ZKP Board Game

組長	111550097	蔣昀成
組員	111550020	方漢霖
組員	111550135	林李奕

Outline

01

Abstract

02

Introduction

03

The Algorithm

04

System Architecture

05

Experiment

06

Contribution

Abstract

Decentralized card game

- Independent decks for each player
- Prove validity of deck
- Prove whether a card exists in a chosen subset of current hand
- All proof done without revealing other information

Introduction

Da Vinci Code

- Speculate number of the card
- Cards are sorted in ascending order, white cards are considered bigger than black ones with same number
- Correctly guessed card must be revealed

Public deck

→ Players draw cards from the same deck

Proof of ordering

→ Cards should be sorted in the correct order

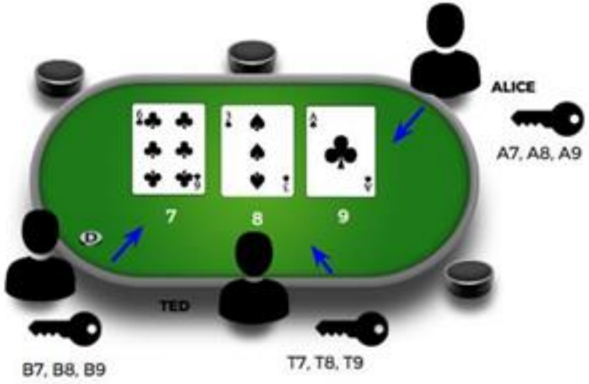
Proof of response

→ Prove the guess is wrong without revealing the card

Introduction

Mental Poker

- Poker game without trusted third party
- First mentioned by Rivest, Shamir and Adleman in a paper published in 1917
- Exist several achievable algorithm



Public deck

→ Players draw cards from the same deck

Proof of ordering

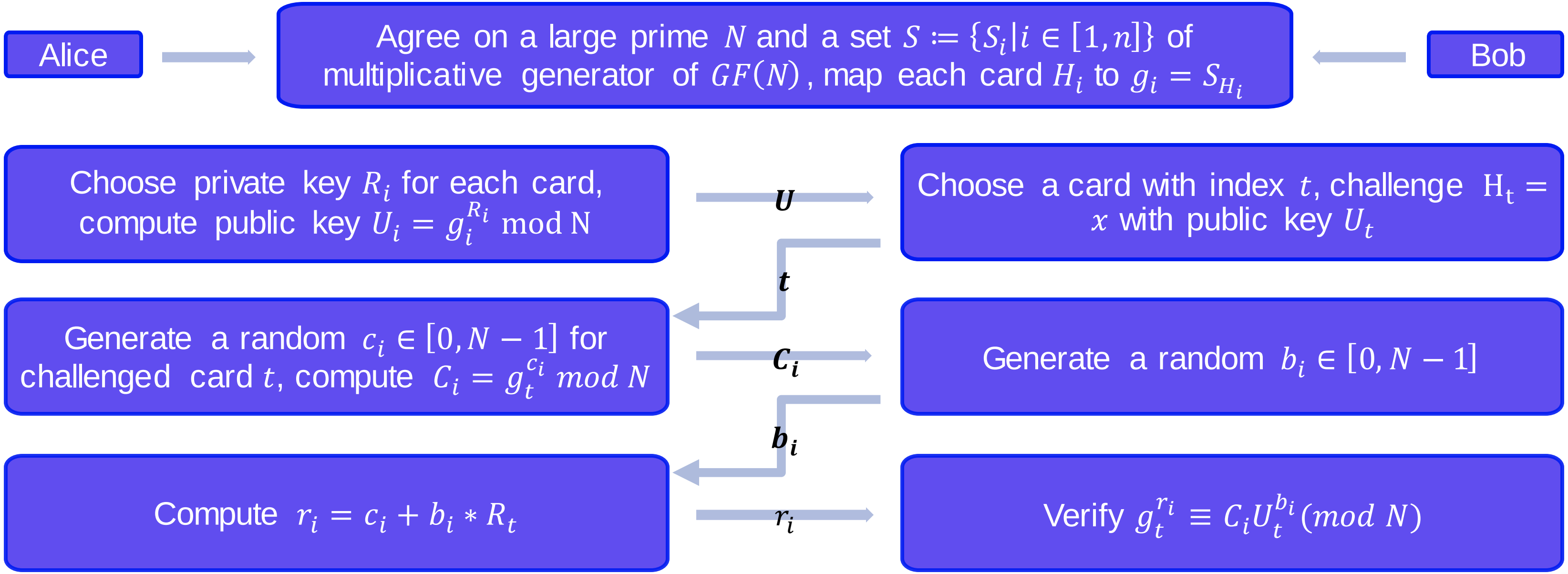
→ Cards should be sorted in the correct order

