Kriptografi Modern Pada Socket Programing dengan Media Device Android

Whilda Chaq 13511601
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13511601@std.stei.itb.ac.id

***Abstract*—pada makalah ini akan menunjukkan implementasi kriptografi modern yang diimplementasikan pada dua buah device android yang terhubung dalam satu jaringan local. Satu device berperan sebagai server yang bertugas menerima koneksi dari client dan menerima file image yang terenkripsi. setelah file yang terenkripsi diterima kemudian didekripsi untuk mendapatkan file image yang asli. Kebalikan dari server, client berperan untuk request koneksi kepada server lalu mengirim file image yang telah di enkripsi. Client server terhubung menggunakan socket. Selain itu, penulis akan menunjukkan perbandingan algoritma enkripsi cipher block dengan mode ECB,CBC, dan mode baru yang mengkombinasikan beberapa teknik enkripsi.**

***Index Terms* — android, socket programming, image, enkripsi, histogram.**

I. PENDAHULUAN

I.I Latar Belakang

Dewasa ini perkembangan teknologi sangat lah pesat. Berbagai device baru muncul seiring kebutuhan aplikasi yang semakin bervariasi. Selain hal itu juga para profesional kerja dituntut cepat dalam memperoleh informasi atau membagi informasi. Dengan dimikian tidak heran jika aplikasi mobile sangat digemari pasar dilihat dari device mobile sendiri yang selalu dibawa dan simpel cara pemakaiannya. Melihat potensi yang menjanjikan di dunia mobile maka banyaklah terlahir start up company yang mulai membangun aplikasi berbasis mobile yang dikembangkan berbagai platform. Aplikasinyapun bervariasi dari aplikasi perhitungan sederhana sampai aplikasi yang menghubungkan antar perangkat untuk menjalankannya (contoh : game, chatting, dll).

Tren saat ini adalah migrasi besar – besaran yang semula aplikasi berbasis desktop menjadi aplikasi mobile. Kebutuhan network menjadi kebutuhan primer untuk membangun sebuah aplikasi terdistribusi. Trafic pertukaran data pada jaringan akan membesar seiring bertambahnya pengguna applikasi mobile. Masalah keaman data dan privasi data menjadi fokus masalah pada tren ini.
Aplikasi yang menyambungkan berbagai device pada suatu jaringan (LAN, WAN, WLAN, dll) memiliki kelebihan dan kekurangan. Kelebihannya adalah aplikasi akan lebih menarik dan lebih interaktif tentunya namun jika sebuah aplikasi yang digunakan adalah aplikasi yang bersifat private dan penting maka banyak cara untuk mengambil data yang saling dipertukarkan pada sebuah jaringan.

I.II Rumusan Masalah

Masalah yang ditemukan adalah bagaimana cara untuk mengamankan data yang mengalir dari suatu device ke device lain dalam sebuah jaringan wireless sehingga tidak ada pihak ke 3 yang dapat mencuri atau mendapatkan data / informasi yang sedang dipertukarkan dan juga mengubah data yang sedang di pertukarkan.

I.III Batasan Masalah

Masalah yang akan difokuskan pada makalah kali ini adalah menerapkan algoritma enkripsi modern dengan metode block cipher pada sebuah aplikasi yang berjalan pada 2 buah device android yang terhubung dalam sebuah jaringan wireless lokal atau 1 buah device android dengan 1 buah laptop sebagai server untuk implementasinya. Wireless yang digunakan adalah wireless jaringan ad-hoc dari laptop. File yang dipertukarkan adalah file image. Sehingga mempermudah untuk membandingkan hasil enkripsi dari kombinasi algoritma enkripsi yang digunakan. Pada makalah ini akan berfokus pada pengujian algoritma enkripsi cipher block sederhana. Algoritma dekripsi tidak dituliskan secara explisit.

