IMPLEMENTASI UJI KORELASI UNTUK PENGUJIAN SUB KUNCI PADA ALGORITMA KRIPTOGRAFI *BLOCK CIPHER PRESENT* MENGGUNAKAN BAHASA PEMROGRAMAN C++

Faizal Achmad
Lembaga Sandi Negara
e-mail : faizal.achmad@lemsaneg.go.id

Abstrak
Kriptografi merupakan salah satu teknik dalam mengamankan suatu informasi berklasifikasi. Penerapan suatu aplikasi kriptografi dalam mengamankan suatu informasi tidak lepas dari pemilihan kekuatan algoritma kriptografi yang digunakan. Pada algoritma kriptografi block cipher, salah satu komponen pembangun yang harus diperhatikan adalah key schedule, untuk menghasilkan sub kunci yang digunakan pada proses enkripsi/dekripsi. Metode uji korelasi merupakan salah satu metode yang digunakan untuk menguji kekuatan algoritma kriptografi block cipher. Penelitian paper ini akan membuat implementasi pengujian korelasi menggunakan bahasa pemrograman C++, yang dterapkan pada block cipher PRESENT sebagai salah satu algoritma yang banyak digunakan saat ini.

***Kata Kunci** : Kriptografi, Uji Korelasi, Block Cipher, PRESENT, C++.*

1. PENDAHULUAN

1. Latar Belakang Masalah

Ancaman yang kerap terjadi dalam penyampaian suatu informasi antara lain pengubahan, penyadapan, dan pemalsuan. Salah satu teknik pengamanan terhadap suatu informasi adalah dengan aplikasi kriptografi. Dalam kriptografi simetrik, kunci untuk enkripsi sama dengan kunci untuk dekripsi. Berdasarkan bentuknya algoritma sistem sandi simetrik terbagi 2 yaitu, *Stream Cipher* dan *Block Cipher*.

Block Cipher memproses satu grup karakter dari *plaintext* dienkripsi secara serentak dengan menggunakan transformasi enkripsi yang telah ditetapkan. Untuk menguji kekuatan komponen pada *Block Cipher* telah tersedia berbagai *tools* dan metode yang dapat digunakan. Salah satunya adalah metode yang dipublikasikan pada tahun 2007 oleh Muhammad Saqib Niaz, yang membuat analisis dan perancangan metode pengujian korelasi untuk sub kunci pada *block cipher*, sebagai salah satu alternatif pengujian sub kunci pada *block cipher*. Penulisan paper ini akan membuat implementasi pengujian korelasi tersebut pada algoritma kriptografi PRESENT, yang merupakan *lightweight block cipher* yang dikembangkan oleh Orange Lab (Perancis), Ruhr University Bochum (Jerman) dan Technical University of Denmark pada tahun 2007. Implementasi pada penelitian paper ini akan dibuat menggunakan bahasa pemrograman C++.

2. Tujuan Penulisan

Tujuan penulisan paper ini adalah implementasi uji korelasi pada pengujian sub kunci algoritma kriptografi PRESENT dengan menggunakan bahasa pemrograman C++. Hasil implementasi ini diharapkan dapat digunakan sebagai alternatif pengujian pada algoritma kriptografi *block cipher* lainnya, terutama pengujian pada sub kunci yang dihasilkan.

3. Perumusan Masalah

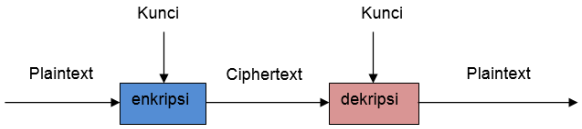
Dalam implementasi uji korelasi pada algoritma kriptografi PRESENT, terdapat beberapa hal permasalahan sebagai berikut :

- Implementasi pembangkitan sub kunci algoritma kriptografi PRESENT dengan banyak sampel yang dibutuhkan.
- Implementasi metode uji korelasi disesuaikan dengan input berupa sub kunci yang dihasilkan oleh algoritma kriptografi PRESENT.

- Pengujian keacakan rangkaian dari sub kunci algoritma kriptografi PRESENT yang dibangkitkan menggunakan metode uji korelasi.

