トポロジーを暗号に応用したい
How to Apply Topology to Cryptology, Hopefully

縫田　光司（Koji NUIDA）

産業技術総合研究所 (AIST)

トポロジーとコンピュータ 2016　　2016 年 10 月 29 日

Commuting Operations

Mr. X "May I ask you to give a talk?"

↓

N. "(Hmm..., oh, I got topic to talk!)"

↓

N. "Sure, I can talk!"

Commuting Operations

Mr. X “May I ask you to give a talk?”

N. “(Hmm..., oh, I got topic to talk!)”

「もしかして」

N. “Sure, I can talk!”

Commuting Operations

Mr. X “May I ask you to give a talk?”

N. “(Hmm..., oh, I got topic to talk!)”

「もしかして」

N. “Sure, I can talk!”

「わたしたち」

Commuting Operations (Not Recommended)

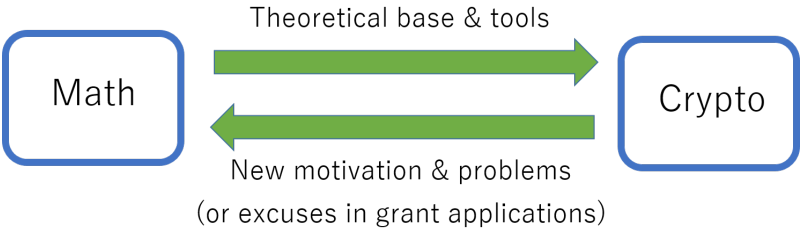
Mr. X “May I ask you to give a talk?”

N. “Sure, I can talk!”

「入れ替わってる〜〜！？」

N. “(Hmm..., oh, I got topic to talk!)”

Relation between Mathematics and Cryptography



Q. How about topology?

Public Key Encryption (PKE)

- To conceal messages from attackers

Public Key Encryption (PKE)

- To conceal messages from attackers
- Encryption: message $\mapsto$ ciphertext
  - using **public** encryption key pk

Public Key Encryption (PKE)

- To conceal messages from attackers
- Encryption: message $\mapsto$ ciphertext
  - using **public** encryption key pk
- Decryption: ciphertext $\mapsto$ message
  - using **secret** decryption key sk

Public Key Encryption (PKE)

- To conceal messages from attackers
- Encryption: message $\mapsto$ ciphertext
  - using **public** encryption key pk
- Decryption: ciphertext $\mapsto$ message
  - using **secret** decryption key sk
- $\mathrm{Dec}_{\mathrm{sk}}(\mathrm{Enc}_{\mathrm{pk}}(m)) = m$

Public Key Encryption (PKE)

- To conceal messages from attackers
- Encryption: message $\mapsto$ ciphertext
  - using **public** encryption key pk
- Decryption: ciphertext $\mapsto$ message
  - using **secret** decryption key sk
- $\mathrm{Dec}_{\mathrm{sk}}(\mathrm{Enc}_{\mathrm{pk}}(m)) = m$
- pk **should not yield information on** sk

The RSA Cryptosystem [1977 ?]

- $N = pq$ (distinct primes)
- e, d with $ed \equiv 1 \pmod{(p-1)(q-1)}$

Given message $m \in (\mathbb{Z}/N\mathbb{Z})^\times$,

- $\mathrm{Enc}(m) := m^e$ (public key: (N, e))
- $\mathrm{Dec}(c) := c^d$ (secret key: d)

The RSA Cryptosystem [1977 ?]

- $N = pq$ (distinct primes)
- e, d with $ed \equiv 1 \pmod{(p-1)(q-1)}$

Given message $m \in (\mathbb{Z}/N\mathbb{Z})^\times$,

- $\mathrm{Enc}(m) := m^e$ (public key: (N, e))
- $\mathrm{Dec}(c) := c^d$ (secret key: d)

d **would be computable if** p, q **were known**

- $N = pq$ (distinct primes)
- e, d with $ed \equiv 1 \pmod{(p-1)(q-1)}$

Given message $m \in (\mathbb{Z}/N\mathbb{Z})^\times$,

- $\mathrm{Enc}(m) := m^e$ (public key: (N, e))
- $\mathrm{Dec}(c) := c^d$ (secret key: d)

d **would be computable if** p, q **were known**

Drawback: Enc is deterministic ("textbook RSA")

- Improved variant is practically used

Computational Assumptions

In PKE, secret should not be found in "practical" (theoretically, probabilistic polynomial) time

- E.g. "Factoring N is hard" for the RSA

Computational Assumptions

In PKE, secret should not be found in "practical" (theoretically, probabilistic polynomial) time

- E.g. "Factoring N is hard" for the RSA
- Theoretically, just "assumption" (cf. P vs NP)
    - Practically, evaluated by experiments
    - Consensus: "(General) Number Field Sieve" would factorize $N \approx 2^{1024}$ in near future

Prior to RSA — Diffie–Hellman Key Exchange [1976]

Protocol between parties P_1 and P_2

1.
2.

