Pemanfaatan Citra Real-Time pada Inisiasi Umpan dalam Pembangkitan Bilangan Acak

Fitrandi Ramadhan - 13508065
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
if18065@students.if.itb.ac.id

Abstract*— Random Number Generator telah menjadi pertanyaan besar bagi setiap computer scientist di dunia. Bagaimana cara menciptakan sebuah angka yang trully random. Random Number Generator yang ada pada saat ini sebagian besar merupakan sebuah generasi dari fungsi yang bersifat chaos. Fungsi bersifat chaos ini maksudnya adalah sebuah fungsi yang dapat memberikan dua output yang berbeda sangat jauh dari 2 input memiliki selisih relatif dekat. Pertanyaan berikutnya muncul dan menantang banyak peneliti. Apa yang terjadi bila suatu algoritma yang menghasilkan suatu angka yang bersifat pseudo-random diberikan umpan yang bersifat trully random? Maka ditemukanlah suatu seed yang dianggap oleh para peneliti sebagai sesuatu yang bersifat trully random yaitu nilai entropi alam semesta. Nilai entropi dunia ini selalu bertambah dan tidak pernah stagnan dalam 1 milisecond sekalipun. Perubahan ini pun terjadi pada kecepatan yang berubah-ubah. Maka dapat dengan aman dinyatakan bahwa entropi dunia ini menjadi sesuatu yang bernilai trully random. Implementasi random number generator dengan umpan entropi dunia ini tentunya bukanlah sesuatu yang murah untuk di realisasikan. Adakah suatu seed yang bisa didapatkan dengan mudah dan memiliki sifat yang trully random? Seperti yang telah kita ketahui dewasa ini penggunaan perangkat seluler di dunia meningkat pesat. Pada setiap perangkat ini telah terpasang suatu perangkat kamera didalamnya. Gambar memiliki atribut yang menarik dimana setiap pixel dari gambar memiliki value yang berbeda satu sama lainya. Dari gambar ini akan dapat digenerasi sebuah numerik yang dapat dijadikan sebuah umpan untuk algritma pembangkitan bilangan acak. Secara umum manusia sulit untuk menghasilkan citra yang memiliki seluruh pixel value yang sama. Sehingga dapat dikatakan bahwa citra yang dihasilkn oleh manusia bersifat trully random. Dari sinilah berangkat topik dari makalah ini yaitu dapatkah sebuah kamera dari perangkat seluler dimanfaatkan untuk menjadi umpan dalam pembangkitan bilangan acak.

***Kata kunci*— Pembangkit bilangan acak, MD5, true-random, perangkat seluler, citra**

I. PENDAHULUAN

Pembangkit bilangan acak telah menjadi pertanyaaan besar bagi setiap computer scientist di dunia. Bagaimana caranya menciptakan angka-angka yang dibangkitkan dari sebuah perangkat yang sama akan tetapi tidak memiliki kemiripan atau hubungan dengan angka yang diciptakan sebelumnya. Pembangkit bilangan random sudah banyak tersedia di berbagai perangkat dalam bentuk perangkat keras maupun perangkat lunak. Akan tetapi fenomena yang ada pada kebanyakan pembangkit bilangan acak adalah bilangan yang dibangkitkan bersifat pseudo-random dan bukan true random. Bilangan yang dihasilkan oleh pembangkit pseudo-random adalah bilangan semi random yang mana adalah bilangan hasil dari suatu operasi tertentu yang dilakukan kepada suatu bilangan umpan tertentu. Bilangan umpan akan menjadi basis utama dari pembangkitan bilangan acak ini, menjadi input utama dari operasi pembangkitan bilangan acak. Sehingga deret bilangan acak yang dihasilkan oleh pembangkit bilangan acak (pseudo-random) akan selalu menghasilkan deret yang sama apabila mendapatkan bilangan umpan yang sama.