2. TINJAUAN PUSTAKA

1. Definisi Kriptografi
   Berikut merupakan istilah-istilah yang terdapat dalam kriptografi [1] :
   - Enkripsi adalah proses menyamarkan suatu pesan sebagai cara untuk menyembunyikan isinya.
   - *Plaintext* (Teks Terang) adalah suatu pesan yang belum terenkripsi.
   - *Ciphertext* (Teks Sandi) adalah suatu pesan yang telah terenkripsi.
   - Dekripsi adalah suatu proses untuk mengembalikan Teks Sandi menjadi Teks Terang.



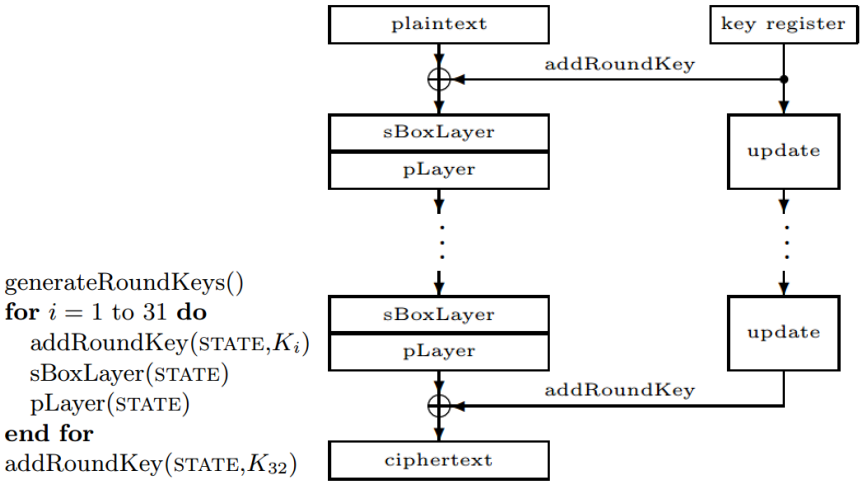
Gambar 1. Proses Enkripsi/Dekripsi

Secara umum dalam kriptografi terdapat dua macam metode sistem sandi, yaitu sistem sandi simetrik dan sistem sandi asimetrik. Sistem sandi simetrik adalah sistem sandi yang menggunakan satu kunci, dimana kunci untuk enkripsi sama dengan kunci untuk dekripsi. Sistem sandi asimetrik adalah sistem sandi yang menggunakan dua kunci yang berlainan, dimana kunci untuk enkripsi berbeda dengan kunci untuk dekripsi. Berdasarkan bentuknya algoritma sistem sandi simetrik terbagi 2 yaitu, *Stream Cipher* dan *Block Cipher*. Pada Stream Cipher, karakter dari *plaintext* dienkripsi satu persatu pada satu waktu. Pada *Block Cipher*, satu grup karakter dari *plaintext* dienkripsi secara serentak dengan menggunakan transformasi enkripsi yang ditetapkan.

2. Block Cipher
   Berikut merupakan beberapa definisi mengenai *block cipher:*
   - *Block cipher* menurut William Stalling (Stalling. 2005) adalah skema enkripsi/dekripsi yang memperlakukan blok *plaintext* secara keseluruhan untuk menghasilkan blok *ciphertext* dengan panjang yang sama.
   - *Block cipher* menurut Alfred J. Menezes dkk (Menezes dkk. 1997) adalah suatu fungsi yang memetakan *n*-bit blok *plaintext* ke *n*-bit blok *ciphertext* dengan *n* adalah panjang blok. *Block cipher* dapat dipandang sebagai sistem sandi substitusi sederhana dengan periode panjang. Fungsi dalam *block cipher* diparameterkan oleh K, yaitu kunci dengan panjang *k*-bit dengan mengambil nilai-nilai dari *subset K* (ruang kunci) dari himpunan semua vektor-vektor *k*-bit *Vk*. Umumnya diasumsikan kunci dipilih secara acak. Pengggunaan ukuran blok *plaintext* dan *ciphertext* yang sama bertujuan untuk menghindari penambahan data.

3. Algoritma Kriptografi PRESENT
   PRESENT adalah *lightweight block cipher* yang dikembangkan oleh Orange Lab (Perancis), Ruhr University Bochum (Jerman) dan Technical University of Denmark pada tahun 2007 (Bogdanov, dkk,.2007).

