

Pemanfaatan Pertukaran Kunci Diffie-Hellman dan Pembangkit Bilangan Acak Semu Mersenne Twister pada Permainan Poker

Hafizh Dary Faridhan Hudoyo
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganessa 10 Bandung 40132, Indonesia
hafizhdary@gmail.com

Abstrak— Di dunia nyata, poker adalah sebuah permainan yang dimainkan oleh dua sampai sembilan orang dengan kesempatan menang yang tak tentu. Kartu diacak sedemikian rupa sehingga pemain tidak dapat menentukan kartu apa yang berada di atas, di bawah, atau di urutan mana pun pada tumpukan kartu. Poker berkembang sehingga dapat dimainkan secara online. Pada makalah ini akan dibahas bagaimana pengacakan tumpukan kartu dan pemanfaatan pertukaran kunci Diffie-Hellman agar pemain tidak dapat melihat kartu pemain lain.

Kata Kunci—Acak, Kunci, Kartu, Poker, Hand

I. PENDAHULUAN

Texas hold 'em (juga dikenal sebagai hold 'em atau holdem) adalah variasi dari permainan kartu poker yang standar. Permainan ini merupakan permainan yang paling terkenal di antara semua varian poker sehingga holdem sering juga disebut sebagai poker. Permainan ini terdiri dari dua kartu yang dibagikan tertutup untuk setiap pemain dan kemudian lima kartu yang ditempatkan di meja secara terbuka oleh dealer, lima kartu ini dinamakan *community card*—terdiri dari serangkaian tiga (*the flop*) kemudian kartu keempat (*the turn* atau *fourth street*) dan yang terakhir kartu kelima (*the river* atau *fifth street*), dengan pemain memiliki kesempatan untuk melakukan bet.

Meskipun sedikit yang mengetahui tentang penemuan Texas hold 'em, Badan Legislatif Negara Bagian Texas resmi mengakui Robstown, Texas, Amerika Serikat sebagai tempat kelahiran permainan ini, permainan lahir awal 1900-an.

Setelah permainan tersebar di seluruh Texas, hold 'em diperkenalkan ke Las Vegas pada tahun 1967 oleh sekelompok penjudi Texas dan para pemain kartu, termasuk Crandell Addington, Doyle Brunson, dan Amarillo Slim. Addington mengatakan pertama kali ia melihat permainan ini adalah pada tahun 1959. “Pada saat itu mereka tidak menyebutnya Texas hold 'em, mereka hanya menyebutnya hold 'em.... Kemudian saya berpikir bahwa permainan itu mampu menjadi populer, dan dapat menjadi sebuah pertandingan. Dalam *Draw* poker, anda bertaruh hanya dua kali. Namun dalam hold 'em, Anda

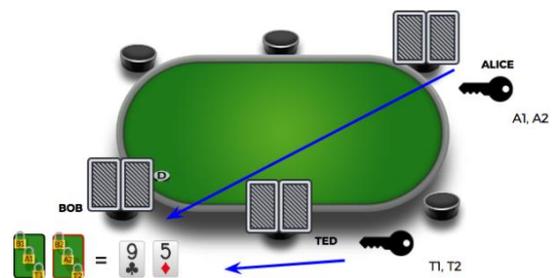
bertaruh empat kali, yang berarti pemain dapat bermain strategis. Ini lebih dari permainan berpikir.”

Kemampuan untuk bermain dengan murah, internet telah diakui sebagai penyebab meningkatnya popularitas Texas hold 'em. Situs poker online memungkinkan orang untuk mencoba permainan dan juga memberikan jalan untuk masuk ke turnamen besar (seperti *World Series of Poker*). Pemenang tahun 2003 dan 2004 (masing-masing adalah Chris Moneymaker dan Greg Raymer) dari *World Series No Limit Hold 'em Main Event* memenuhi syarat untuk bermain dalam turnamen itu.