Getting a common (random) **secret element**

with **no pre-shared secret**

Protocol between parties P_1 and P_2

Choose $G = \langle g \rangle$ (finite cyclic) in public, then

1.
2.

Getting a common (random) **secret element**

with **no pre-shared secret**

Protocol between parties P_1 and P_2

Choose $G = \langle g \rangle$ (finite cyclic) in public, then

1. P_i sends $h_i := g^{a_i}$, while hiding $a_i \in \mathbb{Z}$
2.

Getting a common (random) **secret element**

with **no pre-shared secret**

Protocol between parties P_1 and P_2

Choose $G = \langle g \rangle$ (finite cyclic) in public, then

1. P_i sends $h_i := g^{a_i}$, while hiding $a_i \in \mathbb{Z}$
2. Given h_{3-i}, P_i computes $K_i := {h_{3-i}}^{a_i}$

Getting a common (random) **secret element**

with **no pre-shared secret**

Protocol between parties P_1 and P_2

Choose $G = \langle g \rangle$ (finite cyclic) in public, then

1. P_i sends $h_i := g^{a_i}$, while hiding $a_i \in \mathbb{Z}$
2. Given h_{3-i}, P_i computes $K_i := {h_{3-i}}^{a_i}$

Getting a common (random) **secret element**

$$K_1 = (g^{a_2})^{a_1} = g^{a_2 a_1} = g^{a_1 a_2} = (g^{a_1})^{a_2} = K_2$$

with **no pre-shared secret**

Protocol between parties P_1 and P_2

Choose $G = \langle g \rangle$ (finite cyclic) in public, then

1. P_i sends $h_i := g^{a_i}$, while hiding $a_i \in \mathbb{Z}$
2. Given h_{3-i}, P_i computes $K_i := {h_{3-i}}^{a_i}$

Getting a common (random) **secret element**

$$K_1 = (g^{a_2})^{a_1} = g^{a_2 a_1} = g^{a_1 a_2} = (g^{a_1})^{a_2} = K_2$$

with **no pre-shared secret**

- Can be converted to PKE [ElGamal 1985]

Is DH Key Exchange Secure?

Public: $G = \langle g \rangle$ and $h_i \in G$

Secret: a_i with $h_i = g^{a_i}$

Is DH Key Exchange Secure?

Public: $G = \langle g \rangle$ and $h_i \in G$
Secret: a_i with $h_i = g^{a_i}$

$\Rightarrow$ The **discrete logarithm problem** (DL) in G must be computationally hard:

(DL) $\boxed{\text{Given } g, h, \text{ find } x \text{ with } h = g^x \text{ in } G}$

- Remark: (In)sufficiency is still open

Choice of the Group for Security (1/2)

Choice of the Group for Security (1/2)

Q1. $x \cdot 7 = 15$ in $\mathbb{Z}/16\mathbb{Z}$? ...

Choice of the Group for Security (1/2)

Q1. $x \cdot 7 = 15$ in $\mathbb{Z}/16\mathbb{Z}$? ... $x = 9$

Choice of the Group for Security (1/2)

Q1. $x \cdot 7 = 15$ in $\mathbb{Z}/16\mathbb{Z}$? ... $x = 9$

Q2. $10^x = 6$ in $\mathbb{F}_{17}^{\times}$? ...

