CRYPTOGRAPHICALLY SECURE RANDOM NUMBER GENERATOR (CSRNG) BERDASARKAN BILANGAN PI DAN BILANGAN EULER

Jansen – NIM : 13506028

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if16028@students.if.itb.ac.id

Abstrak

Perkembangan teknologi pada saat ini telah banyak mendorong banyak kemudahan untuk melakukan apapun di dunia maya (Internet): pengiriman *e-mail*, penjualan barang, dan yang lain. Namun karena Internet merupakan dunia yang sangat terbuka, banyak pihak membutuhkan privasi dan hak cipta. Berbagai teknik dan algoritma mulai dari penyandian sampai kepada tanda tangan digital dikembangkan untuk menjaga hal tersebut.

Namun, berbagai teknik dan algoritma tersebut berdasarkan kepada kekuatan kunci. Oleh sebab itu, kunci yang dibangkitkan sebaiknya berupa bilangan yang benar-benar acak (*truly random number*). Namun, *truly random number* tersebut tidak mungkin dapat dibangkitkan oleh *software*, harus dengan *hardware* dengan cara mendeteksi entropi/kekacauan dari radioaktif, *noise*, pergerakan foton atau yang lain. Karena alasan tersebutlah, dibuat suatu standar *cryptographically secure random number* (*CSRN*) atau bilangan sesungguhnya yang merupakan pseudonumber, namun tahan terhadap serangan yang dapat dibangkitkan melalui *software*.

Sudah banyak cara yang dilakukan untuk membuat *cryptographically secure random number* tersebut dan hasilnya juga sudah banyak. Penulis, melalui makalah ini, hendak memperkenalkan sebuah metode baru untuk membangkitkan *CSRN* tersebut menggunakan bilangan Pi (***π***) dan bilangan Euler (**e**).

Bilangan Pi (***π***) dan bilangan Euler (**e**) merupakan bilangan yang disebut-sebut sebagai bilangan yang mungkin tidak akan pernah habis dan berulang dan pengujian sampai saat ini belum ada yang mengabarkan bahwa kedua bilangan tersebut berulang. Berdasarkan fakta tersebut, penulis yakin bahwa menggunakan kedua bilangan tersebut dalam fungsi pembangkitan bilangan acak akan menghasilkan bilangan-bilangan dengan tingkat keacakan yang lebih tinggi.

Penulis melalui makalah ini akan menjelaskan cara baru pembangkitan *CSPRN* dengan menggunakan Bilangan Pi (***π***) dan bilangan Euler (**e**).

***Kata kunci**: cryptographically secure*, pembangkit bilangan acak, bilangan Pi, bilangan Euler, enkripsi, dekripsi.

1. Pendahuluan

Perkembangan teknologi dewasa ini membutuhkan suatu metode untuk menjamin keamanan, keaslian dan keutuhan suatu data baik ketika data tersebut disimpan ataupun dikirimkan melalui Internet dikarenakan Internet merupakan suatu dunia yang sangat terbuka dan setiap orang dapat berpartisipasi di dalamnya. Tidak tertutup kemungkinan bahwa dari orang-orang tersebut ada yang berusaha mencuri dan merusak data tersebut. Untuk menangani hal tersebut, data yang akan dikirimkan tersebut sebelumnya dilakukan proses penyandian (enkripsi dan dekripsi) atau proses penandatanganan digital.

Pada proses tersebut, diperlukan adanya kunci-kunci. Boleh dikatakan bahwa kekuatan sebagian besar algoritma enkripsi-dekripsi dan algoritma tandatangan digital terletak pada kunci yang bersifat acak. Semakin acak kunci, semakin baik pula tingkat keamanannya.

2. Alat Pembangkit Bilangan Acak

Alat untuk membangkitkan bilangan acak ada 2, yaitu:

1. Perangkat keras (*hardware*)
2. Perangkat lunak (*software*)

2.1 Pembangkitan Bilangan Acak Menggunakan Perangkat Keras

Pembangkitan bilangan acak menggunakan perangkat keras dapat menghasilkan bilangan yang benar-benar acak (*truly random*). Pembangkitan tersebut biasanya mengambil model dari lingkungan seperti pergerekan foton, entropi/kekacauan radioaktif, *noise*, dan lain-lain. Namun dikarenakan perangkat keras biasanya mahal dan pembangkitan bilangan acak tersebut membutuhkan waktu yang relatif lebih lama, maka jarang sekali orang menggunakannya.