Bila dirunut dari sejarah, pembangkitan bilangan acak sudah dikenal oleh manusia sejak lama. Manusia telah menggunakan dadu, kepingan koin, kartu dan berbagai benda yang bersifat fisik lainya untuk membangkitan bilangan acak. Benda-benda yang bersifat fisik ini dapat membangkitkan suatu bilangan acak yang bersifat true random, bilangan yang dihasilkan oleh benda ini tidak bisa atau sangat sulit dihasilkan tanpa suatu lingkungan yang terkontrol sedemikan rupa. Sehingga dapat dengan aman dikatakan bahwa bilangan ini merupakan bilangan true random.

Kedua metode ini memiliki kelebihan dan kekuranganya masing-masing. Bilangan acak yang dihasilkan oleh pembangkit bilangan acak klasik ini memiliki ko-domain yang sangat terbatas. Pada dadu misalnya, bilangan yang dapat dihasilkan hanyalah enam buah bilangan integer dari 1 sampai dengan 6. Terlebih lagi pada koin, koin hanya bisa menghasilkan du buah nilai boolean. Tidak hanya sampai disitu, kesulitan lain pun muncul karena mekanisme pembangkitan bilangan acak dengan metode klasik ini membutuhkan waktu yang cukup lama. Pada pelemparan dadu pengguna harus melempar dadu dan menunggu sampai dadu berhenti bergulir. Hal yang sama juga terjadi pada koin dimana koin harus dilempar dan ditunggu sampai koin jatuh ke tanah dan berhenti bergerak.

Kebutuhan akan bilangan acak pada masa sekarang ini sangatlah bervariasi. Dari penggunaanya untuk judi

sampai dengan penggunaan yang lebih advance seperti enkripsi dan dekripsi data. Untuk penggunaan yang advance ini dibutukan proses yang relatif cepat karena jumlah angka random yang diperlukan relatif sangat banyak. Proses ini bila dilakukan dengan menggunakan metode pembangkitan bilangan acak klasis tentunya akan membutuhkan waktu yang tidak masuk akal. Proses pembangkitan bilangan acak yang digunakan untuk enkripsi dan dekripsi seringkali memang membutuhkan suatu bilangan pseudo random yang sewaktu-waktu dapat dibangkitkan kembali dengan hasil yang sama. Akan tetapi, ada proses atau fenomena lain yang membutuhkan bilangan acak yang bersifat true random yang hanya bisa dibangkitkan satu kali. Contoh mudah nya adalah pembangkitan bilangan acak pada pembangkitan seed untuk digunakan pada pseudo-random number generator pada berbagai algoritma enkripsi. Contoh lain yang aplikasinya dekat dengan kehidupan sehari-hari adalah game. Game membutuhkan sangat banyak pembangkitan bilangan acak, sebagian besar dari pembangkitan bilangan acak yang dibutuhkan oleh proses komputasi game adlah bilangan acak yang hanya bisa dibangkitkan sekali dan tidak dapat dimanipulasi untuk dibangkitkan kembali dengan nilai yang sama. Dengan kesulitan yang dihadapi saat ini dimana pada umumnya bilangan acak yang dibangkitkan merupakan bilangan bersifat pseudo-random seringkali bilangan acak terkesan tidak adil atau manipulable.

Manusia saat ini telah menemukan beberapa solusi untuk membangkitkan bilangan yang bersifat true random. Bilangan ini biasanya adalah bilangan yang diproses menggunakan pembangkit pseudo random yang diberikan umpan yang bersifat true random. Tedapat berbagai materi di dunia yang bersifat true random dan dapat digunakan sebagai nilai umpan pembangkitan bilangan acak. Salah satu contoh yang paling populer saat ini adalah nilai entrophi alam semesta. Sumber entropi ini dapat berasal dari berbagai materi seperti peluruhan radioaktif dan kebisingan thermal.

Materi umpan ini biasanya tidak dapat didapatkan dengan biaya yang murah. Kemudian muncul pertanyaan, adakah suatu sumber lain yang memiliki sifat true random dan dapat dengan mudah didapatkan dengan biaya yang relatif murah. Penggunaan perangkat seluler, telah meningkat pesat dibandingkan dengan penggunaanya 20 tahun ke belakang. Dengan memanfaatkan keadaan ini, dapatkah manusia mendapatkan suatu materi yang bersifat true random dengan menggunakan perangkat seluler sebagai sensor. Sebagian besar perangkat seluler memiliki perangkat penangkap citra.