IV. LINGKUNGAN PENGEMBANGAN

Lingkungan pengembangan :

1. Software
    a. Bahasa Pemrograman : Java
    b. IDE : Eclipse dan Netbeans
    c. Android SDK
    d. ADT Plugin
    e. JVM (java Virtual Machine)
2. Hardware
    a. 1 buah Laptop (Wireless adapter)
    b. 1 atau 2 buah device android

V. ARSITEKTUR IMPLEMENTASI

Bagan 1 Arsitektur Implementasi

Pada prinsipnya, terdapat 2 buah device android yang terhubung pada suatu jaringan wireless. Media wireless. Paga gambar diatas dimodelkan ada sebuah file yang bernama x.jpg. sebuah aplikasi ingin mengirimkan x.jpg tersebut ke device yang berbeda. Untuk menjamin kerahasiaan file yang dikirim menggunakan socket programming maka file yang ingin di kirimkan akan di enkripsi terlebih dahulu menggunakan metode kriptografi modern. Ketika file yang terenkripsi sudah diterima, maka pihak penerima akan mendekripsikannya terlebih dahulu sebelum mendapatkan file yang sebenarnya. Apabila di analogikan sebagai program client – server, maka aplikasi yang mengirim file image adalah client dan aplikasi yang menerima file image adalah server.

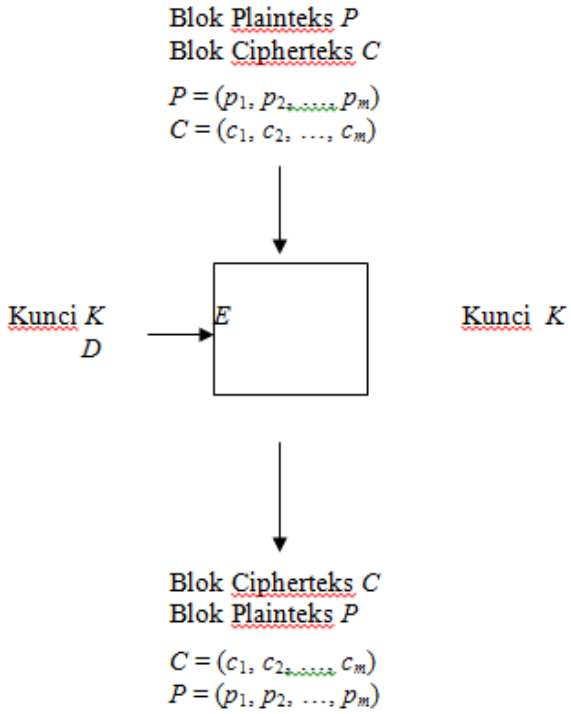
II. DASAR TEORI

II.I *Cipher* Blok (*Block Cipher*)

[1]Pada *cipher* blok, rangkaian bit-bit plainteks dibagi menjadi blok-blok bit dengan panjang sama, biasanya 64 bit. Algoritma enkripsi menghasilkan blok cipherteks yang – pada kebanyakan sistem kriptografi simetri – berukuran sama dengan blok plainteks.

Dengan blok *cipher*, blok plainteks yang sama akan dienkripsi menjadi blok cipherteks yang sama bila digunakan kunci yang sama pula. Ini berbeda dengan *cipher* aliran dimana bit-bit plainteks yang sama akan dienkripsi menjadi bit-bit cipherteks yang berbeda setiap kali dienkripsi.

Skema enkripsi dan dekripsi dengan *cipher* blok digambarkan pada model berikut :



Bagan 2 Skema enkripsi dan dekripsi dengan cipher block

- Mode Operasi Cipher Blok

Plainteks dibagi menjadi beberapa blok dengan panjang tetap. Beberapa mode operasi dapat diterapkan untuk melakukan enkripsi terhadap keseluruhan blok plainteks. Empat mode operasi yang lazim diterapkan pada sistem blok *cipher* adalah:

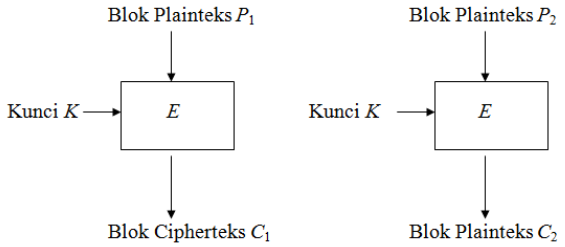
1. *Electronic Code Book* (*ECB*)
2. *Cipher Block Chaining* (*CBC*)
3. *Cipher Feedback* (*CFC*)
4. *Output Feedback* (*OFB*)

Hanya 2 mode operasi saja yang akan diimplementasikan dalam makalah ini, yaitu *ECB* dan *CBC*.

- Electronic Code Book (ECB)

Pada mode ini, setiap blok plainteks dienkripsi secara individual dan independen. Secara matematis, enkripsi dengan mode *ECB* dinyatakan sebagai $C_i = E_K(P_i)$ dan dekripsi sebagai $P_i = D_K(C_i)$ yang dalam hal ini, P_i dan C_i masing-masing blok plainteks dan cipherteks ke-*i*.

Model berikut memperlihatkan enkripsi dua buah blok plainteks, P_1 dan P_2 dengan mode *ECB*, yang dalam hal ini *E* menyatakan fungsi enkripsi yang melakukan enkripsi terhadap blok plainteks dengan menggunakan kunci *K*.



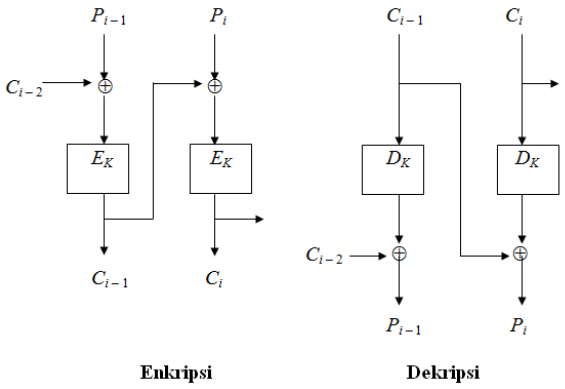
Bagan 3 Skema enkripsi dan dekripsi dengan mode ECB

- Cipher Block Chaining (CBC)

Mode ini menerapkan mekanisme umpan-balik (*feedback*) pada sebuah blok, yang dalam hal ini hasil enkripsi blok sebelumnya di-umpan-balikkan ke dalam enkripsi blok yang *current*.

Dengan mode *CBC*, setiap blok cipherteks bergantung tidak hanya pada blok plainteksnya tetapi juga pada seluruh blok plainteks sebelumnya.

Gambar berikut memperlihatkan skema mode operasi *CBC*.



Bagan 4 Skema enkripsi dan dekripsi dengan mode CBC

II.II *VIGENERE CIPHER*

Setiap baris di dalam bujursangkar menyatakan huruf-huruf cipherteks yang diperoleh dengan *Caesar Cipher*. Kunci: $K = k_1k_2 \ldots k_m$ k_i untuk $1 \le i \le m$ menyatakan jumlah pergeseran pada huruf ke-i.

Karakter cipherteks: $c_i(p) = (p + k_i) \bmod 26$

Konsep key pada vigenere chiper akan digabungkan dengan block cipher sehingga key tidak harus berukuran sama dengan block yang akan dienkripsi.[2]

II.III *SOCKET PROGRAMING JAVA (ANDROID)*

Kelas Jaringan termasuk dalam standar Java dianggap sebagai salah satu solusi yang lebih baik dan lengkap diantara bahasa modern lainnya, dengan sintaks yang sederhana dan semantic.

Kelas Socket pada java didefinisikan sebagai endPoint dalam standard TCP connection. Kelas Socket mengimplementasikan method yang menangani semua kebutuhan untuk keperluan komunikasi jaringan bersama TCP communication.