   Deskripsi
   PRESENT merupakan salah satu contoh dari SP-Network dan terdiri dari 31 *round*. Dengan panjang blok 64 bit dan panjang kunci yang terdiri dari 80 bit atau 128 bit. Setiap dari 31 *round* terdiri atas suatu operasi XOR untuk menghasilkan kunci round *Ki* untuk $1 < i < 32$, dimana K_{32} digunakan untuk *post whitening*, *linear bitwise permutation* dan *non-linear substitution layer*.



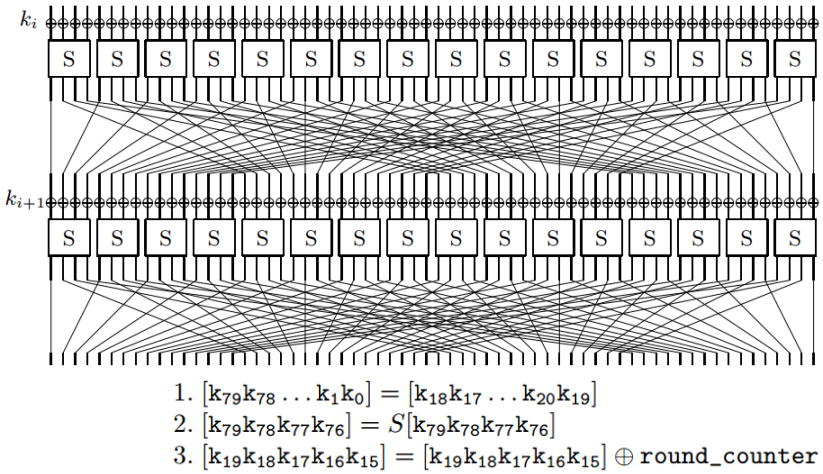
Gambar 2. Skema Algoritma Kriptografi PRESENT

Key Schedule

PRESENT dapat menerima input kunci 80 bit atau 128 bit. Pada penulisan ini akan dijelaskan *key schedule* untuk input kunci 80 bit. Input kunci disimpan dalam suatu kunci *register K* dan direpresentasikan sebagai $k_{79}, k_{78}, \ldots, k_0$. Pada *round i* 64-bit kunci *round* $K_i = k_{63}, k_{64}, \ldots, k_0$ terdiri dari 64 *leftmost bit* dari konten terkini register *K*. Sehingga pada *round i* didapatkan:

$$K_i = k_{63}, k_{62}, \ldots, k_0 \ = k_{79}, k_{78}, \ldots, k_{16}$$

Setelah ekstraksi kunci *round* K_i, *key register* $K=k_{79}k_{78}...k_0$ diperbaharui dengan cara berikut:



Gambar 3. Skema Pembangkit kunci Algoritma Kriptografi PRESENT

4. Metode Uji Korelasi

Perubahan secara simultan dari dua nilai variabel acak numerik disebut dengan korelasi. Sementara rangkaian biner adalah acak, jika digit binernya independen secara statistik dan seimbang. Cara yang sangat dasar untuk memeriksa korelasi antara dua urutan biner adalah dengan melakukan XOR antara dua rangkaian biner, lalu dilanjutkan dengan memeriksa keacakan dari rangkaian baru hasil dari XOR tersebut. Jika rangkaian yang dibangkitkan acak maka dapat dianggap kedua rangkaian yang di XOR tersebut tidak memiliki korelasi. Kita dapat memeriksa keacakan dari suatu rangkaian dengan menerapkan pengujian statistik pada rangkaian yang dibangkitkan melalui operasi XOR.

Untuk memeriksa keacakan dari subkunci, Muhammad Saqib Niaz (Niaz. 2007) telah menyusun empat metode untuk menghasilkan rangkaian XOR. Setelah menghasilkan rangkaian XOR, diterapkan empat uji statistik yaitu *frequency test, poker test, runs test* dan *auto-correlation test* untuk memeriksa keacakan dari rangkaian yang dibangkitkan.