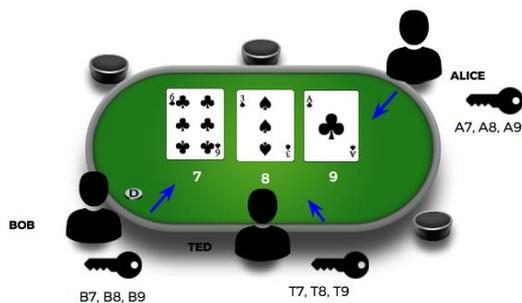
Sampai saat ini, permainan poker online sangat diminati. Terdapat banyak situs web yang menyediakan layanan poker online. *Chip* pada permainan poker online juga berwujud *virtual* namun dibeli dengan uang nyata.

Permainan ini diadaptasi menjadi permainan online yang mensimulasikan poker pada dunia nyata. Kartu diacak dengan algoritma yang bergantung dengan *pseudorandom number generator* (PRNG) dan dibagikan kepada pemain secara rahasia. Kerahasiaan ini dapat menggunakan algoritma pertukaran kunci, salah satunya adalah Diffie-Hellman.

Pada artikel yang dibuat oleh Virtue Poker, poker yang dirancang juga menggunakan pertukaran kunci sebagai kerahasiaan kartu yang ada di tangan pemain. Setiap kartu menggunakan kunci yang berbeda. Bagannya adalah sebagai berikut.



Gambar 1: Bagan pemberian kunci untuk melihat kartu di tangan



Gambar 2: Bagan pemberian kunci untuk melihat kartu di meja/board

Setiap kartu pada Terdapat dua kartu pada setiap pemain dan lima pada meja/board sehingga total maksimal kartu yang dipakai adalah $2 \times 9 + 5 = 23$ kartu dari total 52 kartu pada sebuah tumpukan kartu. Efisiensi ini diragukan karena harus membangkitkan 23 sampai 52 kunci untuk setiap kartu. Pada makalah ini akan dibahas pemanfaatan kunci Diffie-Hellman dari setiap pemain untuk merahasiakan kartu di tangan dan mengacak tumpukan kartu dengan Mersenne Twister melalui algoritma pengacakan Fisher-Yates.

II. DASAR TEORI

A. Texas Hold'em Poker

Texas Hold'em Poker adalah permainan *community card* yang memfokuskan kepada seberapa baik seorang pemain dapat bertaruh terhadap kartu yang dipunyainya. Permainan ini dimainkan oleh dua sampai sembilan orang dengan tujuan mendapatkan *chip* yang dipertaruhkan oleh pemain lain sebanyak-banyaknya. Pemain dikatakan menang jika kombinasi kartu di tangannya dan di meja lebih kuat dibandingkan kombinasi dari pemain lain atau pemain lain tidak ingin bertaruh (*fold*).

Permainan dapat dimulai jika ada seorang *dealer*, yaitu orang yang memegang *deck* kartu dan memastikan jumlah taruhan saat seorang pemain bertaruh, dan minimal dua orang pemain. Dua orang pemain ditunjuk sebagai *small blind*, yaitu orang yang wajib bertaruh setengah dari jumlah taruhan minimal yang ditentukan, dan *big blind*, yaitu orang yang wajib bertaruh sejumlah taruhan minimal. Keduanya dipilih secara berurutan, giliran *small blind* tepat sebelum *big blind*.

Setelah itu, *dealer* mulai mengocok kartu dan membagikan dua kartu untuk setiap pemain. Giliran bertaruh dimulai dari pemain setelah *big blind*. Jika pemain tersebut ingin melanjutkan permainan, maka ia harus bertaruh sejumlah yang dipertaruhkan oleh *big blind* (*call*), atau menaikkan jumlah taruhan (*raise*). Jika ada orang yang menaikkan jumlah taruhan, maka pemain lain harus menaikkannya sesuai jumlah yang dipertaruhkan oleh orang yang menaikkannya. Jika seseorang tidak ingin bertaruh, maka kartu yang dipegang harus dikembalikan (*fold*).