II. LANDASAN TEORI

A. Pembangkit Bilangan Acak

Tidak ada sejarah yang mencatat kapan pertama kali manusia mengenal pembangkitan bilangan acak. Dadu sebagai perangkat pembangkitan bilangan acak klasik telah digunakan sebelum sejarah yang tercatat oleh manusia. Terlebih penggunaan koin sebagai pembangkit bilangan acak diduga telah digunakan jauh sebelum dadu dikenal oleh manusia.

Pembangkit bilangan acak yang terdapat pada komputer masa kini biasanya merupakan pembangkit bilangan acak yang bersifat pseudo-random. Bilangan yang dihasilkan merupakan bilangkan hasil operasi matematis yang dari suatu masukan yang disebut umpan. Fungsi matematika yang digunakan adalah suatu fungsi yang memiliki sifat chaos. Sifat chaos disini maksudnya adalah fungsi tersebut sangat sensitif terhadap masukan atau kondisi awal yang diberikan. Paradigma yang sering juga disebut sebagai butterfly effect, dimana perbedaan kondisi awal yang diberikan akan menghasilkan keluaran yang berbeda relatif jauh.

Dengan penggunaan fungsi yang bersifat chaos ini manusia dapat membangkitkan bilangan acak dengan memberikan suatu input dan membangkitkan bilangan acak yang berbeda dengan memberikan input yang berbeda pula. Karena sifat chaos yang dimiliki ini maka tidak ada masalah pola hasil yang didapatkan menghasilkan pola angka yang mirip karena pola yang dihasilkan merupakan pola yang kacau.

Kesulitan yang dihadapi pada pembangkit bilangan pseudo-random adalah bilangan ini dapat dihasilkan kembali dengan bilangan umpan yang sama. Ini dapat dikatakan tidak adil. Bayangkan bila sebuah pseudo-random number generator digunakan

B. Algoritma MD5

MD5 adalah algoritma message-digest yang umum digunakan sebagai fungsi hash kriptografi. MD5 adalah algoritma yang didesain untuk mengganti algoritma hash sebelumnya yaitu MD4 yang dianggap sudah tidak aman lagi. Algoritma MD5 didesain untuk bekerja cepat pada mesin 32-bit.

Algoritma MD5 dimulai dengan mendefinisikan pesan sebesar b-bit sebagai input yang ingin didapatkan message-digestnya. Disini b adalah bilangan bulat positif sembarang. Tidak harus kelipatan 8 dan dapat sangat besar. Algoritma ini bekerja dengan menerapkan 5 langkah.

1. Menambahkan padding bits
   Pesan diberikan padding agar panjangnya kongruen dengan 448 dalam modulo 512. Pesan diberikan padding sehingga panjang besar tepat kurang 64 bit dari kelipatan 512 bit. Langkah ini dilakukan walaupun panjang pesan sudah kongruen dengan 448 dalam modulo 512. Penambahan padding dilakukan dengan menambahkan satu buah bit "1" dan menambahkan

bit "0" sampai panjang bit tepat kurang 64 bit dari kelipatan 512 bit selanjutnya.

2. Menambahkan panjang
   Representasi 64 bit dari b ditambahkan dari hasil penambahan padding sebelumnya. Apabila b lebih besar dari 64 bit maka akan digunakan 64 bit terkecil dari angka tersebut. (bit ini ditambahkan sebagai dua huruf 32-bit). Pesan akan memiliki panjang tepat kelipatan 512 bit. Dari sini dapat diambil kesimpulan bahwa panjang pesan merupakan tepat kelupatan dari 16 kata (32-bit). Kemudian direpresentasikan sebagai M[0..N-1] dimana N adalah kelipatan 16.