2.2 Pembangkitan Bilangan Acak Menggunakan Perangkat Lunak

Berkebalikan dengan pembangkitan bilangan acak dengan menggunakan pernangkat keras, pembangkitan bilangan acak dengan menggunakan perangkat lunak membutuhkan waktu yang lebih cepat, walaupun bilangan yang dibangkitkan bukan merupakan *truly random*, tetapi merupakan bilangan acak semu (*pseudorandom*).

Namun, bilangan-bilangan tersebut sudah dirasakan cukup aman untuk dipakai sebagai kunci dalam enkripsi-dekripsi apabila merupakan *cryptographically secure pseudorandom number generator* (*CSPRN*G). Persyaratan *CSPRNG* adalah:

1. Secara statistik ia mempunyai sifat-sifat yang bagus (yaitu lolos uji keacakan statistik).
2. Tahan terhadap serangan (*attack*) yang serius. Serangan ini bertujuan untuk memprediksi bilangan acak yang dihasilkan.

Beberapa algoritma yang merupakan *cryptographically secure pseudorandom number generator* (CSPRNG) adalah Blum Blum Shub dan algoritma Yarrow.

3. Bilangan Pi (π)

Bilangan Pi merupakan salah satu konstanta yang paling populer dan paling penting dalam bidang matematika dan fisika seperti menghitung luas lingkaran.
Bilangan Pi telah diketahui selama hampir 400 tahun. Para ilmuwan matematik mulai menggunakan simbil π pada tahun 1700. Diperkenalkan oleh William Jones pada tahun 1706, simbol tersebut dipopulerkan oleh Leonhard Euler pada tahun 1737.

Bilangan Pi pada saat ini telah dihitung sampai ketelitian triliunan digit (10^{12}). 200 digit pertama bilangan Pi adalah
3.14159265358979323846264338327950288419
71693993751058209749445923078164062862089986280348253421170679821480865132823066470938446095505822317253594081284811174502841027019385211055596446229489549303819.

3. Bilangan Euler (e)

Bersamaan dengan bilangan Pi, bilangan Euler merupakan salah satu bilangan yang paling populer dan paling penting dalam bidang matematika. Bilangan Euler juga merupakan bilangan yang tak pernah berulang dan kemungkinan besar tidak pernah habis.

200 digit pertama bilangan Euler adalah
2.71828182845904523536028747135266249775724709369995957496696762772407663035354759457138217852516642742746639193200305992181741359662904357290033429526059563073813232862794349076323382988075319525101901.

4. *CSPRNG* Berbasis π dan e

Pada CSPRNG berbasis bilangan Pi dan Euler ini, terdapat 5 parameter nilai, yaitu :

1. Parameter *idxp*, menyatakan indeks dari bilangan Pi yang akan diambil.
2. Parameter *idxe*, menyatakan indeks dari bilangan Euler yang akan diambil.
3. Parameter *r*, menyatakan faktor pengali.
4. Parameter *p*, menyatakan faktor pemodulo.
5. Parameter *q*, menyatakan faktor pemodulo.

4.1 Algoritma *CSPRNG*

Algoritma yang digunakan adalah sebagai berikut.

1. Pilih dua bilangan *p* dan *q* yang masing-masing merupakan bilangan prima.

2. Hitung n dengan mengalikan p dan q.
3. Pilih r sedemikian rupa sehingga:
   - r merupakan bilangan relatif prima terhadap n.
   - $r < n$.
4. Barisan bilangan acak dapat dihasilkan dengan melakukan iterasi sepanjang yang diinginkan dengan menghitung $x_i = (x_{i-1} * r_i * \pi_{idxp_i} * e_{idxe_i}) \bmod n \bmod 10$ jika $x_{i-1} \neq 0$ dan $x_i = (r_i * \pi_{idxp_i} * e_{idxe_i}) \bmod n \bmod 10$ jika $x_{i-1} = 0$ di mana:
   - x_0 adalah r mod 10
   - $r_i = r_{i-1}^2$ mod *panjang_file*.
   - π_{idxp_i} adalah digit bilangan Pi ke-$idxp_i$ dan $idxp_i = idxp_{i-1}^2$ mod *panjang_file*.
   - e_{idxe_i} adalah digit bilangan Euler ke-$idxe_i$ dan $idxe_i = idxe_{i-1}^2$ mod *panjang_file*.
   - *panjang_file* adalah panjang arsip eksternal hasil pembangkitan bilangan Euler dan bilangan Pi.