Q1. $x \cdot 7 = 15$ in $\mathbb{Z}/16\mathbb{Z}$? ... $x = 9$

Q2. $10^x = 6$ in $\mathbb{F}_{17}^{\times}$? ... $x = 5$

Choice of the Group for Security (1/2)

Q1. $x \cdot 7 = 15$ in $\mathbb{Z}/16\mathbb{Z}$? ... $x = 9$

Q2. $10^x = 6$ in $\mathbb{F}_{17}^{\times}$? ... $x = 5$

Q2 looks more difficult than Q1,
though $\mathbb{Z}/16\mathbb{Z} \simeq \mathbb{F}_{17}^{\times}$ as groups

Choice of the Group for Security (1/2)

Q1. $x \cdot 7 = 15$ in $\mathbb{Z}/16\mathbb{Z}$? ... $x = 9$

Q2. $10^x = 6$ in $\mathbb{F}_{17}^\times$? ... $x = 5$

Q2 looks more difficult than Q1,
though $\mathbb{Z}/16\mathbb{Z} \simeq \mathbb{F}_{17}^\times$ as groups

$\Rightarrow$ Difficulty of DL **does depend** on
a "realization" of the same abstract group G!

Choice of the Group for Security (2/2)

Efficient solution for Q1:
use (Extended) Euclidean Algorithm

Choice of the Group for Security (2/2)

Efficient solution for Q1:
use (Extended) Euclidean Algorithm

- which uses integer **division** (or **ordering**)

Choice of the Group for Security (2/2)

Efficient solution for Q1:
use (Extended) Euclidean Algorithm

- which uses integer **division** (or **ordering**)
- for the DL in **additive group** $\mathbb{Z}/n\mathbb{Z}$!

Choice of the Group for Security (2/2)

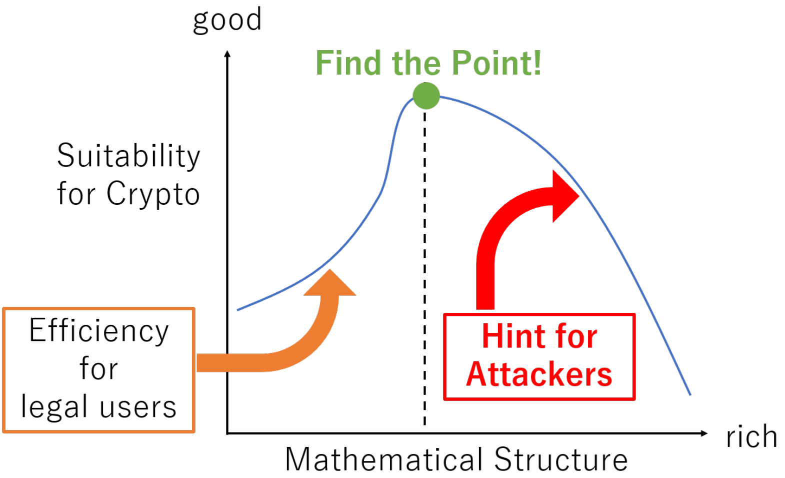
Efficient solution for Q1:
use (Extended) Euclidean Algorithm

- which uses integer **division** (or **ordering**)
- for the DL in **additive group** $\mathbb{Z}/n\mathbb{Z}$!

A lesson: Additional structure for group G makes the DL easier ($\Rightarrow$ break of DH Key Exchange)

- Cf. [Maurer 2005] DL is hard in "generic group"
  - "Oracle access to multiplication table only"

A New Viewpoint from Cryptography



(Mathematician: more structures, more happiness)

State-of-the-Art Candidate from Elliptic Curves

(Subgroups of) groups of rational points on **elliptic curves** (over finite fields)

State-of-the-Art Candidate from Elliptic Curves

(Subgroups of) groups of rational points on **elliptic curves** (over finite fields)

- Additive group structure

State-of-the-Art Candidate from Elliptic Curves

(Subgroups of) groups of rational points on **elliptic curves** (over finite fields)

- Additive group structure
- **Other structures are not known well** (in comparison to $\mathbb{Z}/n\mathbb{Z}$ and ${\mathbb{F}_q}^{\times}$)

(Subgroups of) groups of rational points on **elliptic curves** (over finite fields)

- Additive group structure
- **Other structures are not known well** (in comparison to $\mathbb{Z}/n\mathbb{Z}$ and $\mathbb{F}_q{}^{\times}$)

Current status: $|G| \gtrapprox 2^{160}$

- Cf. $N \gtrapprox 2^{1024}$ for the RSA

Shor's Quantum Algorithms

Quantum computer: framework of fast computation using superimposed quantum states

- Not practically implemented so far

Quantum computer: framework of fast computation using superimposed quantum states

- Not practically implemented so far

[Shor 1994]: Quantum algorithms, implying

Quantum computer: framework of fast computation using superimposed quantum states

- Not practically implemented so far

[Shor 1994]: Quantum algorithms, implying

- integer factoring **in polynomial time**!
- discrete logarithm **in polynomial time**!