Proof of response

→ Prove the guess is wrong without revealing the card

The Algorithm

ZKP of single card(commitment scheme)



Alice

Agree on a large prime N and a set $S \coloneqq \{S_i | i \in [1,n]\}$ of multiplicative generator of $GF(N)$, map each card H_i to $g_i = S_{H_i}$

Bob

Choose private key R_i for each card, compute public key $U_i = g_i^{R_i} \bmod \mathrm{N}$

Choose a card with index t, challenge $\mathrm{H_t} = x$ with public key U_t

Generate a random $c_i \in [0, N-1]$ for challenged card t, compute $C_i = g_t^{c_i}\ mod\ N$

Generate a random $b_i \in [0, N-1]$

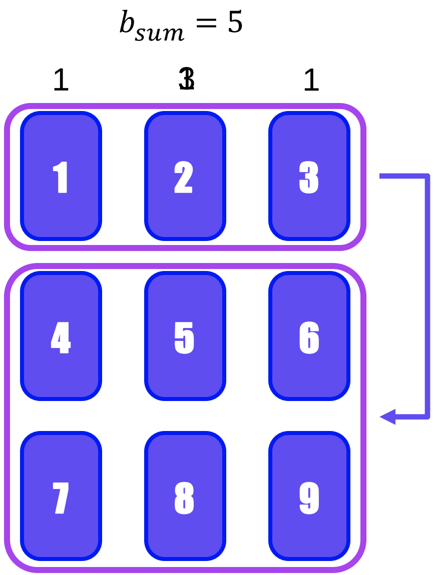
Compute $r_i = c_i + b_i * R_t$

Verify $g_t^{r_i} \equiv C_i U_t^{b_i} (mod\ N)$

The Algorithm

ZKP of disjunctive statements

1. Alice sets arbitrary b_i for all cards
2. Bob challenge a subset with x and $b_{sum} \in [0, N-1]$
3. Alice compute $C_i = \frac{S_x^{c_i}}{U_i^{b_i}}$ for $H_i \neq x$, responds $r_i = c_i$
4. For the designated card $H_t = x$, let $b_t + \sum b_i \equiv b_{sum}(mod\ N-1)$, here b_i are the other cards in the subset, respond $\mathrm{r_t} = c_t + b_t * R_t$
5. Alice sends responds along with b_i for all cards in the subset
6. Bob checks $\sum b_i \equiv b_{sum}(mod\ N-1)$ and $S_x^{r_i} \equiv \mathrm{C_i} * \mathrm{U_i^{b_i}}(\mathrm{mod\ N})$ for all i in the subset including t