Langkah yang diperlukan membangun sebuah komunikasi menggunakan socket programing adalah pihak server yang me-listen koneksi pada sebuah port dan client yang request koneksi kepada address dari server dan port yang digunakan. Setelah koneksi terbentuk, komunikasi dapat dimulai.[4]

II.IV *ANDROID PROGRAMMING (JAVA)*

Android adalah sistem operasi yang berbasis Linux untuk telepon seluler seperti telepon pintar dan komputer tablet. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

Karena Android berbasis Linux dan sudah menjadi pengetahuan kita bahwa linux merupakan operating sistem yang terkenal karena open source. Maka memudahkan pengembang untuk menciptakan sebuah aplikasi. Selain itu, dasar mengembangkan android adalah bahasa pemrograman java yang sudah familiar bagi pengembang aplikasi mobile maupun desktop.[3][5]

II. V *HISTOGRAM*

Sebuah histogram mengilustrasikan bagaimana pixel dalam gambar didistribusikan oleh grafik jumlah pixel pada setiap tingkat intensitas warna. Histogram menunjukkan detail dalam shadow (ditampilkan di bagian kiri histogram), midtones (terlihat di tengah), dan highlight (ditampilkan di bagian kanan) Histogram dapat membantu Anda menentukan apakah gambar memiliki cukup detail untuk membuat penilaian.

Histogram juga memberikan gambaran singkat dari berbagai tonal gambar, atau jenis gambar. Sebuah gambar low-key memiliki detail terkonsentrasi dalam shadow. Sebuah gambar yang high-key memiliki detail yang terkonsentrasi di highlights. Dan, gambar rata-kunci memiliki detail yang terkonsentrasi dalam midtone. Sebuah gambar dengan rentang tonal penuh memiliki

beberapa pixel di semua bidang. Mengidentifikasi area tonal membantu menentukan koreksi tonal yang tepat. [6]

III. IMPLEMENTASI

III.I Target Gambar

Pada makalah ini, image yang menjadi target algoritma enkripsi adalah gambar berikut :

Bagan 5 Target enkripsi dengan nama file Lenna.jpg (512 x 512 px)[7]

Gambar ini dipilih karena pada praktisi IT berpendapat gambar tersebut memiliki syarat sarat gambar yang baik untuk dioleh berdasarkan persebaran warna, contrast warna dan pencahayaan. Berikut histogram dari gambar diatas :

Bagan 6 Histogram Lenna.jpg

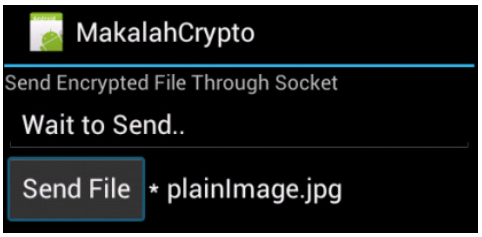
III.II Aplikasi Android

Mengembangkan aplikasi sederhana pada android sebagai client yang bertugas untuk mengenkripsi gambar kemudian mengirimkan ke server melalui socket.

Spesifikasi software client yang digunakan adalah :

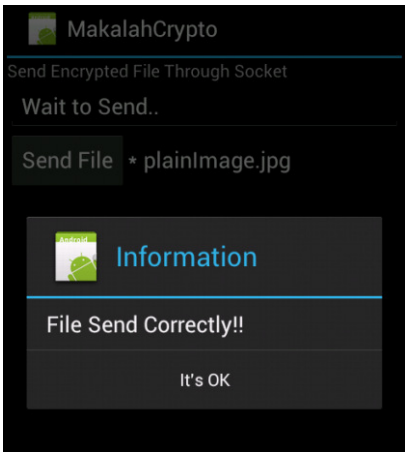
1. Android 4.0
2. API Android 14
3. Jaringan ITB
4. Kebutuhan socket
    a. IP : 167.205.85.177
    b. PORT : 1235