Jika ada dua pemain atau lebih yang melakukan *call*, maka permainan dilanjutkan dengan *flop*, yaitu *dealer* meletakkan tiga kartu pada meja secara terbuka. Setelah itu, pemain dapat melakukan taruhan, dimulai dari *small blind*. Pemain dapat *check*; yaitu melanjutkan giliran ke pemain berikutnya tanpa

taruhan, *bet*; yaitu bertaruh sejumlah *chip* yang diinginkan, mulai dari jumlah minimal sama seperti sebelum *flop* sampai semua *chip* dipertaruhkan (*all-in*), atau *fold*. Jika ada *bet* yang dilakukan, pemain berikutnya harus *call* atau *fold*.

Permainan dilanjutkan dengan *turn*, yaitu giliran di mana kartu keempat diletakkan. Untuk peraturan taruhannya sama seperti sebelumnya. Terakhir adalah *river*, yaitu giliran di mana kartu terakhir diletakkan dengan peraturan taruhan yang sama. Jika terdapat dua orang atau lebih yang melanjutkan gilirannya sampai akhir, maka pemain yang memiliki kartu terkuat dapat membuka kartu di tangannya. Jika pemain lain memiliki kartu yang lebih lemah, maka pemain tersebut boleh tidak membuka tangannya dan dinyatakan kalah (*muck*). Pemain yang memiliki tangan terkuat memenangkan semua *chip* yang dipertaruhkan pada satu ronde ini.

Di bawah ini adalah gambar tingkatan kombinasi kartu terkuat sampai terlemah.





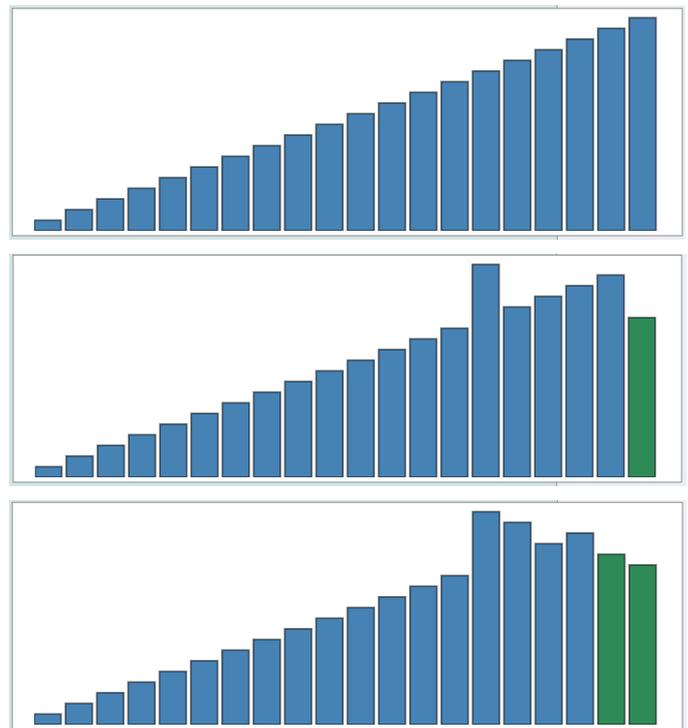
Gambar 3: Tingkatan kombinasi kartu terkuat sampai terlemah

- *Royal Flush* adalah kombinasi yang melibatkan kartu dengan nilai A, K, Q, J, dan 10 dengan simbol (*suit*) yang sama.
- *Straight Flush* adalah kombinasi yang mirip dengan *royal flush* namun urutannya di bawah AKQJ10, seperti K, Q, J, 10, 9 atau di bawahnya.
- *Four of a Kind (Quads)* adalah kombinasi yang mengandung empat kartu dengan nilai yang sama.
- *Full House* adalah kombinasi di mana ada tiga kartu yang bernilai sama ditambah dua kartu yang bernilai sama namun berbeda dengan tiga kartu sebelumnya.
- *Flush* adalah kombinasi di mana semua kartu memiliki *suit* yang sama namun tidak berurutan. *Flush* terkuat ditentukan oleh kartu dengan nilai paling tinggi pada kombinasi tersebut.
- *Straight* adalah kombinasi di mana terdapat lima kartu yang berurutan namun tidak memiliki *suit* yang sama. *Straight* terkuat adalah kombinasi A, K, Q, J, dan 10. Sedangkan yang terlemah adalah 5, 4, 3, 2, dan A.
- *Three of a Kind (trips)* adalah kombinasi di mana ada tiga kartu yang bernilai sama.
- *Two Pair* adalah kombinasi di mana ada dua pasangan kartu yang bernilai sama.