3. Inisiaisasi Buffer MD
   Sebuah buffer berupa 4 kata (A, B, C, D) digunakan untuk mengkomputasi pesan. Disini setiap A, B, C, D adalah 32-bit register. Register ini diinisialisasi dengan nilai dalam hexadesimal.

```
kata A: 01 23 45 67
kata B: 89 ab cd ef
kata C: fe dc ba 98
kata D: 76 54 32 10
```

4. Proses pesan pada 16 blok kata.
   Kemudian didefinisikan 4 fungsi tambahan yang mengambil input 3 x 32-bit kata-kata dan memproduksi output 32-bit kata.

```
F(X,Y,Z) = XY v ~(X) Z
G(X,Y,Z) = XZ v Y ~(Z)
H(X,Y,Z) = X xor Y xor Z
I(X,Y,Z) = Y xor (X v ~(Z))
```

Setiap posisi bit F bertindak sebagai kondisi if X then Y else Z. Fungsi F dapat didefinisikan menggunakan +. Karena XY dan ~(X)Z tidak akan memiliki 1's di posisi bit yang sama. Fungsi G, H, I mirip dengan fungsi F dalam artian seluruhnya bekerja "bitwise parallel" dalam memproduksi keluaran mereka. Langkah ini menggunakan tabel dengan 64 elemen T[1..64] yang diciptakan dari fungsi sinus. Berikut pseudo-code dari Algoritma ini.

```
for i = 0 to N/16-1 do

  //kopi block i into X.
       for j = 0 to 15 do
         Set X[j] to M[i*16+j].
       end //of loop on j

       /* Save A as AA, B as BB, C
as CC, and D as DD. */
       AA = A
       BB = B
       CC = C
       DD = D

       /* Round 1. */
       /* Berikan [abcd k s i]
tunjukkan operasi a = b + ((a +
F(b,c,d) + X[k] + T[i]) <<< s). */

  /* Lakukan operasi ini. */
       [ABCD  0  7  1]  [DABC  1
12  2]  [CDAB  2 17  3]  [BCDA  3
22  4]
       [ABCD  4  7  5]  [DABC  5
12  6]  [CDAB  6 17  7]  [BCDA  7
22  8]
       [ABCD  8  7  9]  [DABC  9
12 10]  [CDAB 10 17 11]  [BCDA 11
22 12]
       [ABCD 12  7 13]  [DABC 13
12 14]  [CDAB 14 17 15]  [BCDA 15
22 16]

/* Round 2. */
/* Berikan [abcd k s i] tunjukkan
operasi a = b + ((a + G(b,c,d) +
X[k] + T[i]) <<< s). */

  /* Lakukan operasi ini. */
       [ABCD  1  5 17]  [DABC  6
9 18]  [CDAB 11 14 19]  [BCDA  0
20 20]
       [ABCD  5  5 21]  [DABC 10
9 22]  [CDAB 15 14 23]  [BCDA  4
20 24]
       [ABCD  9  5 25]  [DABC 14
9 26]  [CDAB  3 14 27]  [BCDA  8
20 28]
       [ABCD 13  5 29]  [DABC  2
9 30]  [CDAB  7 14 31]  [BCDA 12
20 32]

       /* Round 3. */
       /* Berikan [abcd k s t]
tunjukkan operasi
            a = b + ((a + H(b,c,d)
+ X[k] + T[i]) <<< s). */
       /* Do the following 16
operations. */
       [ABCD  5  4 33]  [DABC  8
11 34]  [CDAB 11 16 35]  [BCDA 14
23 36]
       [ABCD  1  4 37]  [DABC  4
11 38]  [CDAB  7 16 39]  [BCDA 10
23 40]
       [ABCD 13  4 41]  [DABC  0
11 42]  [CDAB  3 16 43]  [BCDA  6
23 44]
       [ABCD  9  4 45]  [DABC 12
11 46]  [CDAB 15 16 47]  [BCDA  2
```

```
23 48]

      /* Round 4. */
      Berikan [abcd k s t]
tunjukkan
           a = b + ((a + I(b,c,d)
+ X[k] + T[i]) <<< s). */
      /* Do the following 16
operations. */
      [ABCD  0  6 49]  [DABC  7
10 50]  [CDAB 14 15 51]  [BCDA  5
21 52]
      [ABCD 12  6 53]  [DABC  3
10 54]  [CDAB 10 15 55]  [BCDA  1
21 56]
      [ABCD  8  6 57]  [DABC 15
10 58]  [CDAB  6 15 59]  [BCDA 13
21 60]
      [ABCD  4  6 61]  [DABC 11
10 62]  [CDAB  2 15 63]  [BCDA  9
21 64]


/*  Lakukan penjumlahan ini,
incremental terhadap setiap nilai
register pada iterasi sebelumnya.
*/
      A = A + AA
      B = B + BB
      C = C + CC
      D = D + DD

   end /* of loop pada i */
```