5. Lima Uji Dasar Untuk Keacakan
   Lima uji dasar untuk keacakan adalah 5 buah uji statistik yang biasanya digunakan untuk menentukan apakah suatu barisan biner memiliki karakteristik seperti yang ditunjukkan oleh suatu barisan yang *trully random* (Alfred J. Menezes, dkk,.1996: 181-183).
   Lima uji dasar untuk keacakan tersebut terdiri dari:
   - Uji Frekuensi
     Uji frekuensi bit digunakan untuk menentukan apakah barisan bit mempunyai jumlah bit "0" dan "1" yang relatif sama.
   - Uji Serial
     Tujuan uji serial adalah untuk menentukan jumlah dari pasangan bit "00", "01", "10" dan "11" sebagai sub barisan dari barisan bit yang mempunyai jumlah sama.
   - Uji Poker
     Uji poker digunakan untuk menentukan jumlah barisan bit dengan lebar tertentu muncul dalam barisan bit dengan jumlah yang diharapkan.
   - Uji Run
     Uji run dilakukan untuk mengetahui apakah jumlah runtun dalam barisan bit, baik runtun bit "0" maupun "1", dengan panjang yang bervariasi mempunyai jumlah yang sesuai dengan postulat keacakan Golomb butir R2.
   - Uji Autokorelasi
     Tujuan dari uji autokorelasi adalah untuk melihat kemungkinan hubungan antara barisan bit dengan versi pergeserannya. Sesuai dengan postulat keacakan Golomb butir R3, nilai autokorelasi adalah suatu nilai yang konstan meskipun rangkaian mengalami pergeseran beberapa kali.

3. METODE PENELITIAN

Metode Penelitian yang dilakukan pada penelitian ini adalah sebagai berikut :

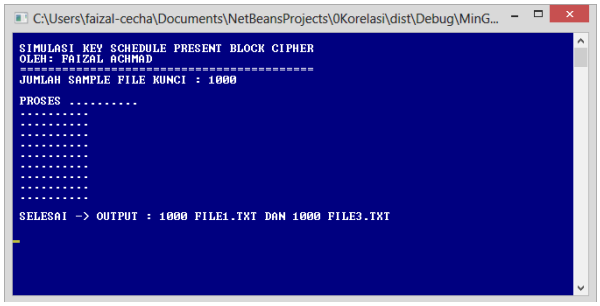
- Melakukan studi literatur dan pengumpulan data dari berbagai sumber seperti buku dan internet mengenai penelitian yang akan dilakukan.
- Melakukan analisis kebutuhan dalam pembuatan aplikasi uji korelasi dan PRESENT *block cipher*.
- Melakukan proses perancangan aplikasi uji korelasi dan PRESENT *block cipher*.
- Membuat *prototype* aplikasi uji korelasi dan PRESENT *block cipher* menggunakan bahasa pemrograman C++.
- Melakukan pengujian keacakan rangkaian menggunakan uji frekuensi, uji serial, uji poker dan uji run.

4. HASIL DAN PEMBAHASAN.

1. Pembangkitan Sub Kunci PRESENT

Uji korelasi digunakan untuk menguji sub kunci yang dihasilkan oleh algoritma PRESENT, sub kunci dihasilkan menggunakan metode *key schedule* dengan input kunci 80 bit, seperti yang telah dijelaskan pada bab sebelumnya. Sub kunci yang dihasilkan untuk suatu *full round* algoritma PRESENT adalah sebanyak 32 sub kunci, yang masing-masing bernilai 64 bit.

Pada penelitian ini diimplementasikan aplikasi pembangkitan kunci algortima PRESENT menggunakan bahasa pemrograman C++ seperti pada gambar 4. dibawah ini:



Gambar 4. Aplikasi Pembangkit Kunci dan Rangkaian Uji Korelasi Algoritma Kriptografi PRESENT

Aplikasi menghasilkan sebanyak 1000 file sampel, dengan input 80 bit kunci yang dipilih secara acak, dan menghasilkan output sebanyak 32 sub kunci.

2. Pembangkitan Rangkaian Uji Korelasi

Untuk memeriksa keacakan dari subkunci, Muhammad Saqib Niaz (Niaz. 2007) telah menyusun empat metode untuk menghasilkan rangkaian XOR. Setelah menghasilkan rangkaian XOR, diterapkan uji statistik untuk memeriksa keacakan dari rangkaian yang dibangkitkan. Pada paper ini hanya membahas Metode I sebagai meotde yang paling dasar dan Metode III sebagai metode yang lebih direkomendasikan.

