COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Fall 2020

Announcements/Reminders

HW4 due Today

HW5 due Nov 10, will be released today
PR2 will be released soon

Heads up: Lecture 18 (next Tuesday) will be pre-recorded

Previously on COS 433…

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

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

Generalizing Cryptographic Groups

Replace fixed family of groups with "group generator" algorithm

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

Group

Generator for group

Group oder

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 λ-bit prime

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

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

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)←GrGen(λ), $a,b \leftarrow \mathbb{Z}_q$]

−Pr[1←  **(g,g^a,g^b,g^c):**

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

Today

Integer factorization
Public key cryptography

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**] ≤ ε(λ)**

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

Ex:

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

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

Pr[y^2=x^2 mod N:

y←**(N,x^2)**

N=pq, p,q←random **λ**-bit primes

x←$\mathbb{Z}_N$ **] ≤ ε(λ)**

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

Proof

To factor **N**:

- $\mathbf{x} \leftarrow \mathbb{Z}_\mathbf{N}$
- $\mathbf{y} \leftarrow$ $\mathbf{(N, x^2)}$
- 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: $\mathbf{\{1,\ldots,(N-1)/2\}}\times\mathbf{\{0,1\}}$
Range: $\mathbf{\{1,\ldots,(N-1)/2\}}$

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

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

Theorem: If the factoring assumption holds, **H** is collision resistant

Proof:

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

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

- 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 $\mathbf{N}$ Necessary for SQ to be hard?

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

Given a QR $\mathbf{x}$ mod $\mathbf{p}$, how to compute square root?

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

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

Solving Quadratic Equations

In general, solving quadratic equations is:

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

Other Powers?

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

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

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

$x^3 \bmod N$

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

- $N = pq$,
- e such that $GCD(e,p-1)=GCD(e,q-1)=1$,
- $y=x^e \bmod N$ 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, , $\exists$ negligible **ε** such that:

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

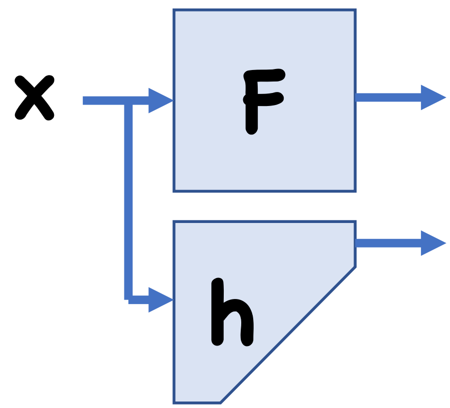
N=pq and **p,q** random **λ**-bit primes s.t.

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

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

Application: PRGs

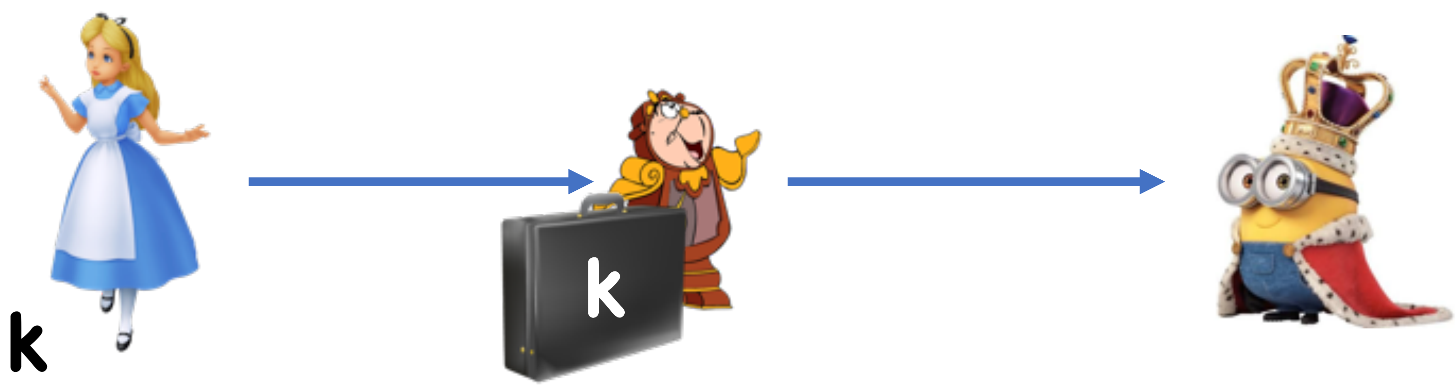
Let $\mathbf{F(x) = x^3 \bmod N}$, $\mathbf{h(x)}$ = least significant bit



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

Public Key Cryptography

How do Alice & Bob get **k**?