   Barisan bilangan acak adalah x_1, x_2, x_3, x_4, x_5, …

4.2 Contoh *CSPRNG* Berbasis π dan e

Misalkan p = 19 dan q = 31 sehingga $n = pq$ = 589. Kita memilih *idxp* = 239, *idxe* = 7 dan r = 13. *panjang_file* adalah 1000000. Barisan bilangan acak didapatkan sebagai berikut:

- $idxp_0$ = 239
  $idxe_0$ = 7
  r_0 = 13
  $x_0 = r_0$ mod 10 = 3
- $idxp_1 = 239^2$ mod 1000000 = 57121
  $idxe_1 = 7^2$ mod 1000000 = 49
  $r_1 = 13^2$ mod 1000000 = 169
  $x_1 = x_0 * r_1 * \boldsymbol{\pi}_{idx\text{-}1} * \mathbf{e}_{\text{idx-1}}$ mod n mod 10 = 3 * 169 * 9 * 9 mod 589 mod 10 = 6.
- $Idxp_2 = 57121^2$ mod 1000000 = 808641
  $Idxe_2 = 49^2$ mod 1000000 = 2401
  $R_2 = 169^2$ mod 1000000 = 28561
  $X_2 = x_1 * r_2 * \boldsymbol{\pi}_{idx\text{-}2} * \mathbf{e}_{\text{idx-2}}$ mod n mod 10 = 6 * 28561 * 7 * 9 mod 589 mod 10 = 7.
- $Idxp_3 = 808641^2$ mod 1000000 = 266881
  $Idxe_3 = 49^2$ mod 1000000 = 764801
  $R_3 = 28561^2$ mod 1000000 = 730721
  $X_3 = x_2 * r_3 * \boldsymbol{\pi}_{idx\text{-}3} * \mathbf{e}_{\text{idx-3}}$ mod n mod 10 = 7 * 730721 * 7 * 8 mod 589 mod 10 = 2.
- $Idxp_4 = 808641^2$ = 468161
  $Idxe_4 = 2401^2$ = 569601
  $R_4 = 28561^2$ = 179841
  $X_4 = x_3 * r_4 * \boldsymbol{\pi}_{idx\text{-}4} * \mathbf{e}_{\text{idx-4}}$ mod n mod 10 = 2 * 179841 * 468161 * 569601 mod 589 mod 10 = 0.
- $Idxp_5 = 468161^2$ = 721921
  $Idxe_5 = 569601^2$ = 299201
  $R_5 = 179841^2$ = 785281
  $X_5 = r_5 * \boldsymbol{\pi}_{idx\text{-}5} * \mathbf{e}_{\text{idx-5}}$ mod n mod 10 = 785281* 721921* 299201 mod 589 mod 10 = 8.

Jadi 5 digit pertama dari hasil pembangkitan bilangan acak dengan parameter di atas adalah 67208.

4.3 Arsip Bilangan Pi dengan Panjang 1000 Digit

3141592653589793238462643383279502884197169399375105820974944592307816406286208998628034825342117067982148086513282306647093844609550582231725359408128481117450284102701938521105559644622948954930381964428810975665933446128475648233786783165271201909145648566923460348610454326648213393607260249141273724587006606315588174881520920962829254091715364367892590360011330530548820466521384146951941511609433057270365759591953092186117381932611793105118548074462379962749567351885752724891227938183011949129833673362440656643086021394946395224737190702179860943702770539217176293176752384674818467669405132000568127145263560827785771342757789609173637178721468440901224953430146549585371050792279689258923542019956112129021960864034418159813629774771309960518707211349999998372978049951059731732816096318595024459455346908302642522308253344685035261931188171010003137838752886587533208381420617177669147303598253490428755468731159562863882353787593751957781857780532171226806613001927876611195909216420198

4.4 Contoh Format Arsip Bilangan Euler dengan Panjang 1000 Digit

271828182845904523536028747135266249
775724709369995957496696762772407663
035354759457138217852516642742746639
193200305992181741359662904357290033
429526059563073813232862794349076323
382988075319525101901157383418793070
215408914993488416750924476146066808
226480016847741185374234544243710753
907774499206955170276183860626133138
458300075204493382656029760673711320
070932870912744374704723069697720931
014169283681902551510865746377211125
238978442505695369677078544996996794
686445490598793163688923009879312773
617821542499922957635148220826989519
366803318252886939849646510582093923
982948879332036250944311730123819706
841614039701983767932068328237646480
429531180232878250981945581530175671
736133206981125099618188159304169035
159888851934580727386673858942287922
849989208680582574927961048419844436
346324496848756023362482704197862320
900216099023530436994184914631409343
173814364054625315209618369088870701
676839642437814059271456354906130310
720851038375051011574770417189861068
739696552126715468895703503 5

4.5 Kelebihan dan Kekurangan *CSPRNG* Berbasis π dan e

Keamanan CSPRNG ini (sama seperti Blum Blum Shub) terletak pada sulitnya memfaktorkan nilai *n*. Semakin besar nilai *n*, maka semakin aman pembangkit bilangan acak ini.

