CSE 539 Project2 Mental Poker

Arjun Magge, Nagarjuna Myla, Varun Kamath Burde
Professor Alan Skousen
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University
Tempe, AZ 85287

Abstract

In this project we implement the game of Mental Poker using an unarbitrated protocol that allows two parties to play a fair game without cheating. We have used Datagram Sockets instead of email to communicate between two programs. This game can be played on same computer or on two computers over a LAN. This project encompasses Socket Communication, Encryption implementation, Card Selection and Exchange using encryption protocol, Verification of non-cheating once all cards are played. The program demonstrates the possibility of playing poker without cards (i.e. over telephone or internet). In this system, there is no Trusted Third Party arbitrator or an adjudicator. We assume two players and a deck fifty-two cards. Five cards are dealt to each player, then one round of betting commences, and then all the cards are shown to reveal the high card in each player's hand to reveal the winner.

Keywords:
Mental Poker, SRA Protocol, unarbitarated protocol, encryption, client, server, game, Public Key encryption

1 Introduction

1.1 Terminology

E_A - Encryption key of player A
D_B - Decryption key of player B
GCD - Greates Common Divisor or Highest Common Factor
inv(A)- Multiplicative inverse of A

1.2 Goal Description

1.2.1 Socket Communication

The game is played between two players where one acts as a server and other acts as a client at different stages of the game. The communication between the two players is established over Datagram Sockets.

1.2.2 Encryption implementation

For encryption and decryption of cards in the game, we implement the SRA [Shamir, Rivest, Adleman] algorithm.

1.2.3 Card selection and exchange

The cards are encrypted and shuffled by the first player (A) and sent to the second player (B). The second player deals the cards between himself and the first player randomly as he does not know the face value of the card due the encryption in place by the first player (A). Players have disjoint hands. As no card is visible to the player while dealing them, any player can have any possible hand and neither player can discover the other player's hand.

1.2.4 Verification

In this task we have to do the verfication of cards and see if any cheating has taken place and announce the winner.

1.3 Assumptions

1.3.1 Socket Communication

We assume that both the players are connected over Wired LAN or Wireless network. As an alternative, the program can be run on same system where two different processes take roles of players A and B.

1.3.2 Encryption implementation

The prime number is choosen at random by the first player and send to the second player. Both players individually generate their encryption and decryption keys which they use subsequently to encrypt or decrypt cards.

1.3.3 Card selection and exchange

The game is initiated by the first player (A) by sending the prime number to the second player (B). The first player A sends the encrypted card deck to B to choose cards for both players. The winner is decided by whoever has the highest card among the two hands.

1.3.4 Verification

After the game is complete and the winner is declared by both players. After A and B agree on the winner based on the highest card, each player will perform verification steps to detect cheating. During this stage, no cards are exchanged. Only decryption keys are exchanged among the players which is used to decrypt the copy of ecrypted cards of the opponent that they possess.

2 Proposed Solution

2.1 Socket Communication

The communication between these two systems will be carried over a Datagram socket. In our current implementation, B starts the server and waits for the Client A to connect so that the initial large prime number can be received.

2.2 Encryption implementation

A prime number is choosen at random by A and the same will be sent to B. A will compute his encryption key and decryption key using the chosen prime number. Similarly B will compute his encryption key and decryption key. In our implementation, this process of key selection is automated to avoid user interaction. The process of creating the keys uses SRA protocol which is invented by Shamir, Rivest and Adleman in 1979. This relies on a commutative encryption scheme i.e. $E_A(E_B(M)) = E_B(E_A(M))$.

2.3 Card selection and exchange

Two players A and B use a common large prime number N, then A chooses his encryption key A such that gcd(A,N-1) = 1 and Bob chooses his key B similarly. A encodes the 52 cards as integers. The decryption key is computed in a similar manner for both players based on the encryption key and N.
Encryption $E_A(M) = M^A \pmod n$
Decryption $D_A(M) = M^{inv(A)} \pmod n$
A permutes the cards to $x_1, x_2, \ldots, x_{52}$ encrypts them and sends them to B $E_B(x_i)$.
B chooses 5 cards for A and sends them to A (without encrypting) $E_A(x_i)$.
B also chooses 5 cards for himself, encrypts them and sends them to A $E_B(E_A(x_i))$.
A can now decrypt her cards to see her hand $D_A(E_A(x_i) = x_i$.
A also decrypts B's cards then sends them back to B removing his encryption factor from those cards.
Here is where we need commutativity so $D_A(E_B(E_A(x_i))) = E_B(x_i)$
A receives her cards and decrypt them seeing her hand $D_A(E_A(x_i)) = x_i$
When B receives the shuffled, encrypted cards he cannot tell which card is which so he picks them randomly i.e, cannot see A's hand. When A receives B's double encrypted hand she cannot read it even when she partially decrypts it.