The Algorithm

Features

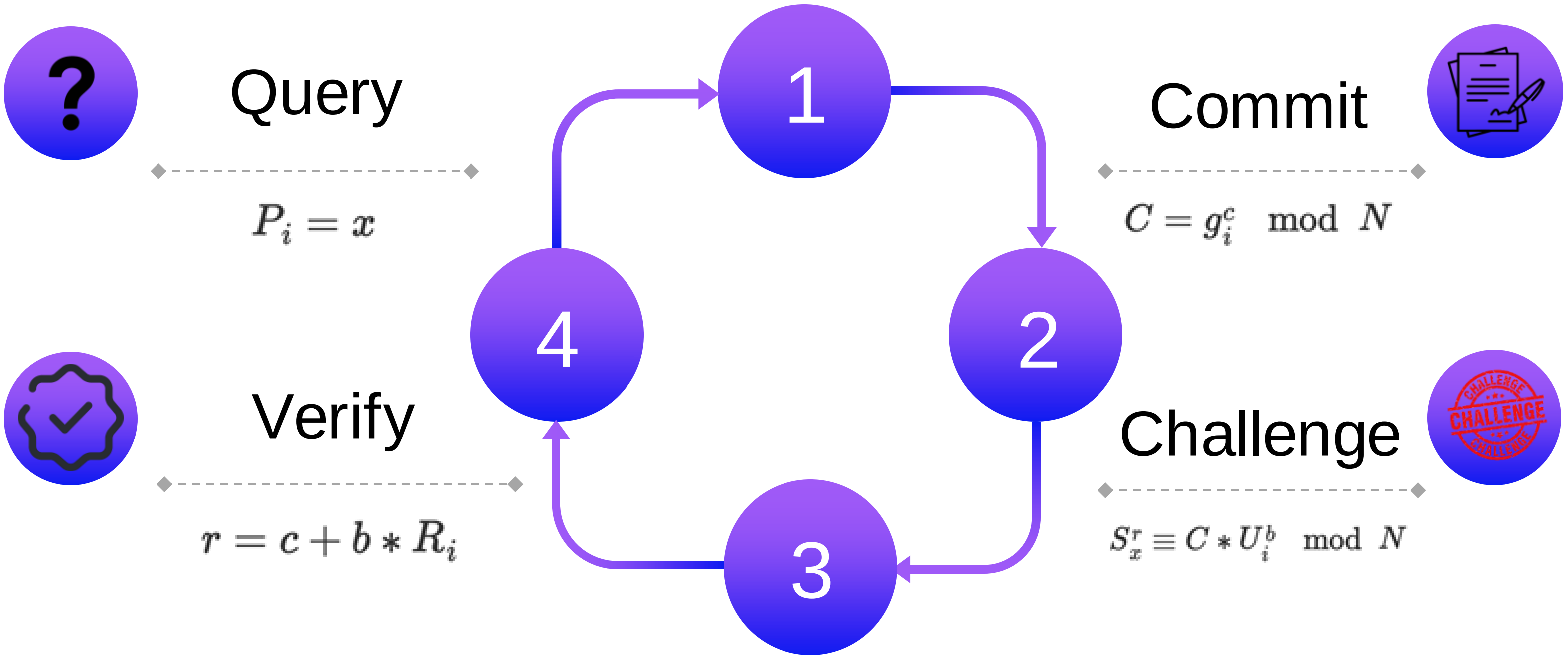
1 **ZKP of valid deck:** by proving every card's existence in the whole deck

2 **Drawing cards:** by agreed public PRNG(pseudo random number generator)