Limitations

Time consuming

Not realistic in many situations

- Do you really want to send a courier to every website you want to communicate with

Doesn’t scale well

- Imagine 1M people communicating with 1M people

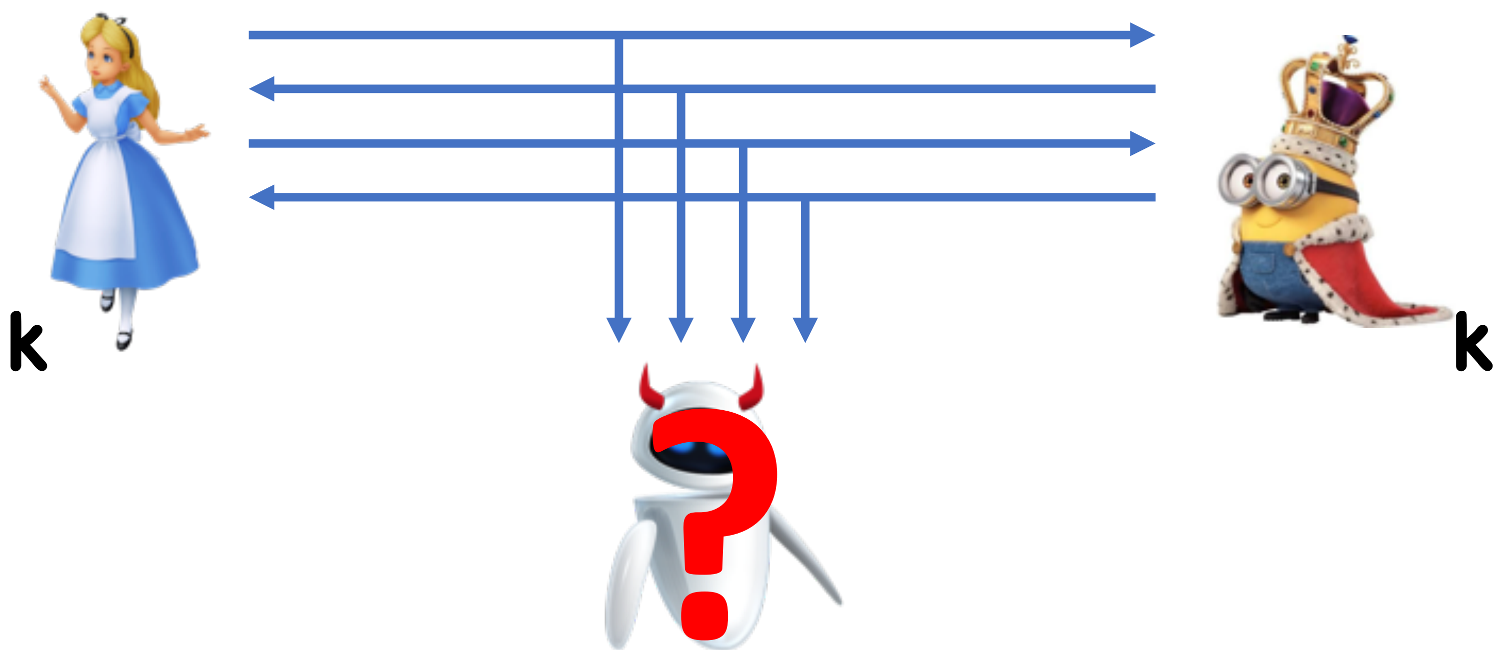
If not meeting in person, need to trust courier