Berikut merupakan Screenshot dari interface pada device android :



Bagan 7 Interface program Android

Ketika user menekan tombol Send File, maka yang dilakukan aplikasi ini adalah mengenkripsi file plainImage.jpg dengan algoritma tertentu menjadi sebuah file baru dengan nama cipherImage.jpg. file plainImage.jpg harus diletakkan sedemikian rupa pada device android sehingga untuk mengakses file tersebut cukup mendefinisikan path "/sdcard/plainImage.jpg". Setelah enkripsi selesai aplikasi akan membuat socket agar terhubung dengan server untuk pengiriman file. akan muncul gambar seperti dibawah ini jika pengiriman telah selesai :



Bagan 8 Pemberitahuan File telah terkirim

Pada sisi server, yang dilakukan cukup melisten sebuah port sampai ada request dari client. Setelah terdapat request koneksi dari client, socket antara client dan server terbentuk kemudian server menunggu file kiriman dari client. File yang diterima disimpan pada root direktori server dengan nama sockImage.jpg kemudian dilakukan dekripsi dan hasil disimpan dengan nama plainImage.jpg. sehingga seolah-olah client dan server mengirim dan menerima file yang sama. Perlu diperhatikan bahwa hasil plainImage.jpg yang diterima juga di hard code pada path "/sdcard/plainImage.jpg",

Spesifikasi software server yang digunakan adalah :

1. Android 4.0
2. API Android 14
3. Jaringan ITB
4. Kebutuhan socket
    a. IP :167.205.56.27
    b. PORT : 1235

III.III Algoritma Enkripsi

Ide enkripsi dan dekripsi pada makalah ini adalah mengenkripsi setiap pixel yang direpresentasikan oleh integer RGB. Kemudian dibagi menjadi 4 block, jadi setiap block berukuran 1 byte atau 8 bit.

Algoritma Umum untuk mengenkripsi sebuah file image adalah sebagai berikut :

```
procedure enkripImage(){
input(image);
input(key);
for(int w = 0 ; w  < width(image) ; w++){
        for(int h = 0 ; h < height(image) ; h++){
              int plainRGB = image.getRGB(w, h);
               int cipherRGB = enkripRGB(plainRGB,key);
              image.setRGB(w, h, cipherRGB);

           }
        }
}
```

```
function enkripRGB(plainRGB : integer, Key : byte){
      byte[] result = plainRGB.toArrayByte(4);
      result[0] = blockChiper(result[0], Key);
      result[1] = blockChiper(result[1], Key);
      result[2] = blockChiper(result[2], Key);
      result[3] = blockChiper(result[3], Key);
      int cipherRGB = result.toInt();
      return cipherRGB;
   }
```

A. ECB (Electronic Cipher Block)

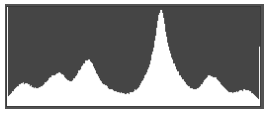
Dengan menggunakan algoritma umum enkripsi file image diatas, kemudian ditambahkan fungsi block chiper sebagai berikut :

```
function byte blockChiper(result : byte, Key : byte){
      result = result XOR Key;
      return result;
}
```

Setelah diterapkan pada target gambar, maka hasilnya adalah sebagai berikut :

Bagan 9 Gambar terenkripsi dengan ECB

Berikut ditampilkan perbandingan histogram Bagan 5 dan bagan 9:



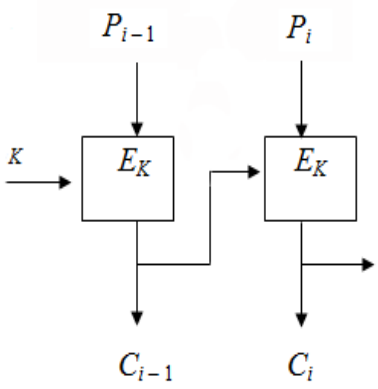
terenkripsi

original

Dari histogram dan hasil gambar menunjukkan gambar sudah mengalami perubahan secara shadow dan highlight namun gambar asli masih dapat dikenali.