3 ZKP of $\mathbf{x} \in$ **some subset** of the deck(or of hand)

4 ZKP of $\mathbf{x} \notin$ **some subset** of the deck(or of hand)

System Architecture



Query

$P_i = x$

Commit

$C = g_i^c \mod N$

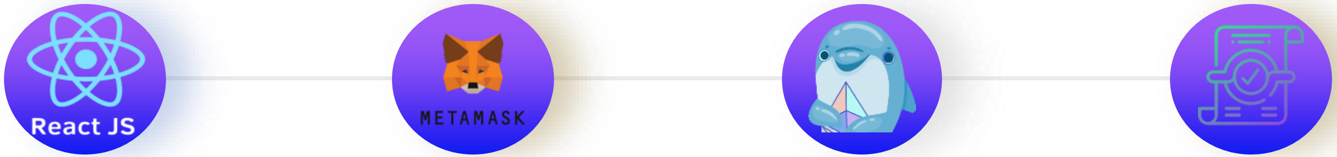
Challenge

$S_x^r \equiv C * U_i^b \mod N$

Verify

$r = c + b * R_i$

System Architecture



Front-end

User Interface
Interacts with MetaMask

MetaMask

Cryptocurrency wallet
Signing transactions and authentication

Sepolia

Blockchain network
Records transactions and game states

Smart Contract

Deployed on the Sepolia
Handles game logic and data storage

Experiment

p

Prover

Generate commit, and responds to arbitrary challenges

v

Verifier

Challenge the prover
and verify his respond

Experiment - ZKP Simulator

Init

User sets private permutation and private keys

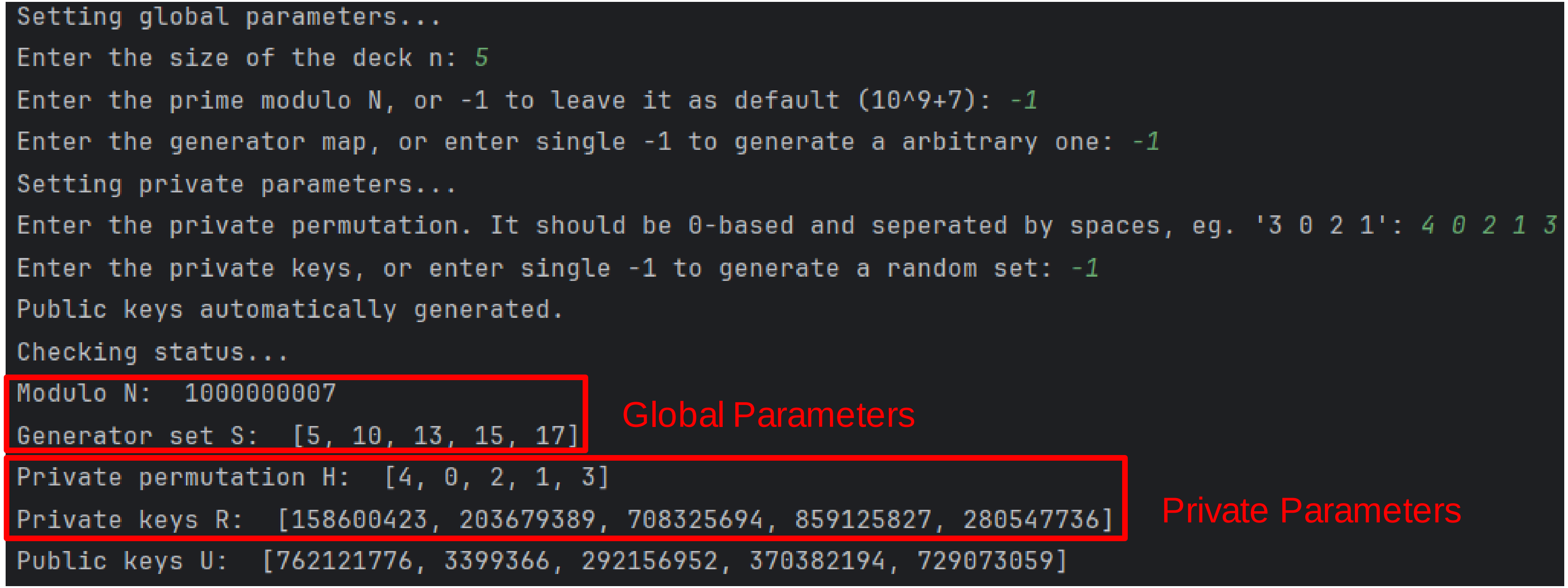
Query

Queries on arbitrary subset and value

Proof & Verify

Simultate proof & verify process, with all details shown

Experiment - Demo

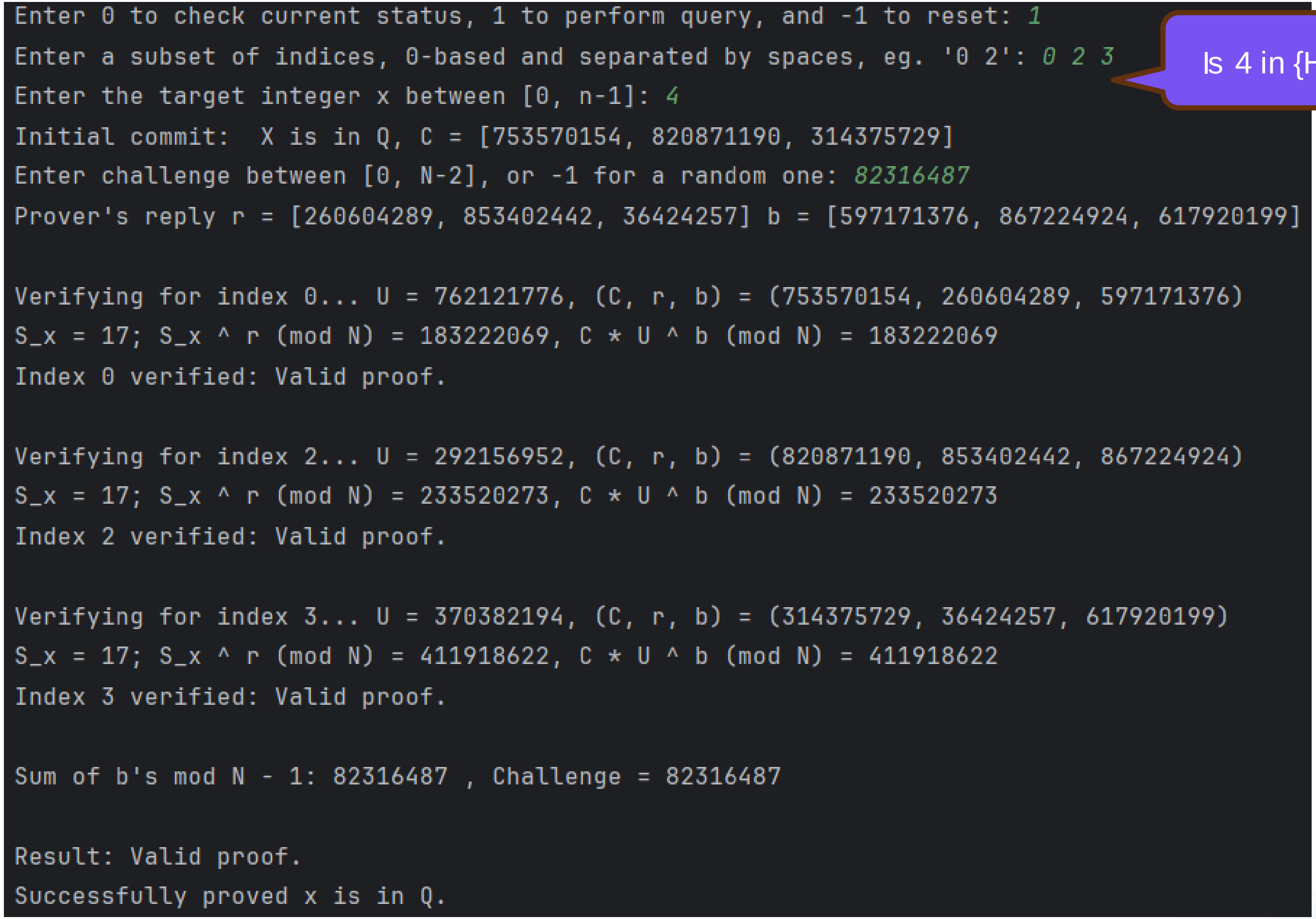


```
Setting global parameters...
Enter the size of the deck n: 5
Enter the prime modulo N, or -1 to leave it as default (10^9+7): -1
Enter the generator map, or enter single -1 to generate a arbitrary one: -1
Setting private parameters...
Enter the private permutation. It should be 0-based and seperated by spaces, eg. '3 0 2 1': 4 0 2 1 3
Enter the private keys, or enter single -1 to generate a random set: -1
Public keys automatically generated.
Checking status...
Modulo N:  1000000007
Generator set S:  [5, 10, 13, 15, 17]
Private permutation H:  [4, 0, 2, 1, 3]