(Cf. [Grover 1996]: Search with quadratic speedup)

Pinpoint Effect by Shor to Cryptography

Shor's main applications: integer factoring and DL

Pinpoint Effect by Shor to Cryptography

Shor's main applications: integer factoring and DL

Main tools of PKE: integer factoring and DL

- Oh, My God!

Pinpoint Effect by Shor to Cryptography

Shor's main applications: integer factoring and DL

Main tools of PKE: integer factoring and DL

- Oh, My God!

$\longrightarrow$ Importance of "quantum-resistant" PKE

- (Believed to be) unbroken by quantum computer

Candidates of quantum-resistant PKE

Based on (conjectural) hardness of solving:

- Knapsack problem
- System of multivariate quadratic equations
- Decoding random linear codes
- Shortest vectors in integer lattices
- Finding sections on algebraic surfaces
- Finding isogeny between elliptic curves
- ...

Candidates of quantum-resistant PKE

Based on (conjectural) hardness of solving:

- Knapsack problem (not good)
- System of multivariate quadratic equations (fair)
- Decoding random linear codes (sometimes good)
- Shortest vectors in integer lattices (hopeful)
- Finding sections on algebraic surfaces (?)
- Finding isogeny between elliptic curves (?)
- ...

Remark: P vs. "Break of Cryptosystem"

Given any algorithm (attacker),

Remark: P vs. "Break of Cryptosystem"

Given any algorithm (attacker),

"non-P" means

- **At least one** problem instance is hard to solve

Remark: P vs. “Break of Cryptosystem”

Given any algorithm (attacker),

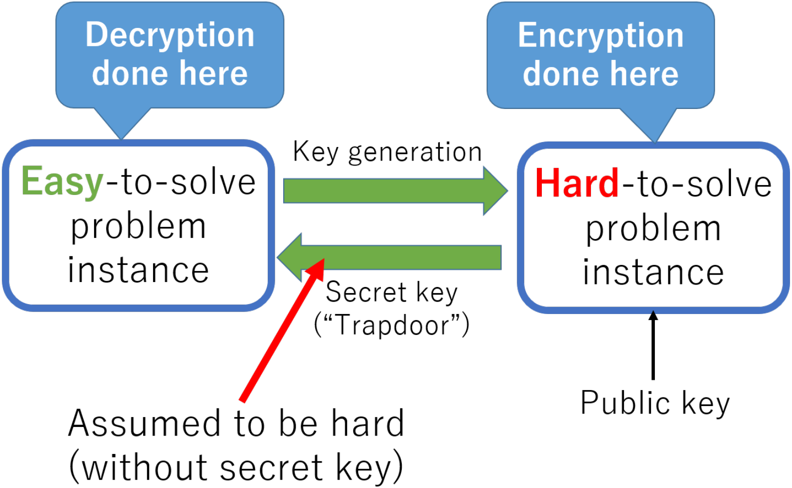
“non-P” means

- **At least one** problem instance is hard to solve

“Security of PKE” means

- “**Almost all**” ciphertexts are hard to break

A Major Strategy for PKE



Multi-Party Computation (MPC): Two-Party Case

Given function $f(x, y)$ (e.g., $f(x, y) = \delta_{x,y}$),

- Party P_1 has secret input a_1
- Party P_2 has secret input a_2
- They want to know $f(a_1, a_2)$ by communication

Multi-Party Computation (MPC): Two-Party Case

Given function $f(x, y)$ (e.g., $f(x, y) = \delta_{x,y}$),

- Party P_1 has secret input a_1
- Party P_2 has secret input a_2
- They want to know $f(a_1, a_2)$ by communication
- while **hiding information on each input!**
  - (except those trivially implied from $f(a_1, a_2)$)

A Tool for MPC: Homomorphic Encryption (HE)

Example: additively-HE

Example: additively-HE

- Message set $\mathcal{M}$ is additive group

A Tool for MPC: Homomorphic Encryption (HE)

Example: additively-HE

- Message set $\mathcal{M}$ is additive group
- A "practical" operation $\boxplus$ for ciphertexts with

$$\mathrm{Dec}(c_1 \boxplus c_2) = \mathrm{Dec}(c_1) + \mathrm{Dec}(c_2) \in \mathcal{M}$$

(called "homomorphic operation")

A Tool for MPC: Homomorphic Encryption (HE)