Karena semua pixel di enkripsi dengan kunci yang sama, maka perbedaan kunci tidak mengubah hasil enkripsi secara signifikan.

B. CBC (Cipher Block Chaining)
Dengan mengubah skema dari CBC yang biasanya, maka digunakan skema sebagai berikut :



Bagan 10 Skema CBC "baru"

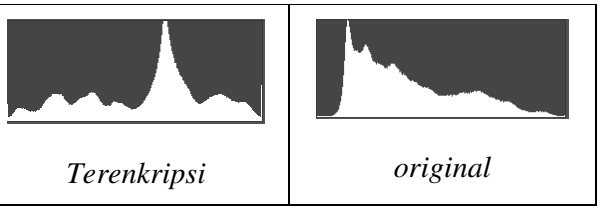
```
function enkripRGB(plainRGB : integer, Key : byte){
      byte[] result = plainRGB.toArrayByte(4);
      result[0] = blockChiper(result[0], Key);
      result[1] = blockChiper(result[1], result[0]);
      result[2] = blockChiper(result[2], result[1]);
      result[3] = blockChiper(result[3], result[2]);
```

```
    int cipherRGB = result.toInt();
    return cipherRGB;
}
```

Hasil yang didapat adalah sebagai berikut :

Bagan 11 Gambar terenkripsi dengan CBC

Berikut ditampilkan perbandingan histogram Bagan 5 dan bagan 11:



Terenkripsi | *original*

Dari histogram dan hasil gambar menunjukkan gambar sudah mengalami perubahan secara shadow dan highlight namun gambar asli masih dapat dikenali.

Histogram untuk hasil gambar dengan metode CBC dan ECB tidak banyak berubah.

Dengan mengganti funsi block cipher menjadi seperti ini,

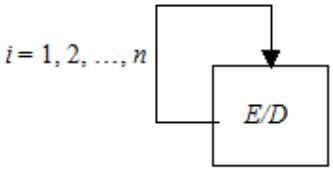
```
function byte blockChiper(result : byte, Key : byte){
    result = (result + Key) mod 256;
    return result;
}
```

Hasilnya pun masih sama, gambar asli masih dapat dikenali.

C. Kombinasi

Untuk memperumit algoritma enkripsi, maka dilakukan modifikasi dan penambahan teknik enkripsi.

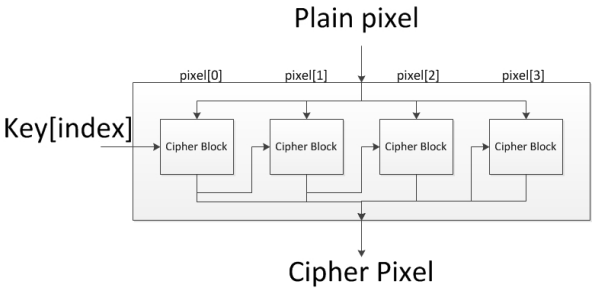
1. Enkripsi Berulang



Bagan 12 Model iterate cipher

Enkripsi berulang diterapkan untuk setiap enkripsi sebuah pixel.

2. Enkripsi Pixel



Bagan 13 Model Enkripsi Pixel

Oerlu diulang adalah Pixel yang direpresentasikan dengan ukuran 32 bit atau setara dengaan 4 byte. Pada implementasinya di pecah menjadi 4 block. Sehingga 1 block berukuran 1 byte.

Enkripsi pixel mirip dengan mode cbc, namun key untuk setiap pixel berbeda, kali ini konsep key yang digunakan seperti konsep key pada vigenere chiper. Namun perlu diingat bahwa panjang block key untuk masuk pada block cipher tetap harus sama dengan panjang block plain.