Private keys R:  [158600423, 203679389, 708325694, 859125827, 280547736]
Public keys U:  [762121776, 3399366, 292156952, 370382194, 729073059]
```

Global Parameters

Private Parameters

Is 4 in {H[0], H[2], H[3]} ?



```
Enter 0 to check current status, 1 to perform query, and -1 to reset: 1
Enter a subset of indices, 0-based and separated by spaces, eg. '0 2': 0 2 3
Enter the target integer x between [0, n-1]: 4
Initial commit:  X is in Q, C = [753570154, 820871190, 314375729]
Enter challenge between [0, N-2], or -1 for a random one: 82316487
Prover's reply r = [260604289, 853402442, 36424257] b = [597171376, 867224924, 617920199]

Verifying for index 0... U = 762121776, (C, r, b) = (753570154, 260604289, 597171376)
S_x = 17; S_x ^ r (mod N) = 183222069, C * U ^ b (mod N) = 183222069
Index 0 verified: Valid proof.

Verifying for index 2... U = 292156952, (C, r, b) = (820871190, 853402442, 867224924)
S_x = 17; S_x ^ r (mod N) = 233520273, C * U ^ b (mod N) = 233520273
Index 2 verified: Valid proof.

Verifying for index 3... U = 370382194, (C, r, b) = (314375729, 36424257, 617920199)
S_x = 17; S_x ^ r (mod N) = 411918622, C * U ^ b (mod N) = 411918622
Index 3 verified: Valid proof.

Sum of b's mod N - 1: 82316487 , Challenge = 82316487

Result: Valid proof.
Successfully proved x is in Q.
```

Yes!

Contribution

- We designed an algorithm and built a protocol for card games on top of it, which should work on any game with separated deck (e.g. Yu-Gi-Oh).

- It extends some features in current Mental Poker Study:
  1. ZKP of initial deck validity
  2. ZKP of card membership or non-membership

感謝聆聽