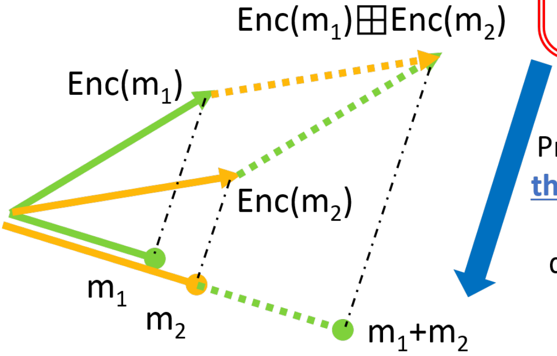
Example: additively-HE

- Message set $\mathcal{M}$ is additive group
- A "practical" operation $\boxplus$ for ciphertexts with

$$\mathrm{Dec}(c_1 \boxplus c_2) = \mathrm{Dec}(c_1) + \mathrm{Dec}(c_2) \in \mathcal{M}$$

  (called "homomorphic operation")
- "Messages can be added in encrypted form"

A "Rough Idea" for HE



Direction should be hard to find without key (art of cryptographer)

Projection in **this direction** yields decryption

Example of (Multiplicative) HE [ElGamal 1985]

Example of (Multiplicative) HE [ElGamal 1985]

Public key: $G = \langle g \rangle$ (prime order), $h \in G$
Secret key: $s \in \mathbb{Z}$ with $h = g^s$

Example of (Multiplicative) HE [ElGamal 1985]

Public key: $G = \langle g \rangle$ (prime order), $h \in G$

Secret key: $s \in \mathbb{Z}$ with $h = g^s$

- Given $m \in G$, $\mathrm{Enc}(m) := (g^r, h^r m) \in G^2$
    - where $r \in \mathbb{Z}$ is random

Example of (Multiplicative) HE [ElGamal 1985]

Public key: $G = \langle g \rangle$ (prime order), $h \in G$
Secret key: $s \in \mathbb{Z}$ with $h = g^s$

- Given $m \in G$, $\mathsf{Enc}(m) := (g^r, h^r m) \in G^2$
  - where $r \in \mathbb{Z}$ is random
- Given $c = (c_1, c_2)$, $\mathsf{Dec}(c) := {c_1}^{-s} c_2$
  - "Project to $(g^0, g^{\mathbb{Z}})$ in direction (g^1, g^{-s})"

Example of (Multiplicative) HE [ElGamal 1985]

Public key: $G = \langle g \rangle$ (prime order), $h \in G$
Secret key: $s \in \mathbb{Z}$ with $h = g^s$

- Given $m \in G$, $\mathrm{Enc}(m) := (g^r, h^r m) \in G^2$
  - where $r \in \mathbb{Z}$ is random
- Given $c = (c_1, c_2)$, $\mathrm{Dec}(c) := {c_1}^{-s} c_2$
  - "Project to $(g^0, g^{\mathbb{Z}})$ in direction (g^1, g^{-s})"
- Homomorphic operation: multiplication in G^2

How to compute δ_{a_1,a_2} (Notation: $[[a]] := \mathsf{Enc}(a)$)

Suppose: additively-HE with $\mathcal{M} = \mathbb{F}_p$

Example of MPC from HE

How to compute δ_{a_1,a_2} (Notation: $[[a]] := \mathsf{Enc}(a)$)

Suppose: additively-HE with $\mathcal{M} = \mathbb{F}_p$

1. P_1 chooses key, sends public key only

Example of MPC from HE

How to compute δ_{a_1,a_2} (Notation: $[[a]] := \mathrm{Enc}(a)$)

Suppose: additively-HE with $\mathcal{M} = \mathbb{F}_p$

1. P_1 chooses key, sends public key only
2. P_1 generates and sends $[[a_1]]$

Example of MPC from HE

How to compute δ_{a_1,a_2} (Notation: $[[a]] := \mathrm{Enc}(a)$)

Suppose: additively-HE with $\mathcal{M} = \mathbb{F}_p$

1. P_1 chooses key, sends public key only
2. P_1 generates and sends $[[a_1]]$
3. P_2 computes $[[a_1]] \boxplus [[-a_2]] = [[a_1 - a_2]]$

Example of MPC from HE

How to compute δ_{a_1,a_2} (Notation: $[[a]] := \mathrm{Enc}(a)$)