5. Output
   Pesan diberikan padding agar panjangnya kongruen dengan 448 dalam modulo 512. Pesan diberikan padding sehingga panjang besar tepat kurang 64 bit dari kelipatan 512 bit. Langkah ini dilakukan walaupun panjang pesan sudah kongruen dengan 448 dalam modulo 512. Penambahan padding dilakukan dengan menambahkan satu buah bit "1" dan menambahkan bit "0" sampai panjang bit tepat kurang 64 bit dari kelipatan 512 bit selanjutnya.

C. Android Platform

Android merupakan platform aplikasi seluler. Perangkat seluler biasanya sudah dilengkapi dengan berbagai perangkat tambahan lainya seperti kamera. Kamera ini dapat diakses dengan permohonan izin dari aplikasi yang akan menggunakanya. Dari penggunaan kamera ini dapat didapatkan sebuah citra hasil tangkapan pengguna aplikasi. Hasil citra dari penangkapan ini sudah dalan bentuk byte array yang mana dapat langsung diproses lebih lanjut oleh aplikasi. Tentunya byte array ini dapat disimpan lagsung kedalam storage dalam format jpg atau lainya.

Khusus pada makalah ini data dalam bentuk byte array yang dihasilkan oleh kamera tidak akan disimpan terlebih dahulu melainkan langsung dihitung nilai MD5 nya. Dari sini akan didapatkan nilai MD5 yang dapat digunakan sebagai umpan dari Pembangkit bilangan acak.

III. ANALISIS DAN IMPLEMENTASI

Permasalahan pembangkitan bilangan acak saat ini dapat diselesaikan dengan penggunaan fenomena fisis random yang terjadi di dunia. Penggunaan atribut dari material fisis yang memiliki sifat random seperti peluruhan radioaktif dapat digunakan sebagai umpan dari algoritma pseudo-random yang mana akan meghasilkan keluaran yang bersifat true random. Materi ini tentunya sulit untuk didapatkan datanya dengan biaya yang murah. Terlebih untuk penggunaan sehari-hari seperti game dan undian yang membutuhkan proses yang cepat, praktis dan murah.

Analisis ini berangkat dari sebuah hipotesis dimana sebuah gambar atau citra yang diambil oleh seseorang mengandung sifat entropi. Apabila sebuah citra diambil maka citra tersebut akan mengandung berbagai atribut fisis seperti ruang dan waktu. Atribut fisis ini tentunya memiliki nilai-nilai tertentu yang mana akan menghasilkan citra yang tertentu pula. Dalam konteks sebuah kamera digital, citra dihasilkan dengan menangkap cahaya yang masuk kedalam sensor dan cahaya tersebut diinterpretasi dan di translasi menjadi sebuah array of pixel di komputer. Citra menangkap gambar dari materi fisis yang tentunya memiliki aribut ruang dan waktu. Seseorang dapat saja mengambil gambar pada lokasi yang sama sehingga orang tersebut dapat mengambil citra dari materi yang berada pada lokasi yang sama. Akan tetapi ada suatu atribut dari benda fisis yang tidak dapat diulang atau direkayasa oleh manusia. Atribut ini adalah atribut waktu. Seseorang yang mengambil gambar ada lokasi yang sama dan mendapatkan atribut ruang yang sama tidak akan dapat mengambil dua buah gambar yang persis sama secara esensi. Hal ini dikarenakan atribut waktu dari benda fisis tidak akan pernah sama. Dimensi waktu terus bergerak maju dan tidak bisa dimundurkan kembali. Analisis ini berangkat dari hipotesis ini dan memuculkan pertanyaan apakah citra yang diambil ini bersifat random. Dengan teknologi saat ini manusia dapat merekayasa dua buah citra yang berbeda menjadi citra yang memiliki atribut sama yang mana menghasilkan dua buah citra yang persis sama. Apakah hal ini menjadikan citra yang digunakan menjadi tidak valid karena dapat saja sebuah citra tersebut sama dan akan menghasilkan sebuah bilangan acak yang selalu sama bila digunakan sebagai umpan dari algoritma pseudo-random.