1) Metode I

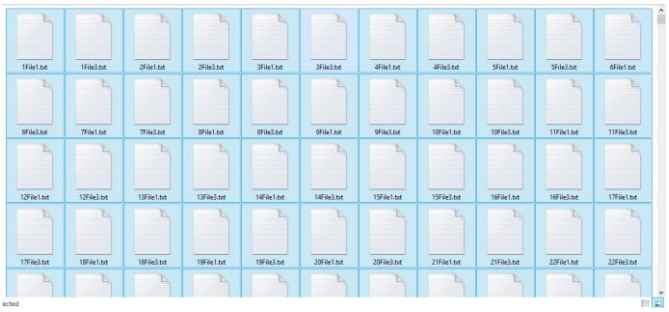
Metode ini adalah metode yang paling sederhana untuk memeriksa korelasi antara sub kunci. Dalam metode ini setiap sub kunci di XOR dengan sub kunci lainnya dan rangkaian biner dibangkitkan dengan cara seperti ini. Setelah membangkitkan rangkaian, pengujian secara statistik diterapkan. Akan tetapi metode ini masih memiliki kelemahan, karena hanya memberikan hubungan antara bit masing-masing sub kunci, yaitu bit pertama dari suatu sub kunci dengan bit pertama dari sub kunci lainnya. Metode ini tidak dapat mengetahui apakah terdapat hubungan antara bit pertama dari suatu sub kunci dengan bit kedua dari sub kunci lainnya.Dengan menggunakan notasi yang diberikan, r adalah sub kunci yang masing-masing memiliki m/8 bytes. Proses dimulai dengan K_1 yang di XOR dengan setiap sub kunci lainnya misal K_2, K_3, ..., K_r. Lalu ambil K2 dan XOR dengan sub kunci lainnya, kecuali sub kunci yang telah di XOR sebelumnya misal K_3, K_4, ..., K_r. Dengan cara ini dilakukan XOR K_{r-1} hanya dengan K_r dan K_r sendiri tidak di XOR dengan sub kunci lainnya.

2) Metode III

Pada metode ini akan dibangkitkan rangkaian dengan cara melakukan XOR pada setiap bit dari setiap sub kunci dengan setiap bit dari setiap sub kunci lainnya. Metode ini lebih baik daripada metode sebelumnya, karena dapat memberitahu korelasi antara setiap bit dari setiap sub kunci, yang membangkitkan rangkaian biner lebih panjang daripada metode sebelumnya.

Metode ini mengambil byte pertama dari sub kunci pertama misal $K_1[1]$ dan melakukan XOR dengan setiap byte dari sub kunci kedua misal K_2. Lalu $K_1[1]$ dirotasi satu kali dan di XOR dengan setiap byte dari K_2. Rotasi dan XOR ini dilakukan sampai $K_1[1]$ dirotasi sebanyak tujuh kali, karena pada rotasi kedelapan telah kembali menjadi byte yang semula. Dengan cara ini $K_1[1]$ di XOR dengan setiap byte K_2 delapan kali, dimana satu kali tanpa rotasi dan tujuh kali dengan rotasi. Lalu pindah ke $K_1[2]$ dan XOR dengan K_2, lalu $K_1[2]$ dirotasi ke kiri satu kali seperti yang telah dilakukan pada $K_1[1]$ dan XOR dengan setiap byte dari K_2. Dengan cara ini setiap byte dari K_1 di XOR dengan setiap byte dari K_2 delapan kali. Setelah selesai dengan K_2, pindah ke K_3. Dimulai dengan $K_1[1]$ dan XOR dengan setiap byte dari K_3, dan keseluruhan proses diulang sampai setiap byte dari K_1 di XOR delapan kali dengan setiap byte dari K_3.

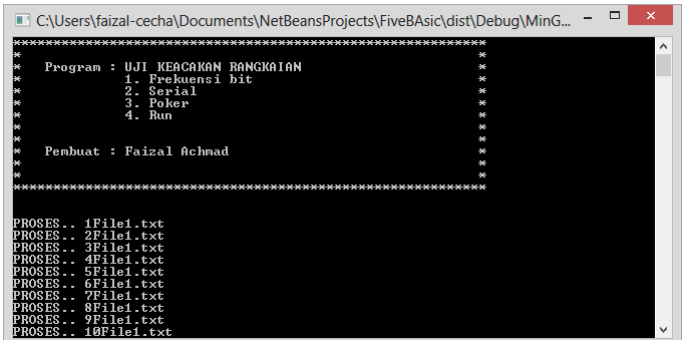
Pada bagian sebelumnya didapatkan sebanyak 1000 sampel yang masing-masing berisi 32 sub kunci, sampel ini kemudian akan digunakan sebagai input untuk membangkitkan rangkaian menggunakan Metode I dan Metode III, rangkaian tersebut disimpan dalam bentuk file, dimana masing-masing metode menghasilkan 1000 file sampel (1000 File1.txt dan 1000 File3.txt), seperti yang terlihat pada gambar 5. di bawah ini.