Suppose: additively-HE with $\mathcal{M} = \mathbb{F}_p$

1. P_1 chooses key, sends public key only
2. P_1 generates and sends $[[a_1]]$
3. P_2 computes $[[a_1]] \boxplus [[-a_2]] = [[a_1 - a_2]]$
4. P_2 computes $[[r(a_1 - a_2)]]$ for random $r \neq 0$
    - by random iteration of $\boxplus$ to $[[a_1 - a_2]]$

Example of MPC from HE

How to compute δ_{a_1,a_2} (Notation: $[[a]] := \mathrm{Enc}(a)$)

Suppose: additively-HE with $\mathcal{M} = \mathbb{F}_p$

1. P_1 chooses key, sends public key only
2. P_1 generates and sends $[[a_1]]$
3. P_2 computes $[[a_1]] \boxplus [[-a_2]] = [[a_1 - a_2]]$
4. P_2 computes $[[r(a_1 - a_2)]]$ for random $r \neq 0$
    - by random iteration of $\boxplus$ to $[[a_1 - a_2]]$
5. P_1 decrypts $[[r(a_1 - a_2)]] \rightsquigarrow 0$ iff $a_1 = a_2$

Fully Homomorphic Encryption [2009]

(Additively-)HE: "addition in encrypted form"

Fully Homomorphic Encryption [2009]

(Additively-)HE: “addition in encrypted form”

Fully homomorphic encryption (FHE):

Any computation in encrypted form

Fully Homomorphic Encryption [2009]

(Additively-)HE: “addition in encrypted form”

Fully homomorphic encryption (FHE):

Any computation in encrypted form

- $\Leftrightarrow$ Ring-HE, when $\mathcal{M} = \mathbb{F}_p$ (p prime)

(Too) Simplified Example [2010] [N. et al. 2015]

$\mathbb{Z}/\ell\mathbb{Z}$ identified with $\{0, \ldots, \ell - 1\}$ by "mod"

Choose $p' \gg p$ primes, $p' \mid N$

(Too) Simplified Example [2010] [N. et al. 2015]

$\mathbb{Z}/\ell\mathbb{Z}$ identified with $\{0, \dots, \ell - 1\}$ by "mod"

Choose $p' \gg p$ primes, $p' \mid N$

$\mathrm{Enc}(m) = r'p' + rp + m \bmod N$ for $m \in \mathbb{F}_p$

(Too) Simplified Example [2010] [N. et al. 2015]

$\mathbb{Z}/\ell\mathbb{Z}$ identified with $\{0, \ldots, \ell - 1\}$ by "mod"

Choose $p' \gg p$ primes, $p' \mid N$

$\mathrm{Enc}(m) = r'p' + rp + m \bmod N$ for $m \in \mathbb{F}_p$

$\mathrm{Dec}(c) = (c \bmod p') \bmod p$

- Decryption works iff r is "not too large"

(Too) Simplified Example [2010] [N. et al. 2015]

$\mathbb{Z}/\ell\mathbb{Z}$ identified with $\{0, \ldots, \ell - 1\}$ by "mod"

Choose $p' \gg p$ primes, $p' \mid N$

$\mathrm{Enc}(m) = r'p' + rp + m \bmod N$ for $m \in \mathbb{F}_p$

$\mathrm{Dec}(c) = (c \bmod p') \bmod p$

- Decryption works iff r is "not too large"

Ring-homomorphic operations: as usual in $\mathbb{Z}/N\mathbb{Z}$

- but iteration of operations is limited! (r grows)

(Too) Simplified Example [2010] [N. et al. 2015]

$\mathbb{Z}/\ell\mathbb{Z}$ identified with $\{0, \ldots, \ell - 1\}$ by "mod"

Choose $p' \gg p$ primes, $p' \mid N$

$\mathrm{Enc}(m) = r'p' + rp + m \bmod N$ for $m \in \mathbb{F}_p$

$\mathrm{Dec}(c) = (c \bmod p') \bmod p$

- Decryption works iff r is "not too large"

Ring-homomorphic operations: as usual in $\mathbb{Z}/N\mathbb{Z}$

- but iteration of operations is limited! (r grows)

"Bootstrapping": refreshing the ciphertext

- possible, but very inefficient

A (hopefully) possible strategy:

Towards "Unlimited" Ring-HE

A (hopefully) possible strategy:

1. "Embed" $\mathbb{F}_p$ into a (non-commutative) group G
   - Operations of $\mathbb{F}_p$ realized by operations of G

Towards "Unlimited" Ring-HE

A (hopefully) possible strategy:

1. "Embed" $\mathbb{F}_p$ into a (non-commutative) group G
   - Operations of $\mathbb{F}_p$ realized by operations of G
2. Take a lift of G (e.g., $G \times H$ for suitable H)

A (hopefully) possible strategy:

1. "Embed" $\mathbb{F}_p$ into a (non-commutative) group G
   - Operations of $\mathbb{F}_p$ realized by operations of G
2. Take a lift of G (e.g., $G \times H$ for suitable H)
3. "Homomorphically hide" the structure of the lift

A (hopefully) possible strategy:

1. "Embed" $\mathbb{F}_p$ into a (non-commutative) group G
   - Operations of $\mathbb{F}_p$ realized by operations of G
2. Take a lift of G (e.g., $G \times H$ for suitable H)
3. "Homomorphically hide" the structure of the lift

$\rightsquigarrow$ hard-to-compute group hom. $\varphi \colon \widetilde{G} \twoheadrightarrow G$

- must be easy-to-compute with secret key
- Public: G and generators of $\ker \varphi$ (for Enc)

How to "Realize" $\mathbb{F}_2$ in $\mathrm{PSL}_2(\mathbb{F}_q)$

[N. 2014 (preprint)]

$G := \mathrm{PSL}_2(\mathbb{F}_q)$, $q \gg 1$

How to “Realize” $\mathbb{F}_2$ in $\mathrm{PSL}_2(\mathbb{F}_q)$

[N. 2014 (preprint)]

$G := \mathrm{PSL}_2(\mathbb{F}_q)$, $q \gg 1$

$X_0 := \{c = (c_1, c_2) \in G^2 \mid c_1 \neq 1, c_2 = 1\}$

$X_1 := \{c = (c_1, c_2) \in G^2 \mid c_1 \neq 1, c_2 = c_1\}$

[N. 2014 (preprint)]

$G := \mathrm{PSL}_2(\mathbb{F}_q)$, $q \gg 1$
$X_0 := \{c = (c_1, c_2) \in G^2 \mid c_1 \neq 1, c_2 = 1\}$
$X_1 := \{c = (c_1, c_2) \in G^2 \mid c_1 \neq 1, c_2 = c_1\}$
$X_b \ni (c_1, c_2) \mapsto (c_1, c_1 {c_2}^{-1}) \in X_{1-b}$ ($\rightsquigarrow$ “NOT”)

[N. 2014 (preprint)]

$G := \mathrm{PSL}_2(\mathbb{F}_q)$, $q \gg 1$
$X_0 := \{c = (c_1, c_2) \in G^2 \mid c_1 \neq 1, c_2 = 1\}$
$X_1 := \{c = (c_1, c_2) \in G^2 \mid c_1 \neq 1, c_2 = c_1\}$
$X_b \ni (c_1, c_2) \mapsto (c_1, c_1 {c_2}^{-1}) \in X_{1-b}$ ($\rightsquigarrow$ "NOT")
For $c, c' \in X_0 \cup X_1$, define (with random $g \in G$)

$$[c, c']^{\dagger} := ([g^{-1} c_1 g, c'_1], [g^{-1} c_2 g, c'_2])$$

[N. 2014 (preprint)]

$G := \mathrm{PSL}_2(\mathbb{F}_q)$, $q \gg 1$
$X_0 := \{c = (c_1, c_2) \in G^2 \mid c_1 \neq 1, c_2 = 1\}$
$X_1 := \{c = (c_1, c_2) \in G^2 \mid c_1 \neq 1, c_2 = c_1\}$
$X_b \ni (c_1, c_2) \mapsto (c_1, c_1 {c_2}^{-1}) \in X_{1-b}$ ($\rightsquigarrow$ "NOT")
For $c, c' \in X_0 \cup X_1$, define (with random $g \in G$)

$$[c, c']^\dagger := ([g^{-1} c_1 g, c'_1], [g^{-1} c_2 g, c'_2])$$

With probability $\approx 1 - q^{-1}$ we have: ($\rightsquigarrow$ "AND")

- If $c, c' \in X_1$, then $[c, c']^\dagger \in X_1$
- Otherwise, $[c, c']^\dagger \in X_0$

How to Realize NAND Gate in Simple Groups

$\mathrm{NAND}(b_1, b_2) = 0$ iff $b_1 = b_2 = 1$
(Compositions of NAND yield $\mathrm{AND}, \mathrm{OR}, \mathrm{NOT}, \dots$)

How to Realize NAND Gate in Simple Groups

$\mathrm{NAND}(b_1, b_2) = 0$ iff $b_1 = b_2 = 1$
(Compositions of NAND yield AND, OR, NOT, ...)