Tujuan utama dari makalah ini mencari materi lain yang bersifat true random dan dapat diambil dengan biaya yang relatif murah. Penggunaan dari bilangan acak yang

dihasilkan pada aplikasi yang diimplementasikan pada makalah ini ditujukan pada penggunaan sehari-hari.

Aplikasi dibuat pada platform android. Pada perancanganya sebuah citra diambil dengan menggunakan perangkat kamera yang telah tersedia pada sebagian besar perangkat seluler berbasis android. Untuk menghindari permasalahan yang muncul karena citra dapat direkayasa lebih lanjut untuk menghasilkan dua citra yang persis sama, maka aplikasi dirancang untuk tidak bisa menyimpan citra yang telah diambil. Citra yang ditangkap akan diambil data nya dalam bentuk byte array untuk kemudian dihitung hash nya menggunakan algoritma hashing tertentu. Algoritma yang diganakan dapat bermacam-macam seperti SHA-1, SHA-256, dll. Untuk penyederhanaan maka digunakan algoritma MD5 yang menghasilkan 128 bit nilai hash yang diekspresikan kedalam 32 digit angka hexadesimal. Sebagai algoritma pembangkit bilangan acak yang digunakan pada aplikasi ini akan digunakan pembagkit bilangan acak pseudo-random yang telah disediakan pada library java yaitu kelas Random. Penggunaan kelas random ini ditujukan untuk membuktikan bahwa citra ini dapat digunakan sebagai umpan dari algoritma pembangkitan bilangan pseudo-random apapun. Untuk menyesuaikan dengan parameter masukan yang dibutuhkan oleh kelas Random ini maka dilakukan persiapan atas nilai String hasil MD5 yang dikeluarkan oleh fungsi hash. Dipersiapkan sebuah fungsi yang menghapus semua karakter yang bukan merupakan bilangan desimal kemudian panjang dari String ini dipotong menjadi hanya berjumlah 16 karakter maksimal. Hal ini dilakukan agar objek dari kelas Random dapat menerima input yang valid yaitu tipe long. long hanya dapat menyimpan 64-byte bilangan bulat (64-byte signed integer) yang mana berarti ia hanya bisa menampung bilangan bulat paling besar 19 digit. Umpan tersebut digunakan pada objek dari kelas Random dan objek dari kelas Random akan menghasilkan sebuah bilangan acak yang sesuai dengan umpan yang diberikan tadi. Pada perancangan ini umpan hanya digunakan sekali karena memang fokus utama makalah ini adalah bagaimana menghasilkan sebuah bilangan yang memiliki sifat true random untuk digunakan pada pembangkit bilangan pseudo-random sehingga akan menghasilkan bilangan yang true random pula.

```
import java.io.InputStream;
import java.io.IOException;
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.DigestInputStream;
import java.security.NoSuchAlgorithmException;

public class HashFile {
  private MessageDigest md;

  public HashFile() {
    try{
      md=MessageDigest.getInstance("MD5");
    }
    catch(NoSuchAlgorithmException e) {
      e.printStackTrace();
    }
  }

  public String generateHash(byte[] data)throws
IOException {
    md.update(data);
    byte[] output=md.digest();
    BigInteger bi=new BigInteger(1,output);
    String hashText=bi.toString(16);

    return hashText;
  }
}
```

```
import java.io.IOException;
import java.util.Random;

public class RandomGenerator {
  public   int   generaterandomintfromhash(String
hashstring, int size) {
    return
modify(generate(prepareseed(hashtopreparedstring
(hashstring))), size);
  }

  public int modify(int _number, int modifier) {
    return Math.abs(_number%modifier);
  }

  public int generate(long seed) {
    Random R = new Random(seed);
    return R.nextInt();
  }

  public    String    hashtopreparedstring(String
hashstring) {
    String temp = "";
    for(int i=0;i<hashstring.length();i++) {
      if(isnumber(hashstring.charAt(i))) temp +=
hashstring.charAt(i);
      }
    return temp;
  }

  public long prepareseed(String _string) {
    if(_string.length()>15)                return
Long.parseLong(_string.substring(0, 15));
    else return Long.parseLong(_string);
  }

  public boolean isnumber(char c) {
    if(c<48 || c>57) return false;
    else return true;
    }
}
```