Gambar 5. Sampel 2000 File dari Sub Kunci Algoritma Kriptografi PRESENT

3. Pengujian Keacakan Rangkaian

Pengujian keacakan rangkaian kunci dilakukan menggunakan aplikasi pengujian seperti yang terlihat pada gambar 6. di bawah ini. Sampel yang digunakan dalam penelitian ini adalah 1000 sampel Key Input yang menghasilkan 1000 file rangkaian dari Metode I dan 1000 file rangkaian dari Metode III. Hasil uji keacakan dari rangkaian Metode I dan Metode III akan dibandingkan pada masing-masing uji.



Gambar 6. Aplikasi Pengujian Keacakan Rangkaian

1) Uji Frekuensi

Uji frekuensi bit digunakan untuk menentukan apakah barisan bit mempunyai jumlah bit "0" dan "1" yang relatif sama. Seperti yang diharapkan untuk barisan bit acak pada postulat keacakan Golomb butir R1. Statistik uji yang digunakan adalah:

$$X_1 = \frac{(n_0 - n_1)^2}{n}$$

dimana n = banyaknya bit dalam rangkaian
n_0 = banyaknya bit "0"
n_1 = banyaknyan bit "1"

Pengujian menggunakan pendekatan distribusi χ^2 dengan derajat kebebasan v=1, sehingga untuk v=1 dan α=0,001 mempunyai daerah kritis $X_1 > 10,8276$, sedangkan untuk v=1 dan α =0,005 mempunyai daerah kritis $X_1 > 7,8794$.

Tabel 1. Hasil Uji Frekuensi
(α = 0,001 dan α = 0,005)

HASIL UJI FREKUENSI BIT				
	α = 0,001		α = 0,005	
KATEGORI SAMPEL	% ACAK	% TIDAK ACAK	% ACAK	% TIDAK ACAK
METODE I	94,1	5,9	90,6	9,4
METODE III	53	47	44	56

2) Uji Serial

Tujuan uji serial adalah untuk menentukan banyaknya pasangan bit 00, 01, 10 dan 11 sebagai sub-barisan dari barisan bit sama. Statistk uji yang digunakan adalah:

$$X_2 = \frac{4}{n-1}\left(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2\right) - \frac{2}{n}\left(n_0^2 + n_1^2\right) + 1$$

Dimana n = banyaknya bit dalam rangkaian
n_0 = banyaknya bit "0"
n_1 = banyaknya bit "1"
n_{00} = banyaknya pasangan bit "00"
n_{01} = banyaknya pasangan bit "01"
n_{10} = banyaknya pasangan bit "10"
n_{11} = banyaknya pasangan bit "11"

pengujian menggunakan pendekatan distribusi frekuensi χ^2 dengan derajat kebebasan v =2. Daerah kritis untuk uji Serial dengan v =2 dan α =0,001 adalah $X_2 > 13{,}8155$ sedangkan untuk v =2 dan α =0,005 adalah $X_2 > 10{,}5966$.

Tabel 2. Hasil Uji Serial
(α = 0,001 dan α = 0,005)

HASIL UJI SERIAL				
	α = 0,001		α = 0,005	
KATEGORI SAMPEL	% ACAK	% TIDAK ACAK	% ACAK	% TIDAK ACAK
METODE I	89,3	10,7	83,9	16,1
METODE III	35,9	64,1	27,5	72,5

3) Uji Poker

Uji poker digunakan untuk menentukan barisan bit dengan panjang tertentu muncul dalam barisan bit dengan jumlah yang sama. Statistik uji yang digunakan adalah :

$$X_3 = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} n_i^2\right) - k$$

Dimana n = panjang rangkaian bit yang diuji,
$k = \left\lfloor \frac{n}{m} \right\rfloor$ adalah banyaknya sub barisan yang *non-overlap* dalam rangkaian bit dengan panjang m
n_i = sub barisan dengan panjang m untuk tiap i dimana $1 \le i \le 2^m$

Pengujian menggunakan pendekatan distribusi χ^2 dengan derajat kebebasan $v = 2^m$ -1. Pada penelitian ini digunakan m=3. Daerah kritis untuk uji Poker dengan v =7 dan α =0,001 adalah $X_3 > 24{,}3219$, sedangkan pada v =7 dan dengan α =0,005 diperoleh nilai c^2 = 20,2777 , sehingga daerah kritis untuk uji Poker v =7 dan α =0,005 adalah $X_3 > 20{,}2777$.