[Ostrovsky–Skeith 2008] For any non-commutative finite simple group G, there exist $g_0 \neq g_1 \in G$ and $F\colon G^2 \to G$ with:

$\mathrm{NAND}(b_1, b_2) = 0$ iff $b_1 = b_2 = 1$
(Compositions of NAND yield $\mathrm{AND}, \mathrm{OR}, \mathrm{NOT}, \dots$)

[Ostrovsky–Skeith 2008] For any non-commutative finite simple group G, there exist $g_0 \neq g_1 \in G$ and $F \colon G^2 \to G$ with:

- $F(g_1, g_1) = g_0$
- $F(g_0, g_0) = F(g_0, g_1) = F(g_1, g_0) = g_1$

$\mathrm{NAND}(b_1, b_2) = 0$ iff $b_1 = b_2 = 1$
(Compositions of NAND yield $\mathrm{AND}, \mathrm{OR}, \mathrm{NOT}, \dots$)

[Ostrovsky–Skeith 2008] For any non-commutative finite simple group G, there exist $g_0 \neq g_1 \in G$ and $F \colon G^2 \to G$ with:

- $F(g_1, g_1) = g_0$
- $F(g_0, g_0) = F(g_0, g_1) = F(g_1, g_0) = g_1$
- **F is composed of group operations in G**

$\mathrm{NAND}(b_1, b_2) = 0$ iff $b_1 = b_2 = 1$
(Compositions of NAND yield $\mathrm{AND}, \mathrm{OR}, \mathrm{NOT}, \dots$)

[Ostrovsky–Skeith 2008] For any non-commutative finite simple group G, there exist $g_0 \neq g_1 \in G$ and $F \colon G^2 \to G$ with:

- $F(g_1, g_1) = g_0$
- $F(g_0, g_0) = F(g_0, g_1) = F(g_1, g_0) = g_1$
- **F is composed of group operations in G**

(Proof idea: $\langle \text{commutators} \rangle_{\mathrm{normal}} = G$)

My recent (very rough) idea:

My recent (very rough) idea:

1. Presentation of $G \times H$ by generators/relations

My recent (very rough) idea:

1. Presentation of $G \times H$ by generators/relations
2. "Shuffle" presentation by randomly applying Tietze transformations

My recent (very rough) idea:

1. Presentation of $G \times H$ by generators/relations
2. "Shuffle" presentation by randomly applying Tietze transformations
3. Apply Knuth-Bendix completion algorithm, to yield normal form of each group element
    - Otherwise, Enc is also hard-to-compute

Towards Homomorphically Hiding the Group

My recent (very rough) idea:

1. Presentation of $G \times H$ by generators/relations
2. "Shuffle" presentation by randomly applying Tietze transformations
3. Apply Knuth-Bendix completion algorithm, to yield normal form of each group element
   - Otherwise, Enc is also hard-to-compute

Current problems:

- Knuth-Bendix algorithm may not terminate
- Is it really secure?

How to Apply Topology, Hopefully

Goal: Hard-to-compute $\varphi\colon \widetilde{G} \xrightarrow{hom.} G$ with G and generators of $\ker \varphi$ public

- Easy-to-compute if secret key is known
- Message set is "embedded" into G

How to Apply Topology, Hopefully

Goal: Hard-to-compute $\varphi\colon \widetilde{G} \xrightarrow{hom.} G$ with G and generators of $\ker\varphi$ public

- Easy-to-compute if secret key is known
- Message set is “embedded” into G

Questions:

How to Apply Topology, Hopefully

Goal: Hard-to-compute $\varphi\colon \widetilde{G} \xrightarrow{hom.} G$ with G and generators of $\ker\varphi$ public

- Easy-to-compute if secret key is known
- Message set is "embedded" into G

Questions:

- Good $\widetilde{G}$, associated to some topological object? (Cf. groups from elliptic curves)

Goal: Hard-to-compute $\varphi\colon \widetilde{G} \xrightarrow{hom.} G$ with G and generators of $\ker\varphi$ public

- Easy-to-compute if secret key is known
- Message set is "embedded" into G

Questions:

- Good $\widetilde{G}$, associated to some topological object? (Cf. groups from elliptic curves)
- Embedding into other topology-related objects? (E.g., quandles from knot theory)