- *One Pair* adalah kombinasi di mana ada satu pasang kartu yang bernilai sama.
- *High Card* adalah kombinasi selain yang disebutkan di atas. *High Card* yang terkuat adalah kartu as, sedangkan yang terlemah adalah 2 (walaupun hal itu tidak mungkin pada *high card*).

B. Fisher-Yates Algorithm

Algoritma Fisher-Yates adalah algoritma untuk mengacak sebuah *list* dengan permutasi yang memiliki probabilitas yang sama dengan permutasi lainnya. Algoritma asli yang dirancang adalah membuat *list* baru dan meletakkan elemen pada *list* sebelumnya yang dipilih secara acak. Sedangkan algoritma modern yang telah diperbarui adalah menukar setiap elemen dengan elemen lain secara acak, contohnya adalah menukar elemen ke-1 dengan elemen ke-7, elemen ke-2 dengan elemen ke-4, dan seterusnya. Lebih jelasnya dapat dilihat pada gambar berikut.



Gambar 4: Proses pengacakan elemen ke-1 dan ke-2

Hal di atas dilakukan terus menerus sampai semua elemen dipertukarkan.

C. Mersenne Twister

Mersenne Twister adalah sebuah algoritma pembangkit bilangan acak semu (*pseudorandom number generator* atau PRNG) yang dibuat oleh Makoto Matsumoto dan Takuji Nishimura di mana *seed* yang digunakan memiliki panjang 19.937 bit, sehingga bilangan acak yang dihasilkan sebanyak $2^{19937}-1$. Semua angka yang dihasilkan diletakkan pada *array* sepanjang 624 elemen dengan satu elemen memiliki

panjang 32 bit. 31 bit terakhir kosong yang bertujuan untuk dipakai untuk *seed* berikutnya. Contoh *pseudocode*-nya adalah sebagai berikut.

```
for i = 0 to 623
  temp = first bit of a(i) followed by
  last 31 bits of a(i+1) ;
  a(i) = temp shifted right one bit xor
  X'9908B0DF' if temp is odd xor
  a(i+397) ;
next i
```

D. Pertukaran Kunci Diffie-Hellman

Pertukaran Kunci Diffie-Hellman adalah sebuah algoritma pertukaran kunci yang digunakan pada sebuah algoritma enkripsi untuk keperluan dekripsi pesan. Pertukaran kunci ini melibatkan minimal dua pihak. Pihak pertama, misalkan Alice, dan pihak kedua, misalkan Bob, memulai dengan menentukan dua buah bilangan prima g dan n , dengan g adalah bilangan prima yang kecil dan n adalah bilangan prima yang besar (contohnya memiliki panjang dua ribu bit).

Langkah tersebut dilanjutkan dengan menentukan angka sembarang untuk setiap pihak (misalkan a untuk Alice dan b untuk Bob). Setelah itu angka tersebut tidak dikirim begitu saja. Mereka harus menghitung sebuah kunci publik yaitu dengan rumus $A = g^a \text{ mod } n$ dan $B = g^b \text{ mod } n$. Untuk mendekripsi pesan, hal yang perlu dilakukan dua belah pihak adalah memangkatkan A dan B dengan $a \text{ mod } n$ atau $b \text{ mod } n$. Hal ini diperlukan untuk menghitung kunci yang ditetapkan untuk mengenkripsi pesan, dan untuk mendekripsinya hanya diperlukan pemangkatan tersebut.