Namun, sulitnya membangkitkan secara *runtime* bilangan Pi dan Euler mengakibatkan bilangan Pi dan Euler harus dibangkitkan terlebih dahulu dan disimpan sebagai arsip eksternal dan kemudian barulah pengambilan indeks digit dilakukan pada arsip eksternal tersebut. Semakin panjang bilangan Pi dan Euler yang dibangkitkan, semakin baik dan acak pula bilangan yang dihasilkan.

4.6 Kode Program

Berikut adalah cuplikan kode program yang dibuat dalam bahasa C#.

```
// mengembalikan digit ke-index dari
// arsip path
private int getNumber(string path, long
   index) {
   try {
      BinaryReader reader = new
         BinaryReader(File.Open(path,
      FileMode.Open));
      reader.BaseStream.Position = index;
      char tmp = reader.ReadChar();
      reader.Close();
      return Int32.Parse(tmp + "");
   } catch (Exception) {
      return (char)0;
   }
}

// mengembalikan angka hasil pembangkitan
// bilangan acak
private int generateRandom(long p, long q,
   long r, long idxP, string pathPi, long
   idxE, string pathE,long iterasi) {

   // inisialisasi
   long n = p * q;

   // untuk inisialisasi x0
   int x = (int)(r % 10);
   r = (r * r) % 1000000;
   idxP = (idxP * idxP) % 1000000;
   idxE = (idxe * idxE) % 1000000;

   for (int i = 0; i < iterasi; i++) {
      // mengambil digit index
      // ke-idxp dari bilangan Pi
      long getP = getNumber(pathPi, idxP);
      // mengambil digit index ke-idxe
      // dari bilangan Euler
      long getE = getNumber(pathE, idxE);

      if (x == 0)
         x = (int)((r * getP * getE) %
            n) %10;
      else
         x = (int)((x * r * getP * getE) %
            n) % 10;

      // memroses bilangan r, idxp, idxe
      r = (r * r) % 1000000;
      idxP = (idxP * idxP) % 1000000;
      idxE = (idxE * idxE) % 1000000;
   }

   return x;
}
```

5. Kesimpulan

Kesimpulan yang dapat dimbil dari metode baru pembuatan *cryptographically secure random number generator* (*CSPRNG*) ini adalah:

1. Pemakaian fungsi *random* di kebanyakan bahasa pemrograman atau sistem operasi tidak aman untuk digunakan sebagai kunci.

2. Tidak semua algoritma *PRNG* (*pseudorandom number generator*) merupakan *CSPRNG* dikarenakan tidak terpenuhinya kedua syarat *CSPRNG*.
3. Pembangkitan bilangan acak dengan menggunakan perangkat keras dapat menghasilkan bilangan yang *truly random*, namun memiliki kendalanya sendiri.
4. Pembangkitan bilangan acak dengan menggunakan perangkat lunak tidak mungkin dapat menghasilkan bilangan yang *truly random*.

DAFTAR PUSTAKA

[1] Sunny Beach. *http://www.sunny-beach.net/random_numbers/manual/173.htm*. Tanggal akses: 21 Mei 2009 pukul 05:00.

[2] Robert J Nemiroff. http://antwrp.gsfc.nasa.gov/htmltest/rjn_dig.html. Tanggal akses: 21 Mei 2009 pukul 05:00.

[3] Munir, Rinaldi. (2004). Bahan Kuliah IF5054 Kriptografi. Departemen Teknik Informatika, Institut Teknologi Bandung.

[4] GAP (Groups, Algorithms, Programming). http://www.gap-system.org/~history/HistTopics/e.html. Tanggal akses: 21 Mei 2009 pukul 05:00.

[5] Eve Astrid Andersson. http://www.eveandersson.com/pi/digits/. Tanggal akses: 21 Mei 2009 pukul 05:00.

[6] Daniel B. Sedory. (2007). http://www.geocities.com/tsrmath/pi/RandPI.html. Tanggal akses: 21 Mei 2009 pukul 05:00.