IV. PENGUJIAN

untuk menguji pola acak dari bilangan yang dihasilkan maka dilakukan pengambilan sampel sebanyak 100 kali pembangkitan. Pengujian ini dilakukan pada randomasi bilangan 0-9. Pengujian dilakukan dengan Metode Chi Squared..

Row	Category	Observed	Expected#	Expected
1	0	7	10	10%
2	1	9	10	10%

3	2	13	10	10%
4	3	8	10	10%
5	4	8	10	10%
6	5	12	10	10%
7	6	10	10	10%
8	7	12	10	10%
9	8	13	10	10%
10	9	8	10	10%

Nilai P dan signifikansi statistic yang didapatkan adalah

- Chi Squared = 4.800 dengan 9 derajat kebebasan
- Two-Tailed P = 0.8514

Dengan kriteria konvensional perbedaan ini dianggap tidak statistically significant.

V. KESIMPULAN

Dari hasil uji chi squared diatas dapat disimpulkan bahwa persebaran angka yang dibangkitkan tidak memiliki perbedaan yang signifikan dari data yang nilai yang diharapkan. Maka dapat disimpulkan bahwa angka yang dibangkitkan dengan aplikasi ini merupakan angka yang bersifat random.

Nilai hash dari file apapun dapat digunakan untuk dijadikan umpan pada pembangkit bilangan pseudo-random. Lebih khususnya file citra, yang diambil secara real-time. Citra yang diambil secara real-time tidak dapat diproduksi kembali dengan hasil yang persis sama tanpa rekayasa lebih lanjut. Hal ini dikarenakan dimensi waktu yang selalu berubah. Dalam aplikasinya di kehidupan sehari-hari bahkan dimensi ruang akan terus berubah seiring dengan perubahan dimensi waktu. Hal ini menjadi dasar dimana sebuah citra dapat dijadikan umpan yang valid dalam menghasilkan sebuah bilangan true random.

Terdapat pembuktian lebih lanjut dimana setiap algoritma pseudo-random akan menghasilkan bilangan true-random apabila diberikan umpan yang bersifat true random.

PUSTAKA

L'Ecuyer, Pierre. Random Number Generation. D´epartement d'Informatique et de Recherche Op´erationnelle, Universit´e de. Montr´eal, C.P. 6128, Succ. Centre-Ville, Montr´eal (Qu´ebec), H9S 5B8, Canada.

www.random.org tanggal akses 14 Mei 2014

Hellekalek, P. Good random number generators are (not so) easy to find. Dept. of Mathematics, Salzburg University, Hellbrunner Straûe 34, A-5020 Salzburg, Austria

developer.android.com tanggal akses 15 Mei 2014

Rivest, R. The MD5 Message-Digest Algorithm. 1992. MIT Laboratory for Computer Science and RSA Data Security, Inc.

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, 16 Mei 2014

Fitrandi Ramadhan 1308065