2.4 Verification

In the verification step of the game, the correctness of data exchanged between the players and the validation of non-cheating in the game is established. The verification stage proceeds as follows:
1) B sends his secret decryption key to A.
2) A sends his secret decryption key to B.
3) B uses A's decryption key to decrypt A's encrypted cards which where chosen randomly.
4) B verifies A's cards and A's high card.
5) A uses B's decryption key to decrypt B's encrypted cards which he received from B for decryption
6) A verifies B's cards and B's high card.

There is no scope for both players to cheat since the only information that is exchanged between the players after the game is the decryption key of the opponent. The information which needs to be decrypted in order to verify non-cheating in the game was established before the game began and hence the data cannot be changed.

3 Execution

Start the game by running MentalPoker with first argument as B and second parameter is optional which takes IP address of the party whom you play with. If second argument is not passed it will be defaulted to localhost. Once program is started it waits to connect with A. Now run the MentalPoker program by passing first argument as A along with the second parameter of B's IP address in case you are playing on LAN.
Now the game will be payed automatically between two connected parties which involves choosing proper random prime number and generating their encryption and decryption keys and cards exchanging. After all these steps A's cards and B's cards are displayed on their respective screens. The high card on their individual hands are exchanged to determine the winner. To verify any occurrence of cheating, the decryption keys are exchanged to reveal the opponent's cards thus verifying if the earlier claim of the highest card was indeed true.

The sample execution results are given below as screenshots:

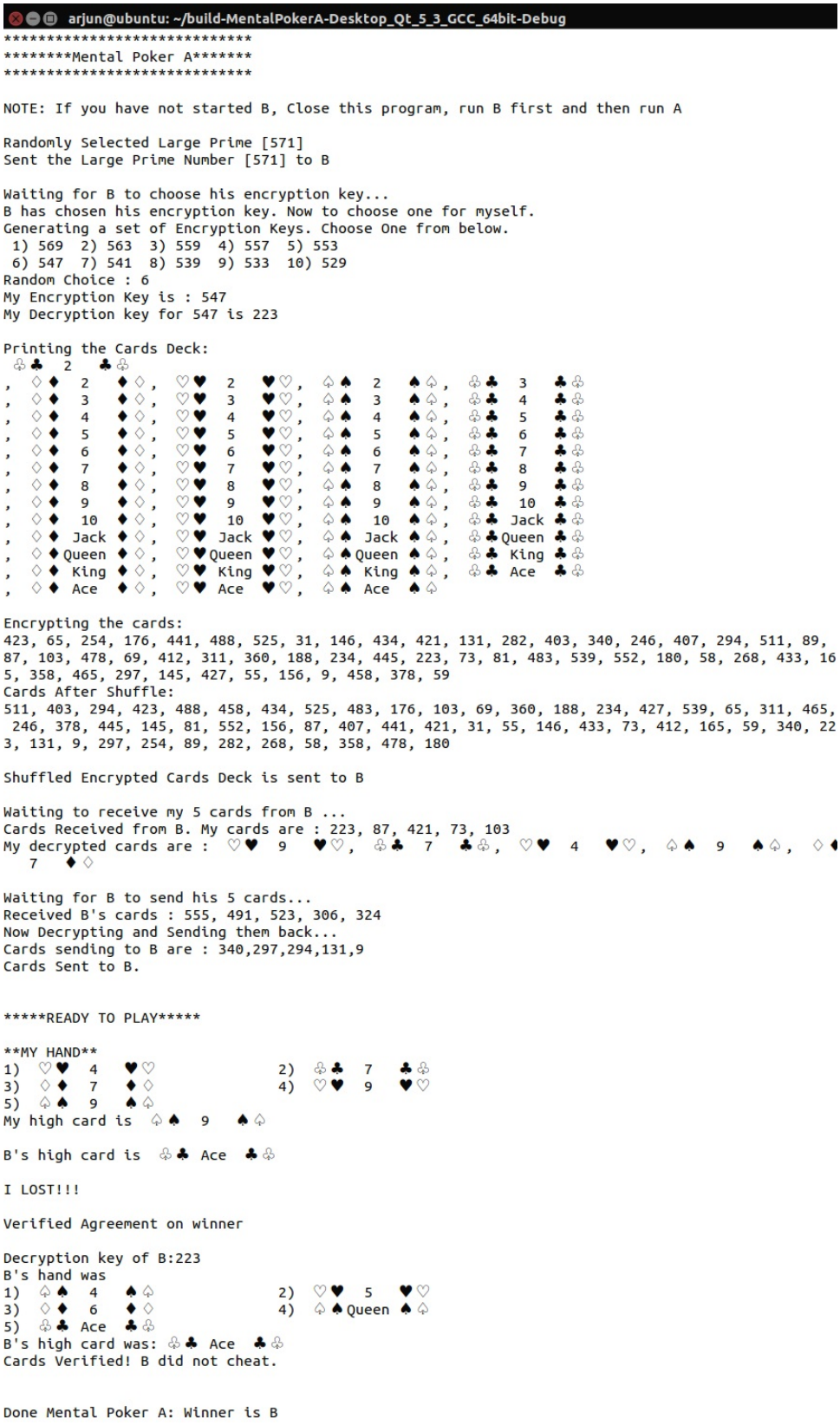

Figure 1: MentalPokerA

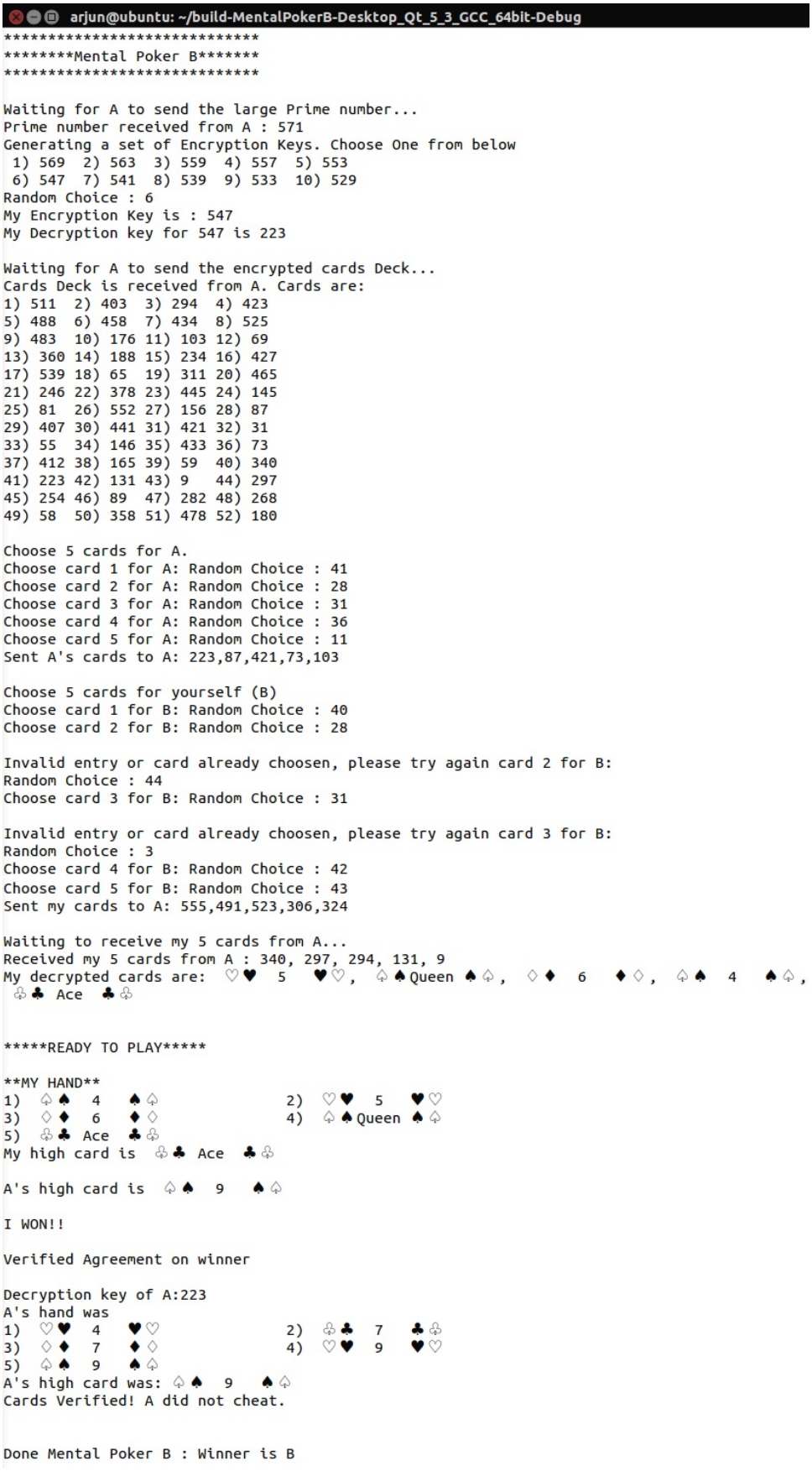

Figure 2: MentalPokerB

4 Conclusion

In this project we have provided a solution developed on the C++ programming language for playing Mental Poker over a network which demonstrates the objective of playing Mental Poker in accordance to the SRA protocol.

5 References

1. SRA Protocol
   http://people.csail.mit.edu/rivest/ShamirRivestAdleman-MentalPoker.pdf

2. Mental Poker
   http://en.wikipedia.org/wiki/Mental_poker