Public Key Distribution

Public Key Distribution

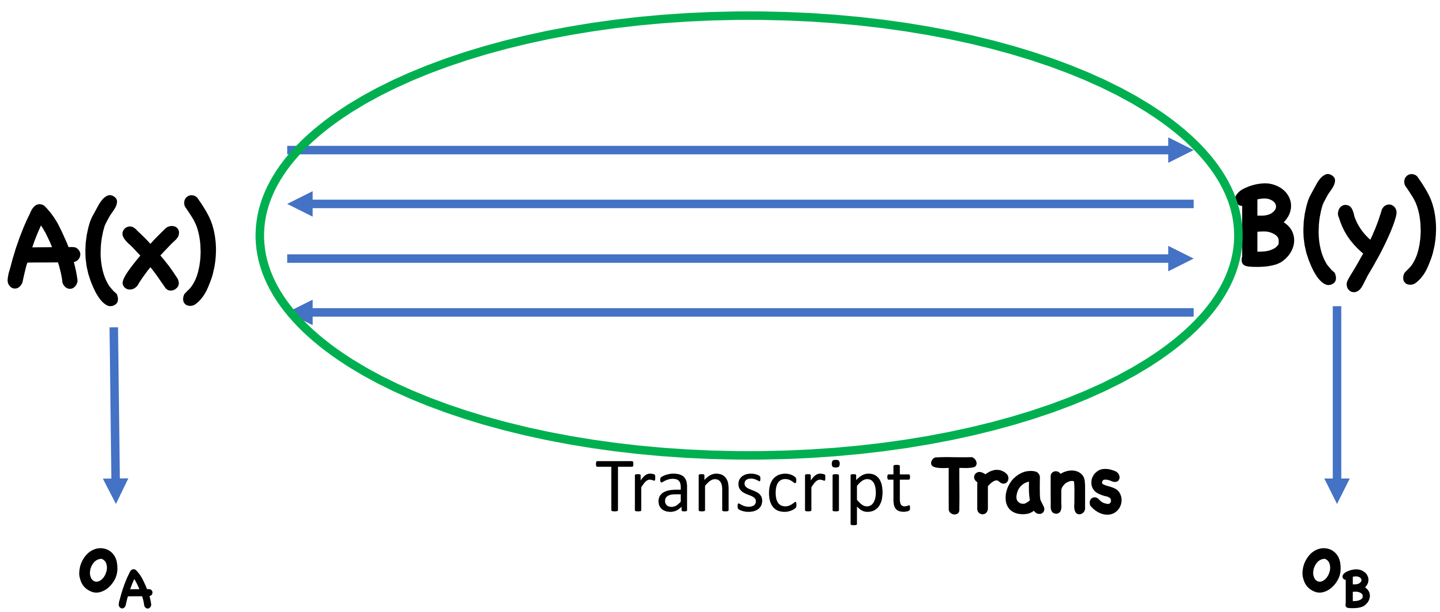
Public Key Distribution

Public Key Distribution



Interactive Protocols

Pair of interactive (randomized) algorithms **A, B**



Write $(\mathbf{Trans}, o_A, o_B) \leftarrow (A,B)(x,y)$

Public Key Distribution

Pair of interactive algorithms $\mathbf{A,B}$

Correctness:

$$\Pr[o_A=o_B:\ (\mathsf{Trans},o_A,o_B)\leftarrow(A,B)()] = 1$$

Shared key is $\mathbf{k} := o_A=o_B$

- Define $(\mathsf{Trans},k)\leftarrow(A,B)()$

Security: (Trans,k) is computationally indistinguishable from (Trans,k') where $k'\leftarrow K$ independent of $\mathbf{k}$

Matrix Multiplication Approach

$\xrightarrow{B}$

$B \leftarrow \mathbb{Z}_q^{\lambda \times \lambda}$

$A \leftarrow B^{-1}$

Matrix Multiplication Approach

$$\xrightarrow{\mathbf{B}}$$

$$\xleftarrow{\mathbf{w}}$$

$\mathbf{B} \leftarrow \mathbb{Z}_q^{\lambda\times\lambda}$
$\mathbf{A} \leftarrow \mathbf{B}^{-1}$

$\mathbf{v} \leftarrow \mathbb{Z}_q^{\lambda}$
$\mathbf{w} \leftarrow \mathbf{B}\cdot\mathbf{v}$