III. RANCANGAN APLIKASI

A. PokerSimulator

PokerSimulator adalah kelas untuk menerapkan simulasi poker. Di dalamnya terdapat atribut privat, yaitu *deck* (sebuah *list* bertipe *PokerCard* yang memiliki panjang 52 elemen), *table* (*list* bertipe *Player* yang memiliki panjang minimal 2 elemen), dan *onTable* (*list* bertipe *PokerCard* yang memiliki panjang 5 elemen, menggambarkan kartu yang telah terbuka pada meja). Sedangkan atribut publik yang dipunya adalah *allIn*, yaitu apakah seorang pemain melakukan All-In atau tidak (dipilih secara acak).

Metode-metode yang terdapat pada kelas ini adalah sebagai berikut.

- *Shuffle*, yaitu algoritma untuk mengacak kartu dengan kombinasi algoritma Fisher-Yates dan bilangan acak semu Mersenne Twister.
- *InitDeck*, yaitu metode untuk membangkitkan *deck*.
- *InitPlayers*, yaitu metode untuk membangkitkan pemain yang didalamnya sudah terdapat pembangkitan kunci untuk setiap pemain.

- *Flop*, yaitu metode untuk melakukan *flop* dan membuka semua kartu di tangan jika *allIn* bernilai benar.
- *Turn*, yaitu metode untuk melakukan *turn* dan membuka semua kartu di tangan jika *allIn* bernilai benar.
- *River*, yaitu metode untuk melakukan *river* dan membuka semua kartu di tangan.

Kode yang ditulis pada program adalah sebagai berikut (jumlah player ditetapkan hanya dua pemain).

```
/*
 * To change this license header, choose
 License Headers in Project Properties.
 * To change this template file, choose Tools |
 Templates
 * and open the template in the editor.
 */
package pokersimulator;

import java.util.Arrays;

/**
 *
 * @author Hafizh Dary <hafizhdary@gmail.com>
 */
public class PokerSimulator {

    public boolean allIn = false;

    private PokerCard[] deck;
    private Player[] table;
    private PokerCard[] onTable;

    public PokerSimulator() {
        InitDeck();
        Shuffle(deck);
        InitPlayers(2);
        System.out.println("-----
        -----");
        Flop();
        System.out.print("Table: ");
        for (PokerCard onTable1 : onTable)
        {
            try {
                System.out.print(onTable1.toString() + ", ");
            }
            catch (Exception e) {
```

```

        }
    }
    System.out.println("");
    System.out.println("-----");
    Turn();
    System.out.print("Table: ");
    for (PokerCard onTable1 : onTable)
    {
        try {
            System.out.print(onTable1.toString() + ", ");
        }
        catch(Exception e) {

        }
    }
    System.out.println("");
    System.out.println("-----");
    River();
    System.out.print("Table: ");
    for (PokerCard onTable1 : onTable)
    {
        try {
            System.out.print(onTable1.toString() + ", ");
        }
        catch(Exception e) {

        }
    }

    private void Shuffle(PokerCard[] array) {
        for (int i = array.length - 1; i >
0; i--) {
            int index =
MersenneTwister.get(i);
            // swap
            PokerCard tmp =
array[index];
            array[index] = array[i];
            array[i] = tmp;
        }
    }

    private void InitDeck() {

```

```

        deck = new PokerCard[52];
        for(int i=0;i<4;i++) {
            for(int j=0;j<13;j++) {
                deck[i*13+j] = new
PokerCard(i,j);
            }
        }

        private void InitPlayers(int pl) {
            table = new Player[pl];
            for(int i=0;i<table.length;i++) {
                table[i] = new Player(i,
deck[0], deck[1]);
                deck
                =
Arrays.copyOfRange(deck, 2, deck.length);
                table[i].generateKeys();

                if(MersenneTwister.get(>0.5) allIn = true;
            }
            if(allIn) {

                table[0].receivePublicKeyFrom(table[1]);
                table[1].receivePublicKeyFrom(table[0]);
                table[0].generateCommonSecretKey();
                table[1].generateCommonSecretKey();
            }
            table[1].seeHands(table[0]);
            table[0].seeHands(table[1]);
        }

        private void Flop() {
            onTable = new PokerCard[5];
            deck = Arrays.copyOfRange(deck, 1,
deck.length);
            for(int i=0;i<3;i++) {
                onTable[i] = deck[0];
                deck
                =
Arrays.copyOfRange(deck, 1, deck.length);
            }
            if(MersenneTwister.get(>0.5)
allIn = true;
            if(allIn) {

                table[0].receivePublicKeyFrom(table[1]);
                table[1].receivePublicKeyFrom(table[0]);