Tabel 3. Hasil Uji Poker
(α = 0,001 dan α = 0,005)

HASIL UJI POKER				
	α = 0,001		α = 0,005	
KATEGORI SAMPEL	% ACAK	% TIDAK ACAK	% ACAK	% TIDAK ACAK
METODE I	93,6	6,4	89,6	10,4
METODE III	43,7	56,3	37,3	62,7

4) Uji Run

Uji run dilakukan untuk mengetahui apakah jumlah runtun dalam barisan bit, baik runtun bit "0" maupun runtun "1", dengan panjang yang bervariasi memiliki jumlah yang sesuai dengan postulat keacakan Golomb butir R2.
Statistik uji yang digunakan adalah:

$$X_4 = \sum_{i=1}^{k} \frac{(B_i - e_i)^2}{e_i} + \sum_{i=1}^{k} \frac{(G_i - e_i)^2}{e_i}$$

Dimana B_i = banyaknya runtun "1" dengan panjang *i* untuk tiap $i, 1 \le i \le k$,
G_i = banyaknya runtun "0" dengan panjang *i* untuk tiap $i, 1 \le i \le k$
e_i = $(n-i+3)/2^{i+2}$ adalah banyaknya runtun dengan panjang *i* yang diharapkan muncul dalam rangkaian
k = nilai terbesar dari *i* jika $e_i \ge 5$

Pengujian dengan menggunakan pendekatan distribusi χ^2 dengan derajat kebebasan $v = 2k-2$. Pada penelitian ini digunakan *k*=15. Daerah kritis untuk uji Run dengan v=28 dan α =0,005 adalah $X_4 > 50,9934$.

Tabel 4. Hasil Uji Run
(α = 0,001 dan α = 0,005)

HASIL UJI RUN				
	α = 0,001		α = 0,005	
KATEGORI SAMPEL	% ACAK	% TIDAK ACAK	% ACAK	% TIDAK ACAK
METODE I	90,3	9,7	86,5	13,5
METODE III	59	41	51,4	48,6

Berdasarkan hasil pengujian keacakan rangkaian yang dihasilkan oleh sub kunci algoritma PRESENT, dihasilkan persentase keacakan yang baik, akan tetapi rangkaian yang dihasilkan oleh Metode I memiliki persentase keacakan yang lebih tinggi dibandingkan dengan Metode III, hal ini disebabkan karena Metode I merupakan metode yang paling sederhana untuk memeriksa korelasi antara sub kunci.

5. KESIMPULAN

Dari hasil penelitian yang dilakukan, sub kunci yang dihasilkan oleh algoritma kriptografi PRESENT dapat digunakan untuk menghasilkan suatu rangkaian baru menggunakan metode uji korelasi dari Muhammad Saqib Niaz. Sampel yang digunakan dalam penelitian ini adalah 1000 sampel Key Input yang menghasilkan 1000 file rangkaian dari Metode I dan 1000 file rangkaian dari Metode III. Hasil pengujian keacakan rangkaian yang dihasilkan oleh sub kunci algoritma PRESENT menghasilkan persentase keacakan yang baik, akan tetapi rangkaian yang dihasilkan oleh Metode I memiliki persentase keacakan yang lebih tinggi dibandingkan dengan Metode III, hal ini disebabkan karena Metode I merupakan metode yang paling sederhana untuk memeriksa korelasi antara sub kunci.

DAFTAR PUSTAKA

A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, Y. Seurin, dan C. Vikkelsoe. 2007. PRESENT: An Ultra-Lightweight Block Cipher.

Bruce Schneier. 2009, *Applied Cryptography, Second Edition*. John Wliey & Sons Inc

Menezes, J. Alfred et al. 1996, *Handbook ofApplied Cryptography*.

Niaz , Muhammad Saqib. 2007, Analysis and Design of Correlation Testing Methods for Expanded Subkeys. Sichuan University, P.R. China.

Stallings, William. 2005, *Cryptography and Network Security Principles and Practices, Fourth Edition*, *Prentice Hall*.