Announcements

HW5 Due April 9th
PR2 Due April 19th
Previously on COS 433...
Discrete Log
Discrete Log

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

Given $g \in \mathbb{Z}_p^*$, $a \in \mathbb{Z}$, “easy” to compute $g^a \mod p$

• Time $\text{poly}(\log a, \log p)$
• How?

However, no known efficient ways to recover $a \pmod{\Phi(p)=p-1}$ from $g$ and $g^a \mod p$
**Discrete Log Assumption:** For any discrete log algorithm running in time polynomial time, there exists negligible $\varepsilon$ such that:

\[
\Pr[a \leftarrow \mathcal{A} (p, g, g^a \mod p): \\
p \leftarrow \text{random } \lambda\text{-bit prime} \\
g \leftarrow \text{random generator of } \mathbb{Z}_p^* \\
a \leftarrow \mathbb{Z}_{p-1}
] \leq \varepsilon(\lambda)
\]
Collision Resistance from DLog

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

To generate key, choose random $p$, $g, h \in \mathbb{Z}_p^*$
- Require $g$ a generator
Blum-Micali PRG

Let $p$ be a prime

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

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

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

Algorithm:
• Let $x_0$ be seed
• For $i=0, ...$
  • Let $x_{i+1} = g^{x_i} \mod p$
  • Output $h(x_i)$
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: $\mathbb{Z}_p^3$

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

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 \((g, g^a, g^b, g^{ab})\)
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 \((g, g^a, g^b, g^{ab})\)
Computational Diffie Hellman: For any algorithm running in polynomial time, there exists negligible \( \varepsilon \) such that:

\[
\Pr[g^{ab} \leftarrow (p, g, g^a, g^b): \\
p \leftarrow \text{random } \lambda\text{-bit prime} \\
g \leftarrow \text{random generator of } \mathbb{Z}_p^* \\
a, b \leftarrow \mathbb{Z}_{p-1} \\
] \leq \varepsilon(\lambda)
\]
Decisional Diffie–Hellman: For any algorithm running in polynomial time, there exists negligible \( \varepsilon \) such that:

\[
|\Pr[1 \leftarrow \sigma(\mathcal{A}, \mathcal{B}, \mathcal{C})] - \Pr[1 \leftarrow \sigma(\mathcal{A}, \mathcal{B}, \mathcal{C})]: a, b, c \in \mathbb{Z}_{p-1}]| \leq \varepsilon(\lambda)
\]
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 $g_0$ be a generator

Suppose $p-1 = qr$ for prime $q$, integer $r$

Let $g = g_0^r$

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

• So $g$ has “order” $q$

Let $G = \{1, g, g^2, \ldots\}$ be group “generated by” $g$
Generalizing Cryptographic Groups

Replace fixed family of groups with “group generator” algorithm

\[(G, g, q) \leftarrow \text{GrGen}(\lambda)\]
Decisional Diffie Hellman for GrGen:
For any algorithm running in polynomial time, there exists negligible $\varepsilon$ such that:

$$\left| \Pr[1 \leftarrow \mathcal{A}(g, g^a, g^b, g^{ab}): (G, g, q) \leftarrow \text{GrGen}(\lambda), a, b \leftarrow \mathbb{Z}_q] - \Pr[1 \leftarrow \mathcal{A}(g, g^a, g^b, g^c): (G, g, q) \leftarrow \text{GrGen}(\lambda), a, b, c \leftarrow \mathbb{Z}_q] \right| \leq \varepsilon(\lambda)$$
Another PRG

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

\[ \text{PRG}(a,b) = (g^a, g^b, g^{ab}) \]

Security almost immediately follows from DDH
Generalizing Cryptographic Groups

Can also define Dlog, CDH relative to general \( \text{GrGen} \)

In many cases, problems turns out easy
Ex: \( G = \mathbb{Z}_q \), where \( g \otimes h = g+h \mod q \)
• 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?

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

• 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: $G$

$F( (a,b_1,b_2,...,b_n), x ) = g^{a b_1^{x_1} b_2^{x_2} ... b_n^{x_n}}$

**Theorem:** If DDH assumption holds on $G$, then the Naor-Reingold PRF is secure
Proof by Hybrids

Hybrids 0: \[ H(x) = g^{a b_1^{x_1} b_2^{x_2} \ldots b_n^{x_n}} \]

Hybrid i: \[ H(x) = H_i(x_{[1,i]})^{b_{i+1}^{x_{i+1}} \ldots b_n^{x_n}} \]
- \( H_i \) 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:

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

Suppose adversary makes $2r$ queries

• Assume wlog that queries are in pairs $x||0$, $x||1$

What does the adversary see?

• $H_i(x)$: $2r$ random elements in $G$

• $H_{i-1}(x_{[1,i-1]})^{b_i x_i} : h_1, ..., h_q$ ($r$ random elements in $G$) as well as $h_1^{b_i}$, ..., $h_q^{b_i}$
Lemma: Assuming the DDH assumption on $G$, for any polynomial $r$, the following distributions are indistinguishable:

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

$$(g, g^{x_1}, g^{b \cdot x_1}, \ldots, g^{x_r}, g^{b \cdot x_r})$$

Suffices to finish proof of NR-PRF
Proof of Lemma

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

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

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:

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

\[(g, g^{x_1}, g^{y_1}, ..., g^{x_{i-1}}, g^{y_{i-1}}, u, v, g^{x_i+1}, h^{x_i+1}, ... g^{x_r}, h^{x_r})\]

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

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

Therefore, \(\text{\textbullet's advantage is the same as \text{\textbullet's}\)}
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.: $O(N^{1/6})$
• 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 \( \varepsilon \) such that:

\[
\Pr[(p, q) \leftarrow \text{R}(N):
\begin{align*}
N &= pq \\
p, q &\leftarrow \text{random } \lambda\text{-bit primes}
\end{align*}
] \leq \varepsilon(\lambda)
\]
Chinese Remainder Theorem

Let $N = pq$ for distinct prime $p,q$

Let $x \in \mathbb{Z}_p$, $y \in \mathbb{Z}_q$

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

- $x = z \mod p$, and
- $y = z \mod q$

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

Definition: \( y \) is a quadratic residue mod \( N \) if there exists an \( x \) such that \( y = x^2 \mod N \). \( x \) 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 \( y \neq 0 \mod p \) and \( y \neq 0 \mod q \). How many square roots? 
**QR Assumption:** For any algorithm running in polynomial time, ∃ negligible ε such that:

\[
\Pr[y^2 = x^2 \mod N: \\
y \leftarrow (N, x^2) \\
N = pq, \ p, q \leftarrow \text{random } \lambda\text{-bit primes} \\
x \leftarrow \mathbb{Z}_N \\
] \leq \varepsilon(\lambda)
\]
**Theorem:** If the factoring assumption holds, then the QR assumption holds
Proof

To factor \( N \):

- \( x \leftarrow \mathbb{Z}_N \)
- \( y \leftarrow \mathcal{R}(N, x^2) \)
- Output \( \text{GCD}(x-y, N) \)

Analysis:

- Let \( \{a, b, c, d\} \) be the 4 square roots of \( x^2 \)
- has no idea which one you chose
- With probability \( \frac{1}{2} \), \( y \) will not be in \( \{+x, -x\} \)
- In this case, we know \( x = y \mod p \) but \( x = -y \mod q \)
Collision Resistance from Factoring

Let $N=pq$, $y$ a QR mod $N$
Suppose $-1$ is not a QR mod $N$

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

$H((N,y), (x,b))$: Let $z = y^bx^2 \mod N$
  • If $z \in \{1,\ldots,(N-1)/2\}$, output $z$
  • Else, output $-z \mod N \in \{1,\ldots,(N-1)/2\}$
Theorem: If the factoring assumption holds, $H$ is collision resistant

Proof:

• Collision means $(x_0, b_0) \neq (x_1, b_1)$ s.t.
  
  $$y^{b_0} x_0^2 = \pm y^{b_1} x_1^2 \mod N$$

• If $b_0 = b_1$, then $x_0 \neq x_1$, but $x_0^2 = \pm x_1^2 \mod N$
  • $x_0^2 = -x_1^2 \mod N$ 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 $(x_0/x_1)^2 = \pm y^{\pm 1} \mod N$
  • $-y$ 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 \( \mathbb{N} \) Necessary for SQ to be hard?

Let \( p \) be a prime, and suppose \( p = 3 \mod 4 \)

Given a QR \( x \mod p \), how to compute square root?

Hint: recall Fermat: \( x^{p-1} = 1 \mod p \) for all \( x \neq 0 \)

Hint: what is \( x^{(p+1)/2} \mod p \)?
Solving Quadratic Equations

In general, solving quadratic equations is:

• Easy over prime moduli

• As hard as factoring over composite moduli
Other Powers?

What about \( x \rightarrow x^4 \mod N \)? \( x \rightarrow x^6 \mod N \)?

The function \( x \rightarrow x^3 \mod N \) appears quite different

• Suppose 3 is relatively prime to \( p-1 \) and \( q-1 \)

• Then \( x \rightarrow x^3 \mod p \) is injective for \( x \neq 0 \)
  • Let \( a \) be such that \( 3a = 1 \mod p-1 \)
  • \( (x^3)^a = x^{1+k(p-1)} = x(x^{p-1})^k = x \mod p \)

• By CRT, \( x \rightarrow x^3 \mod N \) is injective for \( x \in \mathbb{Z}_N^* \)
\[ x^3 \mod N \]

What does injectivity mean?

Cannot base of factoring:
Adapt alg for square roots?

- Choose a random \[ z \mod N \]
- Compute \[ y = z^3 \mod N \]
- Run inverter on \( y \) to get a cube root \( x \)
- Let \( p = \text{GCD}(z-x, N) \), \( q = N/p \)
RSA Problem

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

Find $x$

Injectivity means cannot base hardness on factoring, but still conjectured to be hard
**RSA Assumption:** For any algorithm running in polynomial time, there exists negligible $\varepsilon$ such that:

$$\Pr[x \leftarrow (N, x^3 \mod N) \mid N = pq \text{ and } p, q \text{ random } \lambda\text{-bit primes s.t.} GCD(3, p-1) = GCD(3, q-1) = 1, x \leftarrow \mathbb{Z}_N^{*}] \leq \varepsilon(\lambda)$$
Application: PRGs

Let $F(x) = x^3 \mod N$, $h(x) =$ least significant bit

Theorem: If RSA Assumption holds, then $G(x) = (F(x), h(x))$ is a secure PRG
Reminders

HW5 Due April 9\textsuperscript{th}

PR2 Due April 19\textsuperscript{th}