3. Cipher Block
   Cipher block yang digunakan adalah caesar chiper. Yang dapat dinotasikan

   C = (P + K) mod 256

   Dalam pengujian ini, awalnya penulis memakai operator XOR pada cipher blocknya, namun hasil yang didapat masih kurang maksimal.

Dengan adanya penambahan metode enkripsi, maka algoritma yang didapat adalah sebagai berikut :

```
procedure enkripImage(){
input(image); /* Gambar yang ingin dienkripsi*/
input(key);    /* Key untuk enkripsi */
input(idxKey); /* index dari key */
input(N);   /* Banyaknya perulangan enkripsi*/
for(int w = 0 ; w  < width(image) ; w++){
        for(int h = 0 ; h < height(image) ; h++){
                int plainRGB = image.getRGB(w, h);
                 int cipherRGB =
```

```
                loopEnkripRGB(plainRGB,key[idxKey],N);
                image.setRGB(w, h, cipherRGB);
                idxKey = idxKey + 1;
                if(idxKey == Key.length()){
                    idxKey = 0;
                }
            }
        }
}
```

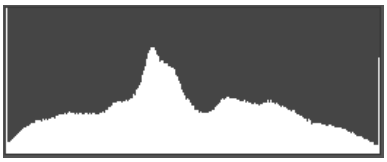
```
function loopEnkripRGB(plainRGB : integer,
                       charKey : byte,
                       nLoop : integer){
    for (int i = 0; i < nLoop; i++) {
        plainRGB = enkripRGB(plainRGB, charKey);
    }
    return plainRGB;
}
```

```
function byte blockChiper(result : byte, Key : byte){
    result = (result + Key) mod 256;
    return result;
}
```

Pengujian dilakukan dengan mengganti nilai N:

- N = 1

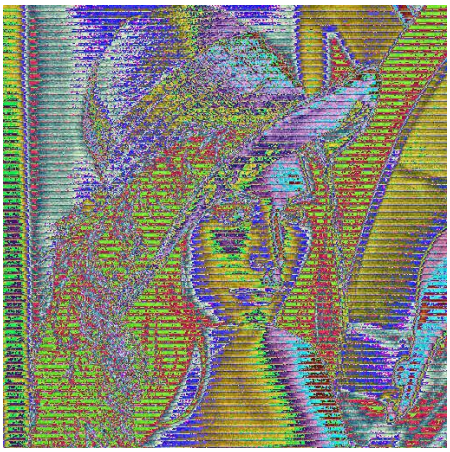
Bagan 14 Hasil enkripsi dengan N = 1



Bagan 15 Histogram bagan 14

Hasil yang didapat dengan N = 1 menunjukkan bahwa hasil gambar terenkripsi masih dapat dikenali. Namun dibandingkan kedua hasil sebelumnya yaitu bagan 9 dan 11. Bagan 14 ini adalah yang paling tidak mirip dengan aslinya. Terlihat juga histogram yang dihasilkan cenderung lebih landai.

- N = 3

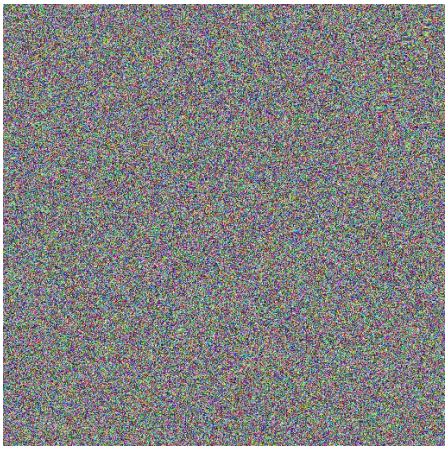


Bagan 16 Hasil Enkripsi dengan N = 3

Bagan 17 Histogram untuk bagan 16

Selanjutnya uji coba menambahkan jumlah pengulangan algoritma enkripsi menjadi 3 kali. Hasil yang didapat semakin jauh dengan gambar asli namun masih mudah untuk dikenali. Histogram yang dihasilkan pun sudah cukup landai.