```

```

table[0].generateCommonSecretKey();

table[1].generateCommonSecretKey();
    }
    table[1].seeHands(table[0]);
    table[0].seeHands(table[1]);
    }

    private void Turn() {
deck.length);    deck = Arrays.copyOfRange(deck, 1,
deck.length);    onTable[3] = deck[0];
deck.length);    deck = Arrays.copyOfRange(deck, 1,
allIn = true;    if (MersenneTwister.get() > 0.5)
        if (allIn) {

table[0].receivePublicKeyFrom(table[1]);

table[1].receivePublicKeyFrom(table[0]);

table[0].generateCommonSecretKey();

table[1].generateCommonSecretKey();
        }
        table[1].seeHands(table[0]);
        table[0].seeHands(table[1]);
        }

    private void River() {
deck.length);    deck = Arrays.copyOfRange(deck, 1,
deck.length);    onTable[4] = deck[0];

table[0].receivePublicKeyFrom(table[1]);

table[1].receivePublicKeyFrom(table[0]);

table[0].generateCommonSecretKey();

table[1].generateCommonSecretKey();
        table[1].seeHands(table[0]);
        table[0].seeHands(table[1]);
        }

/**
 * @param args the command line arguments
 */
public static void main(String[] args) {
    // TODO code application logic

```

```

here
        PokerSimulator ps = new
PokerSimulator();
    }
}

```

B. PokerCard

PokerCard adalah kelas untuk membuat kartu pada deck. Di dalamnya terdapat atribut privat, yaitu theSuit (menggambarkan *suit* pada sebuah kartu) dan theNumber (menggambarkan nilai pada sebuah kartu). Sedangkan atribut publik yang dipunya adalah Suit yang berisi seluruh kemungkinan *suit* yang ada dan Number yang berisi seluruh kemungkinan nilai yang ada.

Metode-metode yang terdapat pada kelas ini adalah sebagai berikut.

- toString, yaitu untuk merepresentasikan kartu ke dalam String.

Kode yang ditulis pada program adalah sebagai berikut.

```

/*
 * To change this license header, choose License
Headers in Project Properties.
 * To change this template file, choose Tools |
Templates
 * and open the template in the editor.
 */
package pokersimulator;

/**
 *
 * @author Hafizh Dary <hafizhdary@gmail.com>
 */
public class PokerCard {
    public enum Suit {Diamond, Clover, Heart,
Spade;}

    public enum Number {Two, Three, Four, Five,
Six, Seven, Eight, Nine, Ten, Jack, Queen, King,
Ace;}

    private int theSuit;
    private int theNumber;

    public PokerCard(int suit, int number) {
        theSuit = suit;
        theNumber = number;
    }
}

```

```
@Override
public String toString(){
    return Number.values()[theNumber] + "
of " + Suit.values()[theSuit];
}
}
```

C. Player

Player adalah kelas untuk membuat pemain pada meja serta membangkitkan kunci privat dan kunci publik. Di dalamnya terdapat atribut privat, yaitu id, Hand (dua kartu di tangan), PublicKey, PrivateKey, SecretKey, ReceivedPublicKey, dan SecretMessage (yang nantinya berisi informasi kartu di tangan).

D. MersenneTwister

MersenneTwister adalah kelas untuk membangkitkan bilangan acak dengan algoritma Mersenne Twister. Bilangan dapat berbentuk *float* (0..1) atau integer (0..*).

