $\mathbf{y = z \bmod q}$

Proof: $\mathbf{z = [py(p^{-1} \bmod q)+qx(q^{-1} \bmod p)] \bmod N}$

Quadratic Residues

Definition: y is a quadratic residue mod N if there exists an x such that $y = x^2 \bmod N$. x is called a "square root" of y

Ex:

- Let p be a prime, and $y \neq 0$ a quadratic residue mod p. How many square roots of y?
- Let $N=pq$ be the product of two primes, y a quadratic residue mod N. Suppose $y \neq 0 \bmod p$ and $y \neq 0 \bmod q$. How many square roots?

QR Assumption: For any algorithm  running in polynomial time, $\exists$ negligible **ε** such that:

$$\Pr[y^2 = x^2 \bmod N : \quad y \leftarrow \mathcal{A}(N, x^2), \quad N = pq,\ p,q \leftarrow \text{random } \boldsymbol{\lambda}\text{-bit primes}, \quad x \leftarrow \mathbb{Z}_N \;] \leq \varepsilon(\lambda)$$

Theorem: If the factoring assumption holds, then the QR assumption holds

Proof

To factor $\mathbf{N}$:

- $\mathbf{x} \leftarrow \mathbb{Z}_\mathbf{N}$
- $\mathbf{y} \leftarrow$ $(\mathbf{N}, \mathbf{x}^2)$
- Output $\mathbf{GCD(x-y,N)}$

Analysis:

- Let $\{\mathbf{a,b,c,d}\}$ be the 4 square roots of $\mathbf{x}^2$
-  has no idea which one you chose
- With probability ½, $\mathbf{y}$ will not be in $\{\mathbf{+x,-x}\}$
- In this case, we know $\mathbf{x=y \bmod p}$ but $\mathbf{x=-y \bmod q}$

Collision Resistance from Factoring

Let $\mathbf{N=pq}$, $\mathbf{y}$ a QR mod $\mathbf{N}$
Suppose $\mathbf{-1}$ is not a **QR** mod $\mathbf{N}$

Hashing key: $\mathbf{(N,y)}$
Domain: $\mathbf{\{1,\dots,(N-1)/2\}\times\{0,1\}}$
Range: $\mathbf{\{1,\dots,(N-1)/2\}}$

$\mathbf{H(\ (N,y),\ (x,b)\ ):}$ Let $\mathbf{z = y^b x^2 \bmod N}$

- If $\mathbf{z}\in\mathbf{\{1,\dots,(N-1)/2\}}$, output $\mathbf{z}$
- Else, output $\mathbf{-z \bmod N} \in\mathbf{\{1,\dots,(N-1)/2\}}$

Theorem: If the factoring assumption holds, $\mathbf{H}$ is collision resistant

Proof:

- Collision means $(x_0,b_0)\neq(x_1,b_1)$ s.t.

$$y^{b0}\ x_0^2 = \pm\ y^{b1}\ x_1^2 \bmod N$$

- If $b_0=b_1$, then $x_0\neq x_1$, but $x_0^2=\pm x_1^2 \bmod N$
  - $x_0^2=-x_1^2$ mod N not possible. Why?
  - $x_0\neq -x_1$ since $x_0,x_1\in\{1,\ldots,(N-1)/2\}$

- If $b_0\neq b_1$, then $(x_0/x_1)^2 = \pm y^{\pm 1} \bmod N$
  - $-y$ case not possible. Why?
  - (x_0/x_1) or (x_1/x_0) is a square root of y

Choosing **N**

How to choose **N** so that **-1** is not a QR?

By CRT, need to choose **p,q** such that -1 is not a QR mod **p** or mod **q**

Fact: if **p = 3 mod 4**, then **-1** is not a QR mod **p**
Fact: if **p = 1 mod 4**, then **-1** is a QR mod **p**

Is Composite **N** Necessary for SQ to be hard?

Let **p** be a prime, and suppose $\mathbf{p = 3 \bmod 4}$

Given a QR **x** mod **p**, how to compute square root?

Hint: recall Fermat: $\mathbf{x^{p-1}=1}$ mod **p** for all $\mathbf{x \neq 0}$

Hint: what is $\mathbf{x^{(p+1)/2}}$ mod **p?**

Solving Quadratic Equations

In general, solving quadratic equations is:

- Easy over prime moduli
- As hard as factoring over composite moduli

Other Powers?

What about $x \rightarrow x^4 \bmod N$? $x \rightarrow x^6 \bmod N$?

The function $x \rightarrow x^3 \bmod N$ appears quite different

- Suppose 3 is relatively prime to $p-1$ and $q-1$
- Then $x \rightarrow x^3 \bmod p$ is injective for $x \neq 0$
  - Let a be such that $3a = 1 \bmod p-1$
  - $(x^3)^a = x^{1+k(p-1)} = x(x^{p-1})^k = x \bmod p$
- By CRT, $x \rightarrow x^3 \bmod N$ is injective for $x \in \mathbb{Z}_N^*$

$x^3 \bmod N$

What does injectivity mean?

Cannot base of factoring:

Adapt alg for square roots?

- Choose a random $z \bmod N$
- Compute $y = z^3 \bmod N$
- Run inverter on y to get a cube root x
- Let $p = GCD(z-x, N)$, $q = N/p$

RSA Problem

Given

- $\mathbf{N = pq}$,
- $\mathbf{e}$ such that $\mathbf{GCD(e,p-1)=GCD(e,q-1)=1}$,
- $\mathbf{y=x^e \bmod N}$ for a random $\mathbf{x}$

Find $\mathbf{x}$

Injectivity means cannot base hardness on factoring, but still conjectured to be hard

RSA Assumption: For any algorithm  running in polynomial time, , $\exists$ negligible **ε** such that:

$\Pr[x \leftarrow$  $(N, x^3 \bmod N)$

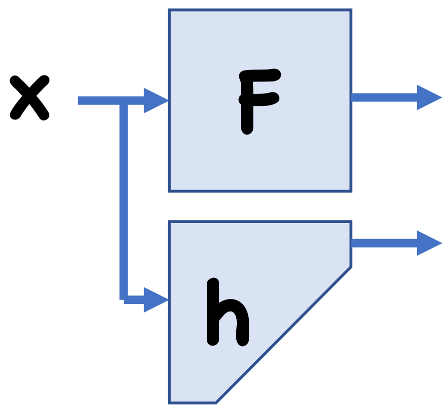
N=pq and **p,q** random **λ**-bit primes s.t.

$GCD(3,p-1)=GCD(3,q-1)=1$

$x \leftarrow \mathbb{Z}_N^*\ ] \leq \varepsilon(\lambda)$

Application: PRGs

Let $\mathbf{F(x) = x^3 \bmod N}$, $\mathbf{h(x)}$ = least significant bit



Theorem: If **RSA** Assumption holds, then
$\mathbf{G(x) = (\ F(x),\ h(x)\ )}$ is a secure PRG

Reminders

HW5 Due April 9^{th}

PR2 Due April 19^{th}