- N = 5 dst.



Bagan 18 Hasil Enkripsi dengan N = 5

Bagan 19 Histogram untuk bagan 18

Pengujian dilanjutkan dengan memasukkan N yang lebih besar. Dengan harapan bahwa gambar yang dihasilkan lebih jauh dari gambar asli dengan kata lain gambar asli tidak dapat di ketahui dari gambar terenkripsi. Dengan algoritma ini penulis menemukan sebuah kesimpulan bahwa semakin besar N maka hasil yang diperoleh akan menjadi lebih bagus(gambar hasil tidak dapat dikenali dan histogram yang dihasilkan tebilang landai). Namun kesimpulan ini hanya berlaku untuk N = 1,2,3,4,5 karena untuk N lebih besar dari 5 hasil yang diperoleh tidak memiliki perbedaan yang signifikan, justru hasilnya terlihat random namun dengan kualitas yang tidak jauh berbeda.

Untuk menyembunyikan file gambar sampai benar benar tidak dapat ditebak gambar aslinya minimal menggunakan 4 kali perulangan enkripsi tiap pixel. Dapat dilihat pula berdasarkan histogram dari hasil enkripsi. N yang semakin besar, file gambar semakin tidak diketahui gambar aslinya seiring histogram yang menjadi lebih landai (tidak memiliki banyak gunung dan lembah).

IV. KESIMPULAN

Setelah menyelesaikan makalah ini, penulis dapat menarik beberapa kesimpulan, antara lain :

1. Dewasa ini menjadi puncak perkembangan aplikasi pada dunia mobile .
2. Algoritma enkripsi CBC dengan menggunakan operasi XOR sangat mudah dipecahkan.
3. Kelemahan menggunakan algoritma XOR pada enkripsi iterate cipher adalah hanya memiliki 2 kemungkinan, jika N ganjil maka mendapatkan hasil yang berbeda g=dengan gambar asli dan jika N genap maka hasil tidak berubah dari gambar asli.
4. Serumit apapun design sebuah algoritma, yang paling dominan terhadap hasil adalah proses perulangan algoritma enkripsi (iterate cipher). Jadi design enkripsi yang sederhanapun dapat mendapatkan hasil yang sulit ditebak jika di terapkan berulang kali pada sebuah plain. Berlaku untuk segala jenis berkas namun tidak berlaku untuk design algoritma dengan operasi XOR seperti yang sudah dijelaskan pada nomor 3.

REFERENCES

[1] http://kur2003.if.itb.ac.id/file/Cipher%20Blok.doc diakses pada 19 Maret 2013 pukul 20.15 WIB

[2] http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2012-2013/Algoritma%20Kriptografi%20Klasik_bag2%20(2013).ppt diakses pada 15 Februari 2013

[3] http://developer.android.com/training/basics/firstapp/index.html diakses pada 7 Maret 2013 pukul 09.00 WIB

[4] W. Richard Stevens: UNIX Network Programming, Volume 1, Second Edition: Networking APIs: Sockets and XTI, Prentice Hall, 1998.

[5] http://blog.shiftyjelly.com/2013/02/20/why-android-first/ diakses pada 5 Maret 2013

[6] http://help.adobe.com/en_US/photoshop/cs/using/WSfd1234e1c4b69f30ea53e41001031ab64-768da.html diakses pada 26 Maret pukul 13.00 WIB

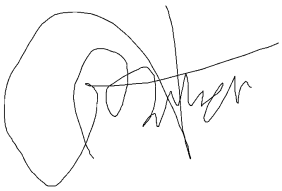
[7] http://www.name-list.net/img/images.php/Lenna_5.jpg diakses pada 24 Maret 2013 pukul 09.00 WIB

[8] http://stackoverflow.com/questions/7122325/socketexception-permission-denied diakses pada 14 Maret 2013 pukul 20.05 WIB

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, 26 Maret 2013



Whilda Chaq (13511601)