IV. PENGUJIAN

Pengujian yang dipakai adalah permainan poker dengan dua pemain. Simulasi ini berjalan dengan sangat cepat pada pengacakan dan pemberian kartu dibandingkan pada artikel yang dibuat oleh Virtue Poker karena memerlukan semua pemain untuk mengacak dan memberikan kunci untuk setiap kartu (membutuhkan koneksi internet untuk setiap pemain). Hasil simulasi yang didapat adalah sebagai berikut.

```
//jika melakukan allIn dari sebelum flop
run:
hand: Ten of Clover, Five of Clover
hand: Three of Spade, Ace of Spade
-----
hand: Ten of Clover, Five of Clover
hand: Three of Spade, Ace of Spade
Table: Six of Clover, Eight of Spade, Three
of Clover,
-----
hand: Ten of Clover, Five of Clover
hand: Three of Spade, Ace of Spade
Table: Six of Clover, Eight of Spade, Three
of Clover, Nine of Clover,
-----
hand: Ten of Clover, Five of Clover
hand: Three of Spade, Ace of Spade
Table: Six of Clover, Eight of Spade, Three of
Clover, Nine of Clover, Ace of Heart, BUILD
SUCCESSFUL (total time: 0 seconds)
//jika tidak melakukan allIn dari awal
```

```
secret
null
secret
null
-----
secret
null
secret
null
Table: Five of Heart, Three of Diamond,
Seven of Diamond,
-----
secret
null
secret
null
Table: Five of Heart, Three of Diamond,
Seven of Diamond, Four of Diamond,
-----
hand: Queen of Heart, Six of Diamond
hand: Ten of Heart, Eight of Heart
Table: Five of Heart, Three of Diamond,
Seven of Diamond, Four of Diamond, Five of
Diamond, BUILD SUCCESSFUL (total time: 0
seconds)
```

Terlihat bahwa kartu tidak dapat dilihat sebelum *river* jika belum ada yang melakukan *allIn* sehingga kartu di tangan aman dari penglihatan pemain lain.

V. KESIMPULAN DAN SARAN

Setelah menerapkan algoritma ini, dapat disimpulkan bahwa simulasi ini cukup baik untuk diterapkan pada permainan poker online. Selain cepat, pengacakan kartu dijamin tidak *uniform* dengan pengacakan sebelumnya karena telah memanfaatkan bilangan acak Mersenne Twister.

Sebagai saran untuk penulis, simulasi ini belum mencakup semua aspek, seperti siapa yang menang dan kalah, jumlah pemain lebih dari dua, atau jumlah taruhan pada sebuah ronde. Namun hal tersebut dapat ditambah jika seluruh aspek pertukaran kunci dan acakan telah dinyatakan baik.

VI. UCAPAN TERIMA KASIH

Pertama-tama penulis mengucapkan terima kasih kepada Allah SWT karena telah memberikan rahmat dan berkat sehingga penulis dapat menyelesaikan makalah ini. Terima kasih juga kepada dosen pengajar Mata Kuliah IF4020 Kriptografi yaitu Dr. Ir. Rinaldi Munir, M.T. atas segala bimbingannya selama perkuliahan di semester 8 ini. Terima

kasih juga kepada teman-teman mahasiswa Teknik Informatika yang selalu mendukung dalam kuliah maupun luar kuliah.

VII. REFERENSI

- [1] <https://www.thelocal.ch/20121219/the-history-of-texas-holdem> diakses tanggal 17 Mei 2018 pukul 11:00 WIB
- [2] <http://www.quadibloc.com/crypto/co4814.htm> diakses tanggal 17 Mei 2018 pukul 10:53 WIB
- [3] <https://hackernoon.com/algorithms-explained-diffie-hellman-1034210d5100> diakses tanggal 17 Mei 2018 pukul 10:58 WIB
- [4] <http://duniabaca.com/asal-usul-sejarah-permainan-poker-texas-holdem.html> diakses tanggal 18 Mei 2018 pukul 10:01 WIB
- [5] <http://www.programming-algorithms.net/article/43676/Fisher-Yates-shuffle> diakses tanggal 17 Mei 2018 pukul 11:04 WIB

VIII. PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 18 Mei 2018



Hafizh Dary Faridhan Hudoyo
13514072