Matrix Multiplication Approach

$\xrightarrow{B}$

$\xleftarrow{w}$

$B \leftarrow \mathbb{Z}_q^{\lambda\times\lambda}$

$A \leftarrow B^{-1}$

$\downarrow$

$v = A\cdot w$

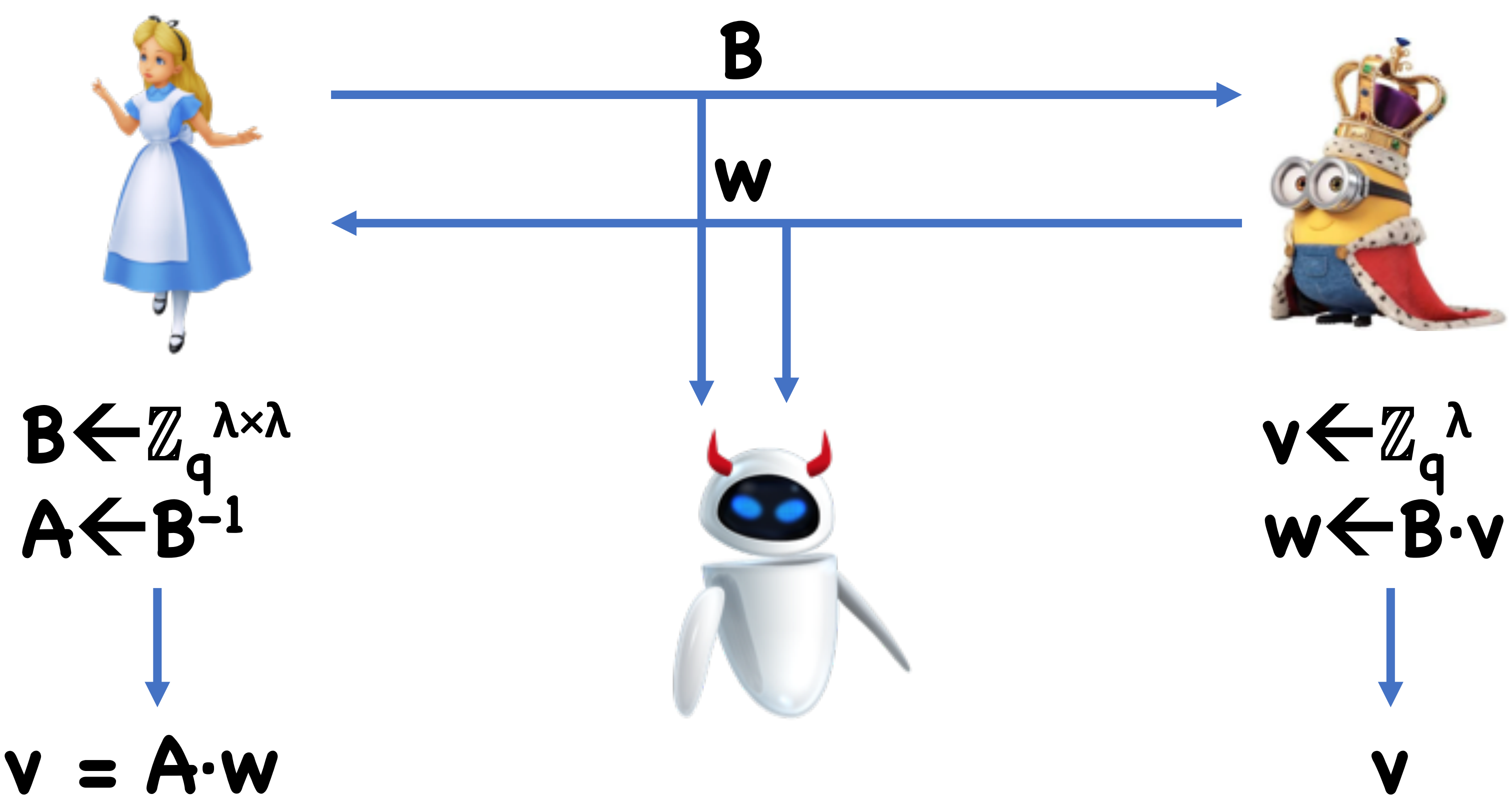
$v \leftarrow \mathbb{Z}_q^{\lambda}$

$w \leftarrow B\cdot v$

$\downarrow$

v

Matrix Multiplication Approach



Running Times?

Bob: $\boldsymbol{O(\lambda^2)}$
Eve: $\boldsymbol{O(\lambda^3)}$

Running Times?

Bob: $O(\lambda^2)$
Eve: $O(\lambda^\omega)$ where $\omega \leq 2.373$
Alice: $O(\lambda^\omega)$

Different Approach:

- Start with $A = B = I$
- Repeatedly apply random elementary row ops to A, inverse to B
- Output (A,B)

Running Times?

Bob: $O(\lambda^2)$
Eve: $O(\lambda^\omega)$ where $\omega \leq 2.373$
Alice: $O(\lambda^\omega)$

Assuming Matrix Multiplication exponent $\omega > 2$, adversary must work harder than honest users

inverse to B